TU WIEN Informatics

Modellgestützte Techniken zur Integration von Data Engineering in Systems Engineering

DISSERTATION

zur Erlangung des akademischen Grades

Doktor der Technischen Wissenschaften

eingereicht von

Simon Rädler, MSc.
Matrikelnummer 12035757

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Prof. Dr. Gerti Kappel
Zweitbetreuung: Prof. Dr. Stefanie Rinderle-Ma

Diese Dissertation haben begutachtet:

Prof. Dr. Gregor Engels

Prof. Dr. Johann Eder

Wien, 9. Oktober 2024

Simon Rädler, MSc.

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

TU WIEN Informatics

Model-Driven Techniques for Integrating Data Engineering into Systems Engineering

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

Doktor der Technischen Wissenschaften

by

Simon Rädler, MSc.
Registration Number 12035757

to the Faculty of Informatics

at Technical University of Vienna

Advisor: Prof. Dr. Gerti Kappel
Second advisor: Prof. Dr. Stefanie Rinderle-Ma

The dissertation has been reviewed by:

Prof. Dr. Gregor Engels	Prof. Dr. Johann Eder

Vienna, 9th October, 2024

Simon Rädler, MSc.

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Ich erkläre weiters, dass ich mich generativer KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Im Anhang "Overview of generative AI tools used" auf Seite 205 habe ich alle generativen KI-Tools gelistet, die verwendet wurden, und angegeben, wo und wie sie verwendet wurden. Für Textpassagen, die ohne substantielle Änderungen übernommen wurden, haben ich jeweils die von mir formulierten Eingaben (Prompts) und die verwendete IT-Anwendung mit ihrem Produktnamen und Versionsnummer/Datum angegeben.

Wien, 9. Oktober 2024

Simon Rädler

Danksagung

Zunächst möchte ich mich bei meiner Betreuerin, Frau Prof. Dr. Gerti Kappel, dafür bedanken, dass sie es mir ermöglicht hat, diese Dissertation zu erstellen. Ihre Erfahrungen und Ihr Netzwerk in der Wissenschaft ermöglichten es mir, wertvolle Kontakte zu knüpfen und Wissen von verschiedenen erfahrenen Wissenschaftlern zu erwerben. Ihr detailliertes Feedback hat mir auch geholfen, meine Arbeit ständig zu verbessern und den heutigen Reifegrad zu erreichen.

Des Weiteren möchte ich mich bei meiner zweiten Betreuerin, Frau Prof. Dr. Stefanie Rinderle-Ma, dafür bedanken, dass sie mich durch ihr ausführliches und kontinuierliches Feedback herausgefordert hat und mich über mich hinauswachsen ließ.

Ich möchte mich auch bei meinen zahlreichen Post-Docs bedanken, die mich immer unterstützt und gefördert haben. Allen voran Dr. Eugen Rigger, der mich über weite Teile meiner Dissertation betreute und mir das wissenschaftliche Arbeiten in Bezug auf Schreibstil und Stringenz beibrachte.

Auch Dr. Jürgen Mangler, durch welchen ich in Kontakt mit Prof. Rinderle-Ma und den I17-Lehrstuhl an der TU München kam und der mich sowohl bei Publikationen als auch bei der Sammlung wichtiger Evaluierungsdaten unterstützte.

Des Weiteren Dr. Luca Berardinelli, der mir Definitionen des Model-Driven Engineering beibrachte und mich bei der Durchführung einer systematischen Literaturrecherche unterstützte.

Ich möchte mich auch bei all meinen Arbeitskollegen bedanken, die mich während des gesamten Dissertationsprozesses auf verschiedene Weise unterstützt haben.

Persönlich möchte ich mich bei meinen Eltern Herbert und Hildegard bedanken, die mir mein Studium ermöglicht haben und mich immer ermutigt haben, meinen Weg zu gehen. Ein Dank gilt ebenso meinen Brüdern Daniel und Nico für Ihre motivierenden Worte über die gesamte Dissertation hinweg.

Abschließend möchte ich mich bei Verena bedanken, die mich in schwierigen und zeitintensiven Phasen der Dissertation unterstützt und Verständnis dafür gezeigt hat, dass ich an den Wochenenden und Abendenden teilweise lange gearbeitet habe.

Acknowledgements

First, I would like to thank my supervisor, Prof. Gerti Kappel, for giving me the opportunity to write this dissertation. Her experience and network in academia enabled me to make valuable contacts and acquire knowledge from various experienced researchers. Additionally, her detailed feedback on my work helped me to improve it so that it eventually became this work.

Furthermore, I would like to thank my second supervisor, Prof. Dr. Stefanie Rinderle-Ma, for challenging me with her detailed and continuous feedback and for letting me grow beyond myself.

I would also like to thank my numerous post-docs who have always supported and encouraged me. First and foremost, Dr. Eugen Rigger, who supervised me for long periods of my dissertation and instructed me in scientific work in terms of writing style and stringency.

Also Dr. Jürgen Mangler, through whom I came to the I17 chair of Prof. Rinderle-Ma at the TU Munich and who supported me both in publications and in the collection of study data.

Furthermore, Dr. Luca Berardinelli, who introduced me to definitions of Model-Driven Engineering and supported me in the execution of a systematic literature review.

I would also like to thank all my work colleagues who supported me in various ways throughout the dissertation process.

Personally, I would like to thank my parents Herbert and Hildegard, who made my studies possible and always encouraged me to follow my path. My thanks also go to my brothers Daniel and Nico, who have always motivated me.

Finally, I would like to thank Verena, who supported me in frustrating and time-consuming phases of the dissertation and showed understanding whenever I had no time on weekends or was working at innovative times.

Kurzfassung

Data Engineering (DE) gewinnt in der Systems Engineering (SE)-Praxis zunehmend an Bedeutung, um Entscheidungen zu unterstützen und voranzutreiben. Gleichzeitig werden Model-Driven Engineering (MDE)-Methoden angewandt, insbesondere im Model-Based Systems Engineering (MBSE), um der steigenden Komplexität moderner Systeme gerecht zu werden. Allerdings bieten MBSE-Methoden nicht genügend Unterstützung für die Datensammlung und -verarbeitung zur Gewinnung von Erkenntnissen und Wissen. Obwohl DE-Methoden vielversprechend sind, um diese Herausforderungen zu bewältigen, spiegelt sich ihr Einsatz in der Praxis und Forschung nicht ausreichend wider. Die Gründe dafür sind mangelndes Wissen über die Möglichkeiten von DE in der Praxis, unzureichende Ausarbeitung der Anforderungen für die Integration in bestehende Prozesse und technische Umgebungen, unklarer Nutzen der Integration von DE und Kommunikationsprobleme zwischen verschiedenen Disziplinen sowie hohe Implementierungskosten von DE-Lösungen.

Um diese Probleme zu lösen, wird in dieser Arbeit eine vierstufige Methode zur Integration von DE in SE vorgeschlagen. Der erste Schritt beinhaltet partizipative Workshops, um relevante Interessengruppen einzubeziehen, Wissen zu sammeln, die interdisziplinäre Kommunikation zu fördern und die Ergebnisse mit grafischen Modellierungsmethoden zu validieren. Im zweiten Schritt wird die Integration der gewünschten DE-Implementierung in bestehende Prozesse unterstützt. Der dritte Schritt beinhaltet die Formalisierung von DE-Aufgaben mit Hilfe von MDE-Techniken, um die Implementierung von DE-Anwendungen zu fördern und die interdisziplinäre Kommunikation während der Definition zu verbessern. Im vierten Schritt werden die formalisierten DE-Aufgaben zerlegt, um Codegenerierung zu ermöglichen, vorhandenes Wissen wiederzuverwenden und die Implementierungszeit zu reduzieren.

Die vorgeschlagene Methode erweitert bestehende State-of-the-Art-Methoden und wird anhand von zwei Anwendungsfällen validiert. Dabei wird die Anwendbarkeit in der Praxis und die Notwendigkeit der Einbeziehung verschiedener Disziplinen als Wissensquellen hervorgehoben. Zusätzlich wird in einer Benutzungsstudie die Anwendbarkeit und Nutzbarkeit für Nicht-Informatiker:innen, wie beispielsweise Maschinenbauer:innen und Datenwissenschaftler:innen demonstriert. Die vorgeschlagene Methode trägt zur Weiterentwicklung des modellgesteuerten Ansatzes für die Implementierung und Integration von DE-Lösungen im Systems Engineering bei.

Abstract

Data Engineering (DE) is gaining importance recently due to its ability to guide and drive decisions, e.g., by increasing efficiency and effectiveness in Systems Engineering (SE). At the same time, Model-Driven Engineering (MDE) methods are applied to SE, referred to as Model-Based Systems Engineering (MBSE), to manage the increasing complexity of modern systems in product development using models as primary artifacts. However, MBSE methods lack sufficient support for collecting and processing data to gain insights and knowledge, respectively. Although DE is promising to solve data collection and analysis challenges of MBSE, current state of practice and state of the art do not reflect the opportunities of DE in SE. Reasons for the gap are: first, a lack of knowledge in practice about the opportunities of DE and insufficient precondition elaboration to integrate with existing processes and technological environment. Second, unclear benefits of integrating DE into SE and communication issues among various disciplines lead to divergent expectations and missing acceptance in practice. Third, high efforts to implement DE applications lead to long duration and a bottleneck of available data scientists.

In response to these issues, this thesis proposes a method with four steps for integrating DE into SE by leveraging MDE techniques. The first step focuses on participative workshops involving relevant stakeholders to gather knowledge, promote interdisciplinary communication, and validate findings using graphical modeling methods. The second step supports the integration of a desired DE implementation into existing processes. The third step supports the formalization of DE tasks using MDE techniques, aiming to drive the implementation of DE applications while fostering communication. The fourth step decomposes formalized DE tasks to enable code generation, reuse existing knowledge, and reduce implementation time.

The proposed method extends existing state of the art methods, consolidated through a Systematic Literature Review. The underlying methods are validated in two use cases, highlighting the applicability in practice and the necessity to involve various disciplines as knowledge sources. Further, a user study indicate applicability and usability for non-programming engineers, e.g., mechanical engineers, and data scientists in practical samples. This method contributes to various research disciplines by introducing and evaluating a model-driven approach to facilitate the implementation and integration of Data Engineering in Systems Engineering.

Contents

CHAPTER 1

Introduction

The application of Data Science (DS) has emerged in various industries to support decision-making [Pow16, PF13], gain additional insights in manufacturing [DB21] or predict specific behavior in future [Sar21]. DS is an umbrella-term to unify various concepts from statistics, data analytics and informatics with the goal to extract information and respectively knowledge by analyzing data to derive for example patterns or trends and report them as human-understandable insights [SLC18], e.g., via text or images generated by Machine Learning (ML) algorithms. The term DS today is mostly used in the context of the intersecting field of ML, aiming to solve a specific problem without the need for being explicitly programmed [KBAK96], but by learning from past data [MRT18]. However, to enable improvement through experience, data is required in a machine-readable format with traceable data relationships. In this context, Data Engineering (DE) includes the steps of collecting, storing, and processing data so that algorithms such as those in the field of ML can be applied to it.

At the same time, Model-Driven Engineering (MDE) is gaining relevance for the engineering of complex systems, such as the design of aerospace systems [MS18]. A key reason for the growing interest of MDE and the related Model-Based Systems Engineering (MBSE), which focuses on support for the engineering of systems, is the rising complexity of systems in development by its number of components, functions, interactions and involved disciplines [BOF+14, MS18]. Recently, the International Council on Systems Engineering (INCOSE) published an updated Systems Engineering (SE) vision for 2035, highlighting that system complexity is not just raising, it will *explode* in the upcoming years due to additional factors, such as environmental sustainability, interconnected systems and the digital transformation that changes products and product development [DFM+22].

SE is a transdisciplinary and integrative approach, enabling to design, integrate and manage complex systems among the Product Lifecycle Management (PLM), while addressing the needs of various disciplines and processes [HdFV19, Mad18, SMM+19]. MBSE is

state of the art and holds promise for improving design performance and addressing the complexity of multidisciplinary systems by providing methods and tools for communicating and managing data related to system development by utilizing models as primary artifacts [HS19, HS21, MP19]. In this regard, key challenge of SE is to improve performance in the collection and analysis of data, information and knowledge, and achieving effective communication between various stakeholders and disciplines at different stages of (technical) product development to present shared information as a common context for discussion [MS18].

Although MBSE and DE for technical product development are both aiming to support an efficient and effective product development, the means of the two branches of methods to achieve this goal is substantially different. MBSE aims to integrate knowledge of various engineering disciplines in product development to establish an authoritative source of truth by formalizing system requirements, behavior, structure and parametric relations of a system using models as primary artifacts. Contrary, DE aims to collect and utilize use-case oriented data to guide and drive decisions, e.g., during the development process of systems, by enabling data-driven decision making. However, as of today, there is no method in literature that supports the integration of DE into MBSE. Additionally, opportunities of DE in technical product development practice are often unknown and the integration in daily operations is cumbersome [RR22]. This lack of integration of DE into MBSE and the lack of awareness of the opportunities of DE in technical product development practice defines the motivation of this research.

In literature, the integration of DE capabilities to guide and drive decisions in the (technical) product development process, e.g., to foster performance in the product development, has been defined as a framework called Data-Driven Engineering (DDE) [TSOO+20]. Due to the entanglement of the terms SE and technical product development [AZ13, SMM+19], in the following, DDE is used as a synonym for the integration of DE into SE, more specifically into the state of the art methods of MBSE. Reasons for the cumbersome application of DDE in practice can be broken down into the following aspects:

First, the elaboration of DDE requires the involvement of multiple disciplines, such as certain domain experts being aware of day-to-day processes and shortcomings in the daily business, data scientists knowledgeable about implementing DE applications, and software engineers experienced to integrate with Information Technology (IT) infrastructure and processes [HSM+19]. Consequently, methodological support must reflect each stakeholder's perspective to allow the identification of DE opportunities in practice. However, in today's practice, the technical understanding of domain-specific processes and the collection of data from the perspective of data scientists is rarely reflected in data mining methodologies [BVR21]. Additionally, a lack of knowledge regarding the elaboration of DE can be observed in practice [RR22]. Therefore, methodological support must reflect

the necessary knowledge of all relevant stakeholders so that potential use cases for DDE can be identified and the development of DDE can be streamlined. Particular focus is required on systematic knowledge gathering and support for data scientists regarding requirements definition and implementation support of DDE in practice.

Second, practitioners, most of whom are not data scientists, report that in day-to-day business, the benefits of DE are unclear and a lack of business models to gain benefit from DE can be observed, leading to issues including financial problems and reduced acceptance [RR22]. Among others, low acceptance originates from divergent (unrealistic) expectations on DE capabilities [Ana22], misleading communication and knowledge issues [Hag21, HPH22], e.g., through different background knowledge, or because DE is often a blackbox for users and therefore, not understandable or trustworthy [KURD22, Shi21]. Consequently, methodological support requires consolidating knowledge and methods from different disciplines and helping to communicate DE problems aligned with the targeted or elaborated support in a neutral format that does not require expertise in a specific domain to be understood.

Third, the implementation of DDE requires considerable effort to achieve sufficient data quality and data availability before implementation, which is often not given from the early development due to missing interfaces, missing sensor specifications to gather data or data acquisition strategies [DB21, RMR22, RR22, WWIT16]. In this context, the definition of data interfaces, data attributes and interconnection of data with respective level of detail must be determined and described before implementation to improve the potential to succeed in the elaboration of DDE. Additionally, automated data collection and analysis must be integrated in actual processes to support the implementation and allow algorithms to be tested on actual data, which can reduce, e.g., concept drifts, which are statistical changes in target variables or data [Tsy04]. Consequently, communication overhead and time-consuming manual data collection are reduced [RMR22]. Additionally, DE knowledge and experiences of data scientists is required from early elaboration of DDE, which leads to an increasing demand for data scientists in practice [MCO16, MH17]. However, the number of available data scientists is too little to meet the demand of the industry [Ana22, RR22]. Therefore, methodological support requires supporting other disciplines to take over tasks, which lead to reduced effort by data scientists and potentially lead to a reduction of the implementation duration. Eventually, taking over responsibility by domain experts leads to increased project support and acceptance. Thus, focus needs to be put on knowledge reuse and communication to foster the understanding and allow to transfer knowledge among involved disciplines.

1.1 Objectives

The goal of this work is to methodologically support the integration of DE into MBSE to increase the usage of ML and other data-driven capabilities in industry and therefore, increase efficiency and effectiveness of SE. The methodological support focuses on support prior to and during the implementation of DDE, whereas the maintenance of the resulting

DDE tool is considered a minor goal. This leads to the following objectives:

First, to allow the successful elaboration of DDE in practice, a systematic decomposition of actual processes, related IT infrastructure and data interfaces of a product or production environment is necessary to integrate elaborated tool support in the development environment/processes of a company. With respect to this, a consolidation of DE and MBSE methodologies is aimed to allow a comprehensive analysis and identification of potentials and opportunities in a use-case oriented development of tool support. Particularly, methodological support requires to focus on collaboration and communication to provide a comprehensive and understandable overview of existing opportunities of DE to practitioners of involved domains.

Second, to enable an efficient and effective development of DDE, prerequisites for the implementation of DE capabilities need to be elaborated to reduce delays during implementation. In this respect, a method is required to identify and define prerequisites from a DE point of view with additional focus on the integration of the developed tool support into daily use. Therefore, focus is put on actual and required data interfaces and the level of detail with the potential to interconnect data artifacts among systems, e.g., if a production process measures a quality attribute of a product, it is necessary to trace the data so to connect one production piece with all generated data among the manufacturing. Consequently, the method requires to take various Cyber-Physical System (CPS) components, such as smart sensor systems, into concern, which can be seen as a key-enabler for digitization and automatic data collection in paradigms such as Industry 4.0 [AH17, MV18].

Third, to allow the definition of DE interfaces and the integration into early product development, the integration of DE into MBSE needs to be elaborated. Due to the collaborative character of the DDE development, focus is put on the integration of DE implementation knowledge into existing modeling languages used in MBSE, such as Systems Modeling Language (SysML). This DE implementation knowledge is also called formalized knowledge about DE. With the representation of different engineering viewpoints and the DE viewpoint using a General-Purpose Modeling Language (GPML), improved collaboration and communication between disciplines is expected [JSD+22]. Additionally, an increase in the acceptance of the solutions and an improved leveraging of the potentials due to the explicit integration of solutions into current processes as well as through the potentially increased understanding is expected.

Fourth, in order to decompose formalized knowledge about DE, the application of MDE techniques, in particular partial and full code generation based on code snippets and formalized DE implementation knowledge will be integrated. With code generation, a reduction of costs and implementation time is expected, while knowledge transfer among various disciplines is fostered.

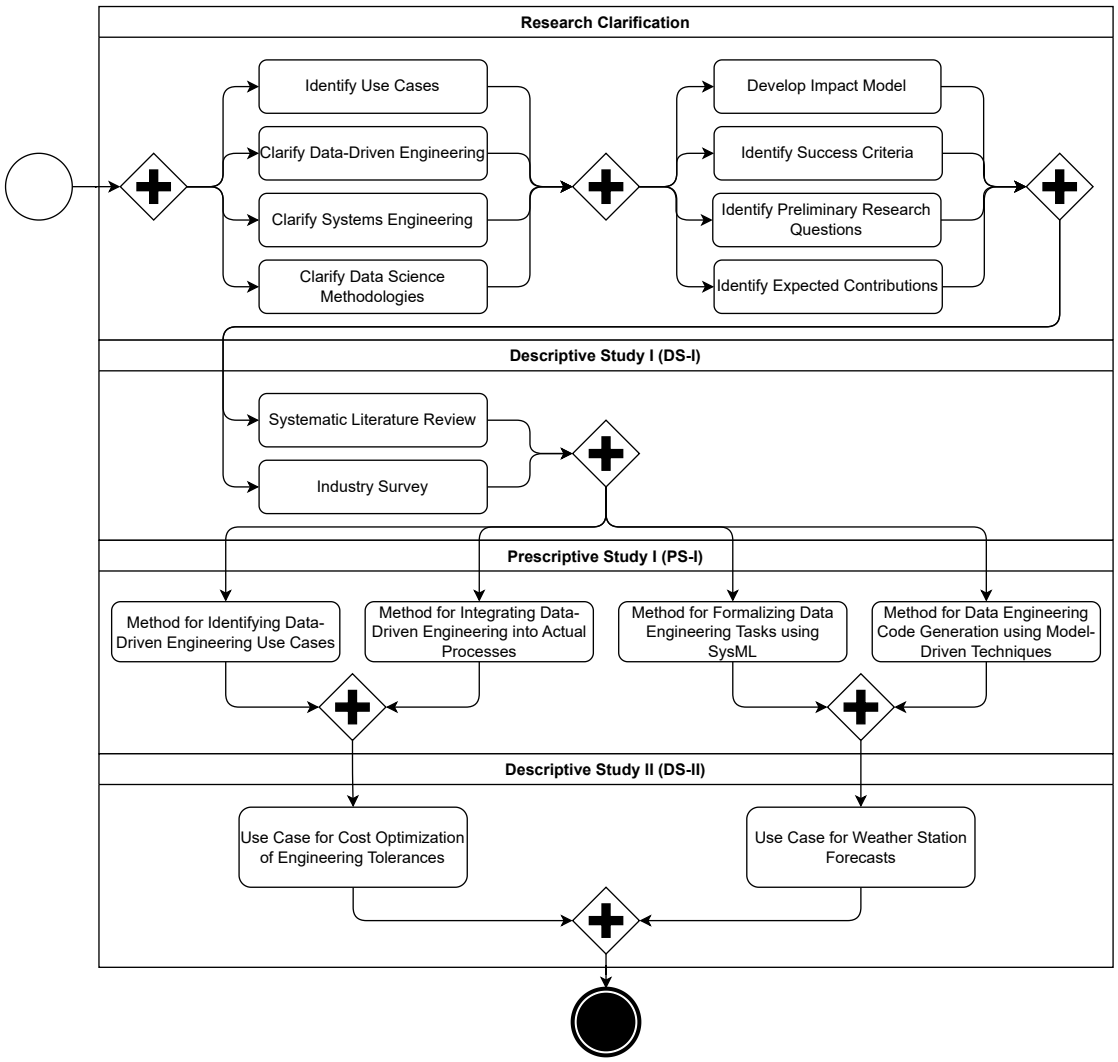


Figure 1.1: Overall research method aligned with [BC09].

1.2 Research Method

The overall research method follows the descriptive-prescriptive-descriptive approach proposed in the Design Research Methodology (DRM) by [BC09]. Figure 1.1 presents the embedded research method in the four-step methodology of the DRM. Each step of the applied research method is represented by a single swim-lane. In each step, main tasks are defined, which can either be processed in parallel or have to be executed sequentially. Although the steps of the research method are sequentially represented, cycles have to be made and various tasks, such as Descriptive Study I (DS-I) and Prescriptive Study I (PS-I) are partly executed in parallel.

The *research clarification* aims to get a grasp idea of the research topic and to sharpen

the research topic towards research questions and expected contributions. Following this, the DS-I is conducted to elaborate on requirements and current state of the art of the research topic. The literature review aims at collecting alternative approaches and sharpen the vision of the aimed support for DE integration in MBSE. With an industry survey, the outcome of the research is expected to be more concise and the focus driven more by practitioner problems, which promises to increase practical applicability.

Based on the results of DS-I, the intended methodological support for DDE is elaborated as a four-step method in PS-I, which are presented as sub-methods due to the possibility of performing the various sub-methods independently. Therefore, in the following, each sub-method is treated as a single method. The first two and the second two methods can be grouped due to their characteristics. Particularly, the first two methods can be characterized as preconditions, requirement and initial preparation steps. The second two methods are implementation support for DE. Due to the interleaving of the first two and the second two methods, the two groups of methods can be developed in parallel.

The developed methods are evaluated in initial case studies during the Descriptive Study II (DS-II). Due to the grouping of the method development, also the evaluation is split in two case studies.

1.3 Research Questions

According to [BC09], the initial step of the DRM supports the research clarification, aiming to derive Research Questions (RQs) and sharpen the scope of research using an impact model to contextualize influences and measurable criteria. The impact model illustrated in Figure 1.2 contextualizes the developed support (blue) with the influencing factors on the overall Research Goal (RG), depicted as success factor. From a methodological perspective, first, factors influencing the *success factor* were identified based on literature [GET19, PBBA23, SYS20] and discussions with experts from the field of DE and MBSE. These experts have been part of the Austrian Center for Digital Production (ACDP) and participants of the industry study, see chapter 4. Next, arrows are drawn to represent influence, with the "+" and "-" at the end and tip of the arrow showing how one factor influences the others, e.g., if the factor at the end of the arrow increases (+), the factor at the tip of the arrow also increases (+) or decreases (-). If a tip of an arrow is not indicated by a "+" or "-" but with a "0", it shows that there is an influence but it cannot be said whether it is generally positive or negative. Finally, measurable success factors and *key factors* were identified to further focus the research interests.

In order to achieve the main objective of this thesis, which is to *increase the efficiency and effectiveness in SE*, the main objectives of MBSE and DE must be consolidated. In this respect, the *rate of decisions based on DE capabilities in SE* is required to be promoted. To increase the rate of DE in SE, three *key factors* are identified:

The first *key factor* is the necessity to *increase the acceptance of DDE* in practice. To

enable an increasing acceptance, the quality of provided support needs to be improved, which is influenced by the level of integration in the processes of the domain experts, the number of identified and valid use cases and the quality of data. However, quality of data is beyond the scope of this thesis due to the complexity and extent required to methodically support, assess, and improve data quality. To increase the number of identified and valid use cases, knowledge on obstacles hindering the implementation and knowledge on DE capabilities is required. Moreover, the level of integration in actual processes can be fostered by collecting knowledge on data attributes and data relationships, e.g., how are data attributes connected with downstream process data artifacts? Particularly, the integration with actual processes in manufacturing is required to enable contextualized data utilization in DDE.

The second *key factor* supporting the rate of DDE is the *maturity of integration of DE into MBSE processes and methodologies*. Since MBSE is state of the art in SE, the knowledge regarding DE tasks needs to be formalized through models, to enable integration within existing methods and foster validation of knowledge. To increase the amount and degree of validated knowledge, the effort for knowledge formalization requires to be minimized. Consequently, a reduced effort in knowledge formalization increases the rate of validated knowledge, which further supports the level of DE integration in MBSE.

The third *key factor* is the *influence of costs on the rate of DDE* usage. Particularly, the higher the costs for the DE implementation, the less the rate of DDE. Among others, the cost for the implementation is determined by the effort for the implementation itself, which can be reduced by knowing preconditions, data artifacts, and hindering obstacles. Furthermore, with a systematic decomposition of formalized and validated knowledge using automatic code generation, the development effort is potentially reduced.

As result of the research impact model, the following **main RQ** can be stated:

Main RQ What means are required to support the implementation and integration of Data Engineering in Systems Engineering?

With respect to the objectives of this thesis and the impact model, the following refined RQs can be posed (Note, RQ1 was formulated as input to the Systematic Literature Review (SLR), which marks the first phase of this thesis work, see Chapter 3. The term Artificial Intelligence (AI) has been used in the broadest sense covering also Data Engineering (DE).):

RQ1 What is the current state of the art of Model-Driven Engineering with extensions to formalize Artificial Intelligence methods and applications?

RQ2 What obstacles hinder the application of Data-Driven Engineering in practice?

RQ3 What is required to promote the integration of Data-Driven Engineering in practice?

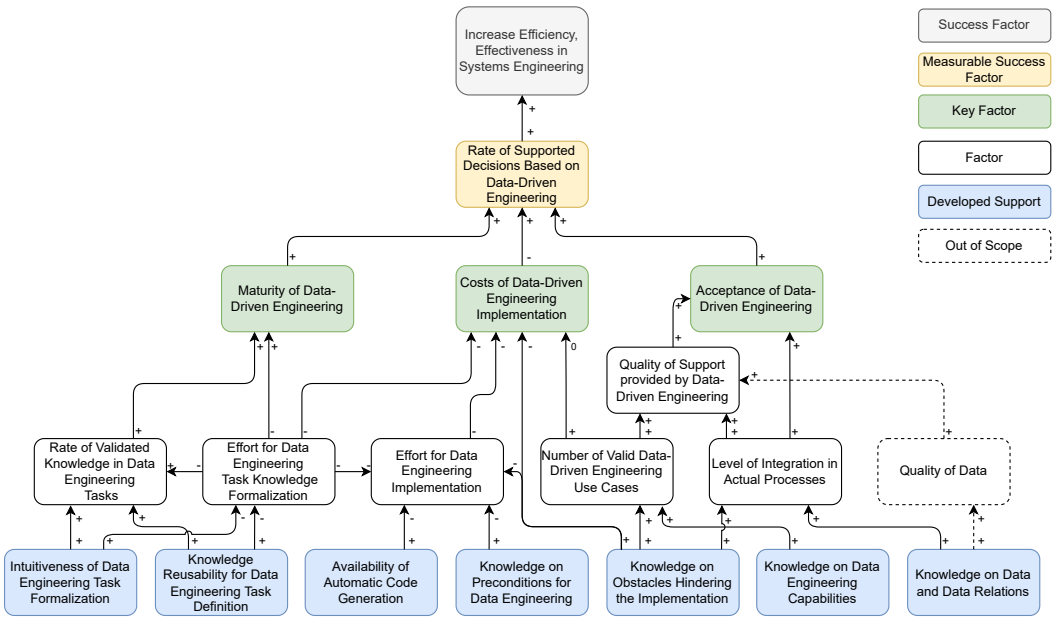


Figure 1.2: Research impact model describing influences on the rate of Data Engineering in Systems Engineering (Data-Driven Engineering).

RQ4 What are appropriate methods to identify use cases for Data-Driven Engineering?

RQ5 What are the prerequisites in a company so that manufacturing data can be leveraged for Data-Driven Engineering?

RQ6 What extensions to graphical modeling languages such as SysML are needed to integrate Data Engineering tasks comprehensively into Systems Engineering processes, to formalize product and process knowledge as well as data artifacts such as data attributes, interfaces, and data transformations?

RQ7 Given a system model that represents data attributes, interfaces, and the formalization of Data Engineering tasks: What model properties can be used to automatically derive an executable Data Engineering model using Model-Driven Engineering techniques?

1.4 Publications

Parts of the described results of this dissertation have been published in peer-reviewed scientific venues, another part is currently under reviewing. Due to the evolution of the thesis and the particular focus of the publishing venues, the term ML used in the publications is referring to the newly adapted terms DE and DDE as depicted in

Section 2.1.4 and Section 2.3.5. To indicate which publications are used as basis in which chapter, at the beginning of each chapter, a reference to the respective publication(s) is given.

An overview of published manuscripts during the development of the thesis is presented in the box „Publications 1: All Thesis Publications“:

Publications 1: All Thesis Publications

[RR20] Simon Raedler and Eugen Rigger. Participative Method to Identify Data-Driven Design Use Cases. In Product Lifecycle Management Enabling Smart X, volume 594, pages 680–694, Cham, 2020. Springer International Publishing. doi:10.1007/978-3-030-62807-9_54.

[RMR22] Simon Raedler, Juergen Mangler, and Eugen Rigger. Requirements for Manufacturing Data Collection to Enable Data-Driven Design. In Procedia CIRP, volume 112 of 15th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 14-16 July 2021, pages 232–237, January 2022. doi: 10.1016/j.procir.2022.09.077.

[RR22] Simon Raedler and Eugen Rigger. A Survey on the Challenges Hindering the Application of Data Science, Digital Twins and Design Automation in Engineering Practice. In Proceedings of the Design Society, volume 2, pages 1699–1708. Cambridge University Press, May 2022. doi: 10.1017/pds.2022.172.

[RRMR22] Simon Raedler, Eugen Rigger, Juergen Mangler, and Stefanie Rinderle-Ma. Integration of Machine Learning Task Definition in Model-Based Systems Engineering using SysML. In 2022 IEEE 20th International Conference on Industrial Informatics (INDIN), pages 546–551, Perth, Australia, July 2022. IEEE. doi: 10.1109/INDIN51773.2022.9976107.

[RRRR24] Simon Raedler, Matthias Rupp, Eugen Rigger, and Stefanie Rinderle-Ma. Model-Driven Engineering for Machine Learning Code Generation using SysML, March 2024. doi: 10.18420/MODELLIERUNG2024_019.

[RBW+23] Simon Raedler, Luca Berardinelli, Karolin Winter, Abbas Rahimi, and Stefanie Rinderle-Ma. Model-Driven Engineering for Artificial Intelligence – A Systematic Literature Review, July 2023. doi: 10.48550/arXiv.2307.04599.

[RMR23] Simon Raedler, Juergen Mangler, and Stefanie Rinderle-Ma. Model-Driven Engineering Method to Support the Formalization of Machine Learning using SysML, July 2023. doi: 10.48550/arXiv.2307.04495.

[RBW+24] Simon Raedler, Luca Berardinelli, Karolin Winter, Abbas Rahimi, and Stefanie Rinderle-Ma. Bridging MDE and AI: A systematic review of domain-specific languages and model-driven practices in AI software systems engineering. Software and Systems Modeling, Sept. 2024. doi: 10.1007/s10270-024-01211-y.

1.5 Thesis Structure

The remainder of this thesis is divided into the following chapters:

Chapter 2 introduces relevant background on utilized methods to enable the understanding of the proposed method. Chapter 2 deals with *Step 1* in Figure 1.1 called Research Clarification. First, relevant background from DE is defined with particular focus on the high-level concepts of Data Mining (DM), ML and the used definition of DDE. Furthermore, the Cross Industry Standard Process for Data Mining (CRISP-DM) is introduced. Second, the basic concepts of MDE, namely *model*, *metamodel*, *model transformation*, the differences between Domain Specific Modeling Language (DSML) and General-Purpose Modeling Language (GPML) are presented. Third, an introduction to Systems Engineering (SE), Model-Based Systems Engineering (MBSE) and its defacto standard language SysML is provided. Finally, various used methods, such as from Lean Six-Sigma toolset, and Enterprise Architecture (EA) are introduced, relevant for the elaboration of use cases for DDE.

Chapter 3 and Chapter 4 cover *Step 2* in Figure 1.1 called Descriptive Study I (DS-I).

Chapter 3 covers the finding of a SLR on the intersection of MDE and AI. The SLR aims to collect and assess the overlap with AI and related terms instead of DE, as there are few publications that use DE to consider MDE approaches related to AI. Within the SLR, methods with focus on Model-Driven Engineering for Artificial Intelligence (MDE4AI) are collected and assessed with regard to MDE and AI concerns. MDE concerns are assessed regarding language engineering characteristics of DSML/GPML, e.g., introduction of metamodel, usage of model transformation and transformation intent or concrete syntax representation. AI concerns are assessed with respect to implementation support of the methods, aligned with the CRISP-DM methodology, e.g., are capabilities of MDE used to describe the *data preparation* or are means of MDE used to describe the integration of e.g., *ML algorithms* with relevant hyper-parameters.

Chapter 4 introduces the findings of a conducted industrial survey, aiming to collect the current state of practice on obstacles hindering the implementation of DS capabilities in practice and the motivational factors of using DS. Furthermore, the experiences on conducted DS projects are collected. In this chapter, DS is used as the core term, as DE is rarely used in practice.

Chapter 5 overviews the developed four-step method. Additionally, the two primary case studies for evaluating the subsequent methods are introduced. Note that each step is presented as independently as possible to enable the application without applying the entire method. However, dependencies and pre-conditions due to the iterative fashion of the method exist. For each method, first, a general context and introduction to related work or research gaps are introduced, followed by main steps of the method. Furthermore, evaluation using the core case studies is given with a compelling discussion highlighting implications for industry and research. The subsequent steps of the method are depicted in the following chapters.

