Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 136 (2025) 290–295

www.elsevier.com/locate/procedia

35th CIRP Design 2025

Fostering Model Reuse in Model-based Systems Engineering using Knowledge Graphs

Roman Stephan[a], Thomas Schumacher[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)531 72 2271. *E-mail address:* inkermann@imw.tu-clausthal.de

Abstract

The increasing spread of MBSE is associated with the issue of reusing the knowledge contained in different system models in different development projects. Existing approaches for the reuse of system models or elements of these can be classified into framework-based, retrieval-based, and pattern-based. While framework- and pattern-based approaches require standardized modelling methods, retrieval-based approaches are based on merging of different system models. This is not supported by current MBSE-tools. In this contribution we investigate how knowledge graphs can be generated from various SysML system models and whether and how new knowledge, e.g., about cause-effect relationships or structural patterns, can be determined.

Keywords: knowledge graph; labeled property graph, model reuse; mode-based systems engineering; transformation concept

1. Introduction

Driven by an increasing amount of functional requirements, more stakeholders and their specific interests and views upon the evolving system must be involved in modern engineering projects [1]. Systems Engineering addresses these challenges as a transdisciplinary and integrative approach to realize complex systems [2]. While traditional Systems Engineering is mainly based on documents, Model-based Systems Engineering (MBSE) focuses on the creation of an integrated and coherent system model as a single source of truth [3] and captures knowledge about systems requirements, architecture, and verification comprehensively [4,5]. Different views of the system are integrated and can be automatically generated and adapted to the specific requirements of the stakeholders [6]. Thus, MBSE improves communication and the common understanding of the evolving system within the engineering process. Moreover, it enables better tracking of information and reuse of models [1]. Despite the advantages, MBSE is facing challenges and limitations which can be classified in a model life cycle. This life cycle consists of model generation, model use & analysis and model reuse & adaption and the challenges and limitations are listed in Table 1.

Table 1. Challenges and limitations of MBSE assigned to the life cycle phases of a model

Model life cycle phase	Challenge/Limitation
Model generation	• Inconsistency between models generated by different engineers e.g., in terms of diagram structure and semantics because no standardized rules for applying the modeling language SysML exist [3,5,7]. Multiple ways to model the same diagram result in challenges when combining or reusing models across engineering projects.
Model use & analysis	• MBSE models alone are insufficient to fully describe the product [4,5,8,9]. • Lack of presentation of conceptual, procedural or metacognitive knowledge like design principles and approaches or development strategies [4].

10.1016/j.procir.2025.08.051

	• Individuals that are not experienced in MBSE struggle to understand or work with the more formal models [4,5,10].
Model reuse & adaption	• As different model views contain a multitude of elements, managing the large amounts of generated data is challenging during the analysis and model-reuse impacting readability and practical representation, especially for complex products [8,9]. • Reuse of models is insufficiently supported in early design stages like system architecture and design validation, highlighting a significant need for knowledge capture and reuse in MBSE [10]. • MBSE specific knowledge is often lost between projects meaning there is insufficient reuse of existing models in new projects [11].

While these challenges and limitations refer to the different life cycle phases of models in engineering processes, this paper focusses on the challenge of model-reuse and adaption. Model-reuse can ensure a common understanding of the system model as well as time efficient and consistence modeling across projects [12]. The objective is to investigate, how knowledge graphs can support the tasks of model reuse and knowledge capturing.

1.1. Knowledge Management and Model Reuse in Model-based Systems Engineering

Knowledge Management (KM) is defined as the identification and utilization of collective knowledge [13]. Efficient KM should establish a systematic process including knowledge creation and capturing, sorting and storing, analyzing and changing as well as retrieving and applying over time [13–15]. Knowledge reuse in MBSE is the process of being able to find and use shared knowledge [13]. Reuse is characterized by the degree of reuse e.g., component or system level, or by the level of preparedness of the reuse e.g., ad hoc or systematic reuse [16].

