Integration of Machine Learning Task Definition in Model-Based Systems Engineering using SysML

Simon Rädler
Business Informatics Group
Technical University of Vienna
Vienna, Austria
simon.raedler@student.tuwien.ac.at

Eugen Rigger
Digital Engineering
V-Research GmbH
Dornbirn, Austria

Juergen Mangler
Information Systems and Business Process Management
Technical University of Munich
Munich, Germany

Stefanie Rinderle-Ma
Information Systems and Business Process Management
Technical University of Munich
Munich, Germany

Abstract—**In order to allow Systems Engineers to utilize data produced in cyber-physical systems (CPS), they have to cooperate with data-scientists for custom data-extraction, data-preparation, and/or data-transformation mechanisms. While interfaces in CPS systems might be generic, the data that is produced for custom application needs has to be transformed and merged in very specific ways, to allow systems engineers proper interpretation and insight-extraction. In order to enable efficient cooperation between systems engineers and data scientists, the systems engineers have to provide a fine-grained specification that (a) describes all parts of the CPS, (b) how they might interact, (c) what data is exchanged between them, and (d) how the data inter-relates. A data scientists can then iteratively (including further refinements of the specification) prepare the necessary custom machine-learning models and components. Therefore, this work introduces a method supporting the collaborative definition of machine learning tasks by leveraging model-based systems engineering in the formalization of the systems modeling language SysML. The method supports the identification and integration of various data sources, the required definition of semantic connections between data attributes and the definition of the data processing steps within the machine learning support. Integrating machine learning-specific properties in systems engineering techniques allows non-data scientists to define a machine learning problem, document knowledge on the data, and further supports data scientists to use the formalized knowledge as input for an implementation.**

Index Terms—**Model-Based Systems Engineering, SysML, Systems Engineering, Machine Learning, Knowledge Formalization, Data-Driven Engineering, PLM**

I. Introduction

Leveraging data to allow experts making informed decisions during the product lifecycle of a product is recently defined as data-driven engineering [1]. With data-driven engineering, engineers are supported in the development and improvement of products and gain more insights during the product lifecycle. The knowledge required for implementing data-driven engineering is two-fold. On the one hand, profound machine learning skills to process data and implement algorithms. On the other hand, the domain knowledge of the product, processes and related data generated during the product lifecycle is required.

To connect the knowledge of the two disciplines, various methods have been proposed in recently [2]–[4]. However, these methods lack support for defining machine learning tasks to drive the implementation from an engineering perspective. Additionally, the methods mainly integrate engineering methods into data science methodologies supporting data scientists rather than from an engineer's perspective.

A recent industrial survey revealed that companies are having fewer experts in the companies and too little knowledge in data science and additionally, a little number of experts are available on the market [5]. Therefore, this work aims to integrate data science knowledge in systems engineering to support engineers in the definition of data science tasks and improve the efficiency and effectiveness of the implementation and, ultimately, support the product development. Particularly, means of model-based systems engineering (MBSE) [6] is adapted to allow to define a task for data-driven engineering by leveraging on data from the product lifecycle of a system. The approach of this work builds upon the systems modeling language SysML in Version 1.6 [7], a general-purpose modeling language allowing to formalize a system from various viewpoints of various disciplines.

This work lays a foundation for an interdisciplinary definition of machine learning tasks by formalizing knowledge from different involved disciplines in a single model-based approach. Additionally, semantic connection of data from various PLM interfaces allows to communicate knowledge on the origination and composition of data relations. With the entire definition of machine learning tasks without programming, this work is the foundation for future work to decompose the SysML model to derive executable machine learning code that is usable as a starting point for data engineers.

The contribution of this work is two-fold:

- A method for a formal model-based specification of

This work has been partially supported and funded by the Austrian Research Promotion Agency (FFG) via the "Austrian Competence Center for Digital Production" (CDP) no. 881843

machine-learning components for systems engineering, utilizing SysML.
- A discussion of perceived advantages and dis-advantages of such an approach.

The following section gives the background regarding model-based systems engineering and data science. In section III, the elaborated method is depicted aligned with a running example. Finally, the findings are discussed in section IV and summarized in a conclusion with future remarks in section V.