Chapter 6 to Chapter 9 cover *Step 3 and 4* in Figure 1.1 called Prescriptive Study I (PS-I) and Descriptive Study II (DS-II).

Chapter 6 presents a method that enables the elaboration of use cases in a realistic environment by leveraging the knowledge of relevant stakeholders in participatory workshops. The chapter shows the integration of participative workshops and the formalization of validated knowledge using model-based approaches, e.g., EA modeling or EA-based modeling of Value-Stream Mapping (VSM).

Chapter 7 builds upon the previous chapter and supports the elaboration of target EA integration of required applications, e.g., integration of automated data collection mechanisms to enable the systematic collection of data using various interfaces of PLM systems or integration of DDE applications in existing processes and IT artifacts.

Chapter 8 facilitates the implementation of DE by extending means of SysML with stereotypes to enable the formalization of implementation relevant knowledge. To incorporate DE-relevant knowledge into MBSE, processes and guidance from CRISP-DM are used to structure required DE knowledge.

Chapter 9 presents a method that builds on the previous chapter to reuse the formalization of DE knowledge by using model transformation to generate code. In particular, the approach builds on template-based code generation that integrates the formalized knowledge of the SysML model into predefined code snippets.

Chapter 10 summarizes the contributions of this thesis aligned with the work's objectives and aims. Furthermore, the applied research method is discussed, the implications for industry and research are highlighted, and relevant future research is proposed.

CHAPTER 2

Background

This section presents relevant background on the two core topics, namely Data Engineering (DE) and Model-Driven Engineering (MDE). Additionally, side-topics and relevant methods from the toolset of Lean Six-Sigma are introduced. The chapter is organized as follows: First, an overview of the key concepts and the relationship between DS/DM and ML is given. Second, the concept of MDE and the implementation with SysML is presented. Next, Lean Six Sigma methods used in the elaborated methods are introduced. Finally, preliminary knowledge on EA is given.

2.1 Data Science, Data Mining, Data Engineering and Machine Learning

Artificial Intelligence (AI) in the broadest sense marks the data-driven and knowledge-driven area of science with numerous practical applications, ranging from image/voice recognition to recommendation systems and self-driving cars [RN21]. Although the topic seems relatively new due to the media presence of AI and especially the emerging trend of chatbots like ChatGPT [Ope23], AI was defined several decades ago and has been expanded and redefined by the evolution of the topic [Eur20]. A side effect of the evolution of the topic is that different definitions are used depending on the underlying methods and the area of application. Similarly, not only the definitions of AI, but also the various terms associated in the literature have various definitions. For example, terms such as Machine Learning (ML), Data Science (DS), and Data Mining (DM) are used differently depending on the application area [ED19].

In this respect, in the following, the terms DS, DM, ML and the de facto standard methodology for supporting the implementation of DS/DM projects, CRISP-DM are defined as they are required to be understood for the particular applied domains of this thesis.

2.1.1 Data Science

The term DS acts as an umbrella term referring to the interdisciplinary field of collecting data, extracting information and knowledge by analyzing data to derive patterns [FPS96], trends, etc., and report them as human-understandable insights that are beneficial for various areas, such as manufacturing [DB21, SLC18, SS18]. In today's applications, DS heavily relies on the application of ML algorithms with its sub-fields such as Deep Learning (DL).

2.1.2 Data Mining

DM is the actual extraction of knowledge from datasets, and processes [PF13]. Due to the growing size and complexity of data, DM became one step within a more holistic DS process [MCF+21]. However, due to the intense entanglement of the terms DM and DS, they will be used interchangeable in this thesis.

2.1.3 Machine Learning

Machine Learning (ML) is a sub-field of the broader area of AI, and allows computer programs to automatically learn from existing data [MRT18]. The whole (pre-)processing of the data is part of DS, and thus ML is intersecting with DS [KD18]. ML algorithms aim to solve a specific problem by eliminating the need for being explicitly programmed [KBAK96]. In today's practice, the application of ML is becoming increasingly important due to the growing amount of available training data and enhanced accuracy [GBC16]. For the rest of this thesis, we do not distinguish between the commonly used categorization of ML, namely *supervised*, *unsupervised*, *semi-supervised*, and *reinforcement learning* [MRT18].

2.1.4 Data Engineering

As previously introduced, the terms DS, DM, and ML refer to the extraction of valuable insights from existing data. Consequently, the data must be available, sufficiently interconnected, and most likely pre-processed, before they can be used by any ML algorithm. These steps are separate within an upstream process called DE [RH22]. According to [RH22], the DE lifecycle consists of *Generation, Storage, Ingestion, Transformation* and *Serving*. The ingestion, transformation and serving of data is equally to the well established term Extract, Transform, Load (ETL). Thus, DE can be characterized as necessary prerequisite to ML, since ML should apply an algorithm only on pre-processed data.

2.1.5 Cross Industry Standard Process for Data Mining (CRISP-DM)

Regardless of the applied methods enabling decision support, standard procedural models for data handling exists [AS08].

The *de facto* standard methodology for data handling in the industry is CRISP-DM [MSMF09]. The first version of CRISP-DM was elaborated in 1999 based on funding from the European Union by an expert team from industry [CCK+00]. The CRISP-DM methodology consists of guidance on four levels: phases (steps), generic tasks, specialized tasks and process instances [CCK+00]. The six phases are executed sequentially but also iteratively and flexibly, and it is not essential to perform all steps in one project. The six phases of CRISP-DM can be summarized as follows:

1. **Business Understanding:** Project objectives, requirements, and an understanding from a business level is achieved. Based thereon, a data mining problem is defined, and a rough roadmap is elaborated.

2. **Data Understanding:** Data is collected to understand the situation from a data point of view.

3. **Data Preparation** The construction of the final dataset for the learning algorithm based on raw data and data transformations.

4. **Modeling:** Various algorithms are selected and applied to the elaborated dataset from the previous step. In this step, so-called hyper-parameter tuning is applied to varying parameter values and achieve the most valuable result.

5. **Evaluation:** The result of the algorithm is evaluated against metrics and the objectives from the first step.

6. **Deployment:** The achievements are presented so that a customer or an implementation team can use them for further integration.

Although CRISP-DM is commonly used in DE projects in recent years [MCF+21], there are several weaknesses, most notably that the methodology has not been updated since 1999 [Sal21].

However, the strengths of CRISP-DM are, among others, the common sense, the flexibility, and the initial focus on business understanding [Sal21]. Given the weakness of being obsolete, several methods have been proposed in the literature to replace CRISP-DM. However, it is still considered the most complete methodology concerning the needs of industrial projects [MCF+21].

As discussed in [MCF+21], CRISP-DM can be readily extended to meet specific new requirements of today's industry. Additionally, CRISP-DM comprises the basic phases of DE. Thus, it will be used as a basic method structuring the methods introduced in this thesis.

2.2 Model-Driven Engineering

Model-Driven Engineering (MDE) uses models as central artifacts in software engineering, enabling the creation and execution of software systems based on models [RDS15]. The core of MDE includes the concepts *model*, *metamodel*, and *model transformation* [BCW17, RDS15].

2.2.1 Model

Models are considered machine-readable artifacts representing an abstraction of one or multiple concerns of interest of a system under study [BCW17]. According to [RDS15], a *model* is a "system that helps to define and to give answers of the system under study without the need to consider it directly". Based on [Sta73], a model is a replication of some system under study with its characteristics. First, it is a mapping and not identical with the original. Second, it is an abstraction covering only relevant characteristics of the original. And third, there is always some pragmatics behind, i.e., what the model is used for.

2.2.2 Metamodel

Metamodels define the modeling concepts and their relationships without defining any concrete representations. The intentional description of all possible models that must correspond to the associated metamodel is given. Therefore, a *metamodel* can also be described as a model of models. The *metamodel* defines the structure of a modeling language [RDS15]. From a language engineering perspective, a metamodel represents the *abstract syntax* of a modeling language [BCW17]. The *concrete syntax* of a language assigns graphical or textual elements to metamodel elements that users can understand and, possibly, edit through model editors [BCW17, RDS15].

2.2.3 Model Transformation

As models in MDE are considered machine-readable artifacts, so-called *model transformations* apply to modify existing or generated new modeling artifacts. These artifacts then are used for particular purposes, realizing the steps of the envisioned engineering process toward the partial or full generation of the software system, most commonly known as code generation in software engineering. In literature, frameworks exist to frame the intent and properties of model transformations [LAD+16]. Furthermore, classifications of existing model transformation tools are given in [KBC+19].

2.2.4 Stereotypes and Model Extension

As defined above, independent of the level of automation and the focus of the modeling language, a metamodel defines the modeling concepts, and their relationships of a modeling language. Models are instances of metamodels describing a specific system and the model characteristics must match all aspects of the associated metamodel.

To support a kind of "light-weight" extrusion of models, still conforming to the given metamodel, so-called stereotype have been introduced in Unified Modeling Language (UML) [BCW17, SSHK15]. A *stereotype* is a means of modeling to extend metaclasses by defining additional semantics to the concept represented by a metaclass [BCW17]. The use of stereotypes in modeling approaches has been proven to support the understanding and standardization of a model [KSW04]. Therefore, stereotype extensions for a specific purpose are common in practice.

2.2.5 General-Purpose and Domain-Specific Modeling Language

In the modeling community, there is controversy and ongoing debate regarding the classification of modeling languages into General-Purpose Modeling Language (GPML) and Domain Specific Modeling Language (DSML) [Fow10, MHS05, KOM+10, RDS15, SCL+21].

A GPML is characterized by more general constructs that can be used for multiple purposes [KOM+10]. Most prominent GPML languages are the Unified Modeling Language (UML) and the Systems Modeling Language (SysML), which are both providing a wide range of constructs and notations allowing to specify and document software systems (UML), and even entire systems (SysML) [RDS15]. The advantages of these modeling languages are their applicability in a broad domain and their understandability by system engineering disciplines [RDS15]. Furthermore, extendability using stereotypes allows to adapt these modeling languages with their toolset to specific domains without inhibiting the properties of the original elements [RDS15].

In contrast to the widespread application of GPMLs, DSMLs are tailored to a specific domain. Therefore, each modeling concept necessary is precisely defined towards supporting a specific concern of interest. Additionally, DSMLs are said to be more expressive, resulting in productivity gains and savings in maintenance costs [KOM+10]. Furthermore, due to the expressive power of few domain concepts, understanding, validation, and reliability is improved [HPv09, RDS15]. However, if only a subset of GPML capabilities are used then DSMLs should not create constructs again and again [BCW17].

2.2.6 MDE4AI

The term MDE intelligence or MDE4AI has been defined recently with a series of workshops [BBGW21, BKWZ21]. The focus of the MDE intelligences workshop refers to both capabilities that support modeling by leveraging AI (Artificial Intelligence for Model-Driven Engineering (AI4MDE)) and applying modeling capabilities to integrate AI concerns in the development process (MDE4AI). MDE4AI focuses on facilitating the AI development by leveraging DSMLs and GPMLs to allow domain experts to design AI artifacts themselves while providing an auditable conversion pipeline [BBGW21, BKWZ21], which contributes to the goals of the European Commission to define ethics guidelines for trustworthy AI [Hig19, Hig20].

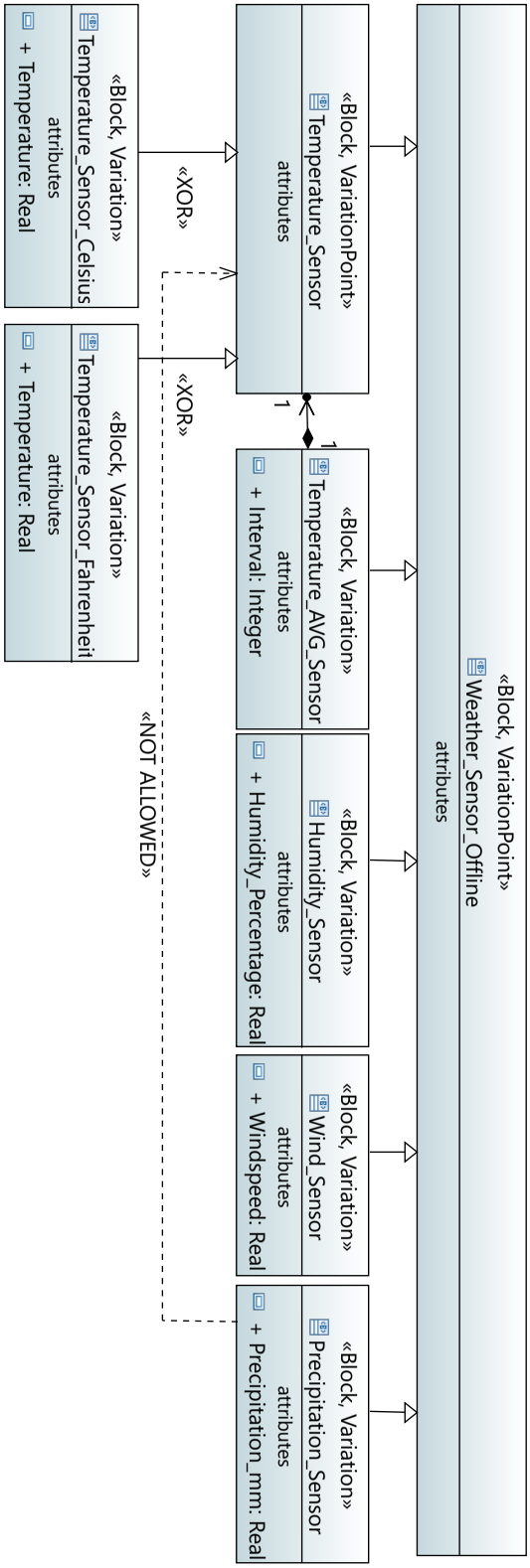


Figure 2.1: Product family sample using the Variant Modeling with SysML (VAMOS) method by [Wei14].

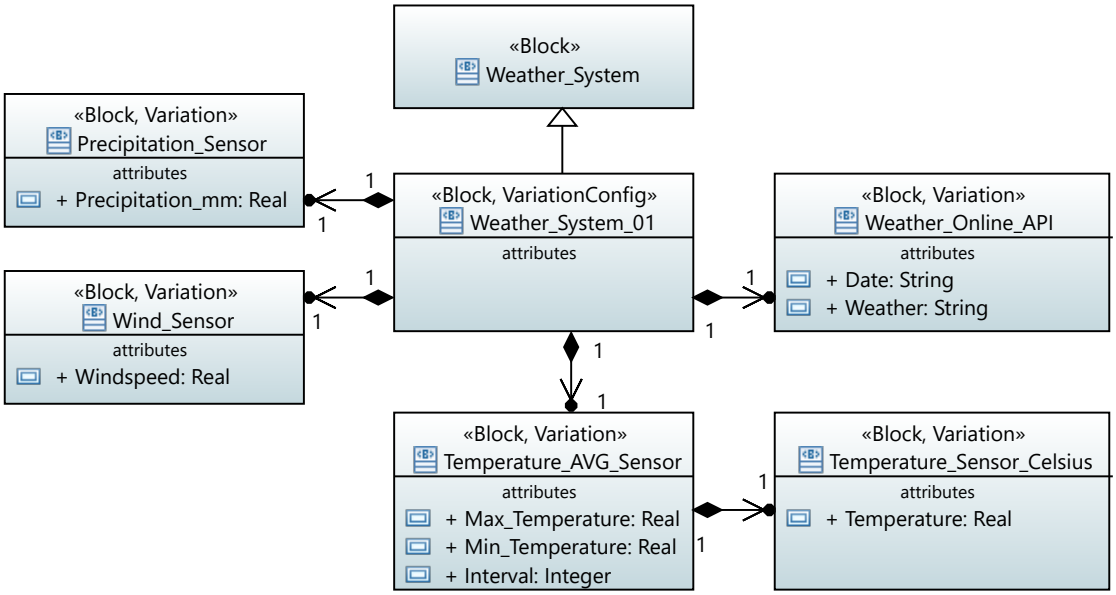


Figure 2.2: Specific configuration of a product using the VAMOS method by [Wei14].

2.3 Systems Engineering

According to INCOSE, "Systems Engineering (SE) is a transdisciplinary and integrative approach to enable the successful realization, use, and retirement of engineered systems, using systems principles and concepts, and scientific, technological, and management methods" [WI23]. Although the definition of INCOSE is often used to define SE, there are various definitions in the literature with slightly different meanings [Möl16]. SE processes are defined in the ISO-15288[1] standard and basically consist of Agreement Processes, Enterprise Processes, Technical Processes and Project Processes. Usually, the SE method itself consists of requirements analysis, functional definition, physical definition, and design validation [KSFB20].

2.3.1 Systems Engineering versus Software Engineering

SE and software engineering are distinct yet interrelated disciplines within the broader field of engineering. SE encompasses the comprehensive management and integration of entire systems, encompassing hardware, software, processes, and human elements, from inception to decommissioning. Software engineering, on the other hand, is concerned with the design, development, testing, and maintenance of software applications, with a particular focus on programming, software architecture, and software life cycle management[PAA+15].

[1] https://www.iso.org/standard/81702.html

2.3.2 Model-Based Systems Engineering

MBSE aims to integrate various engineering disciplines in product development to establish an authoritative source of truth by formalizing system requirements, behavior, structure and parametric relationships of a system using models as primary artifacts for system development [KB19]. MBSE focuses on development support for the discipline of SE, characterized as the collection of system related processes, methods, and tools [Est07]. The main difference between MBSE and conventional SE is that conventional SE focuses on storing artifacts in several documents maintained in case of changes. The main disadvantage of conventional SE approaches is the lack of synchronization and the missing authoritative source of truth or authoritative source of knowledge [KB19]. In a model-based approach, the relevant information to describe an abstract system is stored in models [MS18].

Depending on the degree of automation of model management activities, "model-based" is referred to as a lighter version of "model-driven". However, in practice the difference between the terms Model-Based Systems Engineering (MBSE) and Model-Driven Systems Engineering is blurring. Consequently, the terms are often used interchangeable as discussed in relevant communities[2].

Although MBSE became a major research field [WVMB19], the values and benefits of MBSE are mainly discussed in literature rather than observed, and measured [HS21]. Benefit claims are mainly related to a better communication and an increased traceability [HS21]. The literature concerning graphical MBSE methods promises to increase design performance while supporting the communication of relevant stakeholders of a system [HS19, HS21].

Interestingly, the integration of DE capabilities either for modeling support or the modeling support for the integration of DE capabilities in the context of engineered systems is not available yet [DFM+22, RBGM21]. The challenges to integrate MBSE in the development of DE-enabled systems are among others [RBGM21]:

Availability of data: Restricted access or insufficient amount of data.

Type of data: Necessity to label or pre-process data so that it is usable.

Integration of Processes: Implementation of a process that allows to curate data and define DE in SE

In literature, various methods exist to implement MBSE in practice [Est07, KK16, RFB12, ST18]. A core similarity of the approaches is the integration of SysML as means for modeling. Hence, SysML acts as a de facto standard for MBSE.

[2] See https://modeling-languages.com/clarifying-concepts-mbe-vs-mde-vs-mdd-vs-mda/ for a discussion.

2.3.3 Systems Modeling Language (SysML)

In MBSE, SysML is the most prominent modeling language [AZ13]. SysML is based on the UML standard with a special focus on the modeling of whole systems. The language supports the formalization of structural, behavioral and functional properties [HP13]. Structural diagrams describe the composition of systems and subsystems with their attributes and relationships [BCW17, HP13].

To illustrate the basic concepts of SysML, Figure 2.3 depicts a sample application of core elements of a Block Definition Diagram (BDD) realized in the Eclipse-based open-source software Papyrus[3]. Particularly, the illustration with a sample is chosen here, as it is required in subsequent chapters to follow the elaborated methods closely. On top of Figure 2.3, a Block with the name *Mammal* is defined, consisting of one attribute of type *String* with the attribute name *Mother* and the visibility *public* indicated by the plus (+). Other elements of a block are, e.g., operations, ports, etc. However, these elements are not shown, since they are irrelevant for this work. The name of the *Mammal*-Block is written in italics because it is an abstract class that cannot be instantiated without further derivation. Underneath the *Mammal*-Block, two inheriting elements are defined using white arrows connecting the blocks. The attribute *Mother* is inherited from the parent block, where the arrow head points to. The *Dolphin* child has an additional property *Age*, which only affects this block as long as no further inheritance is modeled in another view. The second block *Human* consists of a subsystem, indicated by the black diamond being a part association (a.k.a. composition). A part association determines that a block describes the whole element and a part of the whole element is additionally described in another element. The 1 and the 0..2 indicate the multiplicity, allowing to define the number of elements. This sample describes that one element *Human* consists of zero, one or two legs, and that each leg is connected to only one *Human*. The white diamond between *Leg* and *Shoe* indicates a shared association, which is a weaker form of the part association. It refers to a relationship where the part element is still valid if the whole element is deleted, e.g., if the element *Leg* is not valid anymore, the *Shoe* is still valid. The multiplicity * indicates that a *Leg* can have arbitrary number of shoes.

In SysML, the execution of multiple activities is modeled using state diagrams. A state diagram requires an entry point and an exit point. Figure 2.4 illustrates a sample state diagram. The arrow between the states indicates a transition and describes that one state has been completed and another is active. Behind a state, the execution of one or multiple activities can be triggered, whereas an activity is a sequential execution of single actions [OMG19].

2.3.4 Variant Modeling

In the SE/MBSE process of developing complex systems, different product lines can be developed. Product lines consist of product families, whereby a family consist of variation that are composed to a system variation. Within a family, various components

[3] https://www.eclipse.org/papyrus/index.php

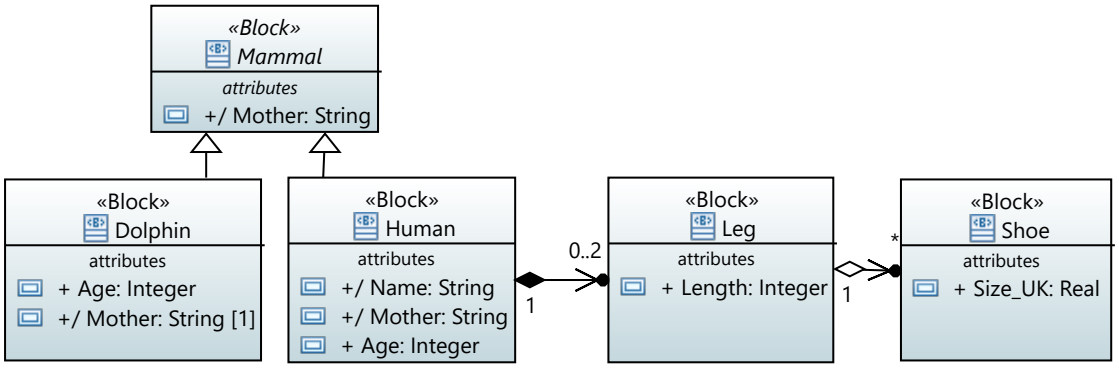


Figure 2.3: Block Definition Diagram sample.

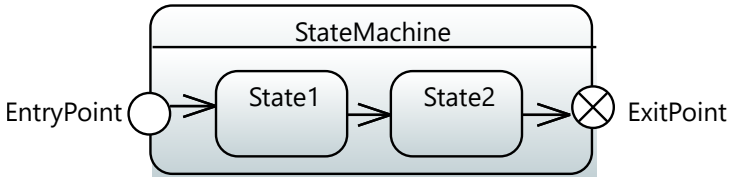


Figure 2.4: State Diagram sample.

are connected so that a specific system is configured. Each system configuration produces various outputs in terms of format, interface or composition.

In literature and practice, different approaches to managing the diversity of product lines during product development are available [EW22, Wei14]. One approach that supports the modeling of variants in the MBSE environment is the VAMOS method [Wei14].

The VAMOS method provides a metamodel to reflect the composition and specification of single subsystems of a complex system model. The approach is mainly based on SysML features like blocks, ports and connections, used to represent the structural differences but also the differences in the components of a system.

Figure 2.1 illustrates an example of a variant model based on [Wei14]. In the figure, the definition of a product family is shown, based on the use case in Section 5.2.1. The product family is indicated by a *VariationPoint* stereotype. Each of the potential variations of the system is indicated by the stereotype *Variation*. As a result of the definition, all possible variants of a product are defined.

A specific configuration of a system is modeled using the *VariationConfig* stereotype, as shown in Figure 2.2. The variation configuration is inherited from the basic system *Weather_System* and the configuration composes a specific subset of *Variations*.

2.3.5 Data-Driven Engineering

Using the result of a DE process to improve product development through decision support is defined by various terms in literature, such as Data-Driven Decision Making [PF13, TSOO+20], and Data-Driven Engineering Design [FZZ+20, LC17, VKPV22]. A similar definition to Data-Driven Engineering Design is found in the literature as Data-Driven Engineering (DDE) and is defined as follows [TSOO+20]:

"Data-Driven Engineering is a framework for technical product development in which the use case-oriented collection and utilization of sufficiently connected product lifecycle data guides and drives decisions and applications in the product development process."

Technical product development processes can be read as a synonym for Systems Engineering (SE) [SMM+19]. Additionally, "use case-oriented collection and utilization of sufficiently connected product lifecycle data" can be summarized under the term DE. Therefore, DDE is the application of DE for SE to enable the application of any algorithm that "guides and drive decisions", which is basically a kind of ML algorithm, respectively.

Consequently, the definition of DDE can be refined as follows:

"Data-Driven Engineering is a framework that enables Data Engineering in Systems Engineering to guide and drive decisions and applications in Systems Engineering processes."

With respect to this, in this thesis, DDE refers to the application of DE and any kind of algorithm that guides and drives decisions in SE.

2.4 Lean Six Sigma Methods

Six Sigma is a set of tools and methods allowing to structure the development and delivering of defect-free products and services by improving a product's value and quality in the design stage [SN12, YE09]. Complementary, lean product development aims to improve product development efficiency and effectiveness by decreasing product development lead time [MLC11, YE09]. Lean Six Sigma combines the advantages of Six Sigma and Lean Management [MJSR13]. Within the context of this thesis, the toolset of Lean Six Sigma is taken into concern since the concepts of Six Sigma and Lean Management have been perceived beneficial for SE and MBSE, respectively [Mul13, ST15, YE09]. Consequently, the methods used from Lean Six Sigma are presented below [Tag10, YE09].

2.4.1 Supplier-Input-Process-Output-Customer

The techniques and tools used in Six Sigma either aim to improve an existing situation or to design a new system by applying Six Sigma methods called Design For Six Sigma (DFSS) [YE09]. As a part of the DFSS, the Supplier-Input-Process-Output-Customer (SIPOC) method is used in the initial definition phase to summarize inputs and outputs

Table 2.1: Supplier-Input-Process-Output-Customer sample using table representation.

Supplier	**I**nput	**P**rocess	**O**utput	**C**ustomer
• Information Supplier 1 • Information Supplier 2	• Input A • Data B	• Task A • Task B	Output AB	Information Receiver

related to one or more processes in a comprehensive form. Table 2.1 depicts a generic template of a SIPOC.

A SIPOC consists of an information supplier (S) that provides certain input data (I) to a process (P) that produces an output (O) for a customer (C). As a rule of thumb, a comprehensive SIPOC constitutes four to five key steps of a process[4]. The main purpose of a SIPOC is to clearly delineate the processes in a process chain and capture an overall process with the actors involved. In literature, approaches exist leveraging the Enterprise Architecture (EA) method to formalize a SIPOC using model-based techniques [RSS22].

2.4.2 Product Development Value Stream Mapping

The Lean Management method VSM aims to identify value streams lacking efficiency and to increase productivity [HR97]. Although the method is typically associated with manufacturing, it can be applied to sources of waste on an information level in engineering [McM05]. In [McM05] the types of waste adapted from VSM to Product Development Value Stream Mapping (pdVSM), which allows to assess waste in information and knowledge. Table 2.2 depicts sources of information waste based on [McM05].

2.4.3 Waste Failure Mode Effect Analysis

To more qualitatively assess the causes and effects of errors, e.g., information waste from the VSM method, the application of Failure Mode and Effect Analysis (FMEA) is proposed in literature [Sta03]. Particularly, in product development, the application of Design Process Failure Mode and Effect Analysis (DP-FMEA) is recommended [CI06]. With respect to the reduction of information waste, Waste Failure Mode and Effect Analysis (W-FMEA) is proposed in the literature, taking VSM into concern [dC14].

To allow the interpretation of an FMEA, the impact of the waste requires to be assessed and prioritized with respect to Occurrence (O), Severity (S) and Detection (D) [Sta03]. Each waste is rated on a scale of one to ten with respect to the three dimensions mentioned above. The resulting ranking are multiplied, with the highest number representing the waste with the most impact that must be addressed first. A guideline for the ranking of the individual dimensions can be taken from the literature [dC14] or defined by the user. For further guideline on conducting FMEA, please refer to literature [CI06, Sta03, dC14].

[4] https://www.isixsigma.com/tools-templates/sipoc-copis/sipoc-diagram/

Table 2.2: Value-Stream Mapping (VSM) dimensions of information waste based on [McM05].

Type	Explanation
Waiting	Late delivery of information; Delivery too early -> leads to rework
Inventory	Lack of control; Too much in information; Complicated retrieval; Outdated, obsolete information
Over-Processing	Unnecessary serial processing; Excessive/custom formatting; Too many iterations
Over-Production	Creation of unnecessary data and information; Information over-dissemination
Transportation	Information incompatibility; Software incompatibility; Communications failure
Unnecessary Movement	Lack of direct access; "Walking" the process
Defective Products	Haste; Lack of reviews, tests, verifications; Lack of interpretation (raw data delivered when information or knowledge needed)

2.5 Enterprise Architecture

This section introduces the concepts of the EA method on a level required as preliminary to understand the here presented method. For a comprehensive introduction to EA, please refer to [BNS03, GP11, Lan09]. Furthermore, in this section, an introduction on the EA To-Be architecture modeling state of the art and methods is given.

2.5.1 Enterprise Architecture Method

The EA principles describe business processes aligned with IT artifacts to enable a more holistic view of an enterprise. The holistic view and the principles of EA support the development of a corporate strategy using the enterprise architecture initiative [GP11, Lan09]. Over the years, several supporting methods and standards emerged, with the Zachman Framework [Zac87] and the TOGAF [TOGAF18] as representative samples [EW22]. The ArchiMate[5] software implements the ArchiMate graphical modeling language, which complements the TOGAF framework [Archi23, Archi22]. ArchiMate is proven beneficial for integration and interoperability of business process with related application and technology layer [IJ07, WC18]. Matching business processes with associated IT artifacts is one of the core differences with other model-driven architecture methods, such as Business Process Model and Notation (BPMN) [WM08].

Figure 2.5 depicts the basic layers of the EA modeling using ArchiMate with representative relationships and used modeling elements. Each layer (e.g., business layer, application layer, etc.) is connected to the underlying layer using links, e.g., by realization links.

[5] https://www.archimatetool.com/

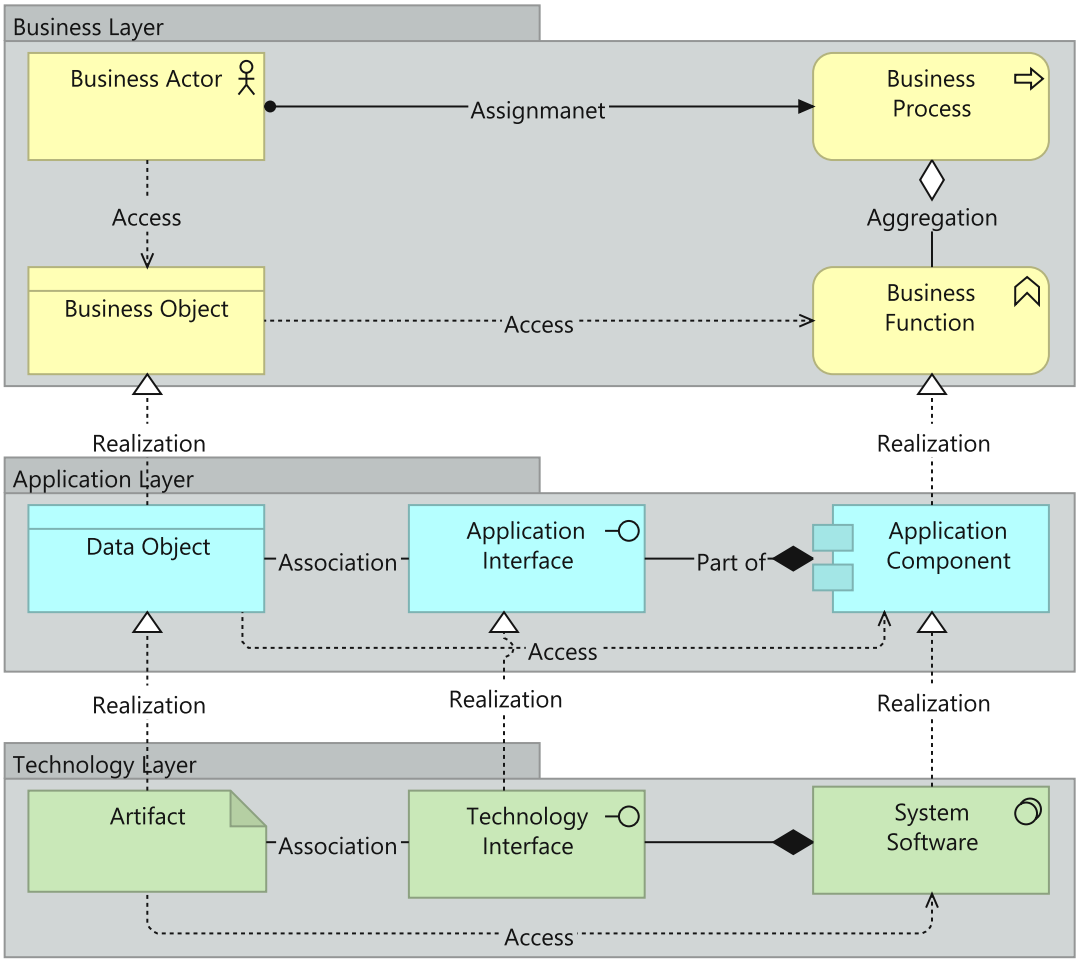


Figure 2.5: The EA model with business process (yellow), related applications (blue) and the artifacts and technologies (green).

To exploit the entire potential of the modeling method, more elements of each layer requires to be modeled, e.g., *Application Interaction* in the application layer. However, only elements relevant to this thesis are introduced in the model to limit the scope, complexity and effort of the modeling. For further details on the modeling of EA, please refer to [Lan09] and [GP11].

2.5.2 Enterprise Architecture To-Be Architecture Modeling

An enterprise's architecture evolves over time to adapt to the market or implement new technologies [MP03]. The enterprise transformation from actual to target architecture is referred to as Assess-Aim-Act approach [GP11]. Using EA, current (As-Is) architecture, target (To-Be) architecture and migration plan from current to target architecture of a

company can be represented [ST06]. The To-Be architecture is also known as desired, future, or target architecture [RMNN13]. In the modeling of EA, there is no visual or standardized distinction between As-Is and To-Be architecture [BRJ17]. For this reason, the method of [BRJ17] introduces a color code aligned with text annotations to identify changed architecture.

However, the modeling of To-Be architecture is still time-consuming and error-prone due to the required knowledge of the As-Is architecture and the prescriptive nature of inventing new processes. Therefore, [NAR+17] introduced a checklist to support the quality assurance for EA To-Be modeling. The checklist consists of various questions on a coarse-grained granularity, e.g., "Is the desired business approach clearly described?" [NAR+17]. Based on the checklist, the readiness for the implementation of a To-Be architecture is theoretically assessed.

