*Article Identifier*
*DOI: 10.22032/dbt.58898*
*URN: urn:nbn:de:gbv:ilm1-2023isc-032:5*

*60th ILMENAU SCIENTIFIC COLLOQUIUM*
*Technische Universität Ilmenau, 4 – 8 September 2023*
*URN: urn:nbn:de:gbv:ilm1-2023isc:1*

# ANALYSIS OF POTENTIAL ERRORS IN TECHNICAL PRODUCTS BY COMBINING KNOWLEDGE GRAPHS WITH MBSE APPROACH

*Faizan Faheem, Zirui Li, Stephan Husung*

Technische Universität Ilmenau

## ABSTRACT

Technical products are developed to meet the demands of stakeholders. Therefore, the product's functions and associated properties are important. Various influencing factors e.g., external disturbances can have an impact on the input flows of the products or its characteristics and thus on the functions. If this leads to deviations between the required and as-is functions, these deviations are called errors.
It is therefore important to analyze errors in product development and implement measures to increase the robustness of the product. Model-Based Systems Engineering (MBSE) supports the development of complex systems. However, MBSE alone has limited ability to identify in-depth errors. This requires knowledge of possible errors from previous products in specific contexts. For this purpose, the method proposed in this paper facilitates identifying errors in the concept phase by combining MBSE approaches with reusable knowledge (i.e., knowledge graph). The approach is presented using an application example for a mobile robot.

***Index Terms -*** Error Analysis, Robustness, Knowledge Graph, Product Development, Mobile Robotics

## 1. INTRODUCTION AND MOTIVATION

Technical products are developed to fulfill specific demands. For this purpose, the product must provide relevant functions in combination with other relevant properties (e.g., reliability, availability, etc.). In this context, the function is a transformation of intended (function-relevant) input flows ($E_f$) into intended (function-relevant) output flows ($A_f$) (cf. Figure 1). The implementation of products leads to restrictions in the realization of the functions, since the product characteristics (according to the CPM model [1]) have deviations or ageing, among other things. In addition, external disturbance input variables $E_n$ (e.g., vibrations) have an impact on the product [2].

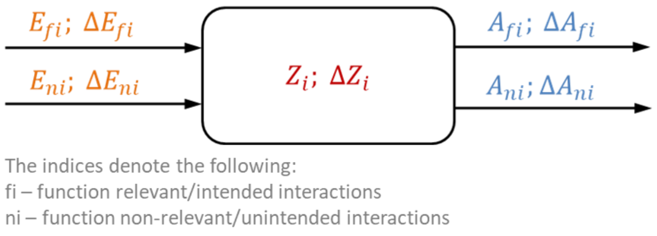

$E_{fi};\ \Delta E_{fi}$
$E_{ni};\ \Delta E_{ni}$
$Z_i;\ \Delta Z_i$
$A_{fi};\ \Delta A_{fi}$
$A_{ni};\ \Delta A_{ni}$
The indices denote the following:
fi – function relevant/intended interactions
ni – function non-relevant/unintended interactions

**Figure 1:** Function depending on the intended and unintended input flows and internal state parameters [2, 3].

## 2. OBJECTIVE

Developing products that are as robust as necessary (see Hansen's harmlessness condition [4]) is a key objective. In this context, robust means that the product must comply with the required properties of the relevant output flows (intended and unintended) within permissible tolerances even in the presence of fluctuations in the input variables and the internal parameters. The deviations between the as-is and required properties are referred to as errors [2, 3].
For the analysis of possible errors in the context of product development, the interactions between external input variables and internal state parameter based on cause-effect relationships in the product must be investigated. As the basis for the analysis, a purposeful system description on basis of system-theoretical concepts is useful.
A widely used approach for modelling complex mechatronic products is described in the so-called Model-Based Systems Engineering (MBSE) [5, 6]. In these models, mainly the content aspects of the product to be developed are represented via model elements (use cases, requirements, function, logical elements, etc.), their parameters and relationships, which can be used as a basis for the analysis [7]. These models must already have a high degree of maturity before they can be used to analyze potential errors. This leads to efforts in product development. In addition, many insights into the behavior of the product to be developed in the planned environments are often not yet available [8].
The approach presented in this paper intends to reduce the modelling effort and increase the model quality by enhancing the system models with existing knowledge from other development processes and products using knowledge graphs.
**The following research question arises:** How can existing knowledge about existing products and their relevant properties be used to enhance system models of the system of interest (SOI) using knowledge graphs as a basis for the analysis of possible errors?

## 3. FUNDAMENTALS

This section briefly presents background knowledge and underlying terms utilized in conjunction with the proposed approach in system modelling, error analysis and knowledge representations using knowledge graphs.

### 3.1 System Modelling using Characteristics-Properties Modelling

A basis for the description of products is the so-called Characteristics-Properties Modelling (CPM) approach by Weber [1]. In this approach, products are described via properties (e.g., behavior) and characteristics (e.g., sizes, arrangements). The classification according to properties and characteristics is used in this paper for system modelling, since the characteristics and properties are an important basis for the direct determination of errors and the basis for the query in knowledge graphs.

### 3.2 Error Analysis based on System Models

MBSE models, especially with the application of SysML, offer the possibility to describe system elements, their parameters and relations. Cause-effect analyses (e.g., FMEA or FTA) are carried out in many scientific investigations on the basis of these system descriptions [9–11]. The knowledge represented in the system model can be reused in a goal-oriented manner. As the models are usually limited to the specific product context, only the context-specific knowledge can be used in these analyses.

## 3.3 Knowledge Representation

### *3.3.1 Knowledge Graph*

For the concept in this paper, knowledge about possible errors is to be represented in knowledge graph. A knowledge graph is a semantic network consisting of nodes representing entities, and edges illustrate interconnected relationships among these entities in the graph form [12]. It incorporates large-scale data representing the knowledge that can be queried. In other words, knowledge graphs provide an efficient way to manage and utilize a huge amount of knowledge [13]. Moreover, it organizes knowledge in a structured graph-based form that is machine-understandable and assists in flexible querying of particular information.

### *3.3.2 Resource Description Framework (RDF) Knowledge Graph*

Different knowledge representation models can be used to store the knowledge graph. These models include Property Graph Model, Resource Description Framework (RDF), conceptual graphs and so on [14]. Each knowledge representation model has distinctive abilities and functionalities; therefore, the selection of a suitable model depends on several factors, such as the application domain and specific requirements etc.