II. Background

First, the concepts of model-based systems engineering and the systems modeling language SysML are introduced. Next, machine learning and the CRISP-DM [8] methodology are introduced. Finally, related approaches are depicted and a summary of the background is highlighted in section II-C.

A. Model-Based Systems Engineering and SysML

Systems engineering and particularly model-based systems engineering (MBSE) aims in integrating different engineering disciplines in the product development to establish a single-source of truth by formalizing system requirements, behavior, structure and parametric relations of a system. With a graphical model-based approach, MBSE methods promise an increased design performance while supporting the communication of relevant stakeholders of a system [9], [10].

The general-purpose language SysML provides means to formalize the required means to enable MBSE graphically. The concepts to describe a model in a modeling language are defined in the metamodel. To create a metamodel for a specific group of use cases, either a new modeling language or an extension using stereotypes can be introduced, often referred to as domain-specific languages [11]. Additionally, approaches solely relying on the metamodel of SysML are possible without extending the metamodel for specific purposes [12]. However, it has been proven that the application of stereotypes supports the understanding of a model [13]. The stereotypes of a metamodel can be applied to blocks so to describe features of a system, subsystem or component of interest. With a state machine and state diagrams, the execution of one or multiple activities can be triggered and detailed in an activity diagram, defining the sequential execution of a block or other structural elements [7].

In literature, methods for the integration of SysML and computational methods have been proposed [14]–[16]. However, these methods introduce a new modeling semantic or are dedicated to specific interests unlike the machine learning task definition. Recently, [12] showed in an approach solely relying on SysML modeling that a constraint satisfaction problem can be entirely formalized. However, the application for machine learning tasks is still different and requires further investigation.

B. Data Science and Methodologies

Data Science and Business Intelligence refer to the extraction of information and knowledge from data through analysis to assist people with various types of insights, such as analysis or prediction, among many others [17], [18]. The digging of such information to derive knowledge is called data mining [19]. One sub-field of data mining is machine learning, which allows computer programs to automatically improve through experience [20]. Machine learning algorithms aim to learn to solve a (specific) problem so to eliminate the need for being explicitly programmed [21]. Machine learning methods are often categorized as *supervised*, *unsupervised*, *semi-supervised*, and *reinforcement learning* [22]. In supervised learning, labeled input and output pairs are used to extract patterns. Contrary, in unsupervised learning the data is not labeled. In this work, the focus lies on supervised and unsupervised learning. Therefore, the others are not depicted.

To support the implementation of machine learning applications, methodologies have been proposed in a general manner [8], [23]. Additionally, extensions of such methods with particular support for data science in the engineering domain are introduced [2], [3]. In CRISP-DM, the core steps are first to get an overview of the existing situation and the business understanding, further initial data is collected to understand the situation from a data point of view. Following this, the so-called data pre-processing is used to prepare data to be usable in a learning algorithm. Further, the algorithm is modeled, evaluated and deployed. In practice, these steps are iterative and multiple forth and back steps are taken.

C. Data-Driven Design and Summary

In literature, various approaches for the support of data-driven engineering are given, e.g. [24]–[26]. These approaches introduce specific metamodels or languages to describe a data science task and eventually enable to derive executable code. However, the approaches are not based on SysML and mainly rely on the implementation aspects of data science tasks without the integration in modeling practices of engineering disciplines. Therefore, the integration in MBSE and the interdisciplinary modeling is not depicted. Therefore, these approaches are rather domain-specific languages than general-purpose languages and do not support the communication between different disciplines.

Although these methods support the implementation of machine learning or data science applications, the support of engineers and the integration in the semantics of SysML is as of the authors' knowledge not available.

III. Method

This section describes a method to integrate machine learning task definition into SysML. First, the metamodel supporting the integration and allowing to reuse of specific parts for the modeling is depicted. Based on the metamodel, the implementation structure aligned with typical data science project structures of CRISP-DM is illustrated. Next, the semantics and syntax of the modeling approach are highlighted. In section III-D, the integration of interfaces and the implementation of the components so to use them in the machine learning task is described. Finally, the integration of predefined stereotypes

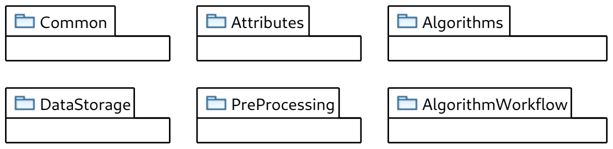

Fig. 1. The Organization of the Metamodel

and the workflow modeling using means of state machines relying on stereotypes is given.