CHAPTER 3

State of the Art - MDE4AI

This section aims to present the actual state of the art on Model-Driven Engineering (MDE) approaches aiming to support the implementation of Artificial Intelligence (AI) capabilities. With respect to this, recently, a series of AI and MDE workshops was initiated, focusing on using MDE techniques for defining AI methods (MDE4AI) and AI support for MDE (AI4MDE) [BBGW21, BCWZ22, BKWZ21]. The adoption of MDE practices to support AI capabilities of the system under study promises to support the development through degrees of automation of the engineering activities, e.g., code generation, and, therefore to increase the number of industrial applications. Still, to the best of our knowledge, the current state of practice and state of the art is not fully elaborated regarding MDE approaches supporting the implementation of AI capabilities.

In this respect, in the following, an SLR is conducted to achieve the RG defined in Table 3.1. The RG definition is aligned to the Goal-Question-Metric approach [BCR94].

Table 3.1: The overall research goal of the conducted SLR.

Purpose	Collection and comparison of studies on
Issue	model-driven approaches that explicitly address the engineering of
Object	artificial intelligence applications
Viewpoint	from the point of view of researchers.

According to the guidelines set out by [KB13], an SLR is conducted to gather and assess existing literature to address the identified research questions [PVK15]. Particularly, this work focuses on the state of the art for MDE approaches that enable the formalization of AI use cases.

A selection of text, figures and tables within this chapter is based on the publications in box „Publications 2: State of the Art“ (It should be noted that an expanded and improved version of this chapter has been published in the Software and Systems Modeling (SOSYM) journal):

Publications 2: State of the Art

[RBW+23] S. Rädler, L. Berardinelli, K. Winter, A. Rahimi, and S. Rinderle-Ma, "Model-Driven Engineering for Artificial Intelligence – A Systematic Literature Review," Jul. 2023, doi: 10.48550/arXiv.2307.04599.

[RBW+24] Simon Raedler, Luca Berardinelli, Karolin Winter, Abbas Rahimi, and Stefanie Rinderle-Ma. Bridging MDE and AI: A systematic review of domain-specific languages and model-driven practices in AI software systems engineering. Software and Systems Modeling, Sept. 2024. doi: 10.1007/s10270-024-01211-y.

The remainder of this section is organized as follows. Section 3.1 introduces the research method, i.e., the paper search and selection process. Section 3.2 presents the approaches aligned with the data extraction strategy of the SLR protocol in Section 3.1. Section 3.3 answers the RQs, discusses the key findings, and depicts implications and future research. Section 3.4 assesses the quality and limitations of the current SLR using threats to validity analysis. Finally, Section 3.5 summarizes the findings of the SLR.

3.1 Research Method

This section introduces the SLR method applied in this work. The SLR study protocol is based on the guidelines by [KB13, KC07, PVK15], introducing the main steps of SLRs to be performed in the Software Engineering domain.

Figure 3.1 depicts an activity-like diagram of the implemented search and selection process protocol workflow. The workflow consists of the following steps:

1. *Identifying the Research Goals and the Research Questions* (RG/RQ): The objective of this work and the research questions are defined to guide the SLR (Section 3.1.1 shows the result of the RG/RQ elaboration)
2. *Search Process*: The literature search is conducted on selected databases collecting scientific publications via the execution of queries based on a search string suitably designed according to the given RGs and RQs (Section 3.1.2).
3. *Study Selection*: The authors define the Inclusion Criteria (IC) and Exclusion Criteria (EC) and apply them to the papers collected in the databases by reading their titles and abstracts. Subsequently, the selected papers are evaluated based on their content (Section 3.1.3).
4. *Data extraction*: Given a set of selected studies that passed the IC and EC application, detailed data are extracted throughout a full-text reading. In the SLR, papers' detailed information is collected in evaluation tables. If a publication is relevant, snowballing is applied to add referenced papers or the one citing the selected publication (see Section 3.1.4).

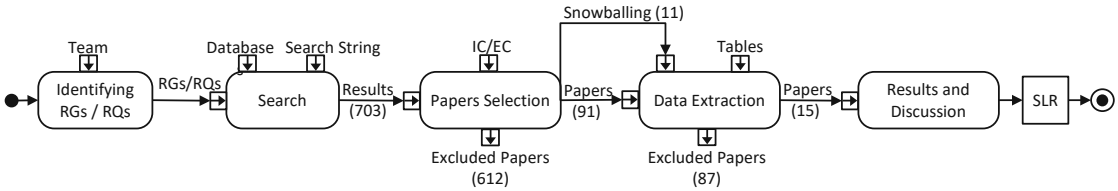


Figure 3.1: SLR method overview.

5. *Results Analysis and Discussion*: Collected results are analyzed, and a discussion occurs among the authors to answer the stated RQs.

The execution of the protocol is documented in a spreadsheet, and bibliographic entries are collected in Zotero Library, published online[1].

3.1.1 Research Questions

The overarching RQ based on the defined RG is the following:

RQ1 What is the current state of the art of Model-Driven Engineering with extensions to formalize Artificial Intelligence methods and applications?

To address these research question RQ1, various refined RQs are defined as follows:

RQ1.1 What Model-Driven Engineering aspects are addressed in the approaches, e.g., abstract syntax (metamodel), concrete syntax etc.?

This RQ aims to assess the pillar concepts of MDE languages concerning comprehensiveness (of modeling) and applicability (maturity).

RQ1.2 Which phases of Artificial Intelligence development aligned with the CRISP-DM methodology are covered by the approaches?

This RQ assesses the extent to which the development phases of CRISP-DM are covered. As a result, implications can be made about the extent of support.

RQ1.3 Which industrial domains are supported by MDE4AI approaches?

This RQ enables finding industries that are using MDE in the context of AI and thus driving the development of MDE4AI using domain-specific tools and methodologies towards the needs of the specific industry.

RQ1.4 What are the used methods and the supporting Model-Driven Engineering tools the proposed approaches rely on?

[1] https://github.com/sraedler/Model-Driven-Engineering4Artificial-Intelligence

This RQ allows assessing the underlying methods and the related tool support, including further development leveraging these underlying technologies to gain maturity.

RQ1.5 To what extent is communication between different stakeholders supported by Model-Driven Engineering?

Communication and business knowledge elaboration are two of the core pitfalls in the development of AI solutions [PPW+21]. Therefore, this question aims to assess the contribution to support fostering AI in the industry.

RQ1.6 Which challenges and research directions are still open?

This RQ will lead to future research directions and challenges for the model-based engineering of AI applications due to a collection of limitations in the proposed approaches based on respective authors or our obtainment.

3.1.2 Search Process

This section describes the search activity in Figure 3.1. According to [KB13], defined search queries are executed on dedicated search engines. In this research, the queries are performed on the following bibliographic sources:

- ACM Digital Library: http://dl.acm.org/
- dblp Computer Science Bibliography: https://dblp.org/
- IEEE Xplore Digital Library: http://ieeexplore.ieee.org
- Google Scholar: https://scholar.google.com
- Springer: https://link.springer.com

To select suitable terms for the search, keywords from known studies, the MDE4AI workshop series[2] [BBGW21, BKWZ21] and the International Journal on Software and Systems Modeling (SoSyM)[3] were selected.

The selected keywords for the search terms are the following:

$S_1(MDE) = \{MDE; Model-Driven\ Engineering; DSL; DomainSpecific\ Language; Metamodeling; Domain\ Modeling\}$
$S_2(AI) = \{AI; Artificial\ Intelligence; ML; Machine\ Learning; Deep\ Learning; Intelligence\}$

Each keyword k_i from the set S_1 and S_2 has been combined in conjunctive logic proposition $p \in P$.

[2] https://mde-intelligence.github.io/
[3] https://www.sosym.org/

Table 3.2: IC and EC.

Type	ID	Type
IC	1	We include system-level DSML (metamodel) with AI extensions
	2	We include data-driven/model-driven approaches with AI extensions
EC	1	We exclude simulation-based (only) approaches
	2	We exclude algorithm-based (only) approaches
	3	We exclude secondary studies
	4	We exclude review papers, but include them in snowballing
	5	We exclude study available only in form of abstract
	6	We exclude study not in English language
	7	We exclude papers with focus on software architecture for MDE4AI, e.g., Hadoop integration in infrastructure
	8	We exclude vision only papers and proposals

$$P = \{p | p = s_i \in S_1 \wedge s_j \in S_2\}$$

$$i = 1, 2, 3, 4, 5, 6,\ j = 1, 2, 3, 4, 5, 6$$

The resulting set P of 36 propositions (p_i) includes the final *search strings*. According to [KB13], the propositions (p_i) should be combined as OR statements. However, for some search engines, a single search term is too complicated, as some search engines limit the length of the search term or do not generate results correctly due to nested search terms. Therefore, each search string is executed as a single query.

The automated search was executed in November 2022. In total, 703 papers have been collected. The search terms and results are archived and are online available[4]. If a result file is unavailable, the search query on the specific search engine did not retrieve any results.

3.1.3 Paper Selection

The IC and EC as outlined in Tab. 3.2 are employed for the paper selection. The IC and EC have been evaluated for each paper collected by queries executed on the selected databases by reading its title and abstract. Additionally, doctoral theses are excluded due to the extensiveness, but the referenced publications of the author are included in snowballing. Although review papers are not considered for the survey, we presented relevant related work in the original publication [RBW+23, RBW+24].

Following the IC and EC application, a full-paper read is applied to select the final papers. Additionally, snowballing is accomplished as suggested by [KB13] to retrieve further results. The relevant papers from the list of snowballing papers were selected

[4] https://github.com/sraedler/Model-Driven-Engineering4Artificial-Intelligence

Table 3.3: List of selected publications with type of publication incl. snowballing results.

Item Type	Year	Author	Title
Conference Paper	2020	[Al-20]	Model Driven Approach for Neural Networks
Conference Paper	2019	[BBK+19]	STRATUM: A BigData-as-a-Service for Lifecycle Management of IOT Analytics Applications
Journal Article	2020	[VGZS20]	Lavoisier: A DSML for increasing the level of abstraction of data selection and formatting in data mining
Journal Article	2022	[GGC22]	A domain-specific language for describing machine learning dataset
Conference Paper	2017	[HMFl17]	The next Evolution of MDE: A Seamless Integration of Machine Learning into Domain Modeling
Conference Paper	2019	[HMS+19]	Meta-Modelling Meta-Learning
Conference Paper	2019	[HMR+19]	Model-based design for CPS with learning-enabled components
Conference Paper	2019	[KMS19]	Realization of a Machine Learning Domain Specific Modeling Language: A Baseball Analytics Case Study
Conference Paper	2019	[KPRS19]	On the Engineering of AI-Powered Systems
Journal Article	2021	[MPN21]	AdaptiveSystems: An Integrated Framework for Adaptive Systems Design and Development Using MPS JetBrains Domain-Specific Modeling Environment
Conference Paper	2022	[MRC+22]	A Model-Driven Approach for Systematic Reproducibility and Replicability of Data Science Projects
Journal Article	2021	[MCBG22]	A MDE Approach to Machine Learning and Software Modeling
Journal Article	2022	[MCC22]	Towards a DSML for AI Engineering Process Modeling
Conference Paper	2021	[RGJ21]	An MDE Method for Improving Deep Learning Dataset Requirements Engineering using Alloy and UML
Conference Paper	2020	[Zd20]	Arbiter: A Domain-Specific Language for Ethical Machine Learning

with the same procedure as the query results. Table 3.3 lists the final list of selected papers. Particularly, 11 papers are added by query selection, and four are added due to snowballing.

3.1.4 Data Extraction

Each selected paper presented in Table 3.3 underwent a data extraction process following the data extraction template in Table 3.4. Additionally, the publication type is assessed as Exploratory (without evaluation, e.g., a pure concept or vision) or Technical (with evaluation).

Table 3.4: Data extraction template.

RQ	**Concern**		**Assessment Description**
RQ1.1	MDE	Metamodels	The metamodel of the approach is either depicted as a diagram in a referenced repository or clearly mentioned and textually described.
		Concrete Syntax	The concrete syntax is given in figures, listings, or tables to illustrate an implementation/use case excerpt or it is indicated whether textual or graphical modeling is applied for a specific aspect.
		Arbitrary Constraints	The approach or the underlying modeling framework (e.g., SysML) allows the specification of arbitrary constraints.
		Model Transformation	The approach uses or introduces model transformation to generate engineering artifacts of any kind.
RQ1.2, RQ1.5	AI	Business Understanding	The model contributes to the understanding of the underlying business. Particularly, the creation of the data and aspects from other disciplines are introduced, such as requirements modeling for AI.
		Data Understanding	The model supports at least two of the following aspects: data description, data attribute relationship, data background, data quality, and data composition.
		Data Ingestion	The model clearly depicts the origin of data and how to load it.
		Feature Preparation	The model allows an understanding of how data needs to be transformed, connected, or preprocessed.
		Model Training	The model depicts the used algorithm with input and output values and potential hyperparameters.
		Metrics/Evaluation	The model depicts metrics for the AI approach or introduces evaluation criteria.
RQ1.3	Others	Problem Domain	The domain of the case study or the mentioned area of application.
RQ1.4		Frameworks	The method and tools used in the approach, e.g., WebGME, Xtext, Xtend, etc.

The extracted data mainly address two concerns of interest, i.e., MDE and AI. Modeling concerns refer to the evidence of sound knowledge and application of *model foundations* [BCF+19] (e.g., abstract syntax/grammar/metamodel, textual/graphical concrete syntax, constraints, model transformations) and supporting tools (e.g., modeling language frameworks). AI concerns [ASX+20] indicate to which extent the publications support ML modeling aligned with the dimensions of the CRISP-DM methodology [WH00]. It should be noted that the assessment dimensions do not correspond exactly to the phases of CRISP-DM to allow for a more detailed categorization of concerns; e.g., in CRISP-DM, *Data Ingestion* is part of the *Data Understanding* phase but separated in the given assessment. An aspect of a concern of interest is assessed as *available* (✓) if the aspect is presented in the approach, or aa *underlying principle* is typically offered by the environment (e.g., constraint modeling might not be presented but is typically offered by the underlying MDE tooling). Finally, it is worth noting that there is no evaluation of the deployment phase of CRISP-DM as it is beyond the scope of this work.

3.2 Literature Assessment

The result obtained from the data extraction process described in the previous section is presented in Tables 3.5, 3.7 and 3.8.

In [Al-20], Al-Azzoni proposes a model-driven approach to describe ML problems addressed by artificial neural networks. The approach enables the description of datasets as well as the consuming Multi-Layer Perception (MLP) Neuronal Networks (NN). With templates and code generators, executable Java programs can be generated. The approach is validated using the Pima Indians Diabetes dataset.

In [BBK+19], Bhattacharjee et al. introduce STRATUM, a model-driven tool that enables dealing with the lifecycle of intelligent component development. The platform addresses design-related concerns such as modeling the ML algorithm pipeline, accessing data streams, allocating and properly sizing cloud-based execution platforms, and monitoring the overall system's quality of service. The primary goal of this work is to support deploying and maintaining various cloud-based execution platforms. The MDE part of this work is minor and less detailed.

In [VGZS20], De La Vega et al. introduce a DSML that describes datasets to select sufficient data on a high level. The approach uses a SQL-like textual language to select, combine and filter various data on an attribute level. The approach aims to increase a dataset's abstraction level to reduce complexity and make using data mining technologies easier.

In [GGC22], the DescribeML DSML is proposed to define ML datasets. From a DescribeML model, a template with basic information is automatically generated, based on a given dataset. The provided DSML allows the definition of metadata, data attributes with statistical features and provenance, and social concerns.

Table 3.5: Result of the data extraction for the MDE and AI concerns.

RQ	Concern	Criteria \ Paper	[Al-20]	[BBK+19]	[VGZS20]	[GGC22]	[HMF17]	[HMS+19]	[HMR+19]	[KMS19]	[KPRS19]	[MPN21]	[MRC+22]	[MCBG22]	[MCC22]	[RGJ21]	[Zd20]	Sum
	General	**T**echnical or **E**xploratory Paper	**T**	**T**	**T**	**T**	**T**	**E**	**T**	**E**	**T**	**T**	**T**	**T**	**E**	**T**	**T**	12 **T** 3 **E**
RQ1.1	MDE	Metamodels	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		14
		Concrete Syntax	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	14
		Arbitrary Constraints	✓	✓					✓	✓			✓			✓	✓	7
		Model Transformation	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓		✓		12
RQ1.2, **RQ1.5**	AI	Business Understanding							✓						✓	✓	✓	4
		Data Understanding			✓	✓	✓	✓		✓			✓			✓		7
		Data Ingestion	✓	✓	✓				✓		✓	✓	✓	✓	✓		✓	10
		Feature Preparation	✓	✓									✓	✓	✓			5
		Model Training	✓	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓			11
		Metrics/ Evaluation		✓					✓		✓			✓				4

This approach aims to improve the understanding of datasets and thus support the replicability of AI projects. Currently, this work is limited to the dataset description. Future work aims to describe AI models and other elements of an AI pipeline.

In [HMFl17], Hartmann et al. present an approach based on so-called micro-learning units at a language definition level. This work proposes to weave the learning units into domain modeling, due to the high entanglement of learning units and domain knowledge. For this purpose, the approach allows the definition of DSMLs with learned attributes (i.e., what should be learned), how (i.e., algorithm and parameters), and from what (i.e., other attributes and relationships).

Hartmann et al. leverage the previous study for meta-learning in [HMS+19]. This study proposes two generic metamodels for modeling i) ML algorithms and ii) meta-ML algorithms (i.e., algorithms to learn ML ones).

In [HMR+19], a comprehensive modeling environment for learning-enabled components in CPS development is introduced. The approach supports training, data collection, evaluation, and verification. It integrates Goal Structuring Notation (GSN) to support assurance and safety cases. The publication is, among others, part of a research project[5] facilitating MDE.

In [KMS19], a DSML is introduced with the goal of proving the plausibility of using MDE approaches to create ML software. The DSML, conceptually sketched by another research group in [Bre14], is realized and applied to a case study in the sports domain. The approach integrates model transformation to generate executable code.

In [KPRS19], an approach describing deep learning using MDE is presented. The approach combines two DSMLs, namely MontiAnna, and EmbeddedMontiArc. The former is a textual modeling framework for designing and training Artificial Neural Networkss (ANNs). It also embeds another DSML, MontiAnnaTrain, for describing the training procedure. The latter, EmbeddedMontiArc, is an architectural description language. It supports the definition of components and connectors, with a particular focus on embedded, automotive, and cyber-physical systems. The frameworks are intended to define deep artificial neural networks, e.g., convolutional neural networks, for processing traffic images to learn how to drive a car in a simulator.

In [MPN21], Meacham et al. propose a set of DSMLs and toolset implemented on top of the Meta Programming System (MPS) language workbench for the design and development of adaptive systems offering MAPE-K and AI in context capabilities. The approach describes an extension and composition of DSMLs that are extended with application-specific concepts.

In [MRC+22], Melchor et al. propose an MDE approach to formalizing ML projects and the associated infrastructure in which the resulting tool will be deployed. The approach aims to increase the reproducibility and replicability of data science projects. Hence a key feature of the approach is to describe processes and datasets in detail.

[5] https://modelbasedassurance.org/

In [MCBG22], Moin et al. present an MDE approach based on ThingML[6] to support the development of IOT devices with the extension of data analytics and ML. The ThingML framework supports defining software parts and components using UML. The communication between the components (things) is defined using ports, messages, and state machines. The approach supports the transformation of the model into executable code.

In [MCC22], Morales et al. provide a DSML to model AI-related processes using Eclipse-based technologies. The approach aims to describe AI processes within an organization and thus contribute to the structured designing, enacting, and automating of AI engineering processes.

A MDE approach for defining dataset requirements is introduced in [RGJ21]. It focuses on the structural definition of requirements using semi-formal modeling techniques.

In [Zd20], Zucker et al. present a very preliminary version of a declarative DSML for ethical AI addressing transparency, fairness, accountability, and reproducibility concerns of ethical machine-learning datasets. The approach describes datasets in a SQL-fashioned language and provides a notation to record how ML models will be trained.

3.2.1 MDE Concerns

In this section, we report the contributions of the selected studies with respect to MDE techniques and practices [BCW17, BCF+19], i.e., metamodels/grammars, graphical/textual concrete syntax, constraints, and model transformations. In particular, we consider whether the proposed approaches leverage language workbenches [ICM+20] to create DSMLs adopted in the presented approaches.

Abstract Syntax

In 14 out of 15 approaches, the abstract syntax of one or more DSMLs is defined by metamodels or grammars. The only exception is [Zd20], where the authors explicitly remark that the proposed DSML is a preliminary ad-hoc implementation for the proposed case study and does not provide any grammar or metamodel specifications.

In the following, we classify the selected studies based on the technologies used to specify the abstract syntax of DSMLs used in the proposed approaches [ICM+20]. A large majority of the selected studies, i.e., eleven, adopt a metamodel-centric language design [Al-20, BBK+19, VGZS20, HMFl17, HMS+19, HMR+19, KMS19, MRC+22, MCBG22, MCC22, RGJ21] by leveraging Eclipse Modeling Framework (EMF) and WebGME language frameworks, two adopts a grammar-centric approach [GGC22, KPRS19] by leveraging Langium and Monticore language workbenches, and one a projectional [MPN21] approach, based on JetBrains MPS.

[6] https://github.com/TelluIoT/ThingML

EMF-based In eight studies, the metamodel is based on EMF [Al-20, VGZS20, GGC22, KMS19, MRC+22, MCBG22, MCC22, RGJ21]. Several EMF metamodels focus on the description of datasets [Al-20, VGZS20, GGC22, RGJ21]. Other studies additionally describe algorithms [Al-20, KMS19, MRC+22] or even further steps of the implementation [MCBG22, MCC22]. In [KMS19], the conceptual metamodel presented as an entity-relationship diagram in [Bre14] is realized as a UML profile, i.e., a lightweight extension of the UML metamodel in Papyrus UML, which leverages EMF.

KMF/Greycat-based Two studies [HMFl17, HMS+19] of the same research group are based on the Kevoree Modeling Framework (KMF) and its successor GreyCat, which results from a research project to create an alternative to the EMF based on Ecore. In [HMFl17], the capabilities of the Greycat metalanguage are presented. In particular, it allows the definition of microlearning units by explicitly declaring *learned attributes* as part of the domain-specific metamodels. In [HMS+19], two metamodels for ML and meta-learning are proposed. The former contains definitions for datasets, metadata, and learning algorithm with hyper-parameters.

WebGME-based Two studies [BBK+19, HMR+19] define metamodels using the WebGME metamodeling framework. While UML and profiles cannot provide the language engineering support typically offered by language workbenches, WebGME allows specifying DSMLs creating a class diagram-based metamodel from which the DSML infrastructure is automatically generated. In [BBK+19], the so-called Stratum approach for BigData-as-a-Service provides a DSML consisting of several metamodels built on top of WebGME (metamodel for ML algorithms, metamodel for data ingestion frameworks, metamodel for data analytics applications, metamodels for heterogeneous resources). In [HMR+19], the metamodel is based on existing metamodel libraries: SEAM, DeepForge, and ROSMOD.

Langium-based In [GGC22], the DescribeML DSML is the only work leveraging the recent Langium open-source language workbench enabling domain-specific languages in VS Code, Eclipse Theia, and web applications, leveraging the Language Service Protocol (LSP)[7]. In [GGC22], three metamodels are described i) metadata model, ii) composition model, and iii) provenance and social concerns model. Such metamodels are then implemented as grammars[8].

MontiCore-based In [KPRS19], all DSMLs, i.e., MontiAnna, MontiAnnaTrain, and EmbeddedMontiArc, are all defined using the MontiCore language workbench [RH18]. One of the main benefit is the reuse of existing C++ code generators for neural network frameworks (MxNet, Caffe2, and Tensorflow).

[7] https://microsoft.github.io/language-server-protocol/specifications/lsp/3.17/specification/

[8] Based on Chevrotain, https://chevrotain.io/docs/.

MPS-based In [MPN21], five different DSMLs are created with JetBrains MPS, an open-source projectional language workbench that allows direct changes to the abstract syntax tree through an editor, without the need for a grammar or parser. [MPN21] leverages MPS' language extension and composition capabilities to deal with domain-independent (e.g., using the AdaptiveSystems DSML to structure the system according to MAPE-K loop by IBM) and domain-specific concerns (e.g., AdaptiveVLE to model concerns of virtual learning environments).

Concrete Syntax

This section assesses the proposed approaches' notations or *concrete syntax*. A concrete syntax is explicitly mentioned by 13 out of 15 approaches.

Seven studies [VGZS20, GGC22, HMFl17, HMS+19, MPN21, MCBG22, Zd20] provide a textual (or tabular) notation; five studies [BBK+19, HMR+19, KMS19, MCC22, RGJ21] adopt a graphical notation; one [KPRS19] offers both a textual and a graphical notation.

No Concrete Syntax Available Two studies [Al-20, MRC+22] do not provide a DSML-specific concrete notation. In particular, Al-azzoni [Al-20] left the definition of a complete DSML as future work while [MRC+22] is conceived to reuse the notations offered by tools defining data science pipelines. However, by leveraging EMF, a tree-based notation is possible by automatically generated editors, and, potentially, compatible technologies can provide textual or graphical concrete syntax options (e.g., via Xtext and Sirius, respectively).

Textual Notation In [VGZS20], De La Vega et al. provide a textual concrete syntax for the Lavoisier DSML defined by an Xtext-based grammar. Similarly, in [MCBG22], the approach is built on top of ThingML and, as such, it provides an Xtext-based textual editor. In [GGC22], the textual concrete syntax is defined by a recent language workbench, Langium. In [HMFl17] and [HMS+19], an Emfatic-inspired textual modeling language is defined. In [MPN21], five different interwoven DSMLs, are proposed, mixing textual and tabular projections, created with JetBrains MPS. In [Zd20], a SQL-like textual notation is proposed. However, they do not provide any grammar, and then the textual notation is just a proposal.

Graphical Notation In [BBK+19] and [HMR+19], the graphical concrete syntax is defined through capabilities offered by the WebGME language framework. [KMS19] implements the metamodel as a UML profile in Papyrus. The UML Class Diagram is chosen as graphical notation since all the stereotypes inherit from the Class metaclass. No DSML-specific customization of the UML graphical notation is offered. [MCC22] provides a web-based graphical editor realized using Sirius Web[9]. In [RGJ21], the DSML

[9] https://www.eclipse.org/sirius/sirius-web.html

provides a graphical concrete syntax and editor realized in Sirius[10]. However, the paper does not discuss or show its graphical elements.

Multiple Notation In [KPRS19], Kushmenko et al. are the only ones proposing a mix of textual and graphical concrete notations to represent AI concerns. However, it is worth noting that the SVG-based hierarchical representation of components and connectors is made for visualization purposes and is not editable[11].

Model Transformation

Twelve selected studies include model transformations as part of the proposed approaches. These model transformations are classified based on their intents, as described in [LAD+16], and the technology they use, as described in [KBC+19]. Table 3.6 summarizes the intents of the model transformation for each paper, as well as the main model-driven technologies used. It is important to note that none of the papers explicitly list or classify their model transformations. The identification of existing transformations and their intents is part of the SLR work to increase possibilities for comparison.

Nine studies leverage model-to-code transformations [Al-20, BBK+19, HMFl17, HMS+19, HMR+19, KMS19, KPRS19, MPN21, MCBG22] to perform refinements on involved artifacts to generate executable code. Three studies [HMFl17, HMR+19, RGJ21] aim at executable models by defining translational semantics for their DSMLs. Five approaches [VGZS20, HMS+19, HMFl17, HMR+19, MPN21] provide more than one transformation with different intents. Two approaches [VGZS20, GGC22] translate artifacts across different modeling languages

The rightmost column in Table 3.6 mentions the main model-driven technology leveraged by the studies to implement model transformations.

[10] https://www.eclipse.org/sirius
[11] https://github.com/EmbeddedMontiArc/Documentation

Table 3.6: Model transformation intent category and concrete intent.

Paper	Intent Category	Concrete Intent	Tool
[Al-20]	Refinement	Model to Code	Epsilon Generation Language (EGL)
[BBK+19]	Refinement	Model to Code	JS Implementation
[VGZS20]	Language Translation	Translation	Xtend
	Abstraction	Restrictive Query	
[GGC22]	Language Translation	Translation	Typescript (Visual Studio Code)
[HMS+19]	Refinement	Refinement	n.a.
	Refinement	Model to Code	
[HMFl17]	Refinement	Model to Code	KMF/GreyCat
	Semantic Definition	Translational Semantics	
[HMR+19]	Refinement	Model to Code	n.a.
	Semantic Definition	Translational Semantics	
	Analysis	Safety Analysis (added)	
[KMS19]	Refinement	Model to Code	EGL Co-Ordination Language (EGX)/ EGL/ Epsilon Object Language (EOL)
[KPRS19]	Refinement	Model to Code	EmbeddedMontiArc /EMADL2CPP
[MPN21]	Refinement	Model to Code	Jetbrains MPS
	Model Composition	Model Merging	
[MCBG22]	Refinement	Model to Code	Xtend
[RGJ21]	Semantic Definition	Translational Semantics	Xtend

The most commonly used platform among the studies is Eclipse, with Epsilon[12] and Xtend[13] being the most popular tools. For example, in [Al-20], the EGL is used in conjunction with templates to define model transformations that generate Java code. Similarly, [KMS19] uses EGL to generate C# code for making predictions on test data. In [BBK+19], WebGME's code generation capabilities are extended with templates for each sub-task. In [VGZS20], two intents of model transformations are reflected: language translation and abstraction using a restrictive query. The model transformation, based on Xtend, transforms dataset descriptions into tabular datasets using low-level data transformation operations, which can then be used in data mining algorithms. In [HMFl17], the GreyCat framework, built on the KMF, provides code generation toolsets for building object-oriented applications. In [HMS+19], the concept of using code generators to generate ML code is mentioned. In [HMR+19], the ALC toolchain enables code generation for data collection or training exercises of learning-enabled components, as well as translational semantics for configuring an embedded Jupyter Notebook that executes the learning model. The approach also allows for the construction of safety cases. In [KPRS19], the MontiAnna2X code generator generates MxNet, Caffe2, or Tensorflow code. In [MPN21], JetBrains MPS language is used to generate Java code. In [MCBG22], Java and Xtend are used to generate Python code. Finally, in [RGJ21], model-to-code transformation is used to complete formal specifications using the Alloy Analyzer.

3.2.2 Artificial Intelligence Concerns

Same as for the MDE concerns, the findings regarding AI development characteristics are presented in the following.

Business Understanding

Industry often faces the problem of missing business understanding and shortcomings in elaborating business values [RR22, BPR21, BPR22, SWZ20]. Therefore, modeling business understanding is essential for mature and comprehensive approaches, e.g., by defining requirements. The assessment revealed that four of the 15 approaches foster business understanding by integrating system-relevant modeling or processes.

In [HMR+19], the business understanding is fostered due to requirements and components modeling using SysML. Particularly, a GSN approach is used to define and structure requirements.

In [MCC22], business-relevant information is modeled through the integration of *Roles*, leading to increased business understanding. Additionally, the metamodel reflects means to model requirements. However, details are currently missing on how the modeling is defined.

[12] https://www.eclipse.org/epsilon
[13] https://www.eclipse.org/xtend

In [Zd20], requirements on ethical ML can be formalized. Particularly, transparency, accountability and fairness are taken into account so that specific attributes are protected during the implementation, e.g., attributes consisting of values such as 'race' or 'age'.

In [RGJ21], a method to describe ML datasets from an requirements engineering perspective is presented. Notably, functional and non-functional requirements are integrated to describe dataset structural requirements.

Data Understanding

The data understanding fosters the downstream processes of CRISP-DM. Additionally, it allows assessing dataset quality and streamlining to form hypotheses for hidden information [WH00]. In the selected literature, seven approaches support modeling some aspects of the data understanding.

[VGZS20] contextualizes dataset properties and improves data understanding by implicitly applying rules on how to select data. In [GGC22], a detailed description of a dataset and data composition is given that fosters the overall data understanding. In [KMS19], data understanding is enhanced due to the input data's graphical representation and the variables' composition. In [MRC+22], data understanding is promoted by describing data attributes such as the data type. Furthermore, the type of ML algorithm is described, allowing the reproduction of an ML project.

In [HMFl17, HMS+19], the enrichment of properties on a metamodel-level is enabled, which contributes to further description of the properties and, therefore, increases data understanding. Moreover, the interconnection of the data properties is highlighted by the underlying principle. Still, the description of the attributes is not very detailed, leading to no support in understanding a single property and its origin. In [RGJ21], the advanced requirements modeling allows for better understanding datasets with specific properties and structured data elements.

Data Ingestion

Ten of the given 15 approaches describe the loading and ingestion of data. Data ingestion, in this sense, refers to the loading or referencing of the input datasets.

In [MPN21], the implementation of data ingestion using a DSML is described. Six other approaches support the specification of a file path, URI, URL, etc., to reference data [MRC+22, Zd20, HMR+19, Al-20, VGZS20, MCC22]. In [VGZS20], the loading of the dataset is described by specifying the name and path of the file or SQL server in combination with SQL selection scripts. Therefore, this approach supports both file and database-related data. In [MCC22], data loading from various sources, such as SQL servers, is supported.

In contrast, to fix data sources, the loading from edge devices or sensors is supported by three approaches [BBK+19, KPRS19, MCBG22]. In [BBK+19], data loading from various edge devices is presented using technologies such as RabittMQ or Kafka. In [KPRS19],

data loading is provided with tagging schemas for EMADL ports. In [MCBG22], two approaches are given, first a black-box approach, where the ML model is imported from a pickle, and second, the paths or URLs of the dataset(s) are passed to the training, validation, and testing of the algorithm.

Feature Preparation

The preparation of features for certain ML algorithms is supported by five of the 15 approaches.

In [BBK+19], the feature preparation is defined in the metamodel. Unfortunately, details on the specific methods, parameters, or the order of execution are missing. In [Al-20], normalization of dataset features is supported. However, other pre-processing methods are not supported in the metamodel. In [MRC+22], data operations contain one or more input or output ports. Each data operation is an atomic operation on the input data to produce certain output data. In [MCBG22], each state allows executing functions. The keyword *DA_Preprocess* is used to apply data preparation methods on a specific dataset. In [MCC22], features can be prepared with specific *feature extraction techniques*, and data can be transformed with data engineering techniques, e.g., *Regression substitution*.

Model Training

The specification of an algorithm and the related training of the model is depicted in 11 of the 15 approaches. The types of algorithms can be separated in Inference [KMS19], ML [HMFl17, HMS+19, MPN21, MRC+22, MCC22] and Deep Learning using Neural Networks [Al-20, BBK+19, HMR+19, KPRS19, MCBG22].

Inference [KMS19] extended the approach of [Bre14] with the required implementation using SysML and Papyrus modeling framework. Within the original approach [Bre14], model training is given by an assignment for each variable, whether it is an observed variable, a random variable, or a standard variable. Details on hyper-parameter tuning are not given.

Machine Learning In [HMFl17, HMS+19], various algorithm models can be used with specific input (learning) and output attributes. In [MPN21], the algorithm (referred to as *approach*) is specified aligned with various hyper-parameters, e.g., *Random Forest Cross Validation Folds*. In [MRC+22], the algorithm type, e.g., *Random Forest*, with a specific task type, e.g., *Classification* can be described. Hyper-parameters are not presented in the metamodel. In [MCC22], hyper-parameters and performance criteria can be specified for each AI model.

Deep Learning In [Al-20], the training is defined using an *MLPDescription* block with certain learning rules like Backpropagation. Further details on other hyper-parameters or the output's facilitation are not given. In [BBK+19], an algorithm for the training

is defined in the metamodel. Moreover, hyper-parameters are defined and applied to a specific algorithm in the editor. In [HMR+19], an experimental model defines the model training. The details of the implementation can be found in the Jupyter Notebooks. In [KPRS19], the training of NN is given with possibilities to specify the network layers and connections. In [MCBG22], state diagrams are used to define various steps of the algorithm. With the state keyword *DA_Train*, various training-related settings are made, and with *DA_Predict*, the trained model can be applied to data.