The RDF model [15] is selected in the proposed method because it follows a triple structure (i.e., subject, predicate and object statements) and represents knowledge in a flexible and expressive manner. The collection of these triples together forms an RDF knowledge graph. These triples should be organized in the form of an RDF dataset. RDF datasets can be perceived as directed labeled graphs in which each triple specifies an edge (i.e., labelled predicate) between the subject and object [16]. Many different formats can be used to represent the RDF dataset based on specific requirements or personal preferences. In the presented method, Terse RDF Triple Language (Turtle) format [17] is used as it provides concise and human-readable syntax for specifying RDF data. After specifying the RDF dataset, it is important to load it into the triple store, which is a special database designed to serve flexible querying of knowledge or data from the RDF dataset (i.e., RDF knowledge graph).

### *3.3.3 SPARQL Query Language and SPARQL Endpoint*

Various query languages have been proposed for querying the knowledge graphs [18] such as Cypher and Gremlin [19, 20], and many more for property Graphs. SPARQL is a standard query language for RDF knowledge graphs [21]. SPARQL is used to retrieve information from the RDF dataset based on graph pattern matching [22]. To achieve this, the variables of triple patterns (i.e., the basic construct of the SPARQL query) are matched with the RDF dataset [16]. Moreover, SPARQL provides different operators (such as SELECT, WHERE, FILTER, etc.) to create expressive queries.

Many SPARQL query services (i.e., SPARQL endpoints) are available that serves thousands of SPARQL queries each time [23, 24]. The SPARQL Endpoint is used to access the RDF dataset loaded in the triple store through the SPARQL query language. The SPARQL endpoint acts as an access gateway between the RDF data stored in a triple store and the application/user. The proposed approach in this contribution enables stakeholders to automatically extract potential errors from their system model by querying the knowledge graph using the SPARQL endpoint in the background.

## 4. STATE OF THE ART

As a basis for the development of the presented approach, scientific contributions were examined that address the analysis of technical products with regard to errors, including the use of models. Selection and evaluation of literature are based on four requirements specified by considering the research question identified in the objective section. These requirements are necessary in order to support stakeholders in identifying potential errors in technical products during the concept phase of development:

- R1: The approach should be applicable at the mechatronic system level, which means that it should support the MBSE approaches (e.g., using SysML modelling language). This is important as MBSE supports stakeholders from different disciplines by providing a holistic view of the overall system through models that lead to efficiently managing the complexity of the overall system [25].
- R2: The approach must consider potential errors identification in the concept phase of development to enhance the overall robustness of system. The analysis of potential errors during the concept phase is crucial for decision-making and finding suitable solution concepts to avoid negative consequences of these decisions during the later product development phases. Various external factors (such as vibrations, humidity, temperature, etc.) can impact the system's functionality resulting in unexpected potential errors in the overall system.
- R3: The approach should facilitate the reuse of existing knowledge regarding potential errors. It is necessary to have an established knowledge base enriched with potential errors of prior products that facilitate engineers in evaluating and identifying potential errors during the concept phase of product development.
- R4: Support in the modelling tool to retrieve the potential error information based on the system model. The approaches should provide a feature in the modelling tool that can automatically retrieve possible potential errors by capturing the system model information.

Among all the examined approaches [2, 9, 10, 25–33], only a few are briefly discussed as an example. The approach presented by Hecht et al. [31] automates the generation of Failure Modes and Effects Analyses (FMEAs) using SysML. Biggs et al. [10] developed a profile to represent safety and reliability relevant information system model. Ebner et al. [26] developed a model-based approach to determine the design space of the mechatronic products considering functional safety and its analysis. These approaches follow requirement R1. The approaches proposed by Brix et al. [2] and Ebner et al. [26] deal with the analysis of functionally related potential errors in the systems during the concept phase of product development and therefore follow requirement R2.
The analysis of the current state of the art shows that several very good research results are available, but that the demands cannot be fully met. This motivates the following explanations of the approach presented.

## 5. APPROACH OVERVIEW

As a basis for the detailed explanations, the overall concept is presented first (cf. Figure 2). Based on this, the individual steps will be discussed in the following sections. The developed approach should support engineers in analyzing potential errors based on MBSE models in conjunction with a knowledge graph during the concept phase of product development. The proposed approach consists of following steps:

1. Definition of the system model of the system of interest (SOI) based on the initial state of knowledge in a project using modelling with SysML.
2. Analysis of the system model to obtain the necessary input information for the query in the knowledge graph.
3. Search for existing knowledge on disturbances or known errors in the knowledge graph and transfer this knowledge to the system model.

Step 1 consist of two steps: creating (a) black- and (b) white-box system models (cf. Figure 2: Right). In the black-box model, the problem space of the SOI in the planned environment is defined. The problem space includes the use cases that the SOI must provide, taking into account existing constraints (i.e., environmental objects). The white-box model is created based on the black-box, in which the system is decomposed into relevant sub-systems considering the interfaces to the identified environmental objects in the black-box model, forming a high-level system architecture model [34]. The black- and white-box models are the basis for the query of possible errors in the knowledge graph, since the combination of black- and white-box model describes the expectations of the system behavior and their realization in the system (e.g. a required detection of disturbance objects (black box) is realized in the system by means of specific measurement functions and sensors).
Steps 2 and 3 are only briefly discussed here, as they will be discussed in more detail in section 6. In step 2, the black- and white-box model is used as a basis for determining the necessary inputs for the search queries. For this, the already defined model elements and their relations are traversed. In step 3, the potential errors can be searched based on the captured model information by querying (e.g., query(q) as shown in Figure 2) into the knowledge graph.
For this contribution, potential errors from already existing products were added into the knowledge graph in order to create the basis for the search (cf. Figure 2: left). This aspect is not described in detail in the paper.
The proposed method is evaluated using an example of a mobile robot system. For this purpose, a knowledge graph framework is coupled with the SysML modelling tool via a plugin, which is further in the development phase, to automate the proposed method. The plugin directly supports stakeholders in the SysML modelling tool (i.e., Cameo Systems Modeler) to analyze potential errors based on the defined system models.