A. Metamodel Extension

In SysML, packages are used to group and organize specific aspects of a system. Figure 1 depicts the organization of the metamodel, in *common*, general stereotypes are defined, usable for other stereotypes as parent. *Attributes* describe basic datatypes like *Integer* or *Float* with additional extension so to allow to describe the data in detail, e.g. the range of values, the format of the datetime. *DataStorage* defines available default interfaces required for the loading and processing of data from the PLM, e.g. SQL Servers, programmable application interfaces or different file formats. *Algorithm* consists of various groups of machine learning algorithms, e.g. Linear Regression or K-Means. In *PreProcessing*, descriptions of functions to convert or manipulate data are introduced, e.g. various data conversions, date converters or cleaning missing values. *AlgorithmWorkflow* consists of a stereotype for the definition of the workflow using the state diagram, more specifically described in section III-E.

Behind each package, specific stereotypes are defined so to describe a specific aspect of the machine learning task. More precisely, properties of interest are described to allow the definition of a specific part of the machine learning task, e.g. specific aspects of the pre-processing like in figure 2. Additionally, figure 2 depicts the hierarchical composition of the stereotypes. On top, there is the *ML* stereotype defined in common, allowing to indicate all derived stereotypes belong to the definition of machine learning tasks. This is useful in case multiple metamodels are integrated in a modeling approach and model transformations shall be applied to only machine learning concerns. The layers below can be defined as abstract (italic letter of the name) or black-box stereotypes, indicating a functionality is not yet defined in detail and require further investigation during the elaboration of the machine learning task. With each layer, the following stereotype inherit the attributes from the parents. With the inheritance and the black-box stereotypes, the iterative investigation of the SysML model can be achieved by various experts from different domains. The described hierarchical definition of the stereotype is applied in the other packages as well.

B. Implementation Structure

In CRISP-DM, each step represents a specific concern required to implement machine learning. The knowledge needed regarding business and data understanding is implicitly formalized within the SysML model. The data understanding