Model-reuse in MBSE focuses on the reuse of the inherent tacit and explicit knowledge within the developed system model. Tacit knowledge summarizes knowledge based on experience and actions such as know-how and mental models [13]. Knowledge which is communicated or written, for instance facts within documents, is summarized as explicit knowledge [13,16]. By using knowledge graphs, both explicit and implicit knowledge can be stored and analysed to a certain extend. Data and facts are represented as nodes or edges. The type and number of nodes and edges of the graph reflect the applied tacit knowledge. The degree of reuse of system models can range from the reuse of the complete model to only single elements of a model [17]. A systematic reuse of models focuses on the intentional design of elements to be reused later meaning they are likely to satisfy a broader range of requirements. It is important to ensure that the adaptation of existing models doesn't generate more effort than the creation of the required new model [16]. When reusing elements without an systematic approach the type, scope and the exact changes in the environment needs to be assessed [7] in order to properly implement the components.

1.2. Objectives and Focus of Research

The focus of this research is to support the ad-hoc model-reuse with the use of knowledge graphs. Knowledge graphs present a promising approach to support model-reuse in early design stages due to their semantic structure and capability to link or expand data. Moreover, the management of knowledge is improved based on a variety of methods available like knowledge graph completion and querying of graphs. The objective of this paper is to investigate suitable approaches for transferring SysML models into a knowledge graph in order to generate new knowledge through knowledge graph modification and querying for model-reuse. This results in the following research questions:

- What is a suitable approach to transfer SysML models into knowledge graphs (labeled properties graphs) and what information needs to be transferred to support model reuse?
- Which methods are necessary to integrate SysML models of different subsystems into one knowledge graph to improve the efficiency of model-reuse?
- How can the enhancement of knowledge of the SysML model based on model integration and interface analysis be achieved?

By answering these questions, the paper contributes to the efficient model reuse in MBSE. The paper is structured as followed. In chapter two a definition of knowledge graphs is given as well as an overview on existing knowledge graphs applications. In chapter three the used process for transforming SysML models into a knowledge graph is described. Chapter four describes the case study applying the concept. In chapter five the limitations of the work are discussed. Chapter six presents the summary and further research work.

2. Knowledge Graphs in Engineering

A knowledge graph is defined as a multi-relational graph consisting of entities and relations [18]. Here entities are visualized as nodes and relation between two nodes as different types of edges. Each edge is represented as a triple consisting of a head entity, a relation and a tail entity, $(h, r, t) \in \mathcal{F}$. Each of these triplets is referred to as a fact $\mathcal{F}$, which in total form the knowledge graph [19]. A knowledge graph acquires and integrates information into an ontology and applies a reasoner to derive new knowledge [20]. The relational data is stored in a graph database system and can be accessed with the use of query languages [21]. This form of storage allows the handling of high amounts of data [22]. Two of the most often used graph data models[1] are Resource Description Framework (RDF) and

[1] In this this model, the data is modeled and presented as a graph. Here the data and the relation between them have the same level of importance. Other data models include relational or object-oriented models [23].

Labeled Property Graph (LPG) which define the structure and relationships of the stored data [24]. Both models allow the connection of data from different sources in an extendable way. The RDF stores data as triplets consisting of subject, predicate and object. Each triplet describes the connection between the subject and the object (nodes) with the use of the predicate (edge) [23]. In an LPG, the nodes as well as the edges can be labelled and thus can have any number of attribute-value pairs. It is also possible to have several edges between any two nodes [23] or edges with one and the same node. To ensure that all data relevant for analysis and reuse is retained in the transformed knowledge graph, LPGs will be used in this research work.

2.1. Established use cases of Knowledge Graphs in MBSE

Knowledge graphs are already used in MBSE. Lu et al. (2022) developed an ontology-meta-model for MBSE models to ensure model consistency. An RDF ontology knowledge graph was created and used to validate the correctness and completeness of the meta-ontology. It also served as a basis for the development of new MBSE models using the ontology [25]. Smajevic et al. (2021) used LPGs to analyze conceptual ArchiMate models and provided a cloud-platform for the transformation, visualization and analysis of graphs [26]. Glaser et al. (2022) analyzed ArchiMate models for bad modeling practices (smells) using a LPG and developed a plugin in ArchiMate visualizing the graph and the smells report [27].