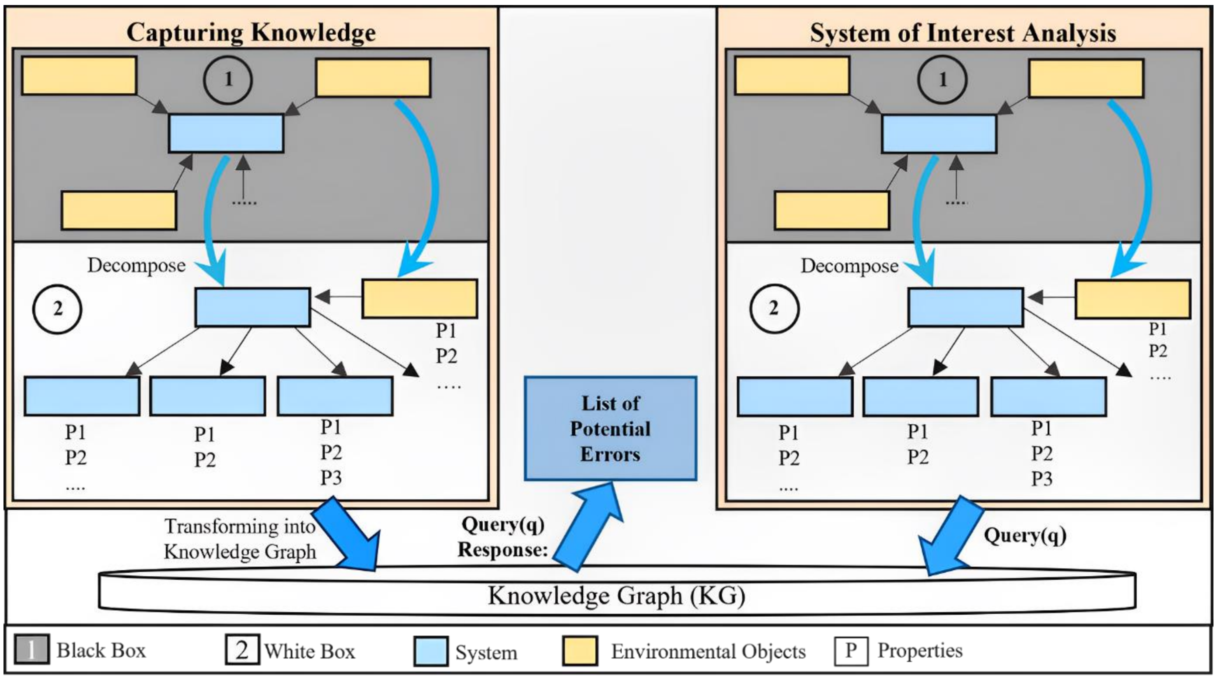


**Figure 2:** Overall method approach. The right-hand side shows the system analysis with query(q) in the existing KG, identifying potential errors. The left side shows; capturing the knowledge of existing products in the KG.

## 6. DETAILED STEPS OF THE APPROACH

This section describes the proposed approach in details that supports engineers in analyzing potential errors in technical systems using a combination of MBSE and knowledge graph. The goal is to enhance the overall robustness of the system during the concept phase of product development by uncovering potential errors that occur due to external influencing factors, properties of the system elements or interactions between the system elements.
Based on the steps mentioned in the approach overview (cf. Section 5), the second and the third steps are the main focus of this paper.

### 6.1 Capturing Information from System of Interest (SOI) Analysis

The approach starts with an analysis of the black- and white-box model of the system of interest (SOI). The captured information later serves as a base to identify potential errors by reusing the insights in the knowledge graph (cf. Subsection 6.2).
**Black Box:** The analysis begins with the black-box model, which specifies the use cases or intended behavior of the system in the planned environment. The black-box model also includes the expected interactions of the system with the environment, taking into account boundary conditions (i.e., external influencing factors or environmental objects). The environment elements, their properties and the influencing factors can have a significant impact on the product behavior.
To illustrate the method, a mobile robot system is chosen as an application example (see Figures 3a & 3b). The use case associated with the robotic system is to reach a specific destination. When implementing the use case, the environmental elements (e.g., moving obstacles), the properties of the environmental elements (e.g., their velocity) and the interactions associated with the robot system must be considered. In order for the robot to move, friction between the robot and the ground is needed. The friction parameters can have variations, which can also affect the behavior of the robot.

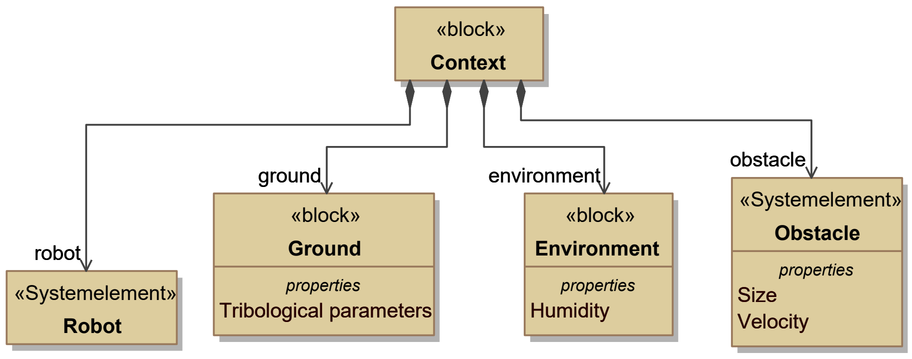


**Figure 3a:** Expected interactions of the mobile robot system in the context of the planned environment in the black-box model.