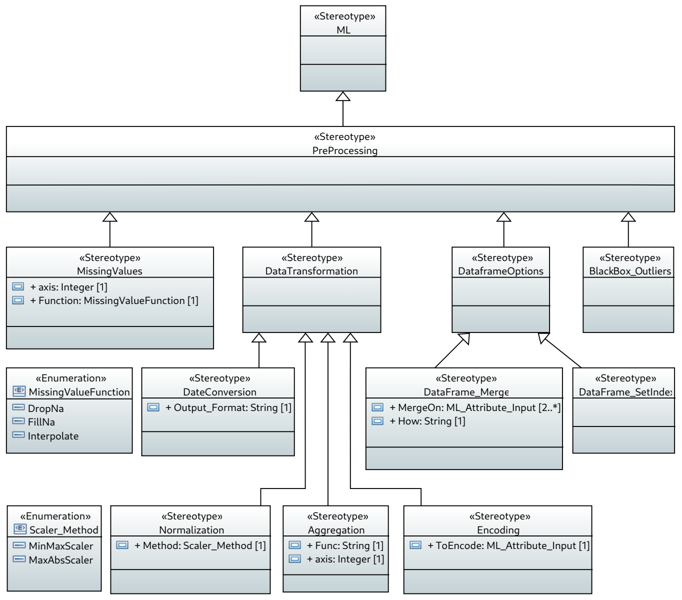

Fig. 2. The Metamodels for Pre-Processing

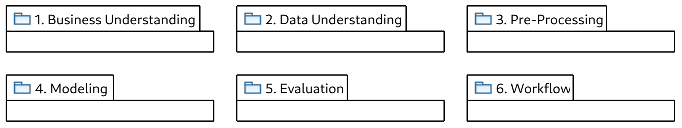

Fig. 3. The Implementation Structure Aligned with CRISP-DM

is partly modeled by the concerns of data interfaces as defined in section III-D. Figure 3 shows the package structure aligned with the phases of CRISP-DM. The business understanding consists of diagrams representing the engineering viewpoint of the system. The data understanding package consists of blocks and block definition diagrams describing the interfaces and related artifacts from the engineering system, e.g. specified from stereotypes in figure 5. The next step of CRISP-DM is one of the most crucial, the data pre-processing. In this step, the data is transformed and rearranged for usage in the modeling. Therefore, in this packages various methods on transforming data are defined. The defined methods in figure 2 are just an example of the definition. The last packages consist of algorithm modeling, describing available machine learning methods, like linear regression. The evaluation package consists of mathematical proofs like mean average error so to measure the success of the machine learning approach. Finally, the workflow package puts the different parts in a logical order of execution using a state machine.

C. Syntax and Semantics of the Implementation

The syntax of the implementation is derived from the SysML syntax and therefore, connections between elements are modeled accordingly [7]. Figure 4 depicts a sample of a shared directed association between two parts of the system. A shared association consists of a white tail diamond indicating the whole component and a head with an arrow showing the

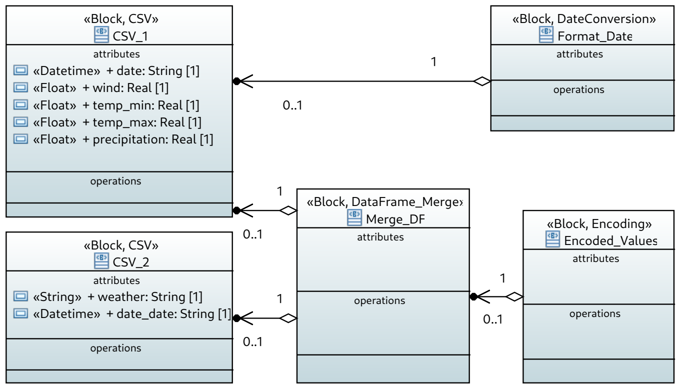

Fig. 4. The applied Syntax derived from SysML and Semantics with custom Stereotypes

connection is directed with the part component [7]. A directed association indicate that the whole component is aware of the part component but not vice-versa. The shared association is used to indicate that a part component can still be valid without the whole component, e.g. the date conversion on top right of figure 4 consists of a CSV file shown on top left, which is still a valid component even if the date conversion is not available anymore. Figure 4 additionally shows the application of a stereotype to define a specific behavior, e.g. output format of the date conversion on top right or the property *weather* in the *encoded_values* block below originating from the *Merge_DF* block. Additionally, the application of specific datatypes in the *CSV* block is shown. In the middle, the *Merge_DF* block shows that multiple connections can be drawn in case multiple inputs are required. If a stereotype misses a property, the one who formalizes the task in the model can add the property to the block without extending the metamodel. Therefore, flexibility and extensibility is given.

D. Sensor and Data Interface Modeling

In systems engineering, the integration of multiple system configurations during the design phase might be required [27]. The configuration of a system is indicated with a variation configuration stereotype as specified by [27] and depicted on top of figure 5. This sample shows the readily integration of multiple metamodels in the given approach. The configuration properties can be interpreted as interfaces of the system since the properties are either sensors or application programmable interfaces (API). The definition of configurations is out of scope. Still, if one adds properties to a configuration without an interface, the property must not be reflected in the realization of the machine learning task. The realization association, depicted as a white dashed arrow defines the output of the system, depicted as CSV files in this sample. With the realization association, the bridge between system components and data artifacts is established and the definition of the content of the file or other data structure like SQL Servers or other PLM systems is given.