Petnga (2019) used an LPG to analyze architecture models and identify defects such as missing satisfied or verified relationships for a requirement within the transformed SysML model. Questions regarding completeness, correctness and consistency of the graph were defined and answered by querying the knowledge graph. Relevant information to be transformed in the graph was determined on the basis of these questions [28]. This work was extended by Schummer et al. (2022) to include structure and behavior diagrams of a satellite. Twenty-two questions about the structure and use of the satellite as well as *what if analysis* were answered using a knowledge graph [29]. Faheem et al. (2023) focused on the robustness of the SysML model. The aim was to identify errors more quickly during the concept phase by performing cause-and-effect analysis of a new knowledge graph enriched by already identified errors [21]. Mandel et al. (2023) developed and assessed a methodological approach to increase of maturity levels of system of objectives in the development process. Here knowledge of prior products is visualized in a RDF graph and severs as a reference system for future product generations [30]. Fu et al. (2021) transformed and integrated some SysML elements in one LPG in order to support the efficient reuse across products. An entity alignment strategy and model fusion were used [31]. Although knowledge graphs have been applied in the MBSE as well as for transforming certain SysML models, there is yet to be an integration of models with the focus on reuse within the product development process. In order to utilize the advantages of the transformed SysML models, a concept for transformation, integration and analysis for reuse is presented after the structured approach for the general creation of knowledge graphs.

2.2. KG Development Process

A knowledge graph development process by Tamašauskaitė et al. (2023) is shown in Figure 1. It provides a unified approach based on a structured literature review in order to capture all relevant steps for creating and maintaining a knowledge graph. The first step consists of identifying the data as well as the domain of interest and methods of data acquisition for example by data mining. In the second step the ontology is defined, the relevant knowledge is extracted in the third step. Entities can be extracted by named-entity recognition (NER), relations by using NLP[2] methods such as semantic role labeling or neural information extractions. In the fourth step the quality of the extracted data is ensured, removing redundancy, contradictions and ambiguity in the data set. After this the data is mapped onto the ontology the graph is enriched establishing new relations for example with the use of sorted neighborhood methods (SNMs). In step five the database is set and the visualization and use of the graph is ensured. The last step consists of methods keeping the graph updated when false knowledge is found or new data is to be integrated. [33]

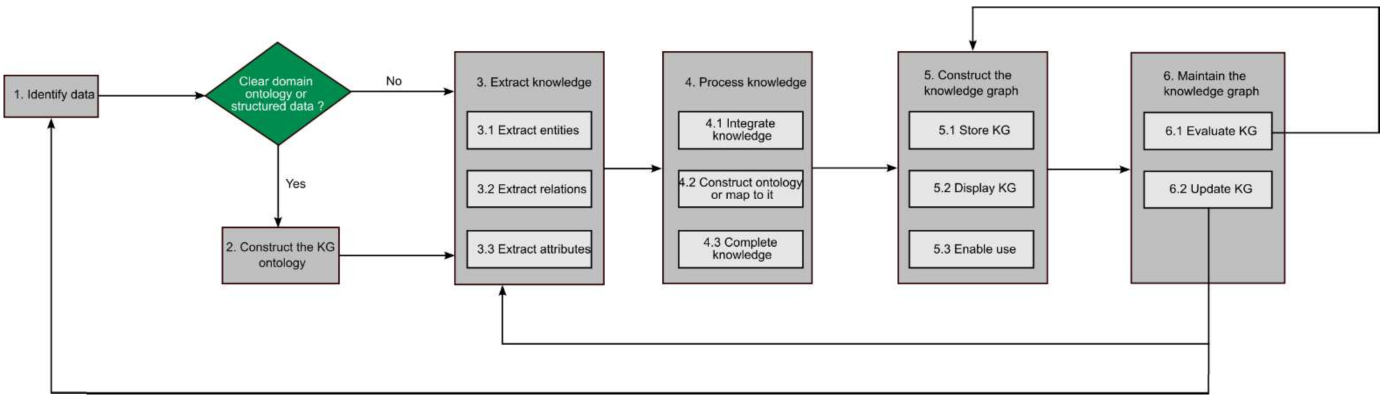

Figure 1: Knowledge graph development process, based on Tamašauskaitė et al. (2023)

[2] The aim of natural language processing (NLP) is to acquire knowledge by extracting semantics from text. The procedure involves processing natural language to analyze, interpret and generate language data models [32].