Metrics/Evaluation

To assess the validity of an algorithm, four of the 15 approaches integrate the modeling of metrics.

In [HMR+19], the metrics are applied directly in the Jupyter Notebooks, which is not actually a modeling approach. Nevertheless, the Jupyter Notebook is integrated into the model. So it can be considered as part of the model.

In [BBK+19], metrics are integrated into the metamodel and can be applied to the training output. In [KPRS19], the evaluation metrics are selected using the name of the metrics, e.g., Mean Squared Error (MSE).

In [MCBG22], basic metrics such as Mean Absolute Error (MAE) or MSE can be applied to the algorithms, such as regression algorithms.

3.2.3 Frameworks (Methods & Tools)

Most of the approaches are based on frameworks and tools. Table 3.7 depicts each approach's used frameworks and tools. Most of the approaches do not particularly mention the underlying methods. Still, similarities can be seen.

3.2.4 Available Artifacts and Domain of Application

Artifacts are a means to enable the replication of research results. Table 3.8 shows whether artifacts are part of the publication, represented as a reference to an online resource, or not given at all. Additionally, the type of application mentioned in the publication or inherently given through the evaluation sample is depicted in the table. If no specific domain is mentioned or derivable, *Unknown* is annotated.

As a result, eight approaches work with datasets, which can originate from any domain. The processing of IOT data is presented in five approaches, whereas one is more specific for image data.

3.3 Results and Discussion

The discussion is organized according to the research questions in Section 3.1.1.

Table 3.7: Used methods and tools (RQ1.4).

Tool or Method \ Paper	[Al-20]	[BBK+19]	[VGZS20]	[GGC22]	[HMFl17]	[HMS+19]	[HMR+19]	[KMS19]	[KPRS19]	[MPN21]	[MRC+22]	[MCBG22]	[MCC22]	[RGJ21]	[Zd20]
Alloy														✓	
BPMN														✓	
DeepForge							✓								
EMF-based	✓		✓								✓			✓	
Epsilon	✓							✓							
GSN							✓								
GreyCat					✓	✓									
Jetbrains MPS										✓					
Jupyter Notebook							✓							✓	
Langium				✓											
MontiCore Workbench									✓						
Papyrus								✓				✓			
Python			✓												✓
ROSMOD							✓								
SEAM							✓								
SQL														✓	✓
Sirius						✓							✓		
ThingML												✓			
WebGME		✓					✓								
Xtext			✓									✓		✓	

Table 3.8: Availability and type of artifacts aligned with the type of application.

Publication	In the Publication	Online (Git, Server)	No Artifacts	Type of Application
[Al-20]	✓			Datasets
[BBK+19]		✓		IOT
[VGZS20]		✓		Datasets
[GGC22]		✓		Datasets
[HMFl17]		✓		IOT
[HMS+19]		✓		Unknown
[HMR+19]		✓		IOT (CPS)
[KMS19]		✓		Datasets
[KPRS19]		✓		IOT (Image)
[MPN21]			✓	Adaptive Systems
[MRC+22]		✓		Datasets
[MCBG22]		✓		IOT
[MCC22]	✓			Unknown
[RGJ21]		✓		Datasets
[Zd20]		✓		Datasets

3.3.1 RQ1.1 - What Model-Driven Engineering aspects are addressed in the approaches, e.g., abstract syntax (metamodel), concrete syntax etc.?

From a language engineering perspective, each dimension is reflected in most approaches. As for concrete syntax, sometimes an example with concrete syntax is given, but the whole definition of the syntax is not presented.

The description of constraints is rarely used. A reason might be that constraints are often rule-based terms, which can be eliminated with specific parameters or algorithms from the AI domain.

Although not all approaches define a model transformation, most artifacts generate execution code in Python or Jupyter Notebooks from the models.

The review revealed that current approaches are quite diverse from a language technology perspective. In addition, most approaches rely on textual rather than graphical modeling.

3.3.2 RQ1.2 - Which phases of Artificial Intelligence development aligned with the CRISP-DM methodology are covered by the approaches?

The CRISP-DM development cycle's supported phases are less balanced than the MDE perspectives. More than half of the approaches support the early phases, such as business understanding. The feature preparation is often not mentioned or integrated with only simple features, e.g., normalization of variables is given but not the subsequent processing of pre-processing tasks. The main focus of the approaches lays in the formalization of model training. However, most of the approaches only support a small range of algorithms. Therefore, the applicability might be very case specific and less flexible.

In summary, it can be seen that multiple approaches depict a specific aspect of the CRISP-DM development cycle, but only a few support more than half of the phases.

3.3.3 RQ1.3 - Which industrial domains are supported by MDE4AI approaches?

Most approaches support processing datasets in specific file formats or using data from SQL servers. Since these datasets can originate from any domain, no focus on a domain can be determined in these approaches.

However, some approaches are rather based on IOT/CPS or sensor data, supporting the integration of production systems or data from the use of e.g., CPS products. Nevertheless, no domain can be clearly defined here since collecting sensor data is possible in any domain.

3.3.4 RQ1.4 - What are the used methods and the supporting Model-Driven Engineering tools the proposed approaches rely on?

The present works are based on a wide variety of tools and methods. One reason for this could be the application domain, e.g., SysML would rather be used as a basis if the integration into a mechanical engineering environment is intended since SysML is used anyway. The advantages and disadvantages of the individual methods and tools are therefore considered application-dependent, and no statement can be made about the quality of the underlying methods. Furthermore, there is a trend towards Eclipse and its products (Papyrus, Sirius, Epsilon, etc). The use of EMF for the definition of metamodels or as a basic modeling construct can also be identified as state of the art.

3.3.5 RQ1.5 - To what extent is communication between different stakeholders supported by Model-Driven Engineering?

Communication in an AI project can be fostered by unifying the language of communication, potentially leading to a better understanding and reduced unknown knowledge among team members. With less unknown knowledge, unrealistic expectations might be reduced, being one of the categories of why AI projects fail [WSS22]. The intersection with other domains is mainly in the initial phases of an AI project, mainly the business, and data understanding. Still, the documentation of other phases of the CRISP-DM cycle supports communication among other AI experts. With respect to interdisciplinary communication, only three approaches support the documentation and integration of business understanding, leading to further research needs. Data understanding and the downstream processes of the CRISP-DM are more often supported. However, still, further integration of MDE techniques is required due to the early development of some of the approaches.

3.3.6 RQ1.6 - Which challenges and research directions are still open?

The researchers' observation selected the direction of future research and open challenges. The first observation is that business understanding needs to be more supported. In literature, experts report needing more business values for AI as a challenge, which potentially originates from the missing understanding of AI experts in the specific business. As a result, the experts may not propose suitable approaches that are realistic and relevant for a particular business use case. Aligned with the business understanding, the requirements of a project need to be formalized to allow the derivation of project metrics and further assess the impact of the computational support [RVSS19]. Considering that the second largest group of supported applications in the existing works is IOT. Therefore, Systems Engineering requires to be considered more in MDE4AI approaches. The definition of requirements or the modeling of the environment could also be borrowed and adapted from these approaches.

Another future work that supports the maturity of MDE4AI is consolidating the advantages of the existing approaches and extending these approaches to fit various use cases. The combination of various approaches to a comprehensive methodology regarding MDE4AI could streamline the research topic and foster the development of MDE4AI toolboxes.

Apart from combining the research workforces, future research needs to focus on the collaboration of engineers using methods designed for concurrently working on models. As of the review's findings, the approaches mainly focus on supporting single editors and do not support collaborative work on a single model. With respect to more extensive or interdisciplinary projects, the live and collaborative work on a single model could increase the development performance and the benefit and acceptance of MDE4AI.

Next, the output of MDE4AI is often a derivation of Python code, etc., based on model transformation. Python is an easy-to-understand, well-known, daily-used language of AI experts that might lead to changes in the Python code rather than the model. Consequently, full code generation is not applied, leading to no single source of information because partial truth of information is stored in the model and partial in the Python code [BCW17]. In this context, it is necessary to elaborate a closed-loop process that feeds the results of the executed algorithm back into the model or adjusts the model in case of changes in the code, e.g., in Python. With this closed-loop approach, the model is always up-to-date, and further, the collaboration with others potentially improves because of the abstracted representation of the actual changes.

Finally, only a few approaches mention user studies to assess the impact and benefits of MDE4AI. For this, user studies are required to identify unused potentials and further streamline the development towards a user-centered MDE4AI methodology.

3.4 Threats to Validity

The study's validity describes the extent to which the results are trustworthy and how biases arising from the subjective views of the researcher are avoided during the analysis. Validity must be considered at all stages of a study, and several approaches have been proposed in the literature. Following [KC07], the following threats to validity are considered:

- *Construct Validity:* Construct validity describes the validity of the concept or theory behind the study design such that the results are generalizable [WRH+12]. In this SLR, construct validity refers to the potentially subjective analysis of the studies and the different ways in which data extraction is conducted. Following the guidelines [KC07], each study analysis is conducted independently by at least two researchers. If the researchers cannot agree on a conclusion, a third researcher evaluates and discusses the literature until there is no disagreement. In addition, each selected literature was evaluated using the quality criteria suggested by [LFB20].

A protocol based on [KC07] was defined for performing the extraction protocol, which was discussed by the performing researchers after each step. Additionally, construct validity relates to the selection of keywords. In particular, keywords related to AI are used without specific reference to DE, which is the main focus of this thesis. However, during the study, it turned out that DE is rarely reflected in literature and yet not a well-known term.

- *Internal Validity:* Internal validity describes the causal relationships of the researcher's investigation of whether a factor influences an aspect under study. The particular danger is that a third factor has an unknown effect or side effect. To avoid this danger, the same behavior as for construct validity applies, that more than one researcher assesses the causal relationships. In addition, the tactic suggested by [KC07] was followed.
- *External Validity:* External validity exists when a finding in the selected literature is of interest to others outside the case under study. In this regard, the SLR uses a quality assessment based on [LFB20], so included papers are published in peer-reviewed. Therefore, third-party investigators pre-assessed the selected studies, and the validity of the initial publication is the responsibility of the external authors.
- *Conclusion Validity:* The validity of the conclusion relates to concerns about the reproducibility of the study. The concerns in this paper relate to the possible omission of studies. In this regard, the concerns are mitigated by the carefully applied search strategy using multiple digital libraries in conjunction with the snowballing system as of [KC07]. In addition, the researchers followed the detailed search protocol as defined in Section 3.1 and applied the quality ratings. However, some concerns might exist due to the interdisciplinary nature of the fields involved and the various definitions of modeling and AI. These were minimized, however, by the background introduction in Section 2.1 and Section 2.2.

3.5 Conclusion

AI is emerging in several disciplines today and has recently attracted the interest of the MDE community, with several workshops being held on the subject. The development of AI requires several development phases, which potentially can be supported using MDE approaches. Currently, the support of AI by MDE is still at an early stage of development. Therefore, it is necessary to understand the existing approaches to support AI to streamline future research and build on existing knowledge.

We conducted an SLR to investigate the existing body of knowledge in MDE approaches to formalize and define AI applications. To this end, we followed a rigorous SLR protocol, selected 15 approaches, and evaluated them for several dimensions of interest, from MDE and AI.

The result showed that the language engineering perspective of MDE4AI is already mature, and some approaches seem applicable in industrial case studies. The MDE

approaches focus on the training phase of the AI approaches, while time-consuming tasks such as data pre-processing are often not considered. Additionally, the focus is not on improving communication, collaboration, or understanding of the business processes to be supported, which is reported in the literature as a core problem in AI development projects. Finally, the review showed that the approaches are case-specific and lack general applicability.

CHAPTER 4

State of Practice

This section aims to depict the actual state of practice regarding DE implementation and integration in engineering and engineering-related industries. The quantitative study results aim to streamline the elaboration of new methods and achieve a more significant impact on practical applications.

To enable alignment of industrial practice with state of the art, an industrial survey is conducted to capture the status and identify obstacles hindering the implementation of digital engineering in industry. Digital engineering is a concept that implements digital technologies to support the engineering design process by taking the entire product lifecycle into account [HGH+20, TDS18]. Particularly, various approaches such as digital twins [TCQ+18], design automation [RVSS19], and data science methods [DB21] are used to incorporate data from PLM and extend engineering methods. The extension of engineering methods is, among others, realized using SE and particularly MBSE techniques [HS21].

Due to the scope of this doctoral thesis, only DS related findings are presented in this section. Findings related to Digital Twins and Design Automation can be found in the original publication [RR22]. Additionally, note that in the original publication, participants are asked about DS rather than DE or DDE. However, due to the development of the thesis, the term DE appeared to be more suitable than DS. Nevertheless, the terms DS and DE are used synonymously in this chapter for the sake of correctness of the presentation of the survey results. The following RQs are defined and answered with the findings of the survey:

RQ2 What obstacles hinder the application of Data-Driven Engineering in practice?

RQ3 What is required to promote the integration of Data-Driven Engineering in practice?

A selection of text, figures and tables within this chapter is based on the publication in box „Publications 3: State of Practice“:

Publications 3: State of Practice

[RR22] 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, doi: 10.1017/pds.2022.172.

In the following, first, the applied research method is introduced with details on the design of the online questionnaire and the experimental setup. Second, the findings of the study are presented in Section 4.2 with particular focus on relevant findings to this thesis. Next, the findings of the survey are discussed. Finally, a conclusion summarizes the findings aligned with the RQs.

4.1 Research Method

This section first presents the survey questions used to answer the RQs. Next, the method, including the experimental set-up, is explained.

4.1.1 Survey Questionnaire

The questionnaire consists of 24 closed questions, which are based on a literature review in collaboration with the IWI[1] Institute. To gain additional insights, some questions allow for open answers. Digital Engineering, as it is defined for the study, consists of concepts regarding Digital Twins, Design Automation and Data Science. In this respect, the participants are asked about the relevance of the topics in their respective companies so to only answer questions related to their interests. Additionally, participants are only asked questions about their previous experience if they respond positively to an initial question about whether experience has already been gained. Consequently, the response rate varies for each topic and question and also the actual number of asked questions. Due to the focus of the doctoral project, only relevant questions regarding DS are outlined below:

1. To what extent is DS used in your company?

[1] Industriewissenschaftliches Institut - IWI: https://iwi.ac.at/

2. What is the motivation behind your company's (planned) application of DS?

3. What are the biggest challenges in elaborating and applying DS in your organization?

4. How would you rate your experience with projects using DS in your company?

4.1.2 The Experimental Setup

The method applied for this research is a mixed-methods study [Cre14]. The chosen strategy is called *sequential explanatory* survey, characterized by the strong quantitative learning and explained with the insights from the qualitative part [Cre14, Pat02]. The quantitative survey is executed as an online survey with closed-answers based on literature review, extended with experts from DS domain and industry. Additionally, some of the questions allow additional open-answers to enable more precise results and insights. However, participants could also omit a question and not give any answer. At the beginning of the questionnaire, the following definition for DS is given so that each of the participants has the same understanding and answers the questions with the same perspective:

Data science *refers to the extraction of information and knowledge from unstructured and structured data. The procedure to analyze and understand the data is based on methods and theories from many fields like mathematics, computer science and statistics. The application results in opportunities to develop solutions based on (large) amounts of data, such as predicting production costs or machine maintenance.*[Cao17]

The online survey allowed gathering answers from more companies compared to an offline questionnaire and thus, enables to identify trends in the industry. To support the interpretation of the given answers, a single-person interview was conducted after the survey. The single-person interview was conducted by an expert for interview studies with a background in market research and myself. Not all participants of the questionnaire were interviewed, but only a small random group. The participants of the study are selected based on NACE[2] categorizations. The selected categories are related to engineering products, e.g., aerospace industry, and industries with a high demand for engineering products, e.g., food companies using machines to produce and fill goods.

4.2 Results

This section presents the findings of the study, starting with study relevant metrics, e.g., response rate or characteristics of the participants. Next, the most important results of the individual survey questions are listed.

[2]The Statistical Classification of Economic Activities in the European Community (short NACE from French) is a system that enables the classification of industries in the European Union.

4.2.1 Response and Participants' Characteristics

The questionnaire was sent to 1842 participants in companies that are either working in the domain of engineering or companies that make heavy use of engineering solutions, such as an airline. The total set of companies is composed as follows: 3,4% of companies with more than 1000 employees, 27,5% between 250 and 1000 employees and 69,1% less than 250 but more than 80 employees. 81 participants answered the survey, yielding a response rate of 4,4% and a margin error of 10,5% [Tan11]. However, in DS only 51 participants were interested. The respondents' positions of the DS interested participants are categorized as CEO (1), CTO (1), CDO (2), CFO (1), Business Unit Manager (3), Head of Department (Research (2), Technical/Engineering (4), Digitalization (4), IT (3), Unspecified (15)), Process Manager (1), and others (3). The survey was comprehensively submitted by 66%, 16% partly and 15% after the first question of interest and 3% without any answer. The survey data is analyzed as a single dataset using univariate analysis [BC09]. Further analysis based on separate economic sectors shown to be not significant (p-value > 0.4). The second, qualitative part of the survey was conducted with 7 companies (4 with more than 1000 employees; 3 with less than 250). The interview guide for the qualitative data was created based on the quantitative survey findings. Although a definition for each domain was given prior to the interviews, respondents were sometimes unable to assign their projects to a single domain, e.g., a CAD configurator with AI correspond to data science and design automation? One reason might be the possible overlapping of the areas, e.g., engineers are increasingly adopting data science for design automation applications [CHR+20, JHWL21].

4.2.2 Implementation Status

Figure 4.1 depicts the implementation status of DS in practice. About a quarter are using DS in daily operation and similar amount are planing to integrate DS within the next 5 years. It should be noted that the survey was conducted before the media breakthrough of chatGPT. Close to 50% of the companies are in early pilots or concepts. However, it was not clear from the study to what extent DS is used, e.g., sales forecasting could be considered a standard function of Enterprise-Resource-Planning (ERP) software and daily used DS, or the use of custom software as a decision support. Consequently, the level of application and integration of daily used DS is dependent on the software tools and definition of DS support. Nevertheless, the responses indicate that companies plan to integrate DS in a short period of time rather than in the long term (only 4%).

4.2.3 Motivation

Figure 4.2 illustrates the companies' motivational aspects for applying DS. Half of the companies or more use or plan to use DS to optimize solutions, solve complex tasks or ensure quality. Optimizing solutions is a general term that can be mapped to arbitrary problems. However, in the in-depth interviews, participants related the optimization of solutions to manufacturing and optimizing during the design phase of products. After the

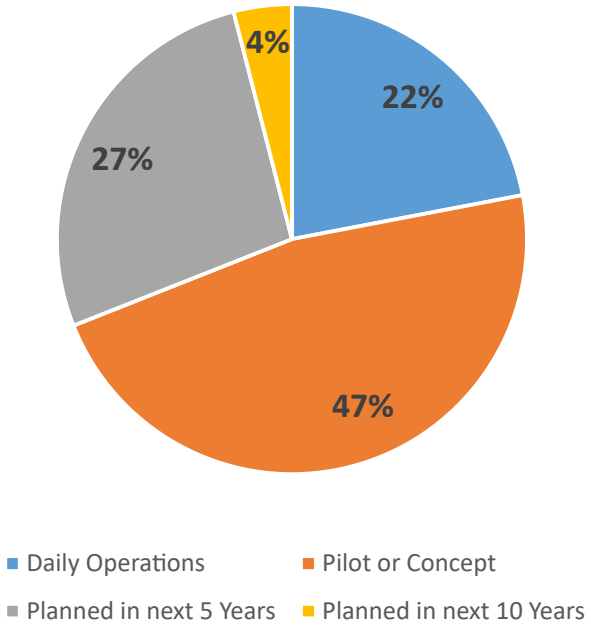


Figure 4.1: To what extent is DS used in your company? (Multiple answers possible; n=51)

three main motivations, the reduction of errors was named most frequently, which is also related to quality assurance in third place. Cost reduction in general is a motivational factor for about 44% of the companies.

4.2.4 Challenges

Figure 4.3 depicts the challenges faced in industry. The main challenges are the the lack of knowledge within the company, shortcomings in data quality and availability as well as the effort of implementation. The problem of sufficient accuracy and integration into the daily processes of the production machines was cited as the reason for the implementation effort. In general, the first challenge with lack of knowledge influences the other top 5 challenges, e.g., lack of knowledge leads to lack of understanding of the existing situation and thus unclear benefits.

Although the collection of sufficient data is a challenge for the companies, the infrastructure requirements, both internal and external, and the lack of suitable tools seem to be no problematical factors. Consequently, data collection and quality also depend on the knowledge of the person using it.

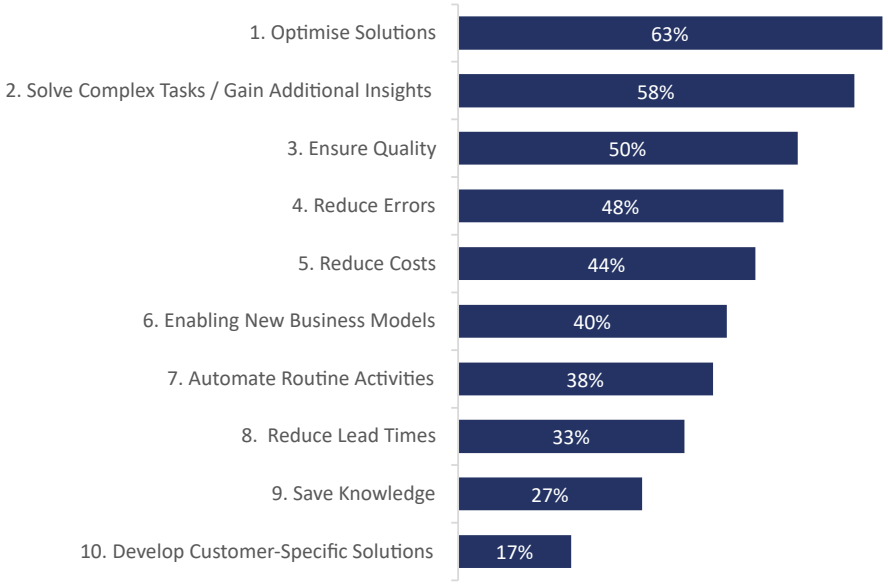


Figure 4.2: What is the motivation behind your company's (planned) application of DS? (Multiple answers possible; n=48)

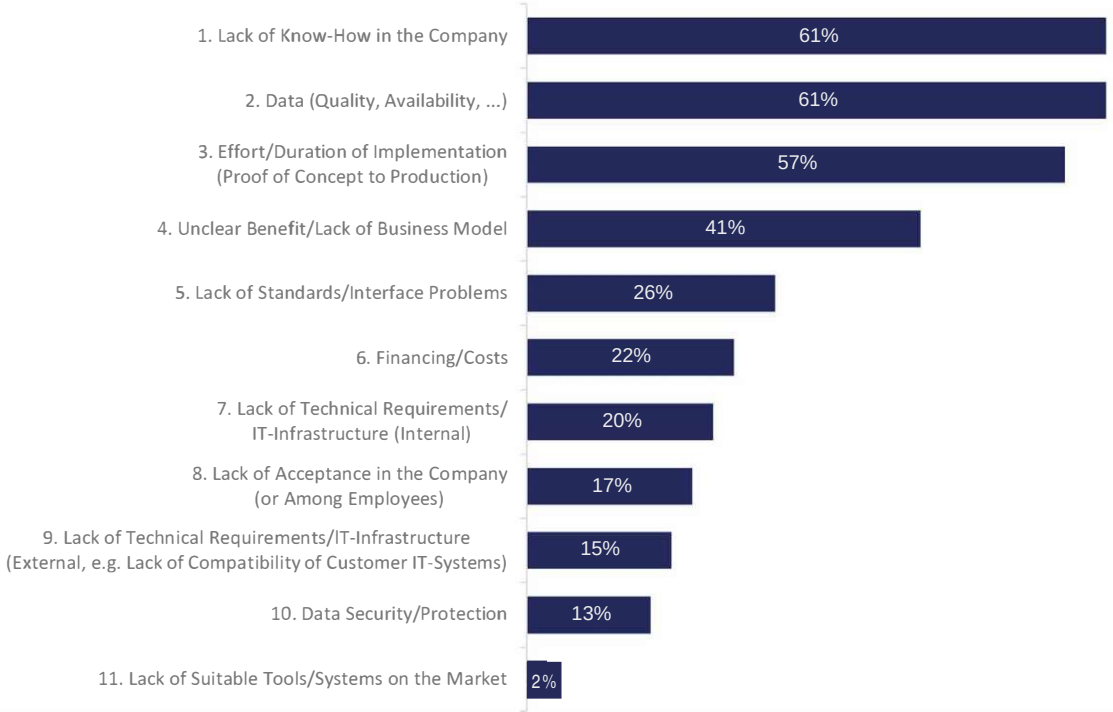


Figure 4.3: What are the biggest challenges in elaborating and applying DS in your organization? (Multiple answers possible; n=46)

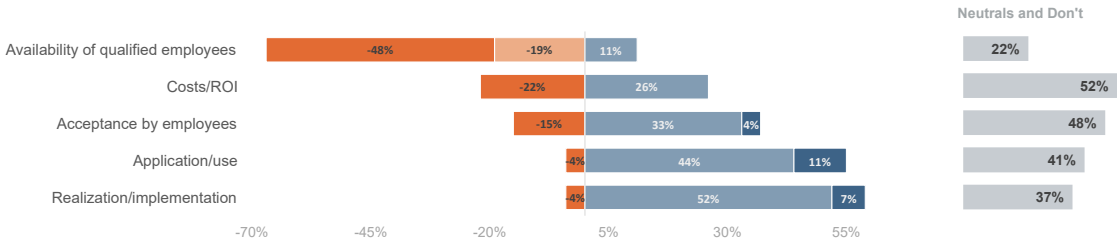


Figure 4.4: How would you rate your experience with projects using DS in your company? (Likert Scale; n=27)

4.2.5 Experiences

A five-point Likert scale from very negative to very positive is used to rate the companies' experiences. Figure 4.4 shows the overall impression on the implementation and integration of DS in practice. Generally, a positive or neutral experience can be recognized except the satisfaction with available experts on the market. Satisfaction with the availability of skilled staff is correlated with cost/Return of Investment (ROI), which might be a reason why the ROI is also not very positive. Consequently, if availability is rated less positively, the cost rating is also less positive.

4.3 Discussion

This section discusses the findings of the industrial survey with respect to open research gaps and potential to foster the applicability of DS in practice. The findings of the survey show that the company's motivation focuses on optimizing solutions, solving of complex tasks and ensure quality. Still, topics such as reducing errors or costs are relevant topics, too.

In contrast to the motivating factors, the companies observe problems with unclear benefits and a lack of business model. Additionally, a lack of data quality and availability is recognized. Both challenges can be traced back to a lack of knowledge about the current situation and the resulting possibilities and potentials of a planned integration. Accordingly, methods must be developed to identify possible use cases and derive business models from them.

Furthermore, human-driven factors like the effort/duration of implementation and the lack of knowledge requires to be solved. The implementation effort and knowledge required to program a DS solution promises to be solvable by integrating non-programming interfaces, e.g., by applying graphical modeling using MDE principles. Additionally, based on the modeling, means of model transformation can be applied to automatically derive DE artifacts. By formalizing knowledge using modeling, knowledge is made reusable, which is not the core focus of companies, but a valuable side effect. Consequently, a step towards standardization of DS interfaces can be achieved by reusing knowledge formalization. On

top, the little number of qualified employees is reduced, if non-programming engineers are empowered to formalize a DS problem using graphical guided modeling tools.

The number of available qualified employees on the market is a challenge that is not just present for DS. Nevertheless, the field of DS is young, which leads to less experience of the employees, as a developer survey conducted by Stack Overflow with about 80,000 participants shows that only students have even less experience on average than data scientists [Sta21]. Additionally, 60% of experts learn programming online, which may require the introduction of best practices to promote and improve the learning of data science methods.

4.4 Conclusion

In this chapter, the current state of DS practice in industry has been discussed based on quantitative and qualitative surveys with Austrian enterprises.

The answer to RQ2 "What obstacles hinder the application of Data-Driven Engineering in practice?" is that a lack of knowledge within the companies leads to unclear benefits and shortcomings in the quality and availability of data. Furthermore, the effort for the implementation in practice is too high and therefore not done. In addition, qualified employees are barely available on the market, which leads to a knowledge gap.

In this respect, RQ3 "What is required to promote the integration of Data-Driven Engineering in practice?" can be answered that methods need to be developed that enable companies to derive use cases from practice and make benefits explicit. Furthermore, the implementation of DS applications requires a methodological and tool support to enable a lower implementation effort. Additionally, methods need to be developed to foster the systematic collection of data with available interconnections to PLM to enable the improvement of data quality in practice. Moreover, the availability of data can be increased through automatic data collection.

CHAPTER 5

Method for Integrating Data Engineering into Systems Engineering

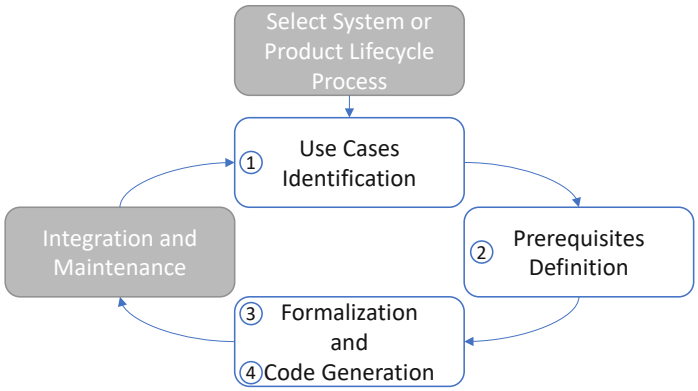
In response to the needs highlighted in Chapter 3 and 4, this section presents a method to support the integration of Data Engineering (DE) into Systems Engineering (SE). Section 5.1 presents an overview of the developed four-step method. It highlights key features, and links to the introduced RQs in Section 1.3 as well as links to the respective method descriptions in Chapter 6 to 9. Section 5.2 industrial use cases from two different application areas, used for evaluating the four-step method.

5.1 Overview of the Method

Figure 5.2 depicts an overview of the four-step method, represented using the ArchiMate modeling language. From top to bottom, there are three levels depicted. The first level depicts the single steps of the elaborated method. The second level highlights the subsequent method to support the respective step, which is related to the values in level three.

Prior to the first step, a system or a process is selected that overlaps with the product life cycle of a system, further referred to as System of Interest (SoI).

The main focus of the method is to support the improvement of an existing SoI within conceptually well-established processes and environments, which is often the case for mature products in industrial practice [WC18]. Although the method focuses on well-established processes and environments, it is still applicable in the early (conceptual) development of systems or production lines to define the integration of DE at an early stage, which allows generating new use cases and strengthening the integration by developing



	Objectives	Research Implications	Challenges	Realization
1	Identify use cases for Data-Driven Engineering	• Identify relevant business processes • Identify data attributes and data interfaces • Identify potential in actual processes	• Diversity of processes • Distributed knowledge sources • Ambiguity of communication	Chapter 6
2	Define prerequisites for integrating Data-Driven Engineering	• Define data associations • Define automatic data collection • Integrate desired tool in actual processes	• Diversity of processes • Legacy hardware support • Diverse data level of detail	Chapter 7
3	Formalize Data Engineering tasks using SysML	• Formalize validated knowledge • Reuse of knowledge from MBSE • Enhanced interdisciplinary communication	• Ambiguous modeling • Complex formalization • Level of detail of formalization	Chapter 8
4	Automatic Code Generation for Data Engineering based on SysML	• Reuse validated and formalized knowledge • Enable flexibility and maintainability	• Decomposition of knowledge • Generation of executable code	Chapter 9

Figure 5.1: Overview of research objectives, implications, challenges as well as the chapter of realization in this thesis.

data interfaces that specifically suite for DE capabilities, e.g., via contextualized data collection [MPRE19].

In terms of focus, the method developed concentrates on *communication* and the *involvement of the various stakeholders* to improve collaboration and *increase acceptance* in practice. The focus is on graphical knowledge representations, as these have proven advantageous over textual knowledge representations [JSD+22].

The objectives, research implications, challenges, and implemented solutions of the four-step method are highlighted in Figure 5.1. At the beginning of each method chapter, Figure 5.1 is depicted and the relevant information are highlighted.

In the following sections, the four steps are briefly introduced, focusing on key characteristics, used/adapted methods and the link to the identified RQs.

Figure 5.2: Overview of the method for integrating DE into SE.

5.1.1 Step 1 - Identifying Data-Driven Engineering Use Cases

The initial step of the method supports to identify use cases for DDE related to a specific SoI. The selection of a SoI is an adjacent topic that is not covered in the scope of this work and is briefly described in Section 5.1.5.

After an SoI is selected, the first step of the developed method is executed. The first step aims to gather knowledge of actual business processes, data interfaces and relationships as well as issues that are potentially supportable with DDE. In this respect, a **Method for Identifying Data-Driven Engineering Use Cases** in the context of SE is developed. The method builds, among others, on methods from the field of SE, Lean Six-Sigma and SysML. The focus of the method is to increase interdisciplinary communication and promote acceptance in practice by conducting participative workshops with various stakeholders from the disciplines involved in the respective scope of the SoI, e.g., mechanical or electrical engineers. Furthermore, the participative workshops are used to enable a collaborative elaboration of potential use cases and therefore allow to validate the findings during the elaboration/workshops. To enable the validation of the findings and to improve communication, graphical modeling methods are used to formalize the gathered knowledge during workshops.

The details of the method to identify use cases based on participative workshops is presented in Chapter 6. Additional information on the evaluation use case can be found in Section 5.2.1. The findings of step 1 answer RQ4 and contribute to answer RQ3.

5.1.2 Step 2 - Integrating Data-Driven Engineering into Actual Processes

The second step of the method enables the elaboration of prerequisites for the integration of automated data collection mechanisms and the integration of the resulting DDE implementation in existing processes. In this respect, a **Method for Integrating Data-Driven Engineering into Actual Processes** is developed. The method mainly builds upon the EA method and SysML. The focus of the method is to increase the amount of defined prerequisites, such as the automatic collection of data as well as the definition of sufficient level of detail of the data collection. Furthermore, changes on actual processes and IT infrastructure with respect to the integration of the elaborated DDE tool is supported. The findings of step 2 aim to support communication, implementation performance and contribute to a reduction of misleading expectations. Furthermore, the transition from current processes to desired processes enables a documentation that allows traceability of changing processes.

The detailed method is presented in Chapter 7. The evaluation use case is discussed in Section 5.2.1. The findings of step 2 answer RQ5 and contribute to RQ2 and RQ3.

5.1.3 Step 3 - Formalizing Data Engineering Tasks using SysML

The previous two steps are preparation steps for the integration of DDE capabilities in an enterprise. The third step enables to formalize DE capabilities and support the implementation of DE tools, being a core step of DDE support. In this respect, a **Method for Formalizing Data Engineering Tasks using SysML** is developed. Within this step, SysML is utilized and extended to enable the formalization of DE tasks on a fine-grained level. With the use of SysML, validated knowledge formalized in MBSE methods becomes accessible and is extendable. Furthermore, the integration of the method in MBSE promotes transfer between different stakeholders and disciplines, aims to improve communication, and creates a pillar for an authoritative source of truth. Additionally, this method lays the foundation for automatic code generation, discussed in the next method step.

The method is evaluated by examining two case studies, and the feasibility and applicability is validated by conducting a user study.

The detailed method is presented in Chapter 8. The evaluation use case is given in Section 5.2.2. The findings of step 3 answer RQ6 and contribute to RQ3.