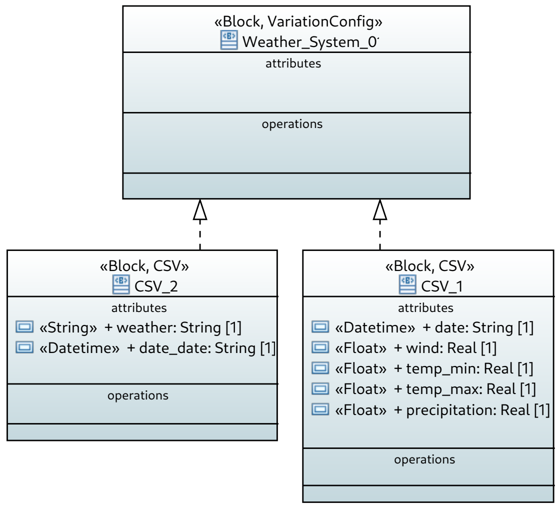
«Block, VariationConfig»
Weather_System_0
attributes
operations
«Block, CSV»
CSV_2
attributes
«String» + weather: String [1]
«Datetime» + date_date: String [1]
operations
«Block, CSV»
CSV_1
attributes
«Datetime» + date: String [1]
«Float» + wind: Real [1]
«Float» + temp_min: Real [1]
«Float» + temp_max: Real [1]
«Float» + precipitation: Real [1]
operations

Fig. 5. Sample of the Mapping between Sensor Data and the File Format

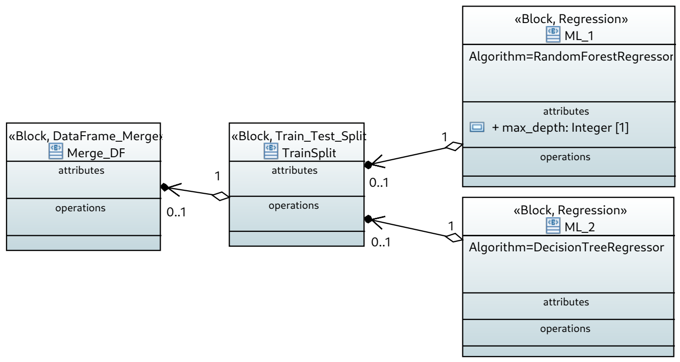

Fig. 6. Sample Implementation of an Algorithm

E. Integration of Predefined Methods and Workflow Modeling

Referring to the description of the metamodel extension in section III-A, specific parts of a machine learning algorithm can be predefined. These parts are used in the modeling of a machine learning task so to define a particular behavior and related activities to be done during the implementation of the application. Figure 6 depicts a sample of the integration of predefined stereotypes. The block on top right shows the application of a regression algorithm, specified as *random forest regressor*. The stereotype optionally defines properties that guide a user to specify concerns of interest. However, it is also possible to add user-defined properties to a block, as shown in the block on the top right with the attribute *max_depth*.

The process of defining specific aspects of a machine learning algorithm is executed iteratively. The last brick of the method consists of modeling the execution path of the machine learning application using workflows, formalized as state machine. In this approach, state machines are used rather than activities due to the level of abstraction required in this development phase and the lower complexity for various stakeholders to specify and understand the model. Each state is

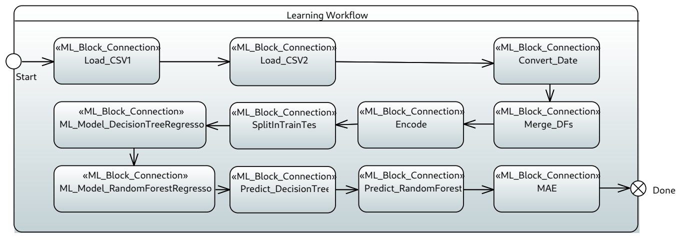
Learning Workflow
Start
«ML_Block_Connection» Load_CSV1
«ML_Block_Connection» Load_CSV2
«ML_Block_Connection» Convert_Date
«ML_Block_Connection» Merge_DFs
«ML_Block_Connection» Encode
«ML_Block_Connection» SplitInTrainTes
«ML_Block_Connection» ML_Model_DecisionTreeRegresso
«ML_Block_Connection» ML_Model_RandomForestRegresso
«ML_Block_Connection» Predict_DecisionTree
«ML_Block_Connection» Predict_RandomForest
«ML_Block_Connection» MAE
Done

Fig. 7. Sample Integration of the Workflow

connected using a stereotype with a block having a stereotype inherited from *ML* from the *common* package. A sample of a workflow for machine learning is depicted in figure 7. Each block consists of a *ML_Block_Connection* stereotype, which consists of a connection to the blocks like in figure 6. The formalization of more detailed workflows can be modeled if necessary using SysML activities and activity diagrams, which is currently not part of this work.