3. Concept to support SysML Model-reuse based on KG

In order to answer the first research question, the development process by Tamašauskaité is now applied to the concept of creating and analyzing an integrated knowledge graph consisting of SysML models for model reuse. Since the SysML models are already available in a structured form, the concept begins with the creation of the ontology. After this the extraction of the data, the integration of the extracted data sets into one as well as the visualization of the knowledge graph and reuse case are described.

3.1. Construction of the Knowledge Graph Ontology

Although SysML models already offer a high degree of structure, the creation of a knowledge graph requires a clear definition of a meta-model, the used ontology. This ontology depends on the number of models to be transferred. It defines the structure of the data to be transferred per diagram of which there are nine in total in SysML. Here the focus is on block definition (bdd) and internal block (ibd) diagrams. The proposed ontology is based on the structure by Schummer et al. (2022) and is shown in Figure 2. In the later constructed knowledge graph, instead of the value of the attribute 'xmi:type' (uml:Port, uml:Association etc.) the value of the attribute 'name' will be shown. This is indicated in the nodes in the diagram but is not fully displayed for a better overview. The legend on the left shows which attributes serve as the basis for creating the ontology.

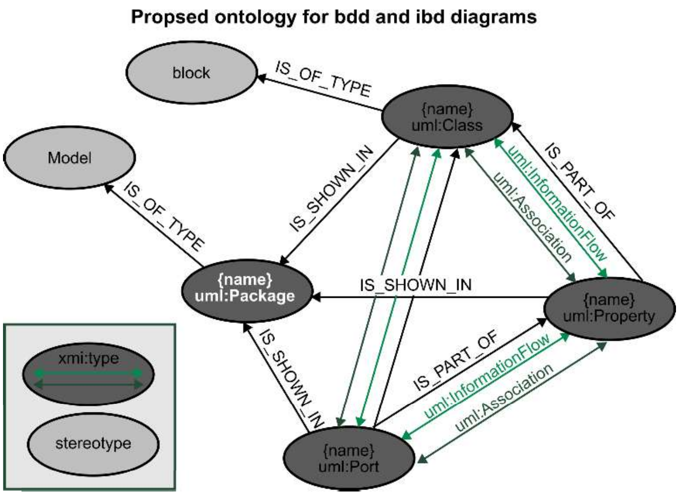

Figure 2: Ontology for bdd and ibd, based on Schummer et al. (2022)

3.2. Extract knowledge – Model Transformation

The structure of the data to be extracted as well as the ontology has been developed on the basis on the XML Metadata Interchange version 2.1 (XMI 2.1) structure. An overview of the relevant date is shown in Figure 3. The selected information serves as a basis for further information which can be extracted from the XML file. Attributes which are not shown in the ontology in Figure 2**Error! Reference source not found.** are still included in the knowledge graph data base but are no longer visible for better clarity.