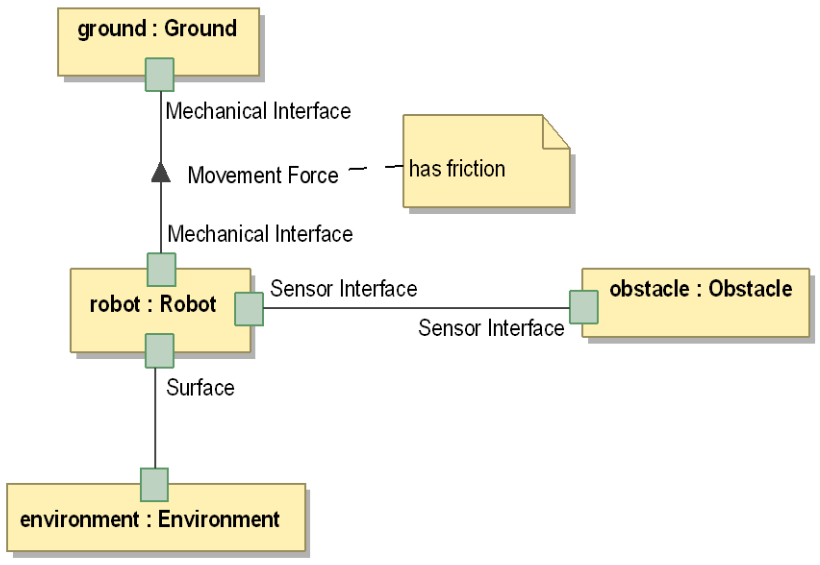


**Figure 3b:** Impact of external influencing factors (due to environmental objects with its associated properties) on the mobile robot system in the black-box model.

**White Box:** The description of the implementation of the robot is done in the white-box model. In this model, the decomposed partial functions (if applicable in relevant states) as well as the decomposed system elements and their allocations are described.
The influence of the environment objects, the properties of the system elements as well as the interactions of the system elements can affect the system behavior. In the SysML model, the relations between the model elements for the functions and the system elements up to the model elements for the environment elements are explicitly described (see Figure 4). For example, for the robot it is explicitly described in the model that the function "Detect Obstacle" is related to the environment element “Obstacle”. The environment element “Obstacle” has properties (such as “Size” and “Velocity”) that can influence the function "Detect Obstacle". It is also described that the function "Detect Obstacle" is realized by an initially abstractly described “Sensor” with a property “Resolution” and by a “CPU”.
The parsing between the function (described as SysML activity), environment elements and system elements enables the determination of the required input information from the SysML model for the search in the knowledge graph. In Figure 4, the realized activity "Detect Obstacle" is parsed with the environment object "Obstacle" and the associated property "Velocity" and serves as the basis for capturing input information to query the knowledge graph for potential errors.

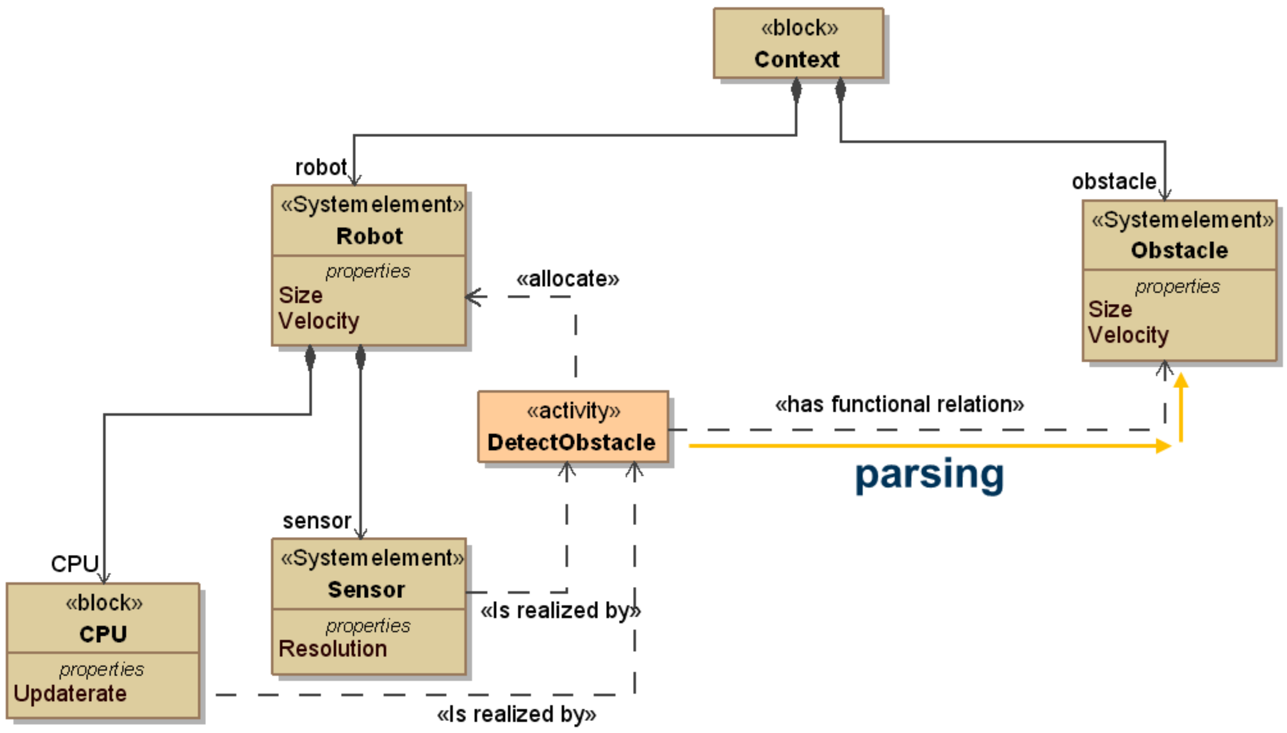


**Figure 4:** High-level system architecture in the white box model.

## 6.2 Querying the Knowledge Graph based on the Captured Information from System of Interest Analysis

In this sub-section, a knowledge graph is queried to retrieve potential errors based on the information captured from the high-level SysML model (cf. Subsection 6.1). To achieve this, an RDF knowledge graph is created by taking into account the knowledge about potential errors. To build an RDF knowledge graph, RDF datasets are filled with triples as an example (cf. Section 3). The defined RDF datasets are included with potential errors based on empirical knowledge from previous projects. These RDF datasets are stored in the knowledge graph database, which supports the visualization of the RDF knowledge graph, as shown in Figure 5. To retrieve potential errors based on the captured input information of the SysML model, the RDF knowledge graph, that is realized by stored RDF datasets, is queried. The query is performed using the SPARQL query language by accessing the SPARQL endpoint generated by the graph database (cf. Section 3). In Figure 4, the captured input information, such as the "Detect Obstacle" function is used to detect "Obstacle" with "Velocity" supports in analyzing the potential errors (e.g., Velocity Too High) by querying the RDF knowledge graph that is realized by RDF datasets via the SPARQL endpoint.