IV. Discussion

In this section, the newly introduced method is discussed. The structure of the section is as follows: First, the extension of the metamodel and the proposed structure of the implementation model are discussed. Next, the benefits and shortcomings of the modeling semantic is assessed. Finally, the integration of PLM interfaces and workflows using state machine diagrams is discussed. Each section highlights necessary work to improve the maturity of the approach by describing tasks for future work.

A. *Metamodel and Structure of Implementation*

The integration of an extended metamodel in SysML has been shown in a small sample. The generic hierarchical structure allows the individual extension without the need for defining any method or functionality that is required in the future. Although various aspects of a machine learning task definition can be reflected using parts from the machine learning community, the complexity of understanding how to program a machine learning application is still required. With the given template for the structure of the machine learning task definition, a first step towards supporting the interdisciplinary and unified definition is done. However, further investigation on the formalization is required to support the advantage of black-box modeling further.

B. *Complexity of Unambiguous Modeling*

With the inherited syntax from SysML, a broad range of rules and possibilities are defined, also applicable for the definition of machine learning tasks. The introduced semantic allows the interpretation of connections between different parts of the model. However, with increasing data properties, the number of blocks and subtasks to process the data increases and therefore unwanted associations can be created. These ambiguous associations harm the understandability and might lead to complicated interpretations, e.g. application of date conversion on a number of attributes without a date attribute. The introduction of model validation to assess the connections of a block with respect to the stereotypes could support to avoid the fuzziness and inconsistency of the model. Although the complexity increases with the number of data properties and PLM interfaces, the formalization of machine learning tasks supports the development of such computational support due to the preservation of knowledge and the possibility to reuse specific parts, which further leads to a reduction of cost and risk in the design [28]. However, the impact of the method in industry still needs to be proven. Additionally, the integration of model transformation to decompose the SysML model and derive executable machine learning code using dedicated programming languages is open for future work. This might lay a basis for the implementation and further support the development of machine learning in practice.

C. *Integration of PLM and Workflow Definition*

The integration of data from various data sources allows using the actual infrastructure. With the stereotype-based definition of high-level interfaces like SQL server connections, files or API interfaces can be integrated. The application of state machines allows the readily definition of an executable workflow of the defined functionalities. However, the programming of the various steps is still required. Therefore, future work consists of defining executable workflows with the integration of workflow engines like centurio.work [29]. With the workflow engine, the automatic gathering of data is enabled, which has been shown to be a core part for data-driven design using data from shopfloors [30].

D. *Potential Disadvantages*

Specific technologies or proprietary interfaces might be hard to describe, leading to additional effort, which can delay the implementation of machine-learning components. In the long-term this disadvantage might be an advantage, as it ensures the proper documentation of the implemented technologies. Furthermore, for huge projects, the complexity of the resulting models might be very high, including potential errors in the model, which might be very hard to find and thus lead to additional communication effort. Again we think that, as the models serve as the foundation for an implementation, and thus as a documentation of the implementation, despite potential additional effort, the benefits for replicability, maintainability and thus flexibility in the long-run will trump short-term delays. Finally, to leverage on the advantages of the model-based approach, the implementation of advanced model lifecycle management [31] is required so to support the working on chunks of a model without blocking further implementation and additionally, to allow the comparative work on various versions of a model.

V. Conclusions

In this work machine learning task definition using means of SysML is depicted. Particularly, the metamodel of SysML is extended so to enable the integration of relevant concerns from the data science community and the CRISP-DM methodology.

With the model-based systems engineering integration and the involvement of various stakeholders from different disciplines, an improvement in communication is expected. The approach is explained using a running sample. Future work will consist of validation in a case study and the applicability of model transformation to derive executable machine learning code as a basis for the implementation.

References

[1] J. Trauer, S. Schweigert-Recksiek, L. Onuma, K. Spreitzer, M. Mörtl, and M. Zimmermann, "Data-Driven Engineering – Definitions and Insights from an Industrial Case Study for a New Approach in Technical Product Development," Jan. 2020.

[2] S. Rädler and E. Rigger, "Participative Method to identify Data-Driven Design Use Cases," in *Proceedings of the International Conference on PLM*. Rapperswil, CH: Springer, 2020.