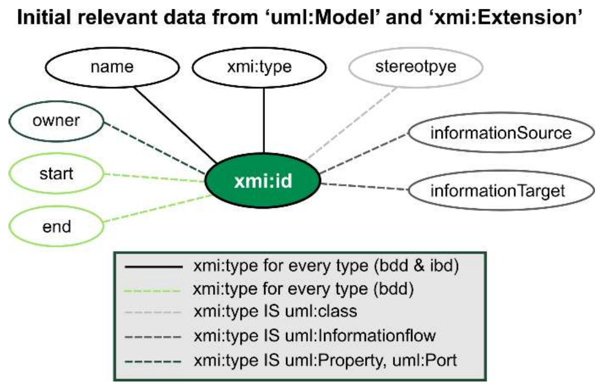

Figure 3: Relevant data extraction from bdd and ibd diagrams

3.3. Process knowledge – Model Integration

Once the relevant data has been extracted from the respective xml document, the data is merged into one data set in the step of processing the knowledge. Here, in reference to the second research question, several data sets with different qualities may exist. Therefore, depending on the quality similarity, certain methods need to be used in order to integrate the data. If the names of the model elements, connections and the types of blocks and connections match, the data integrated data set can be created by combining the data and duplicates are deleted. If this is not the case, there are various steps that are carried out for integration. The first step is to standardize the character formatting. In this way, problems caused by upper- and lower-case letters and possible special characters or spaces can be solved. When inconsistencies are still existing, methods of the Concept of Nearest Neighbors (C-NN) and methods of entity resolution are then applied. C-NN is using a symbolic and instance-based methodology which identifies clusters of similar entities and is done based on shared graph patterns [34]. Profound predictions about similarity can be made and the correct term can be selected. In order to guarantee the quality of the integration, a semantic comparison must be carried out in addition to the C-NN concept. In order to enable integration at a semantically secure level, methods of entity resolution (ER) are used. ER refers to the process of identifying and matching entities in a database that refer to the same real-world entity, despite differences in how they are represented [35]. String matching, as one method for entity resolution, involves identifying occurrences of a substring within a larger string or comparing two strings to determine their similarity [36]. Since the information to be compared is already available as strings, this method can be used to solve conflicts in terms of meaning. The result of this step is a data set which can be displayed as a graph in the model visualization in the next step.

3.4. Construct the knowledge graph – Model visualization

Before the model visualization can be done, the dataset must be transformed into a format suitable for visualization. Initially, a transformation into a tool-independent format should be pursued. This allows for flexibility in the type of visualization. The transformation for model visualization is divided into the suitable transformation into nodes and edges. Nodes can be relatively easily extracted and created from the available list.

The application of the previously defined ontology is more challenging, as different entities are connected depending on the xmi:type. Additionally, data which is not visualized through the ontology must be assigned to specific nodes and edges. To ensure tool independence, edges are stored as triples. Figure 4 shows an excerpt of the code that generates the triples for all values of the element xmi:type, 'uml:Class', 'uml:Package' and 'uml:InformationFlow'.

```python
# Triplet for uml:Class
for element in model_elements3:
    if element['xmi:type'] == 'uml:Class' and element.get('name') and element.get('stereotype'):
        kanten.append((element['name'], 'IS_OF_TYPE', element['stereotype']))

# Triplet für uml:Package
for element in model_elements3:
    if element['xmi:type'] == 'uml:Package' and element.get('name') and element.get('stereotype'):
        kanten.append((element['name'], 'IS_OF_TYPE', element['stereotype']))

# Triplet for uml:InformationFlow
id_to_name = {element['xmi:id']: element['name'] for element in model_elements3 if 'name' in element}

for element in model_elements3:
    if element['xmi:type'] == 'uml:InformationFlow':
        information_source = element['informationSource']
        name = element['name']
        information_target = element['informationTarget']
        # Update ID with name of the Target and Source
        updated_source = id_to_name.get(information_source, information_source)
        updated_target = id_to_name.get(information_target, information_target)
```

Figure 4: Excerpt of the creation of triples

After the nodes and edges have been stored in a tool-independent format, they can be transformed into tool-dependent formats. Since Neo4j is used for the visualization of the knowledge graph in this concept, the transformation of the nodes and edges into cypher queries are carried out. Then the knowledge graph is created using the Neo4j API.

3.5. Maintain the knowledge graph – Analysis for reuse

In the final step, the knowledge graph is evaluated and updated. Here, knowledge graph completion methods can be used to create links between the SysML elements which were not present in the original model. In the course of this, the integrated knowledge can also be reused as a way of evaluating the graph. In the course of this concept this mainly involves querying certain subgraphs and checking the functionality of the ontology. The use-case specific query supports the development of new models that are linked to the existing ones. The evaluation of the ontology is particularly important in the early phases of product development for the development of consistency between different models.