5.1.4 Step 4 - Data Engineering Code Generation using Model-Driven Techniques

The fourth step of the DDE method utilizes the formalized DE knowledge to generate executable code. With respect to this, a **Method for Data Engineering Code Generation using Model-Driven Techniques** is developed. The method focuses on extendability and maintainability of the code generation using open file formats to reduce blackbox knowledge defined in the source code of the generation engine. To enable extension, the code generation relies on a mapping configuration using JavaScript Object Notation (JSON) file format and text-based templates with placeholders that are exchanged with necessary properties from the model during code generation.

The detailed method is presented in Chapter 9. The evaluation use case is discussed in Section 5.2.2. The findings of step 4 answer RQ7 and contribute to RQ3.

5.1.5 Adjacent Steps not Addressed in this Thesis

The first step of the proposed method aims to identify a system or process that requires to be supported by DDE. However, it is not part of the method to identify an SoI that has potential for improvement, nor is it part of the method to develop a business value. Accordingly, prior to applying the method, an identification of the rough processes is required to narrow down the scope.

Furthermore, at the end of the four-step method, the integration and maintenance of the implemented tool is expected, but out of scope of this thesis. The second step of the proposed four-step method supports the conceptual integration of the solution into

actual processes. However, comprehensive support for the integration of DDE tools into actual processes and IT infrastructures very much dependent on the situation at hand and therefore outside the scope of this thesis.

5.2 Industrial Evaluation Use Cases

The method developed in this thesis can be described as a deductive approach, as it combines several steps that aim to build on the predecessor steps and thus deepen the results of the previous steps [BC09]. In a deductive approach, each method step is evaluated independently of the other method step and the overall evaluation of the elaborated method is deduced from this. In this thesis, the evaluation is divided into two case studies due to the dependency of the two groups of methods. Particularly, method 1 is foundation for method 2, and method 3 is foundation for method 4. Therefore, two major use cases are defined for the evaluation of the overall method. The first case study aims to evaluate the preparation for the implementation, and the second case study aims to evaluate the implementation support. The case studies used to evaluate the method steps are described in the following subsection.

5.2.1 Use Case 1 - Cost Optimization of Engineering Tolerances

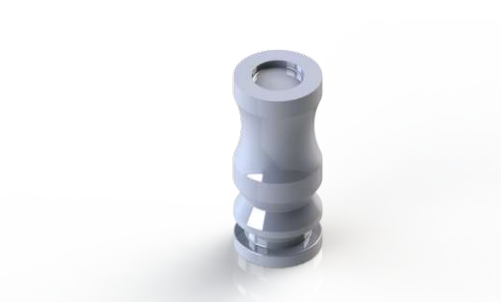
The purpose of the first use case is to evaluate the elaboration of a DDE use case including the definition of the integration of a target DDE application in existing processes. The proposed use case aims to support the production and design of manufactured metal parts based on data of an existing product in a manufacturing environment. This use case is utilized for the evaluation of the method steps steps 1 an 2 as presented in Chapter 6 and 7. The use case is introduced in the following.

In industry, achieving profits and the associated reduction in costs is a key factor for success. In the manufacturing of engineered products, around 70% of product costs are determined in the design phase [EKL07]. Therefore, informed decisions must be made at this stage of product development, taking into account all available data and information.

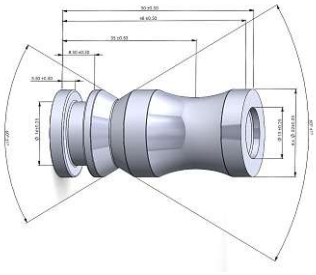
In typical product development processes, the flow of information is directed towards downstream processes. The systematic data backflow, related information and knowledge about the designed product is still manual and requires a lot of effort [CGY+12, SLR17].

For this reason, this use case aims to evaluate the identification of information and data sources that can be utilized by DDE capabilities to contribute to informed decisions in the design phase based on data collected in downstream processes. In addition, the requirements for automatic data collection and the integration of the desired supporting tool shall be planned.

The aimed support is a tool that contributes to make informed decisions in the design of manufactured metal products with special focus on the reduction of manufacturing costs. The design features of the product shall not be changed, e.g., cost reduction due to the



(a) CAD part of the manufacturing part.



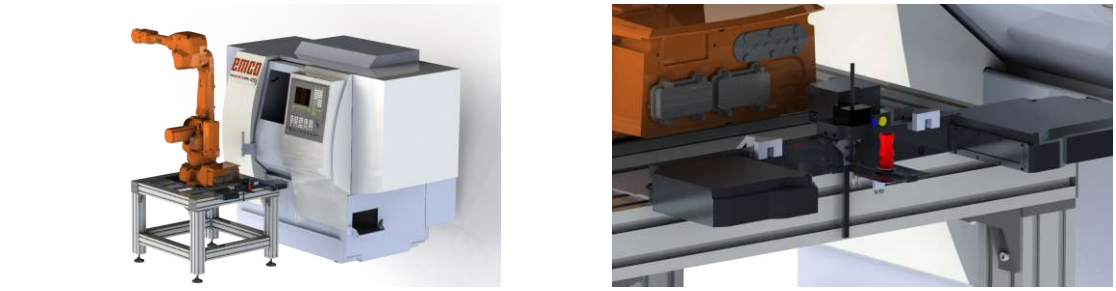
(b) CAD part with MBD dimensions and tolerances.

Figure 5.3: CAD part of the use case.

omission of a borehole. The manufacturing cost reduction particularly focuses on the turning process. Further process steps such as the measurement of the finished part or the assembly can be taken into concern but shall not be optimized.

The selected manufacturing part of this use case is a rotary part with various outside turning features, such as different types of grooves, face features or profiles. Figure 5.3 depicts the Computer-Aided Design (CAD) drawing of the turning part of this study without Model-Based Definition (MBD) (Figure 5.3a) and with MBD (Figure 5.3b) dimensions. MBD is a strategy for using CAD models as primary artifacts containing all relevant information required for e.g., manufacturing, to enable the elimination of two-dimensional paper-based drawings for Geometric Dimensioning and Tolerancing (GD&T) [RZH+17]. From an engineering point of view, the part is not particularly complex and requires little experience for manufacturing.

Depending on the product to be manufactured, various machines are used to produce a part, e.g., milling machine, turning machine, drilling machine. Additionally, arbitrary number of sub-tasks must be performed to produce a specific part. For the manufacturing of the selected part, only a turning machine is required. However, further machines are necessary to measure the result and compare it to the design. Figure 5.4 depicts the case study's infrastructure at the pilot factory of the Technical University of Vienna (TU Wien Pilot Factory Industry 4.0) [HRT+19]. On the right, an **EMCO MaxxTurn45 turning machine** is shown making the actual parts. All parts are produced out of bars, which are feed manually into the machine. The measurement of the parts is made using a **Keyence LS-7000 digital micrometer measuring machine**, depicted in Figure 5.4b with a red colored manufacturing part as specified in Figure 5.3. An **ABB IRB2600 robot** on the left side automatically takes the final parts from the turning machine and place them on a table in the middle of the measurement machine. The table in the middle of the measurement machine is slowly feeding the part through the measurement laser to create a raw point-cloud with a 0.01 micron resolution, at 2400 samples/second. This enables a close-to-machining measuring system without additional programming specific for a single part.



(a) Overview of the infrastructure of the use case in the Pilot Factory at TU Vienna.

(b) Detail view of the manufactured part within the quality assurance.

Figure 5.4: The infrastructure in the Pilot Factory at TU Vienna consisting of a ABB IRB2600 robot on the left, a EMCO MaxxTurn45 turning machine on the right and a Keyence LS-7000 digital micrometer measuring machine in the front with a red colored manufacturing part.

In literature, various approaches are considered for the cost estimation of engineering products, such as parametric, analogous, analytical or bottom-up approaches [CKCR03, HGZ19, HZZ18]. According to the rapid cost model [KCCR02], manufacturing cost components can be categorized as material, manufacturing, assembly and support costs (e.g., rework), amortization of non-recurring and miscellaneous costs. Due to the scope of the case study, the relevant factors are manufacturing and material. The material costs can be determined by the approximated size and the material type of the manufacturing part, which is already known in the conceptual design phase [CKCR03]. The manufacturing costs can be determined by the sum of shopfloor hours multiplied by the wage rate and an additional factor for quality and a factor for contingencies [CKCR03].

However, this is a high-level estimation that does not allow to support the decision making during the product design. Therefore, further assumptions are made as described in the following.

In turning, key parameters affecting the manufacturing are the feed rate, depth of cut and cutting speed apart from fixed costs such as material type or the selected tool [AE12].

These three parameters affect the properties of the yielded output as follows:

1. Time of production - important
2. Dimensional accuracy - important
3. Surface finish - important
4. Tool life - less important
5. Required power by machine tool - less important

Varying the parameters affect the time of production. An improvement of the production time always leads to a degradation in the other properties and vice versa. The outcome varies greatly, again depending on specific (brands of) machines, tools and materials. The evaluation of the properties of the output is performed with regard to the manufactured product. Therefore, tool life and required power are less important than product characteristics and the time needed for production.

Based on the important factors, the following hypotheses are made: Experienced CAD designers and experienced machine operators are rarely the same person and their interaction is time consuming and potentially error inducing. CAD designers tend to set tolerances defensively to give machine operations some wiggle room, or very strictly to ensure that the functionality of the part is given even if the tolerance would not affect the functionality (local knowledge outweighs system knowledge). Machine operators have their own agenda and will strive for a local optimum regarding tool life and production time.

To enable cost-effective product design, production time must be optimized, which requires optimization of dimensional accuracy, i.e., the widest possible margin for manufacturing to allow fast machining, and to meet the required surface finish of the design. Therefore, the goal of the first use case is to identify data that allows to make informed decisions in the design of products in terms of surface roughness and dimensional accuracy without knowing the actual manufacturing costs. Particularly, data from (other) parts that have already been produced will be used to support a designer's decisions in the product development of new or improved products in order to design a cost-efficient product.

5.2.2 Use Case 2 - Weather Station Predictions

The second use case aims to evaluate the formalization of DE concerns using means of MDE. Particularly, means of DE are used to utilize data collected by weather stations to enable weather predictions. The use case enables to validate the method steps 3 and 4 as presented in Chapter 8 and 9.

DE is used among others to analyze data and support informed decision-making in product development, as targeted by the first use case in Section 5.2.1. However, the use of data generated by complex systems, such as Cyber-Physical System (CPS) or Cyber-Physical Production System (CPPS), requires knowledge of various disciplines to collaborate with data scientists, to purposefully use the data and apply it through the resulting algorithms. While the interfaces in CPS systems may be generic, the data generated for custom applications must be transformed and merged in very specific ways so that systems engineers can properly interpret them and gain insights.

To enable efficient collaboration between systems engineers and data scientists, systems engineers must create a fine-grained specification that describes (a) all parts of the CPS, (b) how they might interact, (c) what data is exchanged between them, and (d) how the data relates to each other. CPS and CPPS are just one type of data source. More

generally, any system involved in the product lifecycle of a product and thus part of the PLM is subject to the above considerations.

Even today, the communication of these specifications is complicated due to the strong intertwining with technical and business understanding.

In this respect, surveys show that a third of the data scientists observe a lack of (data) engineering and communication skills [Ana22]. Additionally, data science methods such as CRISP-DM are not able to integrate the technical understanding required for complex technical problems, e.g. in the field of tribology. [BVR21].

For this reason, this use case aims to validate the integration of DE in the context of CPS/PLM data, a system development process that requires the involvement of several disciplines. Consequently, the use case involves multiple interrelated data sources to increase the complexity and potentially lead to necessary data operations such as merging of data-sets. Additionally, support for existing methodologies such as CRISP-DM to enable the transition of knowledge from various disciplines is evaluated.

The system of evaluation in this use case is a weather station equipped with multiple sensors and the purpose of the DE approach is to predict weather conditions. Although the use case is aimed at evaluating a weather station, it can be linked to the first use case regarding tolerance-related costs in manufacturing, since one influencing factor in the manufacturing of milled components is the ambient and material temperature during manufacturing. For this reason, this use case is relevant for evaluating DE tasks. Additionally, in manufacturing, various sensors and data can be used, which in this use case is represented as a composition of weather station sensors.

The selected use case requires the integration of DE for weather predictions based on historical meteorological data. The data used in this study originates from a public weather dataset, collected by a weather station in Seattle. The dataset is contained in a single file, with each line representing the cumulative weather conditions for a single day, e.g., average temperature, maximum temperature, etc. Additionally, a reference implementation of a DE solution is used to compare the result of the formalization as well as the code generation with a ground truth.

As the data was summarized for publication, a single file with all measurements is available. However, to decrease complexity and to represent the original data as much as possible, the single weather data file is split into two source files that contain the same data as the original measurement files. Another reason is that the data originally stems from a local system collecting weather conditions and a second system given by an online Application Programming Interface (API), which must ultimately be two files.

Figure 5.5 illustrates the two CPSs that generate the data for the weather prediction. On the left, the local station is depicted, equipped with various sensors collecting data continuously. However, the weather data is stored only once per day in a file with cumulative values. Similarly, weather prediction for a single day are collected once a day to serve as a marker for the planned supervised learning approach.

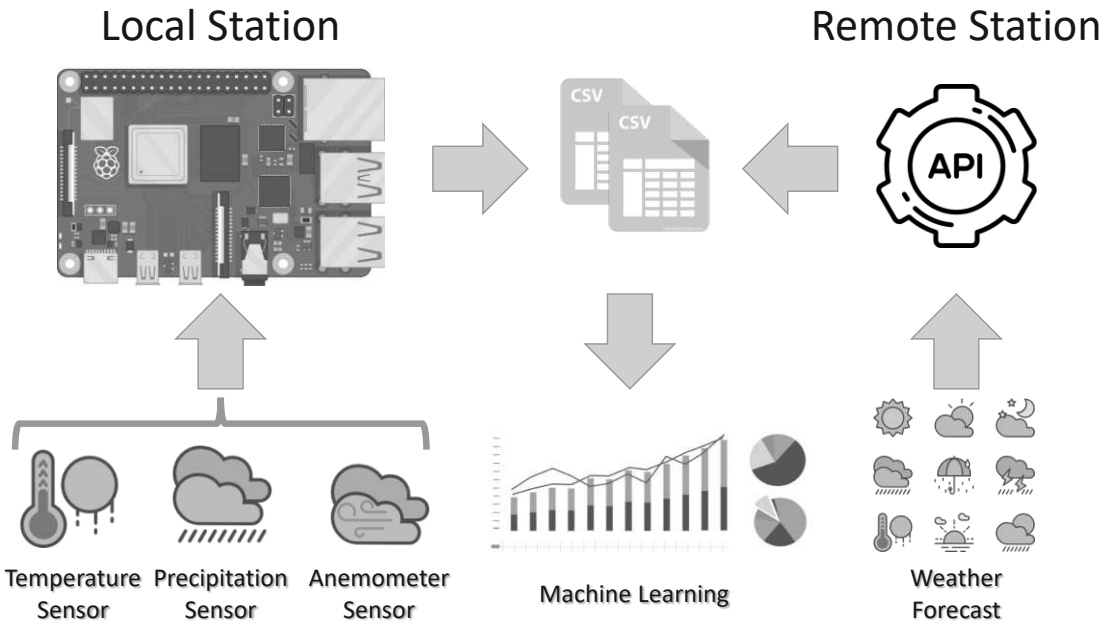


Figure 5.5: Overview of the weather station use case.

The focus of this use case is to show increased communication of business and data understanding among various disciplines by demonstrating the integration of DE into MBSE and facilitate CRISP-DM integration. Build upon the formalization of the DE approach, automatic code generation shall be demonstrated, allowing to reduce the effort and duration for the implementation of DE capabilities.

With the integration of DE task formalization into MBSE, the understanding, support and acceptance of DDE in practice is improved. Finally, with the formalization of DE concerns, knowledge can be reused, maintained and an automatic documentation of sources and DE internal and adjacent processes, such as a DE pipeline, can be reviewed on a conceptual level.

Based on the aforementioned factors, the following hypotheses are made: Integrating the task formalization of DE into MBSE improves the understanding and support of engineers as well as the acceptance of DDE in practice. Furthermore, the use of graphical modeling languages allows formalization and replication of DE knowledge without profound programming knowledge. By automatically decomposing the formalized knowledge, code snippets can be leveraged for code generation, leading to a reduction in implementation effort. Finally, formalizing DE concerns allows knowledge to be reusable, and enables maintenance. Moreover, automatic documentation of the sources and internal processes of DE pipelines can be verified at a conceptual level. Eventually, the lead time of DDE implementations decreases.

CHAPTER 6

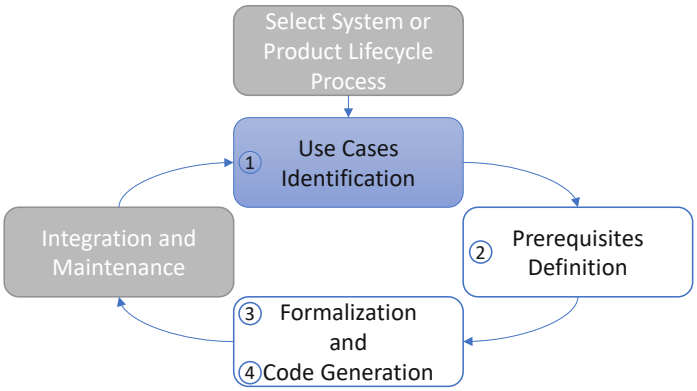
Identifying Data-Driven Engineering Use Cases

This chapter addresses the research objective ① in Figure 6.1 related to the identification of use cases for DDE. As the figure shows, several research implications, such as the identification of business processes, data attributes and interfaces, and the quantification of the impact of the intended DDE support can be deduced.

Right after a system or process that shall be supported by DDE is selected, the identification of potential use cases is performed. A key reason for this method is the lack of business understanding and shortcomings in the elaboration of business values [BPR21, BPR22, RR22, SWZ20]. The lack of business understanding originates among others by lack of knowledge regarding DE [ACMA20, BPR21, RR22]. Consequently, a correlation between knowledge in DE and identification of DDE use cases can be drawn. According to [BPR22], top-down derivation of use cases is beneficial from a business perspective because it has a high degree of alignment with the business and data, resulting in immediate business impact and rapid piloting. However, the proposed approaches lack sufficient integration of data understanding that is a substantial step in the CRISP-DM methodology. Furthermore, there is a lack of focus on communicating business understanding, which can lead to misinterpretation and support unwanted by potential users. Finally, a lack of methodological support with respect to the engineering domain of systems can be observed that have been shown as a shortcoming of the CRISP-DM methodology [BVR21].

In this respect, the following overall RQ is answered by step 1 of the developed method:

RQ4 What are appropriate methods to identify use cases for Data-Driven Engineering?



Objectives	Research Implications	Challenges	Realization
① Identify use cases for Data-Driven Engineering	• Identify relevant business processes • Identify data attributes and data interfaces • Identify potential in actual processes	• Diversity of processes • Distributed knowledge sources • Ambiguity of communication	Chapter 6
② Define prerequisites for integrating Data-Driven Engineering	• Define data associations • Define automatic data collection • Integrate desired tool in actual processes	• Diversity of processes • Legacy hardware support • Diverse data level of detail	Chapter 7
③ Formalize Data Engineering tasks using SysML	• Formalize validated knowledge • Reuse of knowledge from MBSE • Enhanced interdisciplinary communication	• Ambiguous modeling • Complex formalization • Level of detail of formalization	Chapter 8
④ Automatic Code Generation for Data Engineering based on SysML	• Reuse validated and formalized knowledge • Enable flexibility and maintainability	• Decomposition of knowledge • Generation of executable code	Chapter 9

Figure 6.1: Overview of research objectives, implications and challenges addressed in Chapter 6.

Subsequently, the following detailed RQs have been identified:

RQ4.1 How to identify relevant data sources in an enterprise for Data-Driven Engineering approaches?

RQ4.2 How to identify and assess potentials for Data-Driven Engineering based on existing processes and IT infrastructure?

RQ4.3 How to support communication between the involved stakeholders and foster knowledge validation and documentation?

RQ4.4 How to enable the integration of various involved stakeholders in the investigation of Data-Driven Engineering use cases?

Based on the identified RQs, first related work and research gaps are discussed. Second, the elaborated method with a focus on existing process and IT infrastructure is introduced. Next, an evaluation of the approach based on the described use case in Section 5.2.1 is given. Finally, the findings are discussed and summarized.

A selection of text, figures and tables within this chapter is based on the publication in

box „Publications 4: Identifying Data-Driven Engineering Use Cases“:

Publications 4: Identifying Data-Driven Engineering Use Cases

[RR20] S. Rädler and E. Rigger, “Participative Method to Identify Data-Driven Design Use Cases,” in Product Lifecycle Management Enabling Smart X, vol. 594. Cham: Springer International Publishing, 2020, pp. 680–694, doi: 10.1007/978-3-030-62807-9_54.

6.1 Related Work and Research Gaps

The identification of potential use cases for DDE has mainly been discussed in literature by developing processes to support the development of data-driven applications. One sample is a collaboration framework supporting deciding whether DE is sufficient to support a specific use case [HSM+19]. However, the approach starts with discussing whether DE can solve a specific problem without support for identifying potential use cases.

Another approach is proposed to support the identification of so-called intelligent features in conventional mechatronic systems [IGBD15]. The approach is based on SysML and considers methods such as FMEA or Fault Tree Analysis [Eri05]. However, the approach lacks support for the systematic collection and analysis of processes and data gathered among multiple (cyber-physical) systems such as production lines.

In [HJP+20], a five phase approach is proposed consisting of preparing, discovering, understanding, designing and implementing AI use cases. The approach is built on the Technology-Organization-Environment (TOE) framework [Bak12]. According to [HJP+20] there are two possibilities of use cases, first, addressing existing problems and second, finding unknown potentials in the company with the help of AI. Although, the approach considers business understanding for the selection of AI algorithms, only in the last step it is said that "if the required data is not yet available, the organization must plan data acquisition or adapt its data strategy" [HJP+20]. To select a sufficient algorithm, the type and interconnection of data is necessary, which is considered too late and therefore, the approach lacks sufficient knowledge on data in early phases.

In [SFB21], the use cases are either purpose-driven or data-driven. In an purpose-driven approach, the use case is derived by initial identification of potential problems and further existing processes can be revised with AI supported solutions. The data-driven approach builds on existing data that needs to be exploratory investigated. However, both paths of identification lack in sufficient guideline and details on the application.

In [BPR21], two approaches for the identification of use cases based on the business and data understanding phases of CRISP-DM are proposed: a systematic top-down and an exploratory user-centered approach. In the top-down approach, business goals, processes, tasks and decision points are analyzed so to allow the identification of potentials for AI.

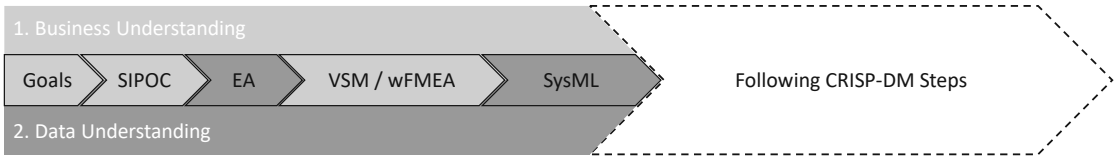


Figure 6.2: Embedding of the use case identification method into the CRISP-DM methodology.

The exploratory approach is based on Design Thinking [HMU+20] and aims to identify potential based on problem formulation of domain experts. Both approaches of [BPR21] are located in the business understanding of CRISP-DM. The authors propose that after use cases are identified, prioritizing and data understanding needs to be achieved to assess the data quality and prioritize the approaches whether they are realistic. However, if multiple use cases are identified in the business understanding, the data understanding phase might be extensive for little available data science experts [RR22], which further lead to long and time consuming processes.

In [DSL21], a study is conducted to identify common use cases in automotive industries. In the survey, experts are asked about existing and conducted AI use cases. The survey is conducted using face-to-face interviews. However, the study does not provide any support for the elaboration of use cases. Additionally, the relevance of the identified common use cases in each company needs to be assessed based on the available data.

6.2 Method

In response to the needs highlighted in Section 6.1 and the fact that AI use case identification requires collaborative work [FMS19], this section proposes a participative method for identification of DDE use cases in engineering processes while comprehensively taking the PLM into account with its technological environments as well as related data and data interfaces. The method builds upon the CRISP-DM methodology and extends its first two steps for business and data understanding as illustrated in Figure 6.2 to make it applicable in an (systems) engineering context.

As shown in Figure 6.2, defining goals is proposed as the first step to create an initial business understanding and formalize the need to improve the existing situation. The target definition forms the basis for the subsequent steps of the identification and enables the streamlining of related processes and thus the evaluation of DDE use cases. Although a goal is defined in the first step, the subsequent steps still requires to be executed independent of the goal to not bias the identification of potentials, e.g., if the goal is defined to narrow, one might specifically search for problems related to the goal instead of other more valuable problems that have to be solved. In this respect, one or multiple SIPOC analyses are conducted in participative workshops to define the processes, stakeholders, and necessary experts for the further steps. Typical workshop participants are engineers, product designers and a workshop leader who guides the re-engineering of

the process steps related to the defined goal. The workshop leader has to be familiar with the here presented method as well as the modeling of EA. The yielded SIPOCs are then further refined using EA modeling to investigate the processes as well as the supporting technological environments and related data with data interfaces. To identify sources of waste and potential to improve the existing processes, pdVSM is executed. Next, W-FMEA is used to quantify the findings and allow to implement the most valuable DDE use case first. W-FMEA additionally enable to derive metrics that lead to an improved evaluation of the defined goal [RV18]. Finally, relevant data objects are identified and contextualized with the identified goal and use case. Particularly, SysML is used to fine-grain model data attributes and enable to interconnect the attributes of data objects with respect to semantic relationships. With the SysML formalization, necessary data understanding can be achieved which is required for the further subsequent steps of the method presented in this thesis. In the following, each step of the proposed CRISP-DM extension is detailed.

6.2.1 Step 1: Definition of Operative Goal

Once a system or a process to optimize is selected, a goal needs to be defined to guide the subsequent steps for identification of use cases for DDE. The definition of the goal is in line with existing approaches to metrics definition, which state that goals must be defined before metrics and corresponding measures are selected, e.g., the Goal-Question-Metric (GQM) [BCR94] method. Hence, goals can refer to specific design artifacts such as "improve lifetime of feature XY" or more generally to (parts of) the design process, e.g., "define less narrow tolerances in the design without losing functionality". Additionally, the desired goal specifies whether DDE can rely on previous design revisions and related PLM data or other designs that feature similar characteristics.

The formalization of the goal is aligned with the GQM proposed by [BCR94]. Table 6.1 depicts the subparts of the goal definition with samples of terms that can be used.

In literature, an approach exists using EA to formalize GQM [CNdST+13]. However, the use of a table seems to be easier to understand and thus contributes to the applicability. Furthermore, the application of EA to formalize GQM requires to be investigated and tested in a user study. Depending on the goal, it might be necessary to formulate more than one goal. Nevertheless, the goal should not be formulated too complex or extensive, otherwise the application of the further methods will become more complex and thus the goal will be more difficult to achieve.

6.2.2 Step 2: Supplier-Input-Process-Output-Customer Analysis

Based on the goal formulation in the previous step, identification of the processes that are related is initiated. More precisely, all aspects of the product lifecycle that impact or are impacted by the investigated artifacts/processes need to be assessed. To acquire the knowledge about the related processes, a SIPOC [YE09] is elaborated within participative workshops to gain a high-level overview of processes and define the scope of investigations.

Table 6.1: Goal definition aligned with the Goal-Question-Metric (GQM) approach [BCR94].

Purpose	Reducing / Improving / Analyzing / Predicting
Issue	manufacturing costs / lead time/quality
Object (Process)	manufacturing process / design process / quality assurance process
Viewpoint	CEO / Department Head / Quality Manager.

With respect to the interdisciplinary nature of the workshop, the participants consist of an expert for modeling who leads the workshop, an expert for the processes adjacent to the defined target, and an expert for data-driven application development, e.g., a data scientist. The necessity for participative workshops originates from the interdisciplinary nature of DDE, a reduction of siloed work and the higher potential to find use cases that benefit to the organization [FMS19]. If more than one process is adjacent, multiple process experts are integrated to create each an SIPOC. To foster the workshop performance, graphical modeling is applied to enable direct validation of the generated models by the participants. The SIPOC process captures the process in three to five main tasks (P), the related input (I) and output data (O). The main suppliers (S) and recipients (C) are connected considering read and/or write access with the input/output data. Within the workshop, typically the process steps (tasks, P) are identified first, since the participants typically are involved in various processes and the focus needs to be clearly stated. Following this, the identification of necessary input (I) and information supplier (S) is conducted, followed by the output (O) and information receiver (C). However, the order can be swapped by the workshop lead as well as the team if another order seem to be beneficial, e.g., the information receiver (C) first, to enable a more customer oriented view [RKR15].

Figure 6.3 shows a generic SIPOC template using the ArchiMate software. The template consists of one process with two subtasks structured according to the SIPOC schema. Additionally, two input, two output, one information receiver and two information supplier are integrated to visualize the composition of the parts using EA syntax. The dashed arrows between the model elements, e.g., actor (*Information Supplier 1*) and business objects (*Input Data 1*) indicate an access relationship, which means that an active element (actor) either delivers or receives information, depending on the direction of the dashed arrow. The arrow between the two tasks indicates a causal dependency, e.g., *Task 2* is executed after *Task 1* is done.

6.2.3 Step 3: Analysis of Actual Processes, Data Interfaces and IT Infrastructure

The results from the SIPOC analysis are used as a basis to further detail the processes by a step-wise decomposition of the tasks to yield single activities [OD05]. Same as for the SIPOC modeling, the ArchiMate modeling language [Archi19] realized as ArchiMate software is applied to graphically model details of the business processes,

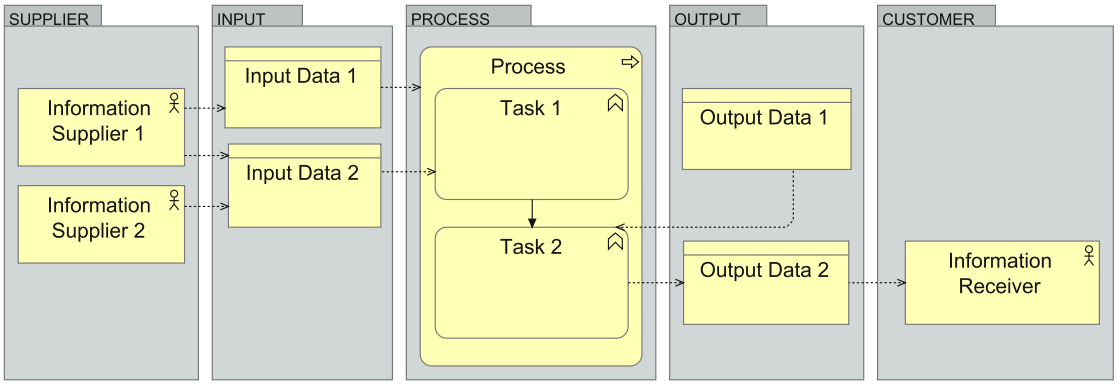


Figure 6.3: Template for SIPOC analysis depicting two generic tasks with respective supplier, input, output and information consumer (customer).

related applications as well as infrastructure including all relationships. To elaborate the detail processes with related applications, the modeling expert guides the workshops by successive detailing of the main tasks from the SIPOC analysis based on a directed question-answer-talk. The workshop leader, asks the question in a fashion that allows to dig deeper in the processes and activities of the experts with a special focus on potential shortcomings, such as the ones proposed in Table 2.2. Still, the workshop leader needs to keep the goal in mind to not overload the model and stay focused on relevant processes and activities related to the defined goal. With respect to the related applications, focus is put on the interaction of the applications within the process. Therefore, questions are posed related to the integrated tools, the used features and interfaces, e.g., API or user interface, as well as the generated and consumed data. The generated and consumed data are presented in a high-level representation only, and may be refined in the final step of the method described in section 6.2.6. To represent a comprehensive model of the EA [Lan09], the relevant infrastructure behind the applications need to be modeled. If an activity is out of the knowledge of the workshop participants, relevant experts requires to be integrated into the modeling process of the specific activity.

Figure 6.4 depicts a sample decomposition of the SIPOC template of Figure 6.3. On top, the business process of the SIPOC is extended with arbitrary subtasks. Similarly, other modeling elements of the ArchiMate language can be integrated if necessary. Underneath the business layer, the application layer is depicted with relevant connections to the business objects and processes. Each application is additionally connected to its technology. The template aims in guiding the decomposition, reducing time in creating a first model of the SIPOC, and providing guidelines on which elements are necessary and how to arrange them. However, other templates provided by the workshop leader (modeling expert) are valid, too.

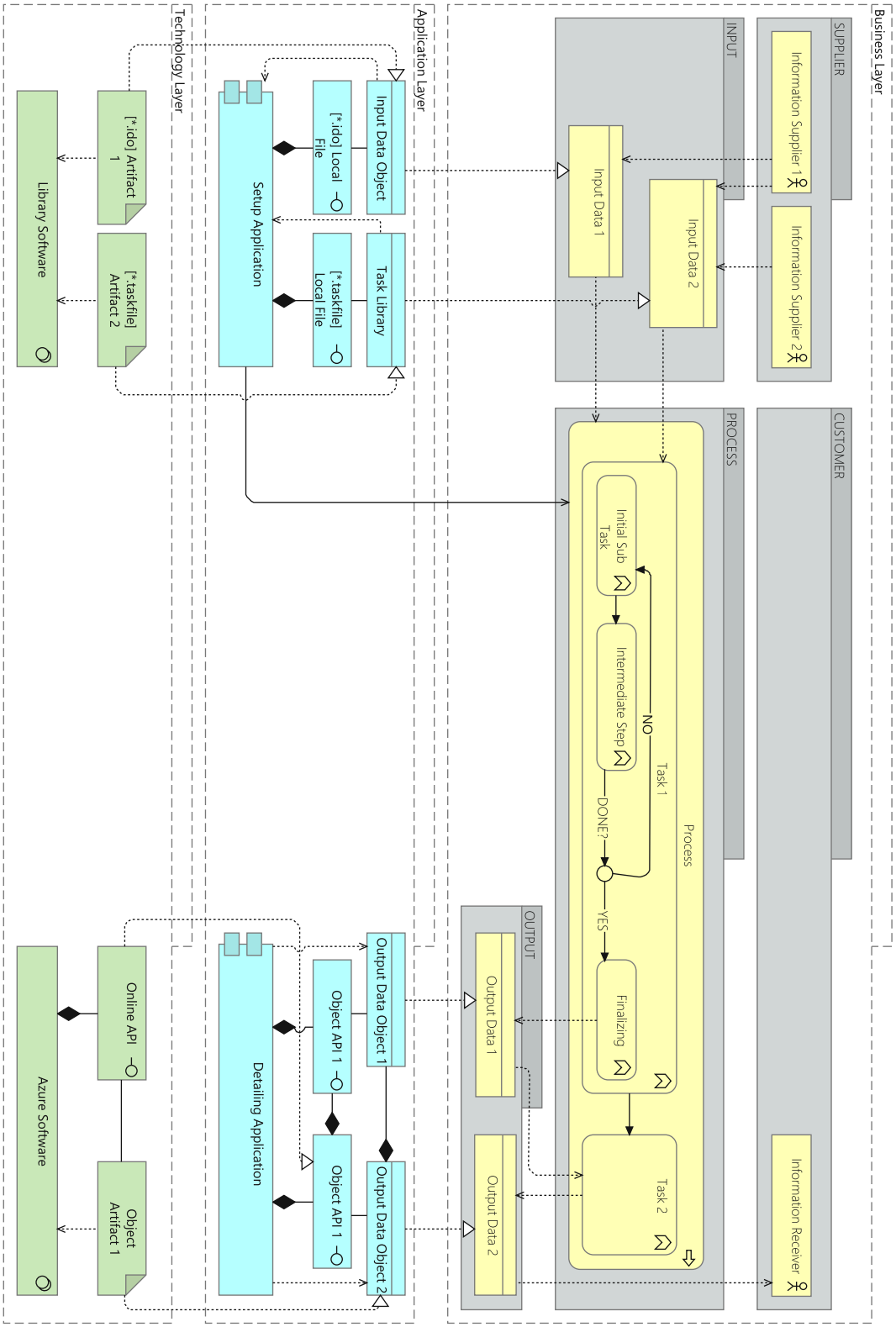


Figure 6.4: Template for modeling a detailed process with IT artifacts based on a SIPOC model.