[3] S. Bitrus, I. Velkavrh, and E. Rigger, "Applying an Adapted Data Mining Methodology (DMME) to a Tribological Optimisation Problem," in *Data Science – Analytics and Applications*, P. Haber, T. Lampoltshammer, M. Mayr, and K. Plankensteiner, Eds. Wiesbaden: Springer Fachmedien, 2021, pp. 38–43.

[4] P. Stanula, A. Ziegenbein, and J. Metternich, "Machine learning algorithms in production: A guideline for efficient data source selection," *Procedia CIRP*, vol. 78, pp. 261–266, 2018. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2212827118310266

[5] S. Rädler and E. Rigger, "A Survey on the Challenges Hindering the Application of Data Science, Digital Twins and Design Automation in Engineering Practice," *Proceedings of the Design Society*, vol. 2, pp. 1699–1708, May 2022, publisher: Cambridge University Press.

[6] J. A. Estefan, "Survey of model-based systems engineering (MBSE) methodologies," *Incose MBSE Focus Group*, vol. 25, pp. 1–70, 2007. [Online]. Available: http://pdf.aminer.org/000/260/416/towards_a_unified_paradigm_for_verification_and_validation_of_systems.pdf

[7] OMG, "OMG Systems Modeling Language (OMG SysML™, Version 1.6)," 2019. [Online]. Available: http://www.omg.org/spec/SysML/1.6/PDF/

[8] R. Wirth and J. Hipp, "CRISP-DM: Towards a standard process model for data mining," *Proceedings of the 4th International Conference on the Practical Applications of Knowledge Discovery and Data Mining*, 2000.

[9] T. Huldt and I. Stenius, "State-of-practice survey of model-based systems engineering," *Systems Engineering*, vol. 22, no. 2, pp. 134–145, Mar. 2019, 65. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/sys.21466

[10] K. Henderson and A. Salado, "Value and benefits of model-based systems engineering (MBSE): Evidence from the literature," *Systems Engineering*, vol. 24, no. 1, pp. 51–66, Jan. 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/sys.21566

[11] M. Brambilla, J. Cabot, and M. Wimmer, "Model-Driven Software Engineering in Practice: Second Edition," *Synthesis Lectures on Software Engineering*, vol. 3, no. 1, pp. 1–207, Mar. 2017. [Online]. Available: http://www.morganclaypool.com/doi/10.2200/S00751ED2V01Y201701SWE004

[12] E. Rigger, R. Fleisch, and T. Stankovic, "Facilitating Configuration Model Formalization based on Systems Engineering," in *Proceedings of the Workshop on Configuration (ConfWS'21)*, Vienna, Austria, Sep. 2021.

[13] L. Kuzniarz, M. Staron, and C. Wohlin, "An empirical study on using stereotypes to improve understanding of UML models," in *Proceedings. 12th IEEE International Workshop on Program Comprehension, 2004*. Bari, Italy: IEEE, 2004, pp. 14–23. [Online]. Available: http://ieeexplore.ieee.org/document/1311043/

[14] C. Bock, R. Barbau, I. Matei, and M. Dadfarnia, "An Extension of the Systems Modeling Language for Physical Interaction and Signal Flow Simulation: SYSML FOR PHYSICAL INTERACTION AND SIGNAL FLOW SIMULATION," *Systems Engineering*, vol. 20, no. 5, pp. 395–431, Sep. 2017. [Online]. Available: http://doi.wiley.com/10.1002/sys.21380

[15] A. A. Shah, C. J. J. Paredis, R. Burkhart, and D. Schaefer, "Combining Mathematical Programming and SysML for Automated Component Sizing of Hydraulic Systems," *Journal of Computing and Information Science in Engineering*, vol. 12, no. 4, p. 041006, Nov. 2012, 00009.

[16] P. Klein, J. Lützenberger, and K.-D. Thoben, "A Proposal for Knowledge Formalization in Product Development Processes," in *Proceedings of the INternational Conference on Engineering Design, ICED15*. Milano, Italy: Design Society, Jul. 2015, pp. 261–272.

[17] L. N. Sanchez-Pinto, Y. Luo, and M. M. Churpek, "Big Data and Data Science in Critical Care," *Chest*, vol. 154, no. 5, pp. 1239–1248, Nov. 2018.