4. Case Study

The application of the concept presented was carried out using two models of a robotic lawnmower. the models on which the transformation and integration are based are shown

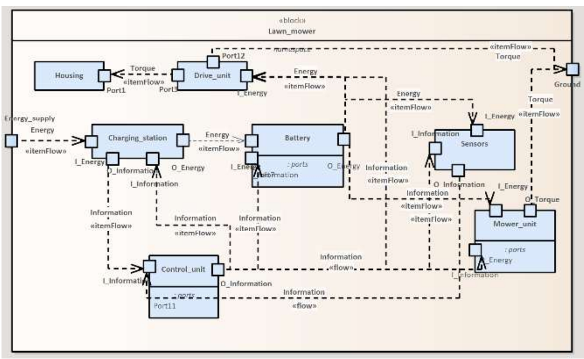

Figure 5: Internal block diagram (ibd) of the robotic lawn mower

in figures 5 and 6. In addition to transformation and integration, the case study also includes an initial reuse case. The ibd and bdd models each contain different information regarding the number of components (sensors, lower housing, upper housing, inner housing only available once) and the number of interfaces

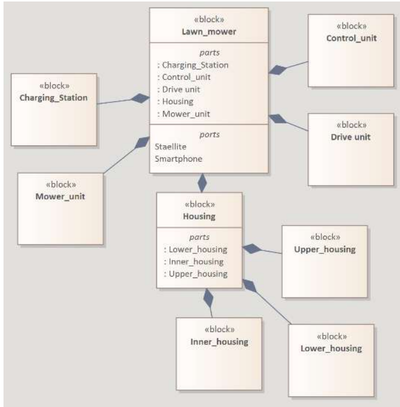

Figure 6: Block definition diagram (bdd) of the lawn mower

between the robotic lawnmower and the environment. By combining the models, the number and type of connection to the environment can be summarized. This information is then used in further work on the detailed modeling of the control unit. This information is relevant because in the next step of modeling the control unit, the number and types of interfaces can be better taken into account. This would not have been possible if the models had been considered in isolation.

Both models were created using Enterprise Architect and

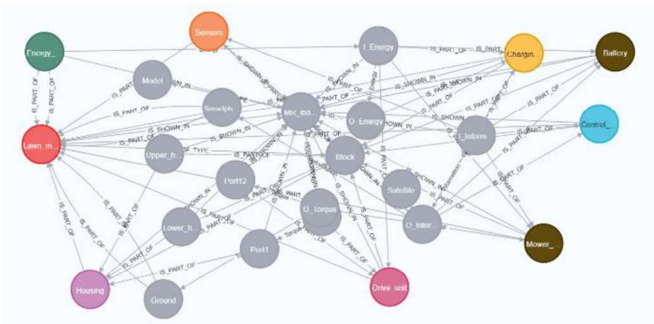

Figure 7: Integrated knowledge graph of the lawn mower

exported as XML Files. They were transformed and integrated using a python script. After the relevant data was extracted and integrated into one dataset, cypher queries were created. For creating the nodes, all values of the attribute 'name' and 'stereotype' were extracted. For creating the edges between the node, triples regarding to the ontology were created. The transformed and integrated knowledge graph is shown in figure 7. All components of the lawn mower have been colored in order to enhance the visibility of the knowledge graph.

For the reuse case the following cypher query was used in order to extract the relevant knowledge shown in figure 8.

The result of the query shows that a total of four interfaces of the robotic lawnmower ('Ground', 'Energy_supply', 'Satellite' and 'Smartphone') are included in the integrated knowledge graph.

```cypher
MATCH (port:uml:Port)-[:IS_PART_OF]→(lawn_Mower {name: "law_mower"})
RETURN port.name AS portName, labels(port) AS portType
```

Figure 8: Cypher query for extracting all interfaces of the lawn mower

5. Discussion and Limitations

In the course of this work, a procedure was developed which can be used for the integration of SysML models into a knowledge graph. The concept was implemented using two models of a robotic mower as examples. In situations where the models have different data quality and cannot be integrated directly by combination, methods from existing literature were presented. These methods are used on the one hand for a comparison of adjacent nodes, the neighboring, and on the other hand for a semantic comparison of inconsistent data. In the course of the case study, a first use case for the reuse of knowledge from the knowledge graph was also demonstrated. The graph was queried with regard to interfaces to the environment in order to be able to model the control unit in more detail on this basis. The difficult visualization of the example, as shown in Figure 7, is already a challenge. It is difficult to derive knowledge from the complete visualization. This can become a further problem with more complex models.