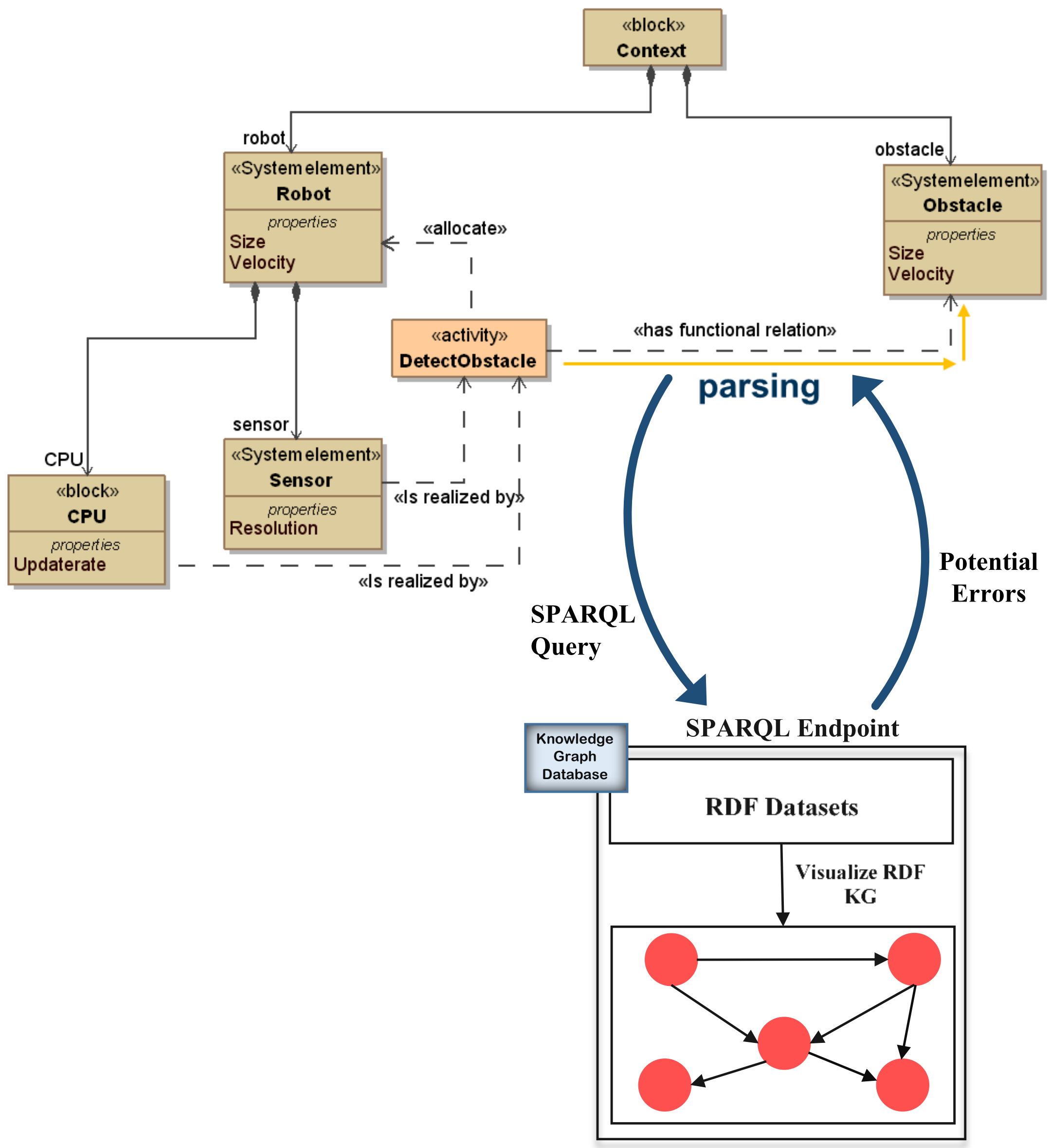


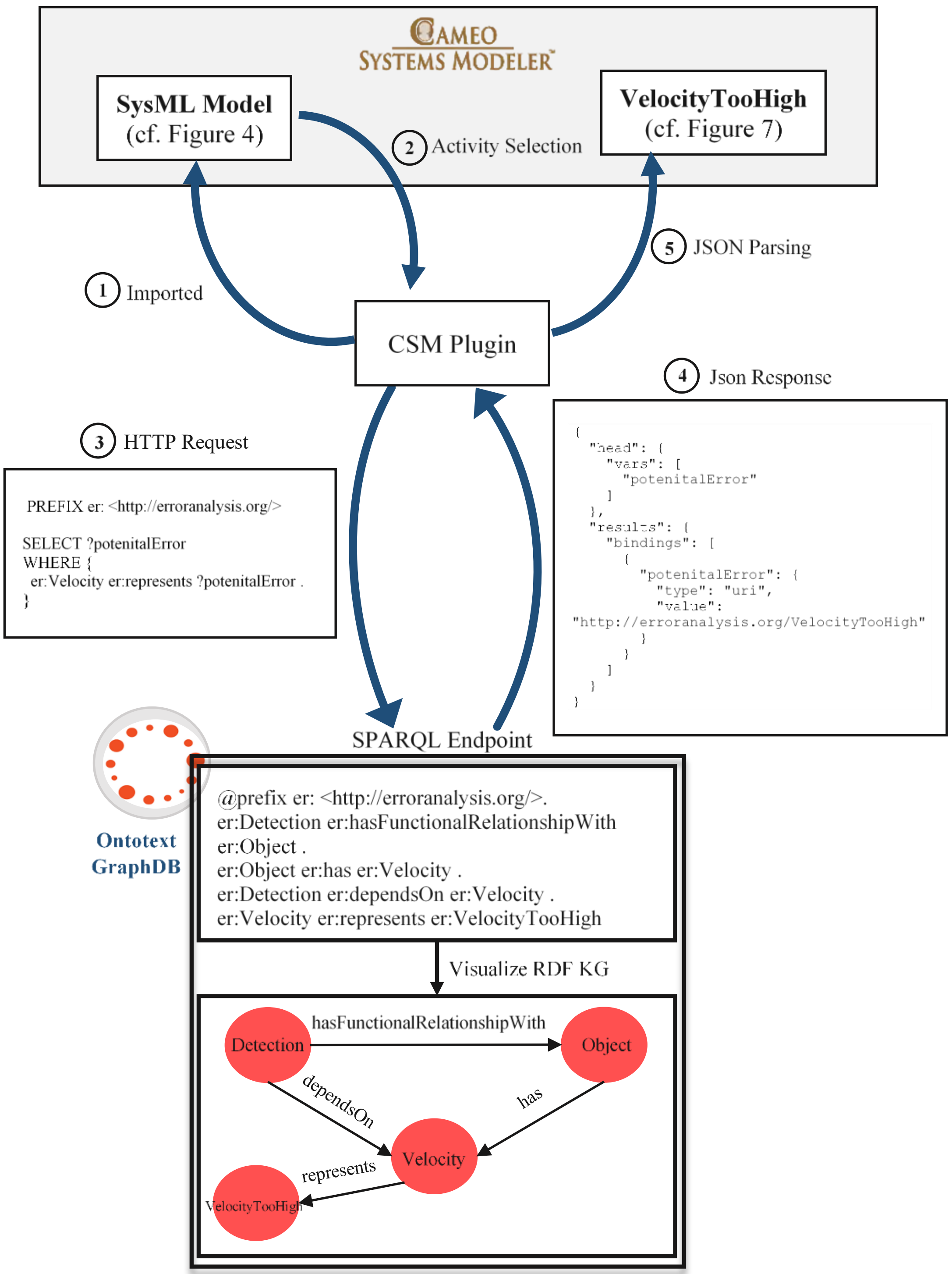
**Figure 5:** Querying the RDF Knowledge graph realized by RDF datasets based on the captured input information from subsection 6.1.