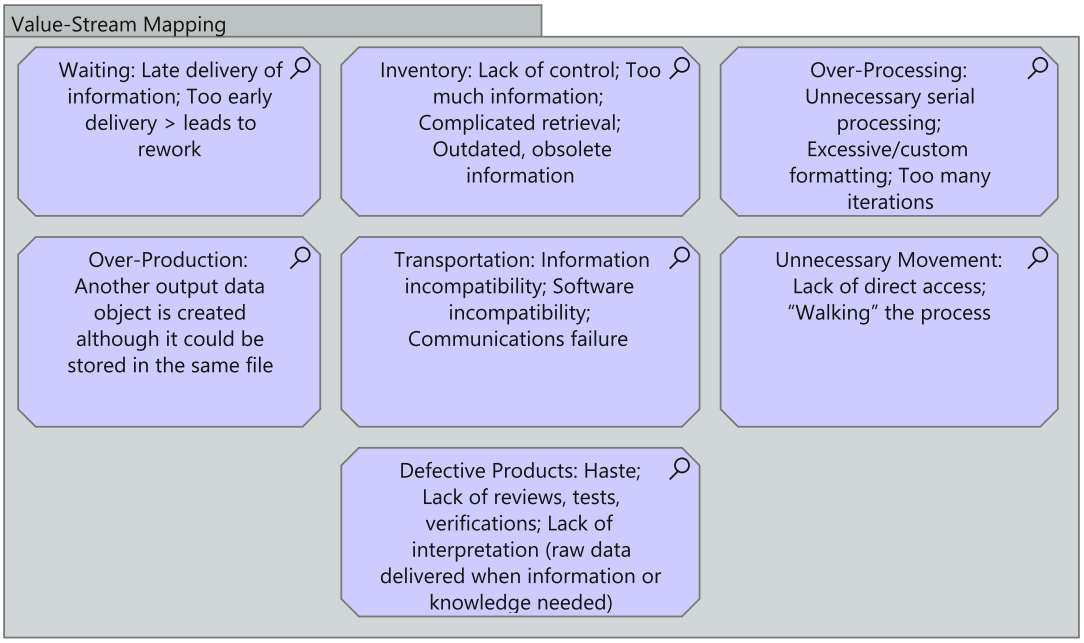


Figure 6.5: Template for Product Development Value Stream Mapping (pdVSM) based on ArchiMate.

The yielded result of this model is the input for the prerequisites elaboration, presented in Chapter 7.

6.2.4 Step 4: Product Development Value Stream Mapping

To further strengthen the business understanding, information wastes are identified within the previously yielded model of the EA using a pdVSM. Particularly, a workshop is conducted with the domain experts to define information wastes as depicted in Table 2.2. To connect information wastes with the EA model, Figure 6.5 depicts assessment templates using the ArchiMate language. The template is used to connect potential wastes in the process. Particularly, the workshop lead asks the participants each dimension of waste and whether a task might be related to one of these issues.

Figure 6.6 depicts the integration of an information waste into the template of the detailed processes using purple assessment objects from Figure 6.5. As depicted, the template content is adjusted so to describe the issue that it can be further used in Section 6.2.5.

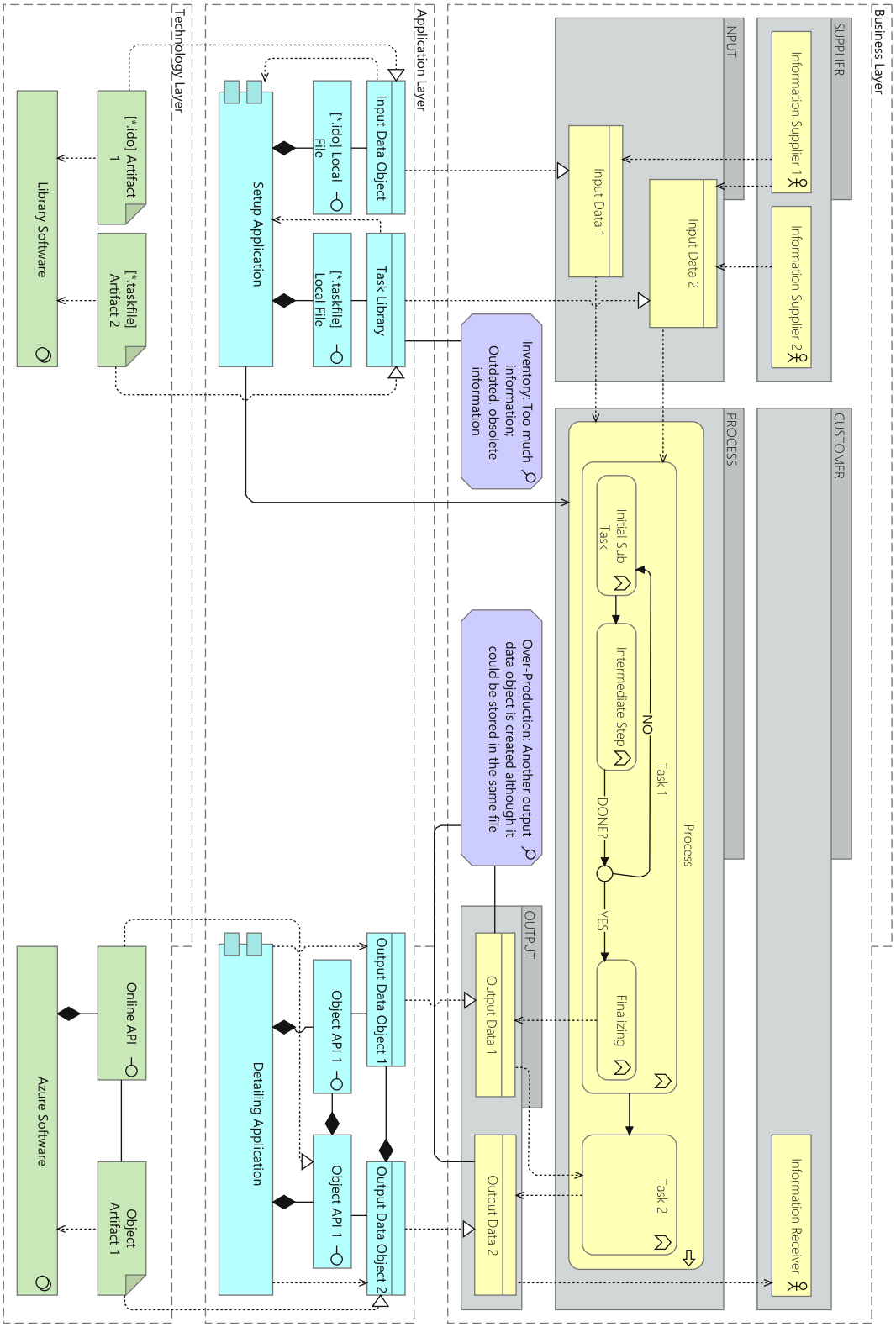


Figure 6.6: Sample integration of the Product Development Value Stream Mapping (pdVSM) Templates.

6.2.5 Step 5: Waste Failure Mode Effect Analysis

To assess the effects and causes of the identified sources of information waste, W-FMEA is conducted based on the result of the previous section. To conduct a FMEA, various templates are proposed in literature [BCJS92, PP14, Pri96, dC14]. To enable a connection between the ArchiMate model and the W-FMEA, a template is created with the following columns (based on [dC14]):

- **Process -** The process related to the information waste.
- **Business Task/Object -** The specific task or business object with information waste.
- **Reference to ArchiMate View -** Where to find the information waste in the ArchiMate model.
- **Waste Mode -** The identified waste mode based on Table 2.2
- **Waste Description -** The description of the identified waste in the ArchiMate model.
- **Cause of Waste Mode -** The cause of the waste.
- **Occurance (O) -** Scoring of occurance from 1-10 as suggested in [dC14].
- **Detection (D) -** Scoring of detection from 1-10 as suggested in [dC14].
- **Effect of Waste Mode -** Description of the effect on the company/department.
- **Severity (S) -** Scoring of severity from 1-10 as suggested in [dC14].
- **Waste priority number (WPN) -** Result of the multiplication of occurance, detection and severity.
- **Cause priority number (CPN) -** The sum of WPN with the same cause of waste.
- **Priority -** The priority based on WPN, taking CPN into concern.
- **AI Supportable -** An identifier that indicates whether AI support is possible or not.

In preparation for the W-FMEA workshop, the table is filled in as comprehensively as possible. The remaining columns, except for the *Potentially AI Supportable* column, are filled in by a DE expert. The result of the assessment is an ordered list of potential issues to be solved. Based thereon, a DE expert assesses each column with respect to potential to be supported using data-driven algorithms and adds a *Yes* if supportable, *No* if not supportable and *?* if further information on the available data, its quality and

amount as well as the relationships between the data is required. If a question-mark is set, the assessment of AI possibilities requires the method step 6 in Section 6.2.6 to be conducted first. In addition to *Yes* or *No*, justification is desirable so that decisions can be understood later.

6.2.6 Step 6: Detailed Data Analysis

Based on the EA model established in Section 6.2.3 and the identified potentials in Section 6.2.5, details on the data understanding is elaborated next. Particularly, one or multiple potential use cases from Section 6.2.5 are selected and influencing input and output data objects are refined to represent a fine-grained level of detail and reflect attributes and their relationships using SysML BDDs [OMG24]. The elaboration of the knowledge regarding data attributes is established by conducting an workshop with an expert that is daily involved in the respective application use. Additional information are collected based on API documentation and other data description documents available.

Based on the BDD with the modeled data attributes, dependencies within the data can be highlighted based on data dependency relationships. Explicitly, the focus is put on the semantic dependencies within data objects with respect to the selected use case, e.g., the machining program of a milling machine in the production phase is dependent on the CAD drawing in the design phase. These data relationships are visualized by adding information flows using the SysML *Item flow* relationship to the BDD. The semantic connection of the attributes are double-checked with domain experts as well as an software engineer or administrator that is knowledgeable with the IT infrastructure. As a result of this step, the influencing data sources can be identified to build the basis for the systematic implementation of DDE. The result of the modeling acts also as basis for the DE task formalization introduced in Chapter 8 as well as for the definition of further preconditions, such as the need for another data interface in Chapter 7.

Figure 6.7 depicts a sample SysML model with data attribute level description of the data objects depicted in Figure 6.6. The figure of the SysML model shows a BDD with data attributes and data types. The PLM block on top is used as binding element among the entire product lifecycle. Each element connected to the PLM is available in the Internal Block Diagram (IBD) that is used to describe semantic connections.

Figure 6.7: Sample of SysML Block Definition Diagram (BDD) with detail data attributes.

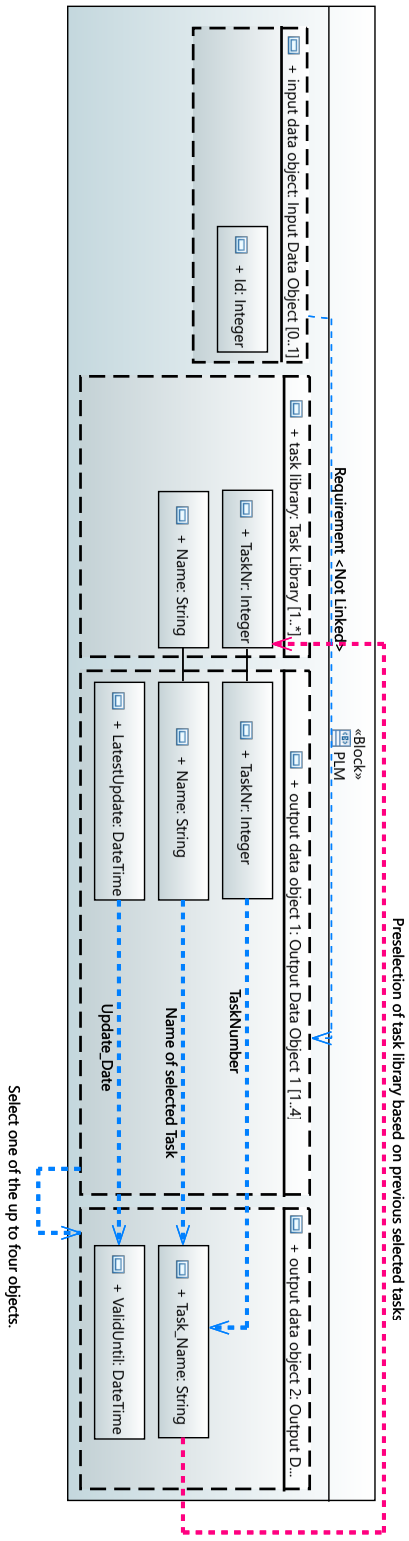
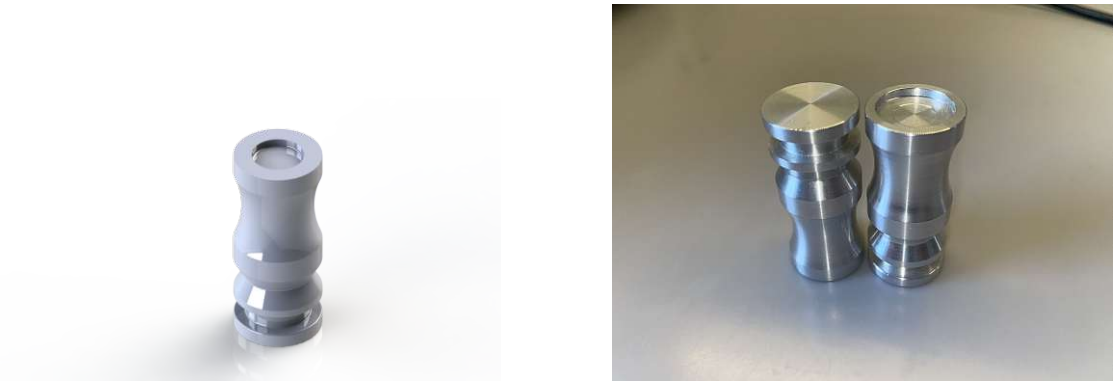


Figure 6.8: Sample of SysML Internal Block Diagram (IBD) indicating item flows.



(a) CAD rendering of the use case bishop chess figure.

(b) Manufactured bishop chess figure using turning process.

Figure 6.9: The chess figure in CAD format and as final manufactured part.

The IBD in Figure 6.8 connects the data attributes on a semantic level using *item flow* feature of SysML. The color of the flows indicate either the flow that is already established (blue) or the potential flow (magenta) of data using tool support such as intended by DDE. The label of an *item flow* indicates the semantics of the information flow, which can also be described in somewhat more general terms. As a sample, CAD attribute influence various downstream process data but are not directly interrelated, e.g., CAD attribute correlation with the feed rate of the Computerized Numerical Control (CNC) program.

As a result of this step, the interim result of the W-FMEA in Section 6.2.5 can be complete. Furthermore, the findings serve as a basis for the definition of the targeted EA integration of DDE support with necessary interfaces for the collection of data in sufficient quality and quantity as described in Chapter 7.

6.3 Evaluation

This section validates the introduced method using a case study presented in Section 5.2.1. The use case involves several steps, including the design and manufacturing of a turned part, more specifically the turning of a Bishop chess figure. The Bishop chess figure manufactured is depicted in Figure 6.9a as CAD and in Figure 6.9b as manufactured part.

6.3.1 Step 1: Definition of Operative Goal

The goal of the evaluation project is to reduce the manufacturing costs of a bishop chess figure during the turning process without changing the functional specifications, the shape of the product or the material using DE techniques. To enable the reuse of the application for other manufacturing parts with the involvement of similar process steps

Table 6.2: The goal definition for the DDE supported reduction of the turning process costs.

Purpose	Supporting the selection of design parameters in product development
Issue	to reduce manufacturing costs
Object (Process)	during the turning process
Viewpoint	from a product manager's point of view.

and applications, the aim is to support the design parameter definition rather than focus on the manufacturing parameters. Consequently, the goal can be posed as shown in Table 6.2.

6.3.2 Step 2: Supplier-Input-Process-Output-Customer Analysis

The pilot factory of the Technical University of Vienna aims to demonstrate demo scenarios for smart manufacturing [HRT+19]. The definition of product designs is not explicitly provided due to the scope of the pilot factory and therefore, no processes are formalized or defined. Consequently, generic processes for product improvement have to be defined and integrated into the factory. In this case study, a process is developed according to the specifications in the literature [PB13]. Additionally, insights from various company projects are used to mimic a realistic process. The applications related to the artificial process are embedded in the pilot factory. The workshop is conducted with the shop floor manager and an engineer programming the manufacturing floor. Figure 6.10 depicts the corresponding SIPOC as specified in Section 6.2.1.

Figure 6.10: SIPOC of the product optimization process with relevant stakeholders.

6.3.3 Step 3: Analysis of Actual Processes, Data Interfaces and IT Infrastructure

To further detail the SIPOC elaborated in Section 6.3.2, a workshop is conducted at the pilot factory with the same participants as in the SIPOC elaboration. To accomplish a detail process model, first, the shopfloor was visited to achieve a grasp idea of the manufacturing facilities and potentially allow the modeler to ask specific questions. The main process tasks and related tools were identified using the question-answer-talk with the experts and the subjective worker view was modeled using EA. The result of the workshops yields in a EA model describing the current adaptive design process with the related application as illustrated in Figure 6.11. Similar as in the template in Figure 2.5, the three layer are vertically depicted to define a well-structured process. Since the structuring and refinement of the EA model is time-consuming, the modeling expert applies a post-processing session to refine the model. The result is discussed with the expert in a second workshop to validate the model and eliminate missing information, such as the location of the file storage or the interfaces of certain applications. To not overload the model, only the most relevant steps of the process are depicted in Figure 6.11. Particularly, the main design and manufacturing processes and corresponding application with respect to the manufacturing are depicted. The identified applications are SolidWorks for CAD, HyperMill for Computer-Aided manufacturing (CAM) and an PDF report for quality measurement protocols (blue layer) which are both stored on the network drive (green layer).

Figure 6.11: The detail Enterprise Architecture (EA) model with relevant processes and applications.

6.3.4 Step 4: Product Development Value Stream Mapping

The modeling expert, a design expert and a shop floor manager participated in a workshop to identify sources of waste using pdVSM method based on the result of the EA model in Section 6.3.3. To guide the workshop, the modeling expert asks each of the waste types in Figure 6.5 and whether one can relate the posed information waste to a process activity or objective in the model. If one identifies a process with the specific information waste that is actually not modeled but closely related to the current processes, the modeling expert decides whether to add the process to the model or not. The workshop yields to the result in Figure 6.12 with certain assessment flags (purple).

Figure 6.12: Product Development Value Stream Mapping (pdVSM) integration in the detail Enterprise Architecture (EA) model with relevant processes and applications.

6.3.5 Step 5: Waste Failure Mode Effect Analysis

The waste assessment flags in Figure 6.12 indicate sources of information waste. To quantify these information wastes and allow to streamline the development of potential DDE support, a W-FMEA is conducted within a workshop to identify causes and effects as well as assessing whether a use case is relevant and potentially solvable using DE methods. Aligned with the template proposed in Section 6.2.5, a table is prepared before the workshop to guide the assessment. Table 6.3 illustrates the identified sources of waste with additional details that are modeled in EA but not visible in Figure 6.11. Table 6.4 depicts the result of the workshop. The identifier (ID) connects the two tables. In practice, the two tables might also be merged. The result in Table 6.4 shows the priority based on the estimations of the workshop participants. The use case with ID two has a lower detection rate and the severity is higher than the first use case, it is not desired as the most impacting since the occurrence is lower. Consequently, use case one has the highest impact and thus is prioritized to be the first to be implemented.

Table 6.3: Waste-FMEA to assess the causes and effects.

ID	Process	Business Task	ArchiMate Reference	Waste Type	Waste Description
1	Adaptive Design	Detail Design	Eval-DetailProcess-VSM	Inventory	Lack of information; static tolerances lead to increased costs in production
2	Adaptive Design / Manufacturing	Manufacturing Planning / Turning Process	Eval-DetailProcess-VSM	Transport	Information from manufacturing not usable for CAM programming

Table 6.4: Waste-FMEA assessment to prioritize the identified information wastes.

ID	Cause of Waste	Effect of Waste	Occurance	Detection	Severity	WPN	CPN	Priority	AI Supportable
1	Rules are defined once and never updated.	Potentially too narrow tolerances and no case specific improvement	8	9	7	504	504	1	Yes
2	Data is not processed and analyzed so that it is usable in the design.	CAM engineers are not aware of the costs related to a specific decision, e.g., selection of a specific tool causes time consuming toolchanges.	6	10	8	480	480	2	Yes

6.3.6 Step 6: Detailed Data Analysis

With the developed ranking of use cases, the selection of the targeted DDE support can be performed. The selection is not necessarily the highest ranked use case, as the highest ranked use case may be too complex to implement, out of budget or the realization is from a management perspective less important. In this evaluation, the detailed data analysis is similar for both problems. Therefore, no specific use case is selected in this method step. The selection is made in Chapter 7 due to demonstration purpose. With the formalization of the business processes and corresponding software artifacts, a detailed data analysis is performed. With respect to this, SysML BDD and IBD are used to represent the attributes of the data and to connect the attributes among various data formats.

Figure 6.13 depicts the BDD with all involved applications and their respective data formats. The connection of the elements to the PLM can be interpreted as "corporate infrastructure" and is relevant for the modeling in Chapter 8.

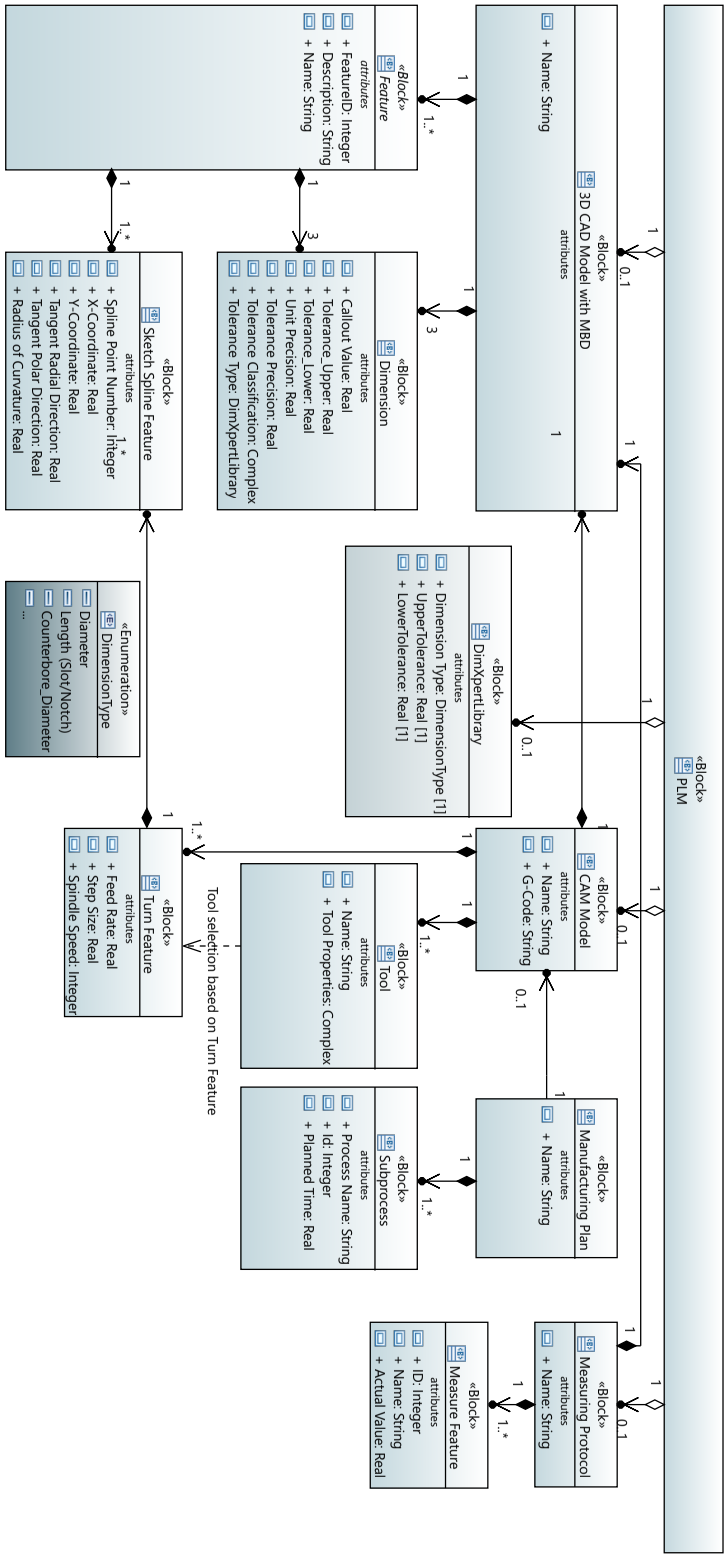


Figure 6.13: Excerpt of a SysML Block Definition Diagram (BDD) with detailed data attributes for evaluation use case.

Based on the PLM in Figure 6.13, an internal data representation is created as shown in Figure 6.14. The *item flows* in the IBD are depicted in blue and magenta respectively as introduced in subsection 6.2.6, indicating whether the relationship is already available or not. It is important to note that an *item flow* from a parent property to an attribute means that all underlying properties have an effect on the child element, such as all properties of *dimensions* have an effect on *turning features* or *tools*.

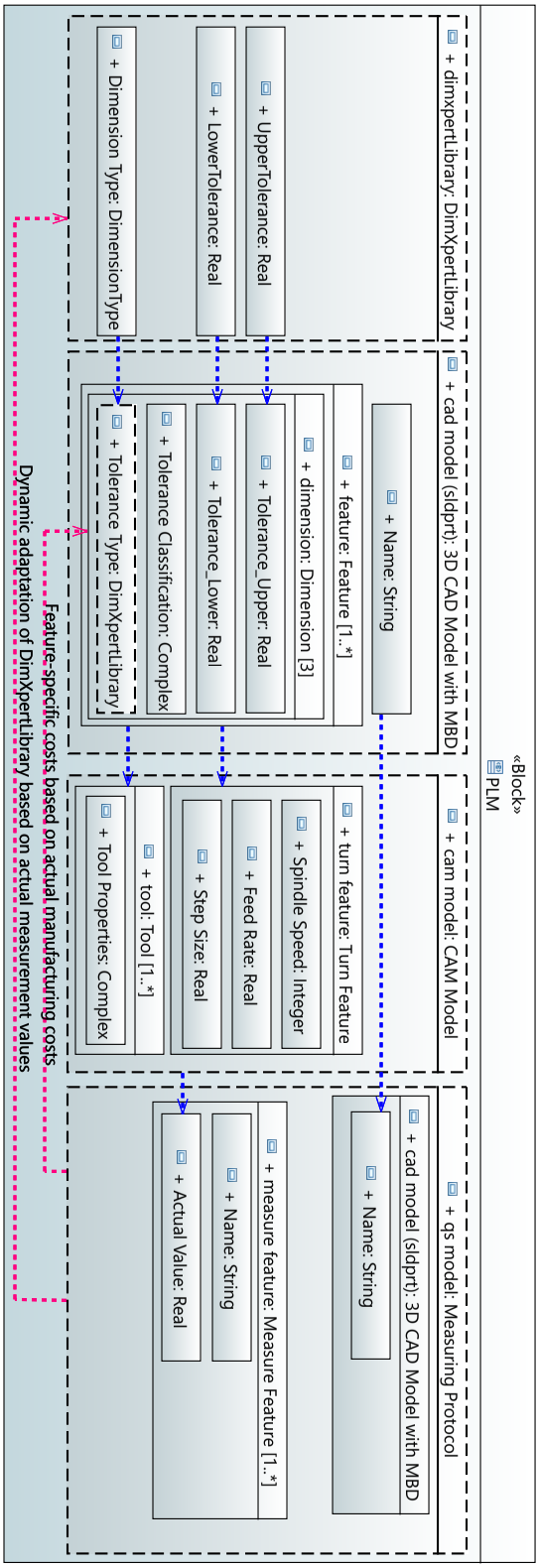


Figure 6.14: Excerpt of a SysML Internal Block Diagram (IBD) with semantic connections of data attributes.

6.4 Discussion

In the following the advantages and disadvantages of the elaborated method are discussed and assessed with respect to future research. In addition, the implications both for industry and research are highlighted.

6.4.1 Validity of the Method

The evaluation of the method has yielded several insights into the benefits and potential pitfalls of the method. In the following, the individual steps of the method are discussed, followed by general implications to evaluate the method comprehensively.

Step 1: The goal definition provided an advantage for streamlining the workshops and following a common plan. The GQM method guided the goal definition, but resulted in very precise goals. This proved to be a disadvantage in the course of the workshop, as the workshop participants kept providing information that further justified the goal, although for some expert a different, more important goal emerged in the course of the workshop. For this reason, the granularity of the goal definition should be examined through a user study, e.g., will better or more use cases be identified if the goal is described in very general terms?

Step 2 & 3: The modeling steps SIPOC and EA are assessed jointly as they are closely related. The participative modeling of the process and IT infrastructure using graphical modeling has proven to be advantageous due to the direct validation and the associated increase in communication between the individual stakeholders. Additionally, the graphical representation encouraged participants to share and discuss their knowledge that further contributes to valid knowledge representation.

However, this led to a somewhat time-consuming evaluation of which processes were appropriate for the goal. Likewise, a discrepancy of processes as well as applications could be observed due to the different levels of knowledge. For this reason, it should be validated in the future whether an expert-separated, sequential workshop would be more advantageous. In this way, the granularity could also be refined step by step.

Step 4 & 5: The evaluation of potentials by means of VSM and FMEA was considered advantageous. In particular, data scientists are assisted in assessing feasibility by evaluating use cases prior to implementation based on high-level specifications, and furthermore, various stakeholders can define the level of importance themselves by evaluating each use case. In contrast to EA modeling, the discussion and joint development or negotiation of a number for the individual criteria was considered beneficial. In this way, the participants were able to justify their assessment once again and a proven evaluation is yielded.

Step 6: The SysML modeling is an advantage for the development of the DE tool and also for the elaborated method in the downstream methods. Nevertheless, the modeling is strongly dependent on the experience of the modeler and thus wrong correlations can be modeled, which can lead to misinterpretations afterwards. Likewise, the level of detail of the modeling is very difficult to estimate, since it is not known what data are actually available and how they can be integrated. Additionally, missing objects to realize a new information flow might be identified but not modeled in this step, e.g., analysis of actual manufacturing costs requires machining time. Nevertheless, the method offers an advantage with respect to the integration and application definition in Chapter 7, since further information is known, which is normally only knowable after some implementation cycles. Future work consists of evaluating the integration of a validation step to prove that an SysML model corresponds to the actual data attributes and to prove the validity of the formalized knowledge.

General: In general, the method appears to be simple to apply in industry, as little prior knowledge is required and the benefit for companies is given. The method also supports the direct validation of knowledge, including the development of possible use cases that might have remained undiscovered. Through the successive revision and refinement of the models, knowledge is continuously enriched and a consistent documentation of the results is given using a model-based representation. In addition, the communication and quantification of results is made possible. Nevertheless, there are some unresolved pitfalls, which have already been discussed in the previous paragraphs. Furthermore, the method is mainly applicable to existing processes and not evaluated for the integration in the development of new products and processes. For use in the product development of a new product, method steps may need to be adapted. Future work consists of evaluating the method in different industries and conducting user studies to exploit the full potential of the method and to determine all advantages and disadvantages.

6.4.2 Implications for Industry

The method contributes to the industry from several perspectives. First, the participative and descriptive approach allows for consideration of different perspectives of the stakeholders involved. As a result, there are fewer unrealistic expectations about the outcome of a project, while understanding of the complexity of tasks associated with the project, such as data distribution, increases. Furthermore, communication increases through the sharing of knowledge during elaboration, which is only made possible by the use of graphical modeling languages. Additionally, the graphical formalization of knowledge allows to evaluate the degree of knowledge formalization in a company and thus contributes to the documentation, validation and standardization of processes during the elaboration of tools.

6.4.3 Implications for Research

A main implication of the method for research consists of the consolidation of knowledge from different scientific communities like data science, engineering and lean management. Furthermore, the consolidation features the development of shared methods among the involved research domains and promises to improve communication. Due to the clear guidance and readily applicability, a transition of methods from academia to the industry is fostered.

6.5 Summary

This chapter contributes with a new method to identify DDE use cases based on existing processes and well-established graphical modeling languages. The method is presented and validated using a case study in a pilot factory with respect to the design and manufacturing of a turned chess figure. The method builds on a systematic decomposition of the associated PLM-processes using SIPOC-analysis and EA-modeling to analyze the business processes and the associated infrastructure. Additionally, a systematic analysis of systems and related data features is conducted using the graphical modeling language SysML. The elaboration of the knowledge formalization is conducted using participative workshops so to directly validate the graphically formalized knowledge with the engineers.

The findings from the evaluation of the method lead to the answer to RQ4.1 *"How to identify relevant data sources in an enterprise for Data-Driven Engineering approaches?"* is that stakeholders need to communicate relevant processes with underlying applications to enable experts to describe data relationships at the attribute level. In this respect, the application of participative workshops to build EA models reflecting business processes and related IT applications appears beneficial.

Based on the knowledge decomposition using EA, the identification of information waste using pdVSM and W-FMEA is conducted. The pdVSM contributes by categorization and guidance for the identification of information waste. Based thereon, the W-FMEA allows to assess the identified information waste and further describe the potentials with its impact. The assessment of the feasibility by an data science expert allows to categorize the potential use cases and further select the most promising approach.

Consequently, the answer to RQ4.2 "How to identify and assess potentials for Data-Driven Engineering based on existing processes and IT infrastructure?" is that stakeholders have to be guided to apply quantitative assessment methods so that the potential use cases are validated with respect to the impact based on the knowledge of involved disciplines.

Due to the participative fashion of the workshops, knowledge of various experts is captured and validated during the modeling. The result of the modeling acts as a documentation and the communication is fostered due to the shared knowledge acquisition. Hence, this method establishes a shared business and data understanding required to successfully identify and implement DDE in industry.

Therefore, RQ4.3 "How to support communication between the involved stakeholders and foster knowledge validation and documentation?" can be answered with the integration of participative elaboration of knowledge in a graphical modeling environment so that a domain independent representation of knowledge is given and the validation can be made implicitly during the workshops. Additionally, the graphical modeling acts as a documentation without extra effort.

Finally, RQ4.4 "How to enable the integration of various involved stakeholders in the investigation of Data-Driven Engineering use cases?" is implicitly answered by the present workshops, which are conducted in the context of participative modeling workshops and domain-independent representation using graphical models.