6. Summary and Further Research

In this work, the transformation and integration of bdd and ibd diagrams into a knowledge graph was described conceptually and demonstrated in the case study. Knowledge gained through the integration was obtained for the further detailing of the electric lawnmower model by querying the knowledge graph. Further work can focus on adding diagram types for example use case or requirement diagrams. This would allow the architectural development to be fully represented in a knowledge graph. Furthermore, the methods presented can be used to integrate models of lower quality. in this way, the applicability of the concept can be increased.

References

[1] A.L. Ramos, J.V. Ferreira, J. Barcelo, Model-Based Systems Engineering: An Emerging Approach for Modern Systems, IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 42 (2012) 101–111.

[2] INCOSE, INCOSE systems engineering handbook, John Wiley & Sons, 2023.

[3] M.C. Berschik, T. Schumacher, F.N. Laukotka, D. Krause, D. Inkermann, MBSE WITHIN THE ENGINEERING DESIGN COMMUNITY – AN EXPLORATORY STUDY, Proc. Des. Soc. 3 (2023) 2595–2604.

[4] P.J. Younse, J.E. Cameron, T.H. Bradley, Comparative analysis of a model-based systems engineering approach to a traditional systems engineering approach for architecting a robotic space system through knowledge categorization, Syst. Eng. 24 (2021) 177–199.

[5] A.M. Madni, M. Sievers, Model-based systems engineering: Motivation, current status, and research opportunities, Syst. Eng. 21 (2018) 172–190.

[6] L. Delligatti, SysML distilled: a brief guide to the systems modeling language, Addison-Wesley, Upper Saddle River, NJ, 2014.

[7] A. Trujillo, O.L. De Weck, A.M. Madni, An MBSE Approach Supporting Technical Inheritance and Design Reuse Decisions, in: ASCEND 2020, American Institute of Aeronautics and Astronautics, Virtual Event, 2020.

[8] L. Boggero, P.D. Ciampa, B. Nagel, An MBSE Architectural Framework for the Agile Definition of System Stakeholders, Needs and Requirements, in: AIAA Aviat. 2021 FORUM, American Institute of Aeronautics and Astronautics, VIRTUAL EVENT, 2021.

[9] R. Baduel, M. Chami, J.-M. Bruel, I. Ober, SysML Models Verification and Validation in an Industrial Context: Challenges and Experimentation, in: A. Pierantonio, S. Trujillo (Eds.), Model. Found. Appl., Springer International Publishing, Cham, 2018: pp. 132–146.

[10] International Council on Systems Engineering (INCOSE), Systems Engineering Vision 2035, (2021).

[11] T. Bayer, Is MBSE helping? Measuring value on Europa Clipper, in: 2018 IEEE Aerosp. Conf., IEEE, Big Sky, MT, 2018: pp. 1–13.

[12] A. Trujillo, O. de Weck, Towards a Comprehensive Reuse Strategy for Space Campaigns, in: 70th Int. Astronaut. Congr. IAC, Washington D.C, 2019.

[13] M. Alavi, D.E. Leidner, Review: Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues, MIS Q. 25 (2001) 107.

[14] L.F. Lai, A knowledge engineering approach to knowledge management, Inf. Sci. 177 (2007) 4072–4094.

[15] Q. Wu, D. Gouyon, P. Hubert, É. Levrat, TOWARDS MODEL-BASED SYSTEMS ENGINEERING (MBSE) PATTERNS TO EFFICIENTLY REUSE KNOW-HOW, INSIGHT 20 (2017) 31–33.

[16] C. Lange, J.T. Grundmann, M. Kretzenbacher, P.M. Fischer, Systematic reuse and platforming: Application examples for enhancing reuse with model-based systems engineering methods in space systems development, Concurr. Eng. 26 (2018) 77–92.

[17] M. Hussain, N. Masoudi, G. Mocko, C. Paredis, Approaches for Simulation Model Reuse in Systems Design — A Review, SAE Int. J. Adv. Curr. Pract. Mobil. 4 (2022) 1457–1471.

[18] Q. Wang, Z. Mao, B. Wang, L. Guo, Knowledge Graph Embedding: A Survey of Approaches and Applications, IEEE Trans. Knowl. Data Eng. 29 (2017) 2724–2743.