[18] W. Grossmann and S. Rinderle-Ma, *Fundamentals of Business Intelligence*. Springer, Jun. 2015, google-Books-ID: vX3MCQAAQBAJ.

[19] F. Provost and T. Fawcett, "Data Science and its Relationship to Big Data and Data-Driven Decision Making," *Big Data*, vol. 1, no. 1, pp. 51–59, Mar. 2013. [Online]. Available: http://www.liebertpub.com/doi/10.1089/big.2013.1508

[20] J. G. Carbonell, R. S. Michalski, and T. M. Mitchell, "1 - AN OVERVIEW OF MACHINE LEARNING," in *Machine Learning*, R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, Eds. San Francisco (CA): Morgan Kaufmann, Jan. 1983, pp. 3–23. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780080510545500054

[21] J. Koza, F. Bennett III, D. Andre, and M. Keane, "Automated Design Of Both The Topology And Sizing Of Analog Electrical Circuits Using Genetic Programming," Sep. 1998, noCitationData[s0].

[22] Q. Zhang, T. Kecskes, J. Mathe, and J. Sztipanovits, "Towards Bridging the Gap Between Model- and Data- Driven Tool Suites for Cyber-Physical Systems," in *2019 IEEE/ACM 5th International Workshop on Software Engineering for Smart Cyber-Physical Systems (SEsCPS)*, May 2019, pp. 7–13.

[23] U. Shafique and H. Qaiser, "A Comparative Study of Data Mining Process Models (KDD, CRISP-DM and SEMMA) | Semantic Scholar," 2014. [Online]. Available: https://www.semanticscholar.org/paper/A-Comparative-Study-of-Data-Mining-Process-Models-Shafique-Qaiser/a07e3fbc5f70eae7e9b586cb3a32d909370f75df

[24] E. Kusmenko, S. Pavlitskaya, B. Rumpe, and S. Stuber, "On the Engineering of AI-Powered Systems," in *2019 34th IEEE/ACM International Conference on Automated Software Engineering Workshop (ASEW)*. San Diego, CA, USA: IEEE, Nov. 2019, pp. 126–133, 6. [Online]. Available: https://ieeexplore.ieee.org/document/8967413/

[25] A. Bhattacharjee, Y. Barve, S. Khare, S. Bao, Z. Kang, A. Gokhale, and T. Damiano, "STRATUM: A BigData-as-a-Service for Lifecycle Management of IoT Analytics Applications," in *2019 IEEE International Conference on Big Data (Big Data)*, Dec. 2019, pp. 1607–1612.

[26] T. Hartmann, A. Moawad, F. Fouquet, and Y. L. Traon, "The Next Evolution of MDE: A Seamless Integration of Machine Learning into Domain Modeling," *2017 ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pp. 180–180, Sep. 2017.

[27] T. Weilkiens, *Variant Modeling with SysML*. Leanpub, Jul. 2014, noCitationData[s0]. [Online]. Available: https://leanpub.com/vamos

[28] B. Beihoff, C. Oster, S. Friedenthal, C. Paredis, D. Kemp, H. Stoewer, D. Nichols, and J. Wade, "A World in Motion – Systems Engineering Vision 2025," INCOSE, San Diego, California, Tech. Rep., 2014.

[29] F. Pauker and J. Mangler, "centurio.work - Higher Productivity Through Intelligent Connectivity," in *Wiener Produktionstechnik Kongress*, vol. 4. new academic press og, 2018.

[30] S. Rädler, J. Mangler, and E. Rigger, "Requirements for Manufacturing Data Collection to Enable Data-Driven Design," in *14th CIRP Conference on Intelligent Computation in Manufacturing Engineering*, Gulf of Naples, Italy, Jul. 2021, noCitationData[s0].

[31] A. Fisher, M. Nolan, S. Friedenthal, M. Loeffler, M. Sampson, M. Bajaj, L. VanZandt, K. Hovey, J. Palmer, and L. Hart, "3.1.1 Model Lifecycle Management for MBSE," *INCOSE International Symposium*, vol. 24, no. 1, pp. 207–229, 2014. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/j.2334-5837.2014.tb03145.x