CHAPTER 7

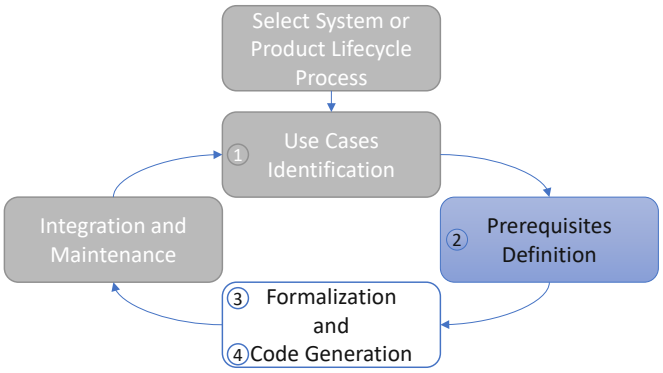
Integrating Data-Driven Engineering into Actual Processes

This chapter addresses the research objective ② in Figure 7.1 related to defining prerequisites and integrating automated data collection. Figure 7.1 depicts the research implications, such as the definition of required data associations among processes, automated data collection mechanisms, and the integration of the intended DE tool support in actual processes. Additionally, integrating data collection mechanisms is elaborated, to allow the collection of data on a fine-grained level and enabling to interconnect data, e.g., a single manufacturing part with various production steps must be traceable to enable the merging of the generated data. The integration of data collection mechanisms and the integration of the DE application are part of the integration of DDE into actual processes.

The requirements definition and prerequisites determination are based on the use case identified in Chapter 6. The method in Chapter 6 supports analyzing current processes and applications. However, existing processes and applications might not support sufficient data collection interfaces or do not represent the relationships between the data for the desired use case. In this respect, an adaptation of currently implemented processes might be necessary.

This method integrates the targeted DE application and the associated definition of preconditions such as new data interfaces. The literature supports the definition of target architectures by describing that changing processes significantly support the chances of successful AI projects [RKK+20], respectively DE projects.

With respect to the defined method goals, the following overall RQ is answered by step 2 of the developed method:



Objectives	Research Implications	Challenges	Realization
1 Identify use cases for Data-Driven Engineering	• Identify relevant business processes • Identify data attributes and data interfaces • Identify potential in actual processes	• Diversity of processes • Distributed knowledge sources • Ambiguity of communication	Chapter 6
2 Define prerequisites for integrating Data-Driven Engineering	• Define data associations • Define automatic data collection • Integrate desired tool in actual processes	• Diversity of processes • Legacy hardware support • Diverse data level of detail	Chapter 7
3 Formalize Data Engineering tasks using SysML	• Formalize validated knowledge • Reuse of knowledge from MBSE • Enhanced interdisciplinary communication	• Ambiguous modeling • Complex formalization • Level of detail of formalization	Chapter 8
4 Automatic Code Generation for Data Engineering based on SysML	• Reuse validated and formalized knowledge • Enable flexibility and maintainability	• Decomposition of knowledge • Generation of executable code	Chapter 9

Figure 7.1: Overview of research objectives, implications and challenges addressed in Chapter 7.

RQ5 What are the prerequisites in a company so that manufacturing data can be leveraged for Data-Driven Engineering?

Subsequently, the following detailed RQs have been identified:

RQ5.1 What are the requirements to enable the traceability of data relationships where the respective data are gathered in different stages of the product lifecycle?

RQ5.2 What data collection mechanisms and architectures can be used to automate data gathering required for rapid iterations in Data-Driven Engineering development?

RQ5.3 What means of graphical modeling can facilitate the integration of data collection mechanisms into current business processes and applications?

RQ5.4 What means of graphical modeling can facilitate the integration of Data-Driven Engineering applications into current business processes and applications?

Based on the identified RQs, related work for the systematic data collection is introduced. Second, the elaborated method for identifying and defining prerequisites is presented, which integrates the automatic data collection mechanism and the intended application in actual processes. Furthermore, an evaluation based on the use case described in Section 5.2.1 and Chapter 6 is given. Finally, the results are discussed and the answers to the posed RQs are given.

A selection of text, figures and tables within this chapter is based on the publications in box „Publications 5: Integrating Data-Driven Engineering into Actual Processes“:

Publications 5: Integrating Data-Driven Engineering into Actual Processes

[RMR22] S. Rädler, J. Mangler, and E. Rigger, “Requirements for Manufacturing Data Collection to Enable Data-Driven Design,” Procedia CIRP, vol. 112, pp. 232–237, Jan. 2022, doi: 10.1016/j.procir.2022.09.077.
[RR20] S. Rädler and E. Rigger, “Participative Method to Identify Data-Driven Design Use Cases,” in Product Lifecycle Management Enabling Smart X, vol. 594. Cham: Springer International Publishing, 2020, pp. 680–694, doi: 10.1007/978-3-030-62807-9_54.

7.1 Related Work and Research Gaps

Integrating DDE into existing processes requires systematic data collection to enable rapid iteration cycles, support the traceability of relationships between data that might be distributed among multiple processes, and enable further improvement of the desired DE application based on up-to-date data.

In this respect, recent trends such as Industry 4.0 [AH17, MV18, MYK+09] are pushing enterprises towards the implementation of smart factories to enable the collection of data from Industrial Internet of Things (IIOT) [Gil16] devices during the manufacturing process [CWS+18]. Particularly, the integration of CPS [WTO15] and CPPS [Mon14] are an enabler for Industry 4.0 [LO20]. However, challenges regarding the collection of data from multiple CPPS across various processes of the PLM with multiple levels of detail remain open [Ger17]. In this respect, the orchestration of data collection and related processes appears beneficial [GMM+22]. Existing solutions for the orchestration of data collection typically depend on the hierarchy level in the automation pyramid [DIN03]. For orchestration of processes on control level, graphical languages are used as a basis. Graphchart [Årz96] is one example for a graphical language for sequential supervisory control of systems. It is based on Sequential Function Charts (SFC), one of the programming languages for Programmable Logic Controller (PLC) described in IEC 61131-3 [DIN03]. Other languages like BPMN or Business Process Execution Language (BPEL) are used on higher organization levels. For lower levels, traditional programming is prevalent. One requirement of CPS based automation is the need to orchestrate different services, which

is implemented, e.g., by the Manufacturing Service Bus (MSB) [Min13], a specialization of the Enterprise Service Bus (ESB). Currently, only one implementation [SHS+18] exists, and the MSB requires extensive tooling for managing its set-up and evolution due to the tight semantic coupling it imposes on its components.

The BPMN process orchestration-based approach tries to balance the rigidly structured and monolithic implementation of traditional approaches based on the automation pyramid, and the CPS approach with an interacting bag of adapter and translation services. Approaches such as centurio.work [MPRE19, PM18] and the open source tool Cloud Process Execution Engine (CPEE) [MR14, MR22], build centralized interaction models and fully decoupled adapters. Data formats such as IEEE eXtensible Event Stream (XES) [AVDS+17, GMM+22], allows for the utilization of common data analysis tools such as Celonis [VGM+17].

In this work, the focus is put on the integration of automated data collection using CPEE. The core reasons for selecting CPEE is the lightweight integration of any data source using Representational State Transfer (REST) services and the graphical modeling using BPMN, which is similar to process modeling using EA, as used in the previous method. Figures 7.2 depict an executable process's BPMN notation. Figure 7.2a shows the steps involved from the viewpoint of the cell orchestration, e.g., how the machines and the robot interact with each other (further called Cell View). As shown in the Cell View, task *a8* spawns the Part View as a sub-process, which then runs in parallel. The Part View is depicted in Figure 7.2b and involves steps required to produce a single part. The two sample views interact through the exchange of signals, e.g., tasks *a4* and *a6* in Figure 7.2b. With the separation of the views, each process or sub-process data is collected in its own logging artifact (e.g., a log-file). Thus, the Part View contains all the machining and measuring data for exactly one part, the information does NOT have to be collected from individual data tanks connected to the machines themselves. Also, while the production and measurement of different parts run in parallel in the Cell View, in the Part View, everything is a sequence. All the measuring and machining data collected during the production is pushed to a logging facility, compliant with the IEEE XES standard [AVDS+17] facility. +

(a) BPMN for the cell orchestration; automatic NC-program generation, robot handling of parts and automatic keyence measuring.

(b) BPMN for producing a work piece.

Figure 7.2: BPMN notation of an execution process of CPEE.

7.2 Method

This section presents a new method to elaborate prerequisites for the application and integration of DDE into existing processes. The method is based on the findings of Chapter 6 and contributes by utilizing means of EA to define the integration of the desired DE application into existing processes and applications. Additionally, the integration of automated data collection mechanisms is modeled using EA. Furthermore, desired data relationships are modeled by extending the existing SysML model. In the following, the different steps of the developed method are introduced.

7.2.1 Step 1: Define Goal, Requirements and Assumptions

The first step of the introduced method is to define the desired *goal* of the DE application aligned with *assumptions* and *requirements* that have to be made to enable the implementation of DDE.

The *goal* is derived or refined from the target definition and the identified use case in Chapter 6. Again, the goal is formalized using GQM to support understanding and structuring of the goal definition. The definition of the goal can also be modeled using EA means of requirements modeling [TIK+21]. The purpose of the refined goal is to update contributing experts and streamline the elaboration and integration of the intended DDE approach into existing processes.

Assumptions aim to narrow down the scope of the investigation and allow to define prerequisites so that the implementation potentially can be realized, e.g., the *task library* in the sample in Figure 6.8 is only derivable from existing *output objects*, if templates are defined and applied to the output object. Based on the assumptions, the project scope can be reduced and potential risks can be assessed a priori, e.g., by applying FMEA.

Requirements are categorized as functional and non-functional requirements. Functional requirements define what the system does, whereas non-functional requirements describe global requirements, such as costs, performance, reliability etc. [AFG+21, CNYM12]. The requirements are hierarchically organized and modeled using ArchiMate software. A main reason for the implementation using ArchiMate is the potential to link requirements in the EA model with other elements and further allow for evaluation during the implementation. Figure 7.3 depicts a sample hierarchy for the organization of the requirements.

Further details on the definition of non-functional requirements is well defined in literature [AFG+21].

7.2.2 Step 2: Identify Data and Interface Prerequisites

The second step of the method aims to assess the traceability of data attributes. Missing traceability harms data linking and is a core issue in manufacturing data analytics use cases [BB22]. Traceability is the "ability to access any or all information relating to that 'which' is under consideration, throughout its entire lifecycle, by means of recorded

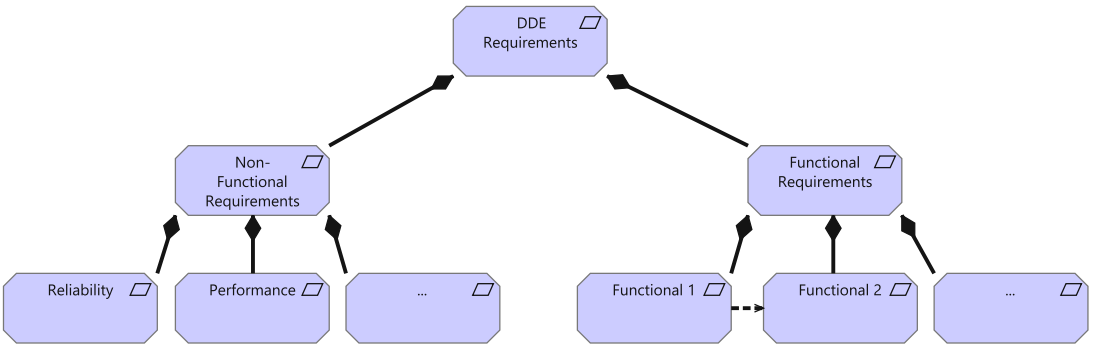


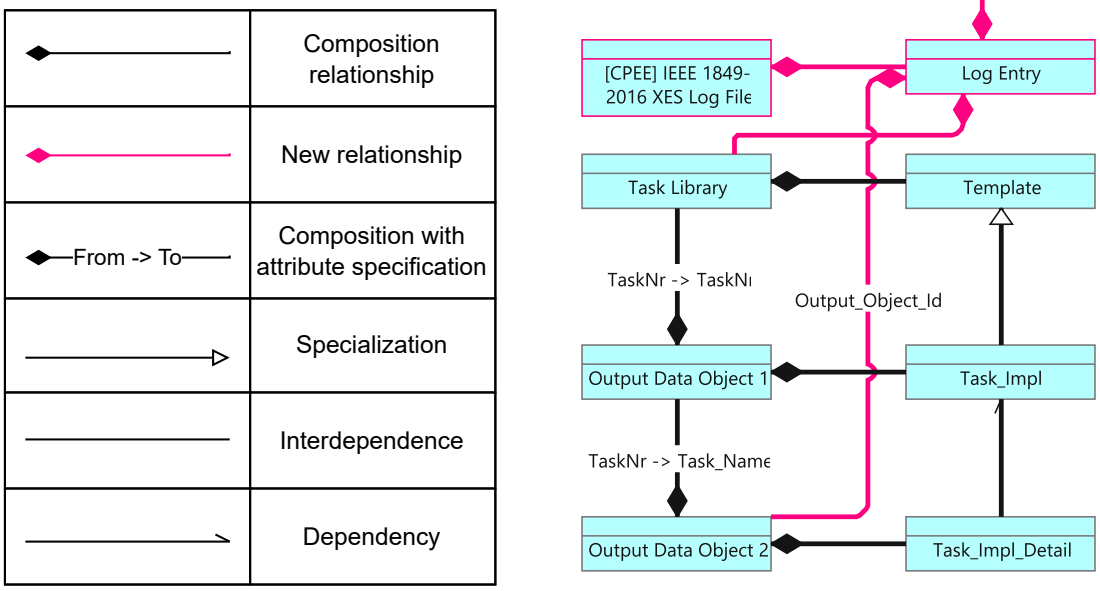
Figure 7.3: Template of hierarchical organized requirements.

identifications" [OB13]. Consequently, in this step, all identified and for the use case potentially relevant data sources are assessed with respect to forward and backward traceability. It is sufficient if one attribute of a data source enables the identification of predecessor and successor data objects. All other attributes can be traced through the association within a data source, similar to foreign key relationships in SQL.

The proposed approach is based on EA and is depicted in Figure 7.4b. The example of Figure 7.4b is based on Section 6.2.

The traceability model can be read as follows: From top to bottom, the upstream and downstream process objects are presented in a causal process aligned order. Between data objects, composition relationships indicate existing traceability, labeled with attributes enabling the tracing. If downstream objects enable to trace backward, compositions are modeled from bottom to top, respectively. Horizontal composition relationships depicts more fine-grained data necessary for tracing, e.g., a CAD file consists of CAD features. Dependencies between data objects are represented using association connections. Particularly, directed and undirected associations are applicable, e.g., *Task_Impl_Detail* is dependent on *Task_Impl* but not vice-versa. Furthermore, specializations are used to indicate inheritance or more detailed representation. Figure 7.4a depicts the different types of modeling elements.

If specific data attributes are not traceable forwards or backward, the integration of a new data object allowing to interconnect the data object needs to be desired, e.g., *Input Data Object* in Figure 7.4b is not traceable among the process, which leads to the necessity to integrate a new data object or attribute, to allow tracing. Particularly, integrating a data collection mechanism that links the data is potentially necessary. The newly introduced relationships are highlighted using *magenta* color code. The integration of potentially integrated data collection mechanism is depicted in the following method step.



(a) Traceability modeling elements.

(b) Sample EA view to assess traceability.

Figure 7.4: Traceability elements and sample application.

7.2.3 Step 3: Integration of Data Collection Mechanisms

Automated data collection mechanisms are required to link data among processes and in-time data processing. For this, integrating a framework with capabilities to integrate arbitrary sources with various interfaces is necessary. The framework of choice is the lightweight orchestration framework CPEE, shown in the literature to be beneficial for industrial data collection [PMRE21, RMR22]. The integration of CPEE potentially replaces and changes existing data objects to allow the integration. Consequently, related business objects must be updated to achieve a valid EA model. Therefore, the integration of CPEE into actual processes requires the modeling of a target architecture with adaptation of the existing EA modeling. In this respect, a template is created with a standard representation of CPEE, allowing to improve modeling performance. Figure 7.5 depicts the framework template with related artifacts.

Since changes from the current model to the target model cannot be highlighted with EA, a transition model is created, depicting grayed-out applications and the new CPEE integration with connections to the process (magenta) and serving connections between the old application and CPEE to indicate replacement as proposed in literature [BRJ17]. Figure 7.6 illustrates an example of the transition with magenta-colored frames around new elements.

In addition to the transition model, integrating the CPEE framework in the actual

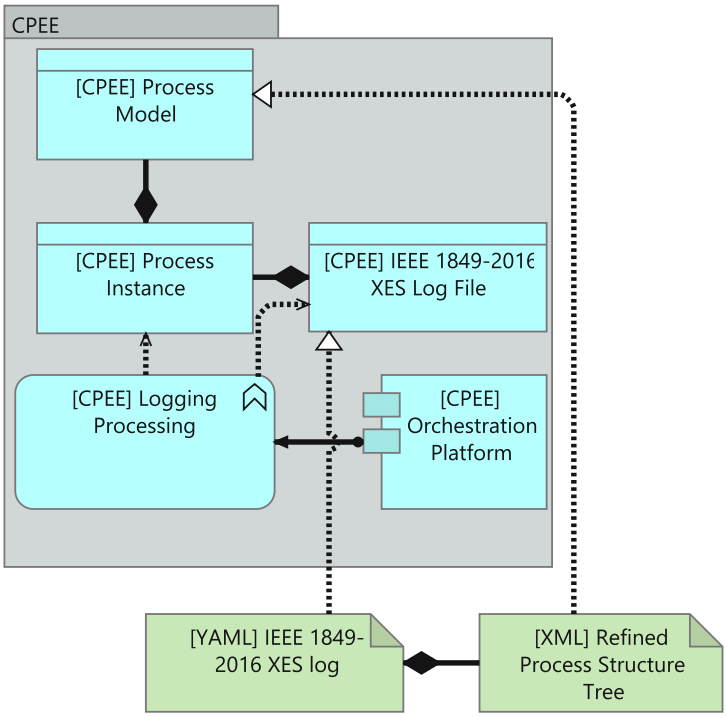


Figure 7.5: Reference model depicting the CPEE application used to integrate various data sources automatically.

processes requires to be modeled. Figure 7.12 depicts a sample of integrating CPEE into the EA model. Changes are highlighted by magenta-colored frames, too.

The result of this step comprises the integration of CPEE into existing processes and enables the collection of all necessary data for in-time data processing.

7.2.4 Step 4: Update of Semantic Connections

The penultimate step of the method aims to incorporate the data collection mechanism integrated in the previous step into the SysML model elaborated in Section 6.3.6. Semantic links are changed, and a target data representation is presented at an attribute level. Aligned with the EA model, changes in the SysML model are graphically highlighted using *magenta* color code.

Figure 7.7 illustrates an example integration of the CPEE workflow engine into the sample from Figure 6.7.

Similarly, Figure 7.8 illustrates the magenta colored changes in the semantic relationships based on Figure 7.8.

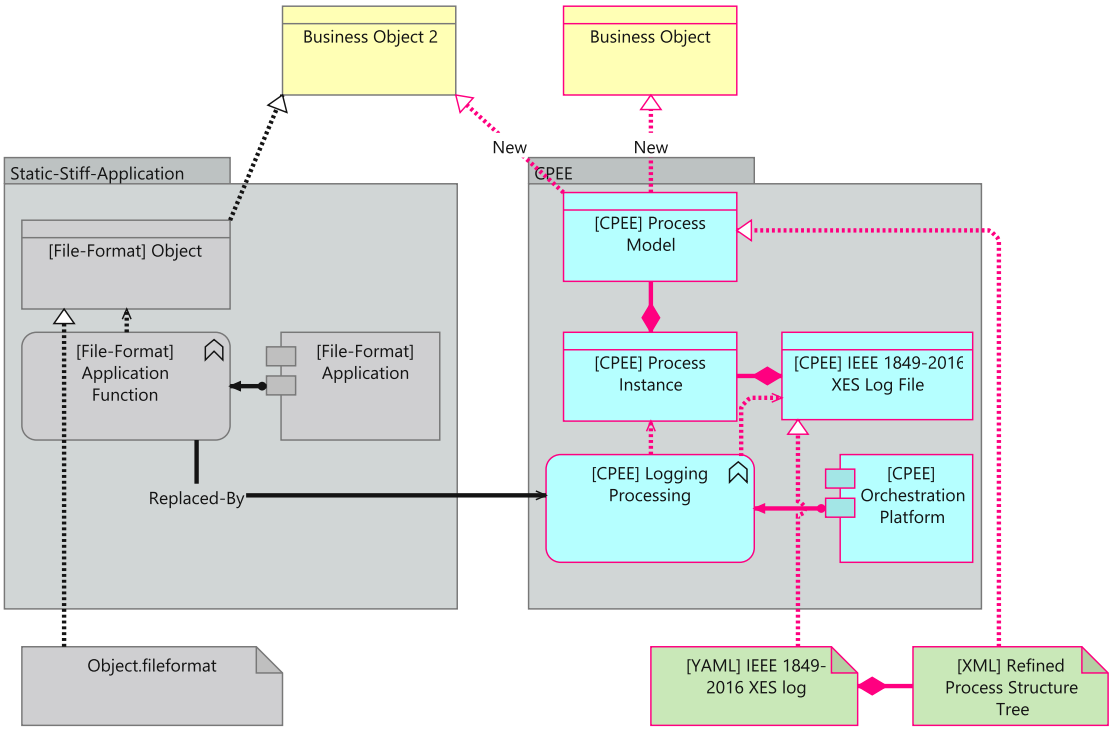


Figure 7.6: Sample of model transition model indicating replaced and extended models.

Figure 7.7: Sample of updated SysML Block Definition Diagram (BDD) integrating automated data collection mechanism.

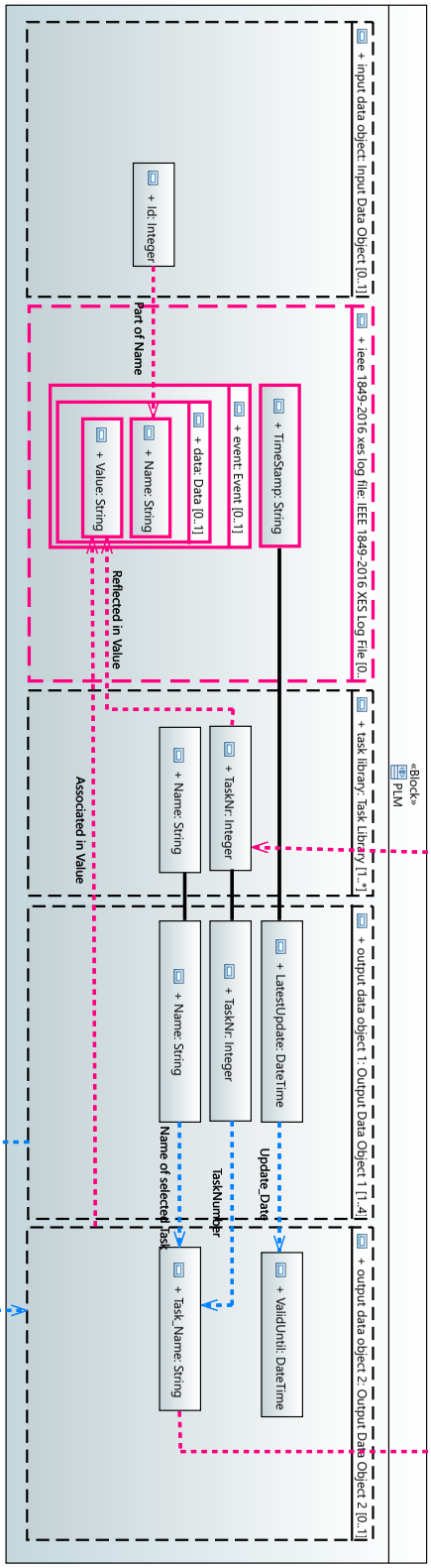


Figure 7.8: Sample of updated item flows using SysML Internal Block Diagram (IBD).

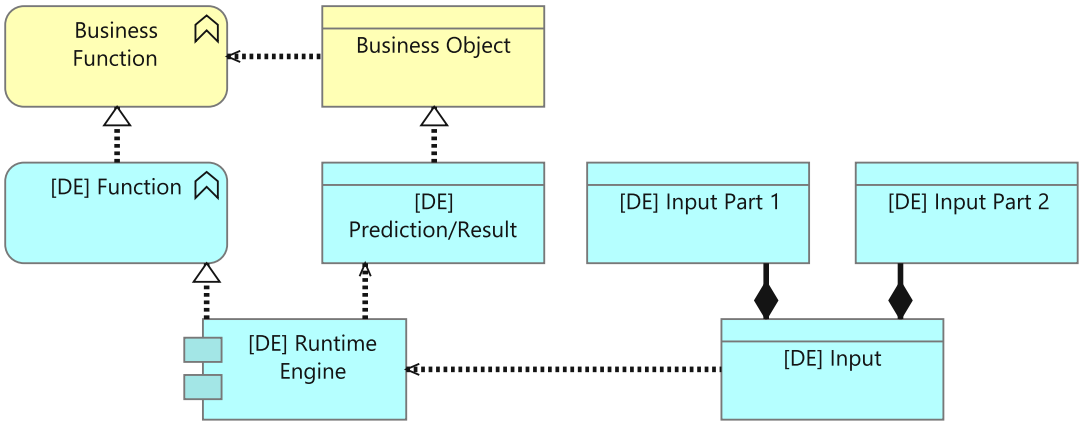


Figure 7.9: Template for the modeling of DE applications.

7.2.5 Step 5: Integrate Data-Driven Engineering Application

Based on the newly accessible data collected through an automated data collection mechanism, all data required for the implementation of the DE application is available in the required contexts so that the integration of the DE application can be carried out.

The following step comprises modeling and integrating the DE application into existing business processes and IT applications. To support the implementation, Figure 7.9 depicts a template highlighting necessary information for integrating the DE application. On the right of the template, the input for the DE tool is defined. Particularly, arbitrary data objects from the existing EA model necessary for the DE approach are connected with the input data object using *part of* associations. The data object with the result/prediction is linked to the business object or function it is desired to be beneficial.

In literature, approaches are describing the modeling of DE (AI) more precisely using EA [TINI21]. In practice, however, it turned out that the specific details did not support the implementation as much as the additional effort would have been worth. Still, the approach from literature can be applied if more extensive and detailed modeling is necessary.

The result of this step is the To-Be integration architecture of the DE application.

7.2.6 Step 6: Quality Assurance

The application of quality assurance in the modeling of To-Be architecture is a complex task and requires guidance. Therefore, this method integrates a checklist for To-Be architecture assessment based on literature [NAR+17]. The actual checklist for the method is an adapted version and is based on the findings from the evaluation of the method. Still, the checklist is not comprehensive and requires to evolve form project to project. Table 7.1 depicts the actual checklist.

Table 7.1: Checklist for the integration of DDE into existing processes.

- ☐ Is the DE application integrated into the data structure?
- ☐ Is the DE application integrated into the business process with sufficient description?
- ☐ Is the data required for the DE application collected using automated mechanisms?
- ☐ Is all data necessary data interconnected with the DE application?
- ☐ Is all data semantically linked to allow traceability?
- ☐ Is the goal of the DE application understandable without further explanation?
- ☐ Are all application function transitions documented and colored in the model?
- ☐ Is the integration of the DE application aligned with the requirements and assumptions?

Table 7.2: The goal definition for the DDE supported reduction of the turning process costs.

Purpose	Support to make informed decisions for tolerance-related costs of design features
Issue	to reduce manufacturing costs
Object (Process)	based on design decisions
Viewpoint	from the designer's perspective.

If all items of the checklist are sufficiently fulfilled, the elaboration of the prerequisites for the implementation of the application is completed and the implementation, as suggested in Chapter 8, may start.

7.3 Evaluation

The newly introduced method is validated in this section using the case study presented in Section 5.2.1. The evaluation builds upon the results presented in Section 6.3.

In the following, each step of the method is applied separately for evaluation purpose.

7.3.1 Step 1: Define Goal, Requirements and Assumptions

Table 7.2 depicts the refined goal of the aimed DDE support. It builds upon the GQM method aligned goal in Table 6.2.

Following the goal definition, the aligned requirements are depicted in Figure 7.10.

Figure 7.10: Requirements on the intended DDE approach.

To enable implementation, several assumptions are defined with domain experts. The assumptions support the understanding of domain relevant knowledge from the perspective of DE experts.

In the following, the assumptions for the DDE approach:

- The design of a turned part is composed of several CAD features
- CAD features have fixed characteristics that cannot be changed, e.g., depth, width, diameter, etc.
- Each CAD feature has an identifier that does not change among parts
- Each CAD feature is created through a predefined sequence of manufacturing steps, e.g., a thread is first drilled, then phased and finally machined by a tap
- The sequence of manufacturing steps are reflected in CAM and depend on the fix characteristics of a CAD feature as well as on tolerance-specific aspects, e.g., deep holes require two drilling steps or a hole with fitting requires additional steps
- The sequence of manufacturing steps is always executed in the same order
- A CAD feature can be mapped to a quality assurance feature using a unique identifier

7.3.2 Step 2: Identify Data and Interface Prerequisites

Based on the requirements in the previous step, the traceability analysis of the data artifacts is conducted. Figure 7.11 depicts the data traceability based on Figure 7.4.

The data objects framed with magenta indicate additional data objects, required for linking CAD features with the actual values of the quality assurance and the actual machining time. Since the machining time is yet not measured in the implemented process, an additional integration of the processing timestamp is necessary to enable time tracking. The following step illustrates the integration of the CPEE with the collection of machining time.

Figure 7.11: Traceability analysis from CAD features to quality assurance based on the existing process.

7.3.3 Step 3: Integration of Data Collection Mechanism

The integration of CPEE into existing processes is defined using EA. Figure 7.5 depicts the template of CPEE which is integrated in existing processes as illustrated in Figure 7.12.

To further reduce the scope of information, Figure 7.13 depicts only changed and added elements. For example, the production plan will in future be displayed via workflows instead of via a PDF file. Additionally, the results of the measurement data will in future be collected by the workflow engine and not in the form of an excel output. Thus, the collected data can be directly processed and linked to the manufacturing times of a feature.

With the results of this step, the To-Be architecture of the data collection mechanism is defined and necessary additional data artifacts to allow the connection of data are integrated. Following this, the actual SysML model with relationships is updated to align with the integrated data collection mechanism.

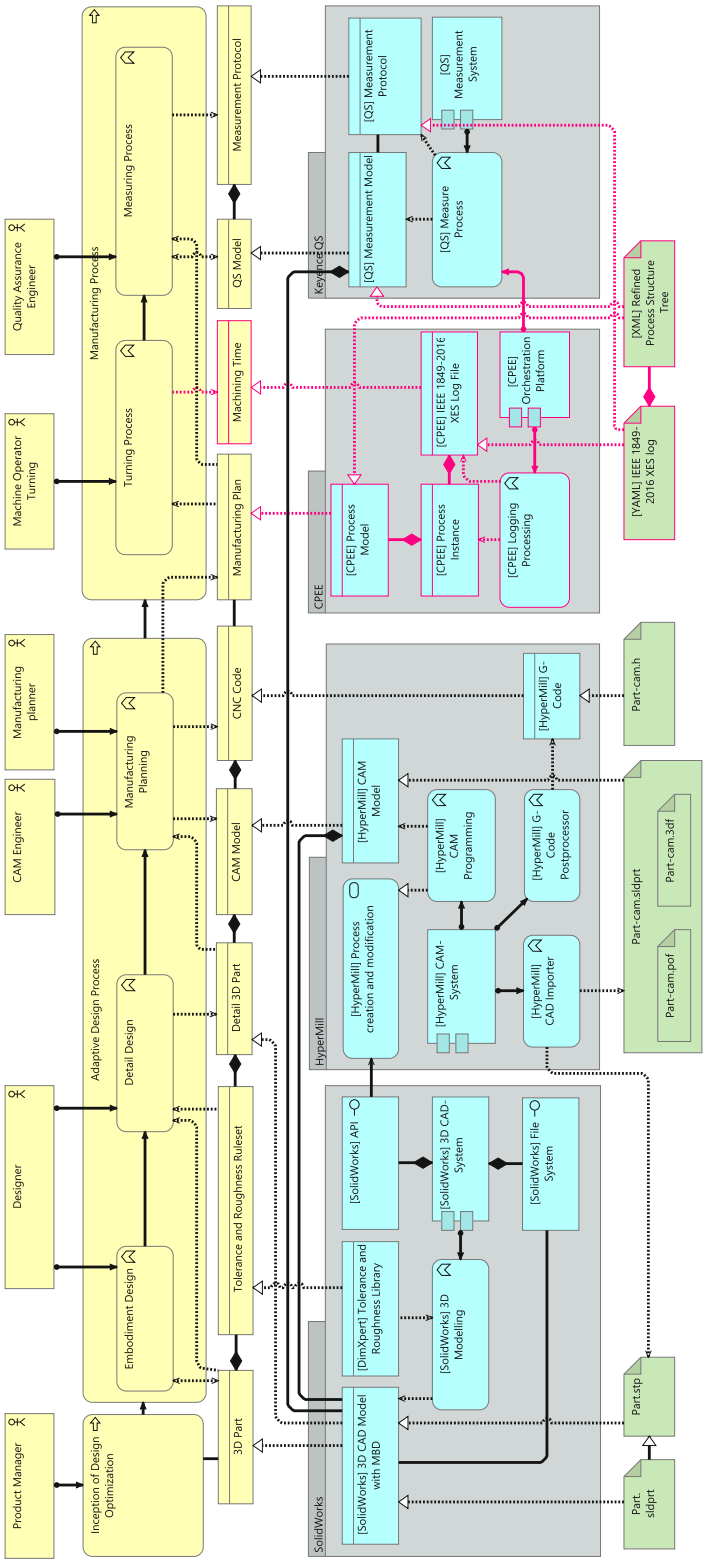


Figure 7.12: Integration of data collection mechanisms and additional required data objects.

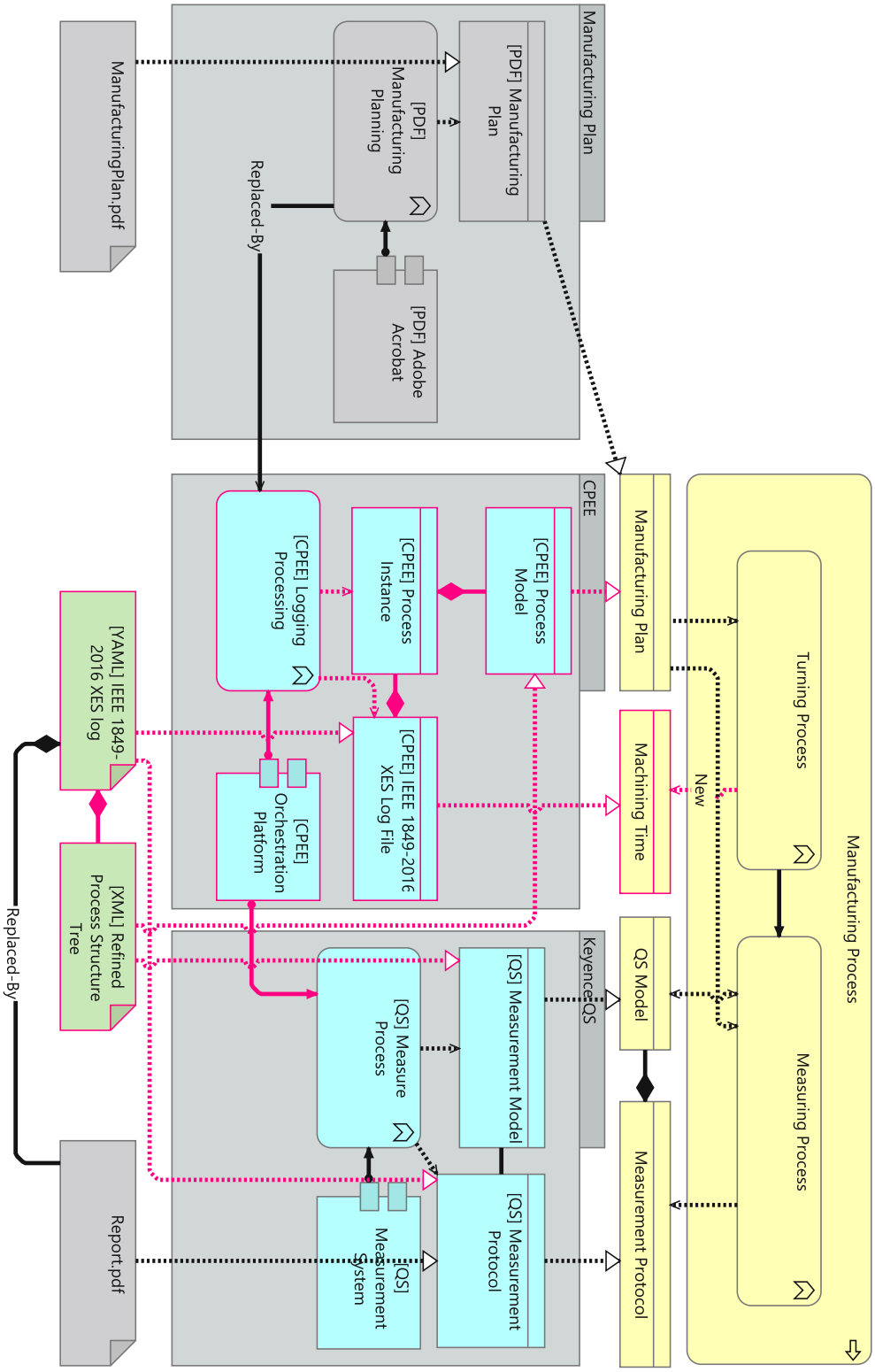


Figure 7.13: Transition model to indicate which applications have been replaced and newly integrated.