[19] S. Ji, S. Pan, E. Cambria, P. Marttinen, P.S. Yu, A Survey on Knowledge Graphs: Representation, Acquisition, and Applications, IEEE Trans. Neural Netw. Learn. Syst. 33 (2022) 494–514.

[20] L. Ehrlinger, W. Wöß, Towards a definition of knowledge graphs., Semant. Posters Demos SuCCESS 48 (2016) 2.

[21] F. Faheem, Z. Li, S. Husung, Analysis of potential errors in technical products by combining knowledge graphs with MBSE approach, Eng. Chang. World Proc. 60th ISC Ilmenau Scientific Colloquium (2023) 2023.

[22] L. Xie, Z. Hu, X. Cai, W. Zhang, J. Chen, Explainable recommendation based on knowledge graph and multi-objective optimization, Complex Intell. Syst. 7 (2021) 1241–1252.

[23] R. Angles, C. Gutierrez, An Introduction to Graph Data Management, in: G. Fletcher, J. Hidders, J.L. Larriba-Pey (Eds.), Graph Data Manag., Springer International Publishing, Cham, 2018: pp. 1–32.

[24] M. Besta, R. Gerstenberger, E. Peter, M. Fischer, M. Podstawski, C. Barthels, G. Alonso, T. Hoefler, Demystifying Graph Databases: Analysis and Taxonomy of Data Organization, System Designs, and Graph Queries, ACM Comput. Surv. 56 (2024) 1–40.

[25] J. Lu, J. Ma, X. Zheng, G. Wang, H. Li, D. Kiritsis, Design Ontology Supporting Model-Based Systems Engineering Formalisms, IEEE Syst. J. 16 (2022) 5465–5476.

[26] M. Smajevic, D. Bork, Towards Graph-Based Analysis of Enterprise Architecture Models, in: A. Ghose, J. Horkoff, V.E. Silva Souza, J. Parsons, J. Evermann (Eds.), Concept. Model., Springer International Publishing, Cham, 2021: pp. 199–209.

[27] P.-L. Glaser, S.J. Ali, E. Sallinger, D. Bork, Model-Based Construction of Enterprise Architecture Knowledge Graphs, in: J.P.A. Almeida, D. Karastoyanova, G. Guizzardi, M. Montali, F.M. Maggi, C.M. Fonseca (Eds.), Enterp. Des. Oper. Comput., Springer International Publishing, Cham, 2022: pp. 57–73.

[28] L. Petnga, Graph-based Assessment and Analysis of System Architecture Models, INCOSE Int. Symp. 29 (2019) 922–936.

[29] F. Schummer, M. Hyba, An Approach for System Analysis with MBSE and Graph Data Engineering, (2022).

[30] C. Mandel, S. Schwarz, T. Brix, A. Albers, S. Husung, Improving system of objectives maturity through systematic reuse of knowledge using ontology-based knowledge representations, Eng. Chang. World Proc. 60th ISC Ilmenau Scientific Colloquium (2023) 2023.

[31] C. Fu, J. Liu, S. Wang, Building SysML Model Graph to Support the System Model Reuse, IEEE Access 9 (2021) 132374–132389.

[32] P.M. Nadkarni, L. Ohno-Machado, W.W. Chapman, Natural language processing: an introduction, J. Am. Med. Inform. Assoc. 18 (2011) 544–551.

[33] G. Tamašauskaitė, P. Groth, Defining a Knowledge Graph Development Process Through a Systematic Review, ACM Trans. Softw. Eng. Methodol. 32 (2023) 1–40.

[34] S. Ferré, Application of concepts of neighbours to knowledge graph completion1, Data Sci. 4 (2021) 1–28.

[35] L. Zhong, J. Wu, Q. Li, H. Peng, X. Wu, A Comprehensive Survey on Automatic Knowledge Graph Construction, ACM Comput. Surv. 56 (2024) 1–62.

[36] M. Equi, V. Mäkinen, A.I. Tomescu, R. Grossi, On the Complexity of String Matching for Graphs, ACM Trans. Algorithms 19 (2023) 1–25.