## 7. IMPLEMENTATION DETAILS

For the coupling of the SysML tool Cameo Systems Modeler (CSM) with the knowledge graph, the programming interface of the CSM is recommended. A plugin was created in the CSM for this purpose. CSM provides a plugin development framework that supports customizing or extending the functionality of the modelling tool through its application programming interface (API). The API offers interfaces, classes, methods, etc. to interact with the main features of the CSM using the JAVA programming language. In this paper, an initial prototype of the CSM plugin is presented, and a fully functional plugin is currently in the development phase.
The initial prototype involves establishing a connection between the CSM modelling tool and the knowledge graph database. The developed connection retrieves potential errors within the CSM by querying the RDF knowledge graph obtained from RDF datasets. Before explaining the implementation details of initial plugin prototype, the construction of the RDF knowledge graph is discussed. The Ontotext GraphDB is selected as knowledge graph database because it

efficiently stores, retrieves, and manages the RDF datasets. An RDF is a data model that represents information in triples. The relationship between these triples forms a graph structure specified as an RDF knowledge graph (cf. Section 3 provides details). The implementation details comply with the proposed approach (cf. Section 6); therefore, the RDF datasets are defined based on the input information captured from the existing SysML model. Moreover, the defined datasets are loaded inside the Ontotext GraphDB repository that provides the feature to visualize the RDF knowledge graph (cf. Figure 6: Ontotext GraphDB part). In addition, Ontotext GraphDB provides a SPARQL endpoint (i.e., via the Ontotext GraphDB API) that acts as a means of interacting with the RDF knowledge graph realized by RDF datasets.

The implementation details of the initially developed prototype begin by exporting the required configuration files and loading them into the CSM plugin directory (cf. Figure 6). The predefined SPARQL query written in the initial prototype plugin is performed based on the selection of activity of the SysML model through the customized icon. The formulated SPARQL query is sent to Ontotext GraphDB (i.e., SPARQL endpoint) via an HTTP request. The Ontotext GraphDB processes the SPARQL query and retrieves the corresponding information regarding potential errors from the RDF knowledge graph by realizing the stored RDF datasets. The SPARQL query response is sent back through the HTTP request protocol in JSON serialization format. Moreover, the potential error information is represented using the CSM tool interface after parsing the JSON response (cf. Figures 6 & 7).



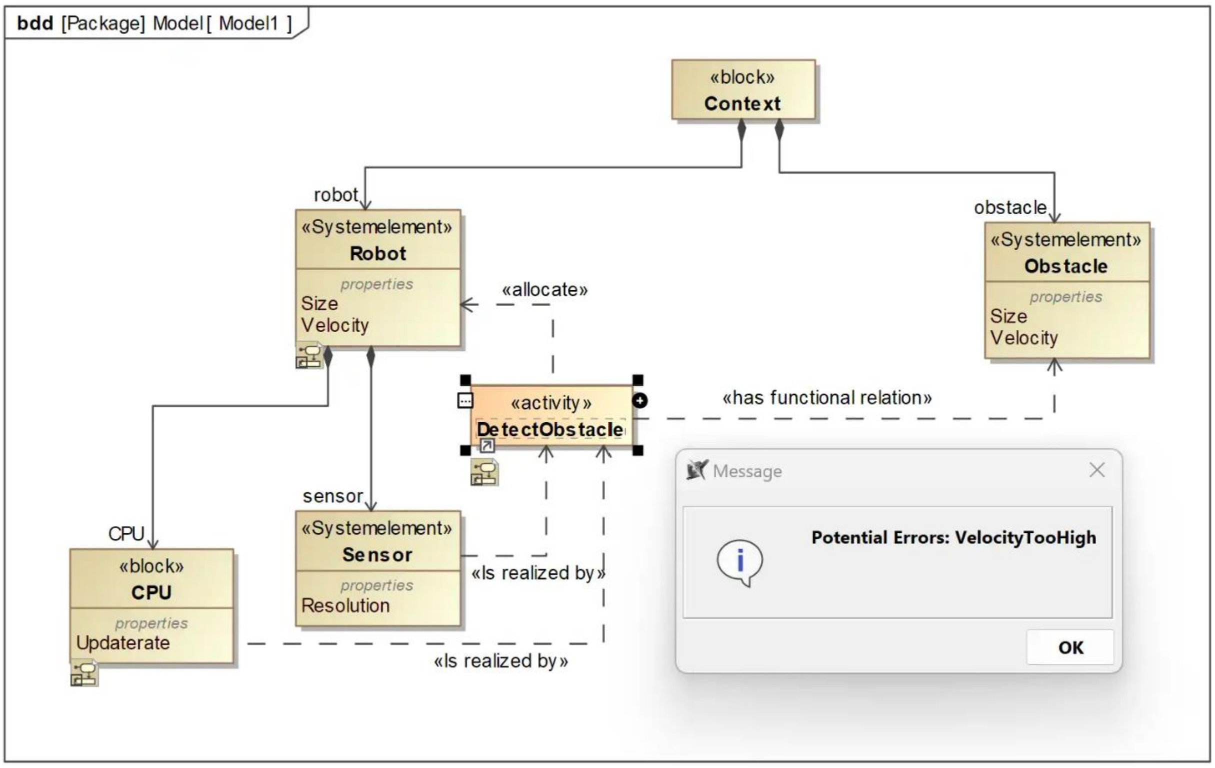
```
PREFIX er: <http://erroranalysis.org/>

SELECT ?potenitalError
WHERE {
 er:Velocity er:represents ?potenitalError .
}
```

```
{
  "head": {
    "vars": [
      "potenitalError"
    ]
  },
  "results": {
    "bindings": [
      {
        "potenitalError": {
          "type": "uri",
          "value":
"http://erroranalysis.org/VelocityTooHigh"
        }
      }
    ]
  }
}
```

```
@prefix er: <http://erroranalysis.org/>.
er:Detection er:hasFunctionalRelationshipWith
er:Object .
er:Object er:has er:Velocity .
er:Detection er:dependsOn er:Velocity .
er:Velocity er:represents er:VelocityTooHigh
```

**Figure 6:** Implementation overview of the initial prototype plugin that supports retrieval of potential errors by coupling the SysML model with the RDF knowledge graph.



Potential Errors: VelocityTooHigh

**Figure 7:** Analyzing potential errors on CSM modelling tool based on parsing JSON response (cf. Figure 6: Step 5).

## 8. SUMMARY AND OUTLOOK

The approach described in this paper demonstrates that potential errors in evolving technical products can be analyzed using the existing reusable knowledge base of established systems. To achieve this, a SOI SysML model is built to capture the necessary input information. The captured information serves as a basis for querying the insights of the knowledge graph to retrieve potential errors.

The contribution addresses the research question mentioned in the objective section and identifies the potential errors by coupling the MBSE (i.e., SysML model) with the existing reusable knowledge in the knowledge graph.

As part of future work, the existing knowledge graph will be further enhanced with the potential errors of many more established products. In addition, a fully functional magic draw plugin will be provided to support engineers in identifying potential errors based on their SysML model. Further research questions deal with considerations of fuzziness in search, necessary modelling guidelines, ontologies for model building, etc.

## REFERENCES

[1] C. Weber and S. Husung, “Solution patterns - their role in innovation, practice and education,” in *14th International Design Conference (DESIGN 2016)*, Cavtat, Dubrovnik, Croatia, vol. Design Theory and Research Methods, 2016, pp. 99–108.

[2] T. Brix and S. Husung, “Research and Teaching on Robust Design in early Design Phases,” RD SIG Seminar Series, 2022.

[3] S. Husung *et al.,* “Systemic Conception of the Data Acquisition of Digital Twin Solutions for Use Case-Oriented Development and Its Application to a Gearbox,” *Systems*, vol. 11, no. 5, 2023, doi: 10.3390/systems11050227.

[4] F. Hansen, *Adjustment of precision mechanisms*. London: Iliffe Books, 1970.

[5] INCOSE, *Systems Engineering Vision 2020: Technical Report INCOSE-TP-2004-004-02*.
[6] A. Morkevicius, A. Aleksandraviciene, D. Mazeika, L. Bisikirskiene, and Z. Strolia, "MBSE Grid: A Simplified SysML-Based Approach for Modeling Complex Systems," in *INCOSE International Symposium*, 2017, doi: 10.1002/j.2334-5837.2017.00350.x.
[7] S. Husung, C. Weber, and A. Mahboob, "Model-Based Systems Engineering: A New Way for Function-Driven Product Development," in *Design Methodology for Future Products*, D. Krause and E. Heyden, Eds., Cham: Springer International Publishing, 2022, pp. 221–241.
[8] C. Mandel, J. Böning, M. Behrendt, and A. Albers, "A Model-Based Systems Engineering Approach to Support Continuous Validation in PGE - Product Generation Engineering," in *IEEE ISSE International Symposium on Systems Engineering 2021*, 2021. Accessed: Sep. 14, 2021.
[9] A. Korsunovs *et al.,* "Towards a Model-Based Systems Engineering Approach for Robotic Manufacturing Process Modelling with Automatic FMEA Generation," *Proc. Des. Soc.*, vol. 2, pp. 1905–1914, 2022, doi: 10.1017/pds.2022.193.
[10] G. Biggs, K. Post, A. Armonas, N. Yakymets, T. Juknevicius, and A. Berres, "OMG standard for integrating safety and reliability analysis into MBSE: Concepts and applications," in *INCOSE International Symposium*, vol. 29, 2019, pp. 159–173, doi: 10.1002/j.2334-5837.2019.00595.x.
[11] S. Husung, C. Weber, A. Mahboob, and S. Kleiner, "Using Model-Based Systems Engineering for need-based and consistent support of the design process," in *23rd International Conference on Engineering Design (ICED21)*, 2021, doi: 10.1017/pds.2021.598.
[12] J. Pujara, H. Miao, L. Getoor, and W. Cohen, "Knowledge Graph Identification," in *Advanced Information Systems Engineering* (Lecture Notes in Computer Science), D. Hutchison et al., Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 542–557.
[13] L. Xie, Z. Hu, X. Cai, W. Zhang, and J. Chen, "Explainable recommendation based on knowledge graph and multi-objective optimization," *Complex Intell. Syst.*, vol. 7, no. 3, pp. 1241–1252, 2021, doi: 10.1007/s40747-021-00315-y.
[14] S. Tiwari, F. N. Al-Aswadi, and D. Gaurav, "Recent trends in knowledge graphs: theory and practice," *Soft Comput*, vol. 25, no. 13, pp. 8337–8355, 2021, doi: 10.1007/s00500-021-05756-8.
[15] "RDF 1.1 Concepts and Abstract Syntax." https://www.w3.org/TR/rdf11-concepts/ (accessed Jun. 26, 2023).
[16] M. Schmidt, M. Meier, and G. Lausen, "Foundations of SPARQL query optimization," in *Proceedings of the 13th International Conference on Database Theory*, Lausanne Switzerland, L. Segoufin, Ed., 2010, pp. 4–33, doi: 10.1145/1804669.1804675.
[17] "RDF 1.1 Turtle." https://www.w3.org/TR/turtle/ (accessed Jun. 26, 2023).
[18] R. Angles, M. Arenas, P. Barceló, A. Hogan, J. Reutter, and D. Vrgoč, "Foundations of Modern Query Languages for Graph Databases," *ACM Comput. Surv.*, vol. 50, no. 5, pp. 1–40, 2018, doi: 10.1145/3104031.
[19] N. Francis *et al.,* "Cypher," in *Proceedings of the 2018 International Conference on Management of Data*, Houston TX USA, G. Das, C. Jermaine, and P. Bernstein, Eds., 2018, pp. 1433–1445, doi: 10.1145/3183713.3190657.
[20] M. A. Rodriguez, "The Gremlin graph traversal machine and language (invited talk)," in *Proceedings of the 15th Symposium on Database Programming Languages*, Pittsburgh PA USA, J. Cheney and T. Neumann, Eds., 2015, pp. 1–10, doi: 10.1145/2815072.2815073.

[21] “SPARQL 1.1 Query Language.” https://www.w3.org/TR/sparql11-query/ (accessed Jun. 26, 2023).

[22] R. Angles and C. Gutierrez, “Querying RDF Data from a Graph Database Perspective,” in *The Semantic Web: Research and Applications* (Lecture Notes in Computer Science), D. Hutchison et al., Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 346–360.

[23] S. Malyshev, M. Krötzsch, L. González, J. Gonsior, and A. Bielefeldt, “Getting the Most Out of Wikidata: Semantic Technology Usage in Wikipedia’s Knowledge Graph,” in *The Semantic Web – ISWC 2018* (Lecture Notes in Computer Science), D. Vrandečić et al., Eds., Cham: Springer International Publishing, 2018, pp. 376–394.

[24] M. Saleem, M. I. Ali, A. Hogan, Q. Mehmood, and A.-C. N. Ngomo, “LSQ: The Linked SPARQL Queries Dataset,” in *The Semantic Web - ISWC 2015* (Lecture Notes in Computer Science), M. Arenas et al., Eds., Cham: Springer International Publishing, 2015, pp. 261–269.

[25] S. Japs, H. Anacker, and R. Dumitrescu, “SAVE: Security & safety by model-based systems engineering on the example of automotive industry,” *Procedia CIRP*, vol. 100, pp. 187–192, 2021, doi: 10.1016/j.procir.2021.05.053.

[26] C. Ebner, K. Gorelik, and A. Zimmermann, “Model-Based Design Space Exploration for Fail-Operational Mechatronic Systems,” in *2021 IEEE International Symposium on Systems Engineering (ISSE)*, Vienna, Austria, 2021, pp. 1–8, doi: 10.1109/ISSE51541.2021.9582505.

[27] *Implementation of MBSE Approach for Developing Reliability Model to Ensure Robustness of Sounding Rocket Program Using MADe Modeling Tool*, 2020. [Online]. Available: http://ieworldconference.org/content/sise2020/papers/odita.pdf

[28] M. Hofbaur, J. Köb, G. Steinbauer, and F. Wotawa, “Improving Robustness of Mobile Robots Using Model-based Reasoning,” *J Intell Robot Syst*, vol. 48, no. 1, pp. 37–54, 2007, doi: 10.1007/s10846-006-9102-0.

[29] R. B. Stone, I. Y. Tumer, and M. E. Stock, “Linking product functionality to historic failures to improve failure analysis in design,” *Res Eng Des*, vol. 16, 1-2, pp. 96–108, 2005, doi: 10.1007/s00163-005-0005-z.

[30] R. Krishnan and S. V. Bhada, “A Systems Approach towards Developing a Diagnostic System for Complex Robots,” *INCOSE International Symposium*, vol. 28, no. 1, pp. 1682–1690, 2018, doi: 10.1002/j.2334-5837.2018.00576.x.

[31] M. Hecht, A. Chuidian, T. Tanaka, and R. Raymond, “Automated Generation of FMEAs using SysML for Reliability, Safety, and Cybersecurity,” in *2020 Annual Reliability and Maintainability Symposium (RAMS)*, Palm Springs, CA, USA, 2020, pp. 1–7, doi: 10.1109/RAMS48030.2020.9153708.

[32] M. Hecht, E. Dimpfl, and J. Pinchak, “Automated Generation of Failure Modes and Effects Analysis from SysML Models,” in *2014 IEEE International Symposium on Software Reliability Engineering Workshops*, Naples, Italy, 2014, pp. 62–65, doi: 10.1109/ISSREW.2014.117.

[33] S. Japs, F. Faheem, H. Anacker, S. Husung, and R. Dumitrescu, “MODEL-BASED SYSTEMS ENGINEERING USING SECURITY DESIGN PATTERNS IN THE CONTEXT OF ISO/SAE 21434,” in *24th International Conference on Engineering Design*, Bordeaux, France, 2023, pp. 2675–2684, doi: 10.1017/pds.2023.268.

[34] A. Mahboob and S. Husung, “A Modelling Method for Describing and Facilitating the Reuse of Sysml Models During Design Process,” in *INTERNATIONAL DESIGN CONFERENCE – DESIGN 2022*, online, vol. 2, 2022, pp. 1925–1934, doi: 10.1017/pds.2022.195.

## CONTACTS

| | |
|---|---|
| M.Sc Faizan Faheem | email: faizan.faheem@tu-ilmenau.de<br>ORCID: https://orcid.org/0009-0009-1014-5389 |
| M.Sc. Zirui Li | email: zirui.li@tu-ilmenau.de<br>ORCID: https://orcid.org/0009-0007-7983-7901 |
| Univ.-Prof. Dr.-Ing. Stephan Husung | email: stephan.husung@tu-ilmenau.de<br>ORCID: https://orcid.org/0000-0003-0131-5664 |