7.3.4 Step 4: Update of Semantic Connections

The update of data relationships is intended to support the implementation of the DE application. In this respect, the actual model from Section 6.3.6 is extended.

Figure 7.14 depicts the updated PLM with integrating the data collection mechanism. Parts without magenta color code are not changed.

Figure 7.15 illustrates the newly elaborated relationships on an attribute level. Remarkably, there is a part of the element flow at the bottom right indicating that the *Value* property of the data collection mechanism contains, but is not limited to, the *Actual Value* of the measurement.

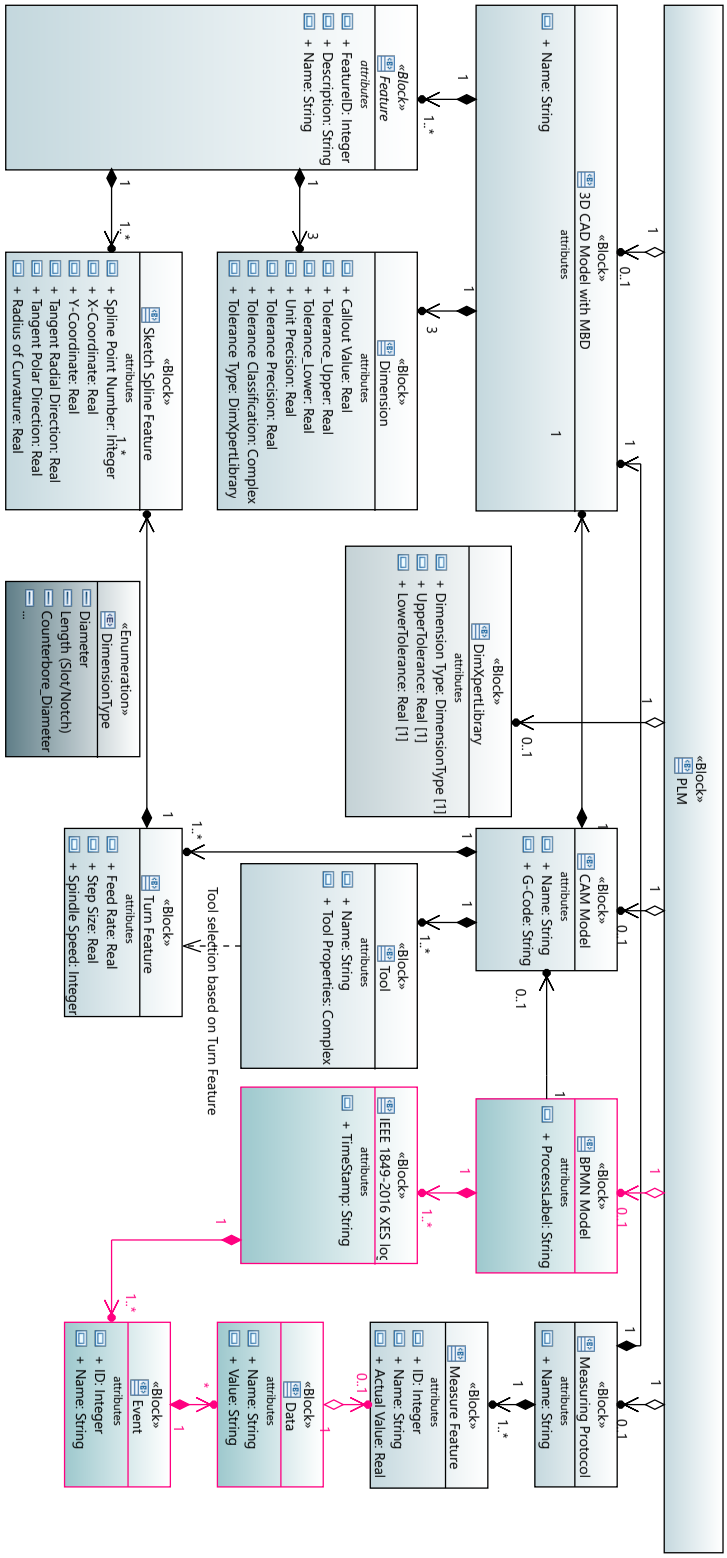


Figure 7.14: SysML BDD with updated data sources and integration of automated data collection mechanism.

Figure 7.15: SysML IBD with updated relationships between the data artifacts with integration of automated data collection mechanism.

With the updated SysML model representing traceability information between data, all prerequisites are defined for the implementation of the DE application.

7.3.5 Step 5: Integrate Data-Driven Engineering Application

With the integration of the CPEE workflow engine, data is interconnected and readily available for further integration in various use cases for DDE. Still, the integration of the intended DE application into existing processes and the data integrated into the DE application are not explicitly depicted in the models. In this respect, Figure 7.16 illustrates the integration of the defined template in Figure 7.9 into the To-Be EA model representation. Particularly, the defined input and output objects of the application are integrated and aligned with a business object that is either created for this purpose, as shown in Figure 7.9, or an existing one is reused. In case the business object is newly created, it is linked to a business function to enable integration into an existing process.

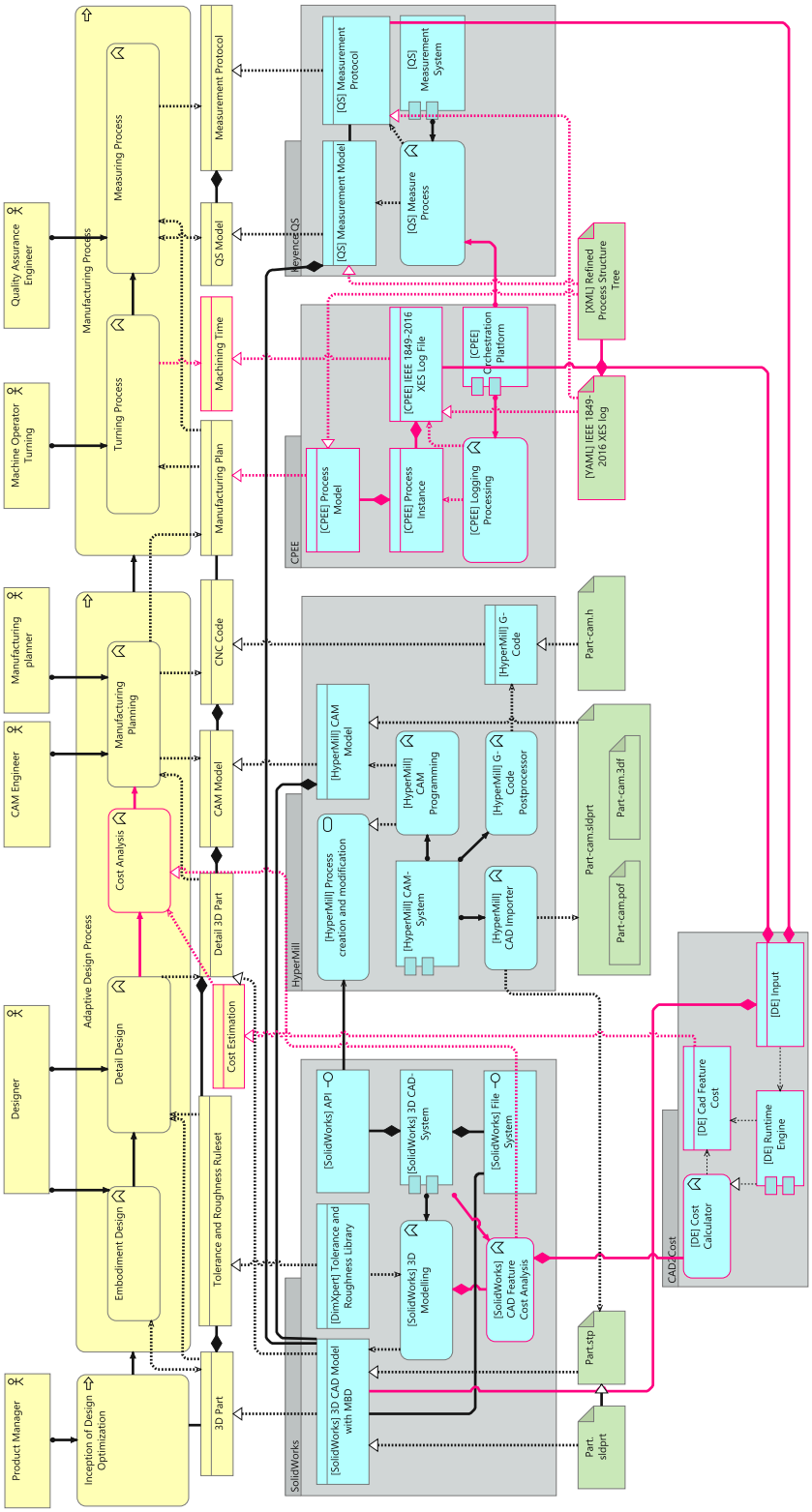


Figure 7.16: Integration of the new DE application into existing processes and applications.

This step wraps up the modeling of the To-Be architecture and initiates the quality assurance process.

7.3.6 Step 6: Quality Assurance

The final step of the method presented is quality assurance based on the data in Table 7.1. The entire checklist is assessed based on the models elaborated during the previous steps and no other sources of knowledge are required. Additionally, another reviewer assesses some items on the checklist to avoid bias, e.g., "Is the goal of the DE application understandable without further explanation?". With the review of all quality assurance features, the development of prerequisites is completed. Further support for the implementation of the DE application is given by the MDE method presented in Chapter 8.

7.4 Discussion

This section critically discusses the newly introduced method. First, the validity of the method itself is discussed in terms of potentials and pitfalls. At the end of the validity assessment, future work is presented. Finally, implications for industry and research are highlighted to show the method's contribution.

7.4.1 Validity of the Method

The evaluation of the method and the method development progress yielded insights into the benefits and pitfalls of the method. In the following, each step is separately discussed, followed by a general discussion with future work.

Step 1: Adjusting the originally defined goal of a use case supports the refinement of the scope definition and the achievement of a common, updated understanding of the planned DDE integration. Aligned with defined requirements and assumptions, this step enables to derive metrics to assess the success of the resulting implementation. A hierarchical definition of requirements allows templating necessary dimensions of requirements, while the modeling using ArchiMate enables connecting the requirements with related processes and achieved architecture. However, the requirements itself are not traceable in the current setup. Therefore, the integration of application lifecycle management systems such as Polarion[1] or Jira[2] might be necessary.

Same for the definition of assumptions. Currently, assumptions are documented by bullet points in the text. Therefore, the traceability of assumptions or the linking of assumptions is not possible. Future work requires elaborating a method to integrate requirements and assumptions in a model or framework that enables traceability.

[1] https://polarion.plm.automation.siemens.com/
[2] https://www.atlassian.com/de/software/jira

Step 2: The elaboration of traceability is beneficial to define interconnections on a data level. The application of this method step initially was made with SysML (see [RMR22]). However, it turned out that the representation using EA is easier for non-software engineers and thus promotes communication and enables to validate traceability. Additionally, the traceability on an attribute level with explicit definitions of influences turned out to be complex with several methods and data objects. Therefore, in this updated method, the modeling is based on EA to improve modeling performance and applicability in practice. Still, future work is required to extend and improve the approach, e.g., the modeling of relationships between data entities might be to high-level. Particularly, certain behaviors besides dependencies, subgroupings, or derivations, such as information influence, i.e., a certain attribute of an object influences the decisions of another attribute, might be necessary. In addition, the modeling of traceability associations between objects is modeled similarly to *part of* associations, which may lead to misinterpretation.

Step 3: The integration of a visual representation of changed elements is advantageous to highlight changes in the EA model without extensive effort. Additionally, omitting unchanged elements to visualize a transition model improves the focus of the integration without a heavy model. The adaptation of the method described in the literature [BRJ17] with colored frames around added elements supports the distinction between existing, unchanged elements relevant for understanding, and newly integrated elements. Furthermore, integrating an automated data collection mechanism is considered beneficial for analyzing data in-time. However, the integration of data collection mechanisms can also be overloaded, resulting in large data lakes with terabytes of unused data. In this regard, the method needs improvement to support the selection of the data collected by the automated mechanisms.

Step 4: The integration of To-Be representation of data relationship supports to guide the implementation of the DE application. In particular, it enables the pre-selection of input variables and also supports the evaluation of the DE tool based on the formal representation of data connections. Additionally, an increase in the understanding of the DE application is expected. Nevertheless, the model is not entirely unambiguous. For example, the representation does not clearly show why the connection between two attributes exists. Likewise, the development of the model is not comprehensible, which is harmful to the documentation. In terms of improved documentation, future work is to develop enriched semantics that provides further details about the relationships and support for subsequent implementation.

Step 5: The contribution of this method step is two-fold. First, it validates the previous step by linking relevant input data to the DE application. Second, it represents the integration of the DE application into the existing processes and data objects and allows (1) documenting the integration, (2) distribution of the required implementation effort, and (3) simulate the integration, thus enabling the identification of potential pitfalls of the application before the integration started. However, integrating the DE

application with data objects is again done at a data level and not at an attribute level. Therefore, validation of data and potentially missing data is not guaranteed. Additionally, a model with omitted elements could be useful not to overload the model and improve understandability, maintainability and validity.

Step 6: A first validation of the integrated automated data collection mechanism is given with the integration of the DE application in the previous step. Still, applying a checklist is beneficial to evaluate whether all necessary steps are comprehensively completed. In practice, the checklist proves to be a useful tool, even though it has some pitfalls. The key pitfall is that the granularity of the checklist elements is too high to prove that all steps are correctly completed. Additionally, the checklist does not evaluate the content of the various steps. Therefore, the integration of a participative workshop with relevant stakeholders is proposed for validity checking, as done in the first method, see Chapter 6. By involving workshops, several stakeholders who are not familiar with the development have to prove that the To-Be architecture is promising, and everyone can judge whether certain aspects are still missing or lack in detail.

General and Future Work: Same as for the previous method in Chapter 6, the method is applicable in industry and allows for immediate validation of knowledge due to the communication aspects of the EA modeling. With the explicit definition of a target implementation, the approach supports documentation, maintainability, and reproducibility for future users and developers of the DE integration. Furthermore, communication and the involvement of several stakeholders is potentially fostered. Nevertheless, the maturity of the method is improvable by conducting the following future work: First, method step one must be adapted so that requirements and assumptions are traceable and metrics can be derived. Secondly, the modeling and assessment of data traceability requires further work regarding related approaches in the literature as well as an extension of the representation of dependencies. Third, the validation of the formalized knowledge requires to be validated within participative workshops so that the result of the six-step method is proven. Next, integrating the derivation of metrics, as proposed in related design automation literature [RVSS19], could be beneficial to validate the impact of the DE application. Finally, industrial case studies must be conducted to validate applicability, comprehensiveness and benefits in practice.

7.4.2 Implications for Industry

The here presented method contributes to industry by fostering the integration of DE application in actual processes. Additionally, communication and documentation is improved with validated knowledge. Due to the simplicity of the application and little DE knowledge required for the application, most of the steps can be conducted by non-experts in DE that reduces the impact of missing available experts on the market, and missing knowledge in the companies [Ana22, RR22]. Moreover, a step-by-step guide for integrating DE applications is given, potentially leading to increased practical use

of DDE. Finally, these steps are done prior to implementation, leading to increased knowledge of the potential implementation and thus promises to reduce the number of failing implementations.

7.4.3 Implications for Research

The method presented here contributes to the consolidation of the different scientific communities and to the development of common methodologies for the involved scientific fields. Especially for the DE community, tools are made available which have not been integrated so far. Furthermore, the accurate documentation of the application promotes the transition of methods from academia to industry.

7.5 Summary

This chapter proposes a new method to support the elaboration and formalization of DDE integration into existing processes. The method builds on the modeling of EA and extends it to include modeling of (1) data traceability, (2) transition of applications and processes into the planned architecture within a EA model, and (3) templates to drive the integration of automated data collection mechanisms and DDE into the planned processes. Furthermore, a definition of goals is proposed that is aligned with the elaboration in Chapter 6, which is completed by a formulation of requirements and assumptions.

Hence, the answer to RQ5.1 *"What are the requirements to enable the traceability of data relationships gathered in different stages of the product lifecycle?"* is that data needs to be collected at a feature (attribute) level to allow the semantic connection of data. Particularly, each data attribute must be assessed regarding traceability among the lifecycle. Additionally, data object collection needs to be automated. It is not particularly necessary to allow the tracing of any data object to each other object forward and backward. Still, it is necessary to enable tracking through the product lifecycle (PLM), even if this requires several steps.

Based on the finding of RQ5.1 that automated data collection mechanisms are required, the answer to RQ5.2 *"What are the requirements to enable the traceability of data relationships where the respective data are gathered in different stages of the product lifecycle?"* is that a modular, simple, and lightweight architecture is required to enable the integration in any subprocesses or devices. Modular means extending and adapting to integrate any data source, e.g., legacy hardware, SQL servers or other systems. Simple describes the integration of a new data source and the processing of data. The evaluation showed that potentially multiple systems and data sources must be integrated using the workflow engine. If the application of a data collection mechanism requires much effort, it will not be automated even if it is necessary. Lightweight is related to integrating small systems that might be unable to execute heavy systems. In this method, the evaluation is based on the open-source workflow engine CPEE due to the readily integration of any

data source by using REST services for communication. However, the method is still applicable for other workflow engines.

To allow the integration of arbitrary frameworks and data sources, the answer to RQ5.3 *"What means of graphical modeling can facilitate the integration of data collection mechanisms into current business processes and applications?"* is that the data collection mechanism has to be integrated using EA while changes on business processes, objects and data artifacts need to be highlighted. Particularly, changes and the data transition need to be documented to assess the effort before the integration.

The answer to RQ5.4 *"What means of graphical modeling can facilitate the integration of Data-Driven Engineering applications into current business processes and applications?"* is that the DDE capabilities need to be integrated into existing processes using EA to visualize the impact on actual processes and highlight used data sources. To foster the understanding of the To-Be integration, changes need to be highlighted to enable documentation. Moreover, means of SysML BDD and IBD are needed to represent the various relationships between data and hence, streamline the implementation, which further leads to a reduced effort of the scarce resource of DE experts.

CHAPTER 8

Formalizing Data Engineering Tasks using SysML

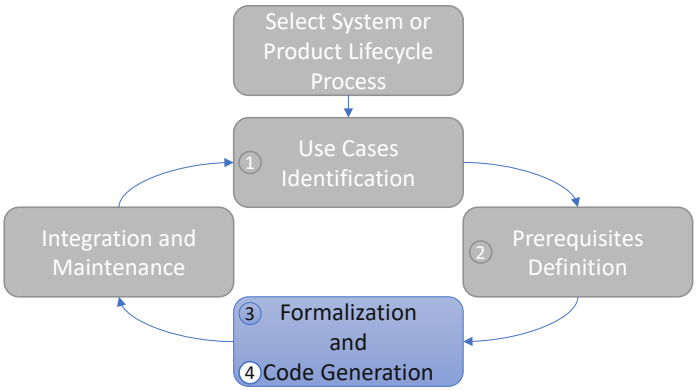
This chapter addresses the research objective ③ in Figure 8.1 related to the formalization of DE tasks using SysML. As the figure shows, several research implications, such as the formalization of validated knowledge, reuse of formalized knowledge or enhancement of interdisciplinary communication can be deduced.

The need for a more comprehensive method that incorporates the entire development cycle of the CRISP-DM methodology is highlighted in the results of the SLR presented in Section 3. For example, the integration of business understanding is not reflected in state of the art methods due to the focus on DSMLs, which leads to a reduced level of integration into existing MBSE methodologies. Additionally, the lack of generalization and use case specific applicability requires special focus on maintainability and extendability. Although the previous methods introduced in Chapter 6 and 7 are beneficial as prerequisites for this method, it is applicable independently from the aforementioned methods as well.

The general RQ for this method is as follows:

RQ6 What extensions to graphical modeling languages such as SysML are needed to integrate Data Engineering tasks comprehensively into Systems Engineering processes, to formalize product and process knowledge as well as data artifacts such as data attributes, interfaces, and data transformations?

Subsequently, the following detailed RQs have been identified:



Objectives	Research Implications	Challenges	Realization
1 Identify use cases for Data-Driven Engineering	• Identify relevant business processes • Identify data attributes and data interfaces • Identify potential in actual processes	• Diversity of processes • Distributed knowledge sources • Ambiguity of communication	Chapter 6
2 Define prerequisites for integrating Data-Driven Engineering	• Define data associations • Define automatic data collection • Integrate desired tool in actual processes	• Diversity of processes • Legacy hardware support • Diverse data level of detail	Chapter 7
3 Formalize Data Engineering tasks using SysML	• Formalize validated knowledge • Reuse of knowledge from MBSE • Enhanced interdisciplinary communication	• Ambiguous modeling • Complex formalization • Level of detail of formalization	Chapter 8
4 Automatic Code Generation for Data Engineering based on SysML	• Reuse validated and formalized knowledge • Enable flexibility and maintainability	• Decomposition of knowledge • Generation of executable code	Chapter 9

Figure 8.1: Overview of research objectives, implications and challenges addressed in Chapter 8.

RQ6.1 What means of SysML can be used to represent a sequence of Data Engineering statements?

RQ6.2 What means of SysML can be used to represent the order of execution?

RQ6.3 What stereotypes and associated structure need to be defined to enable reuse, extensibility and simplicity?

RQ6.4 What means of graphical modeling can be used to represent and guide the development of Data Engineering tasks?

To address the identified RQs, the elaborated method enabling graphical formalization of DE tasks is presented in the following section. For evaluation purpose, the method is applied to a use case that facilitates sensor data to represent a weather station as illustrated in Section 5.2.2. Finally, the findings are assessed with respect to benefits and shortcomings. For related work and state of the art approaches, please refer to Section 3.

A selection of text, figures and tables within this chapter is based on the publication in box „Publications 6: Formalizing Data Engineering Tasks using SysML“:

Publications 6: Formalizing Data Engineering Tasks using SysML

[RRMR22] S. Rädler, E. Rigger, J. Mangler, and S. Rinderle-Ma, "Integration of Machine Learning Task Definition in Model-Based Systems Engineering using SysML," in 2022 IEEE 20th International Conference on Industrial Informatics (INDIN). Perth, Australia: IEEE, Jul. 2022, pp. 546–551, doi: 10.1109/INDIN51773.2022.9976107.
[RMR23] S. Rädler, J. Mangler, and S. Rinderle-Ma, "Model-Driven Engineering Method to Support the Formalization of Machine Learning using SysML," Jul. 2023, doi: 10.48550/arXiv.2307.04495.

8.1 Method

In response to the need highlighted in Section 3 and the identified RQs, this section describes a method to formalize DE tasks based on SysML and the application of an extended metamodel. Particularly, the metamodel of SysML is extended with specific stereotypes and functions to interpret the modeling. Apart from SysML, the method is aligned with CRISP-DM to allow the structuring of the implementation.

In the following, first, the extension of the SysML metamodel using stereotypes is described. Special attention is given to the package structure for organizing the stereotypes, extensibility for different purposes, and generalization so that stereotypes can be used for multiple use cases. Second, a package structure aligned with the CRISP-DM methodology is presented, enabling to guide the application of the newly defined stereotypes. Next, a syntax and definition of functions is introduced, allowing to interpret the formalized DE model enriched with the introduced stereotypes. Finally, means of SysML state diagram is used to define the tasks' execution order.

8.1.1 Metamodel Extension using Stereotypes

In the following subsections, six packages are introduced, which allow to group stereotypes that semantically describe required functionalities. Subsequently, an exemplary stereotype hierarchy for defining higher-order functions for domain-specific data transformation purposes is described in detail.

8.1.1.1 Stereotype Package Structure

SysML packages are used to group and organize a model and to reduce the complexity of system parts. Similarly, it can be applied for the organization of stereotypes, as depicted in Figure 8.2.

The organization of the newly introduced stereotypes is as follows: in *Common*, general stereotypes are defined that are used in other packages as basis, e.g., a stereotype *DE* is defined in *Common*, each defined stereotype related to DE inherits from this stereotype

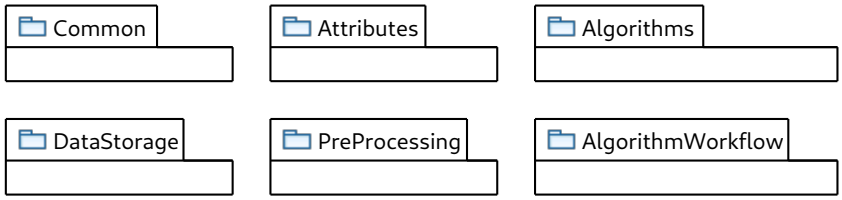


Figure 8.2: SysML package structure to organize stereotypes for DE concerns.

to indicate that it is a DE stereotype. Additionally, stereotypes can be defined allowing to categorize other stereotypes, e.g., a *Pre-Processing* stereotype allows to identify that all inheriting stereotypes are introduced for the data preparation step of the CRISP-DM methodology.

In *Attributes*, stereotypes for a more detailed definition of attributes are defined. These attribute stereotypes cannot be applied to blocks, only to attributes of a block. Thus, the stereotypes extend primitive data types such as *Integer* or *Float*. The purpose of the extension are additional characteristics to describe the data, e.g., valid ranges of a value or the format of a datetime property or a regular expression to collect or describe a part of a text value.

The package *DataStorage* defines available data interfaces from a general perspective required for the loading and processing of data from various data sources, e.g., SQL servers, API or other file formats (e.g., CSV). The purpose of the stereotypes are to support the *data understanding* of the CRISP-DM methodology. Additionally, it allows to bridge the gap between business and data understanding due to the explicit definition of data. Further details in Section 8.1.3.

In the *Algorithm* package, various ML algorithms are defined and grouped with respect to types of algorithms, e.g., regression or clustering algorithms. Particularly, the focus is put on key characteristics of an algorithm implementation, such as mandatory hyper-parameter or the stereotype description. Optional algorithm parameters are not described in the stereotype, but can be added during the modeling, as illustrated in Figure 8.5.

The *PreProcessing* package (a.k.a. as data preparation) is the most complex and extensive package due to the number of functionalities required. Additionally, a survey revealed that computer scientists spend the most effort in preparing and cleaning data [Ana22]. Within this package, functions are defined allowing to transform data so that a cleaned and applicable dataset for the DE algorithm is defined.

Finally, the *AlgorithmWorkflow* package, consisting of stereotypes for states of a *state diagram*, allowing to define the implementation order of the DE tasks. Typically in SysML, states are connected to activities, which are a sequence of execution steps. However, in practice, it turned out that it is very time consuming to prepare activities first. Additionally, a function abstracted as a single block can be considered as a set of activities. Consequently, *state diagrams* are used instead of *activity diagrams* to reduce the implementation effort and complexity.

8.1.1.2 Stereotypes Hierarchy

As mentioned in Section 8.1.1.1, each package represents a specific hierarchy of stereotypes, allowing to describe various aspects of DE subtasks. An example definition of stereotypes related to data pre-processing is depicted in Figure 8.5. As introduced in Section 2.3.3, stereotypes can be hierarchically composed to describe specific attributes only once for a set of stereotypes. On top, the *de* stereotype defined in the *Common* package is depicted, indicating that all inheriting stereotypes are related to DE. Formalizing a DE task is intended to be iteratively, which is why some stereotypes are abstract, illustrated by italic letters. If a stereotype is abstract, it means that the stereotype requires further details or that a child stereotype with additional information is required, e.g., *DataTransformation* cannot be used without further details as it can be arbitrary transformation of data. The purpose of abstraction is to support the early definition of tasks in the product development without details already known, e.g., the final file-format used to store the data.

From top to bottom in Figure 8.3, the level of detail increases and the task is more fine-grained. Consequently, leaves are the most fine-grain stereotypes. The inheritance additionally allows to group functions of a specific kind, e.g., functions regarding outlier detection etc. Due to the grouping of functions, the composition of stereotypes strongly depends on the preferences of the implementing expert and the purpose of the composition in terms of inheritance of attributes. However, note that attributes defined in a parent stereotype are also available in a child stereotype, respectively. Therefore, each level should only represent mandatory attributes. This especially applies for algorithms with a lot of hyper-parameters, e.g., logistic regression with more than 20 parameters and attributes[1]. In case a parameter is not defined in the stereotype, it sill can be added during the modeling and application of the stereotypes. A sample can be found in Section 8.2.

Additionally, it is possible to add a set of values using *Enumerations* for a single attribute, e.g., *MissingValueFunction* highlighted in green. In this respect, modeling is more precise and guided by a fixed set of valid options. Similarly, specific stereotypes can be used as an attribute, which means that only blocks or attributes that apply the specific stereotype can be assigned, e.g., *Method_Attribute_Input* indicating that only properties with a stereotype defined in the package *Attributes* can be applied because each attribute stereotype inherit from that specific stereotype.

Finally, the application of the keyword *BlackBox* can be used if a function shall be hidden due to security reasons or the implementation is unknown, e.g., *BlackBox_Outliers* on the right side of Figure 8.3.

[1] https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

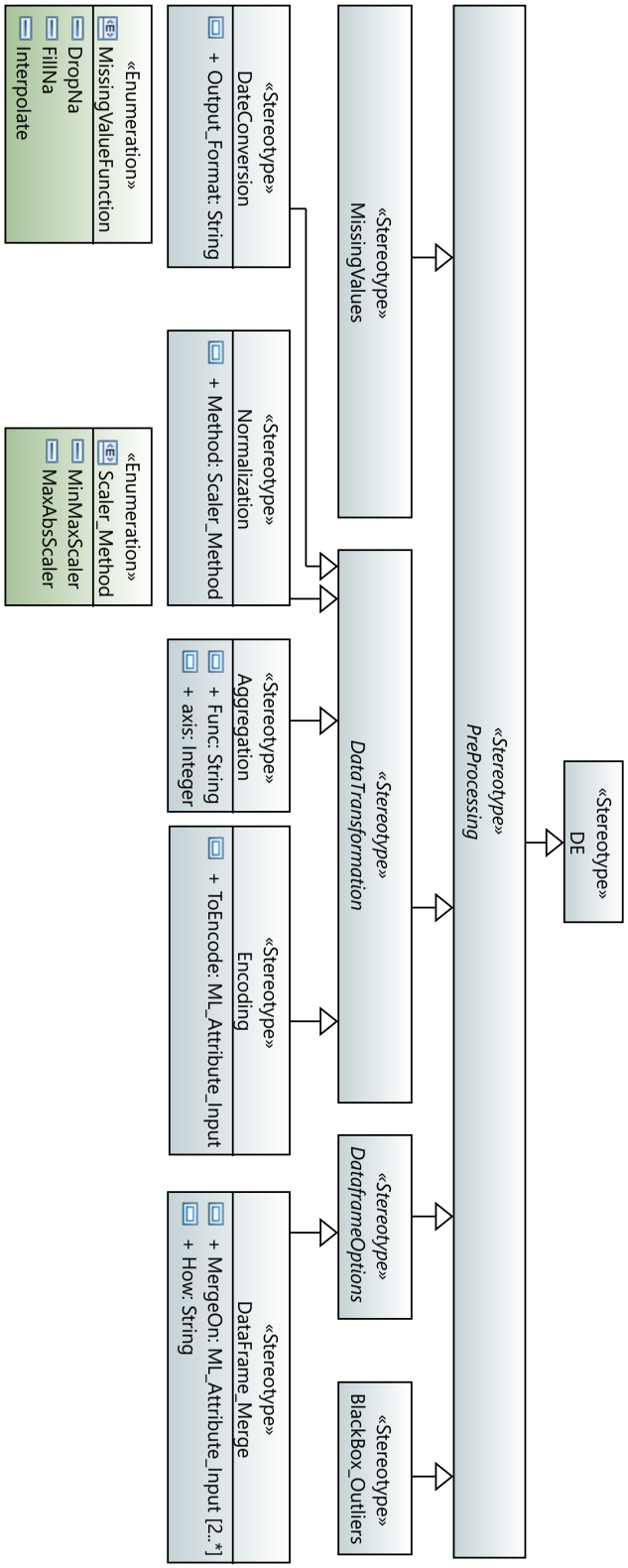


Figure 8.3: Example hierarchy of stereotypes related to data pre-processing/preparation.

8.1.2 Package Structure Guiding the Implementation

CRISP-DM as described in Section 2.1.5 consists of six steps, each describing a specific aspect required for the development of a DE project. Figure 8.4 illustrates the package structure aligned with the CRISP-DM methodology. *Business Understanding* consists of BDDs describing the system under study with the composition from a system configuration point of view. In this respect, the VAMOS method [Wei14] is integrated to describe a specific system configuration. The integration of the VAMOS method focuses on the data interfaces and attributes of a particular configuration of a system, as different configurations of a system might lead to changed data output. In this method, the VAMOS method is used to focus on data interfaces. Therefore, other SE knowledge is presented in other diagrams, which is out of the scope of this work. Still, the knowledge modeled in other diagrams is connected to the instance of a block used in the VAMOS method and therefore, multiple disciplines are enabled to work on the same model. Assuming that collaborative working is supported by the underlying modeling tool.

The second step, *Data Understanding*, details the *Business Understanding* with the definition of delivered data on an attribute and data format level. Particularly, the data type and the name of the delivered data attribute are described using BDDs. Additionally, attribute stereotypes are used to describe the data in detail as described in Section 8.1.1.1. With the application of stereotypes on a block level, the type of data interface is defined, e.g., CSV files or SQL servers. As a result of the formalization of the interfaces in this package: The information exchange between the SE and the data engineering can be considered as completed.

Based on the *Data Understanding*, the *Pre-Processing* is applied to transform and prepare the data in a final dataset that can be used in the *Modeling*. In the *Pre-Processing*, the most effort is required due to the possible number of required data transformations to create a dataset usable for DE. The result of the *Pre-Processing* is a final dataset, considered to be ready for the ML algorithm.

Within the *Modeling* package, algorithms are applied to the final dataset. Additionally, train-test-splitting and other required functions on the ML algorithm are applied. In the *Evaluation* package, various metrics are used to asses and prove the validity of the algorithm result of the *Modeling* package.

Finally, the *Workflow* package, which describes the execution order of the formalization in the previous packages using state diagrams. For each state, a custom stereotype is applied allowing to connect a block that is connected to a stereotype inherited from *de*. The method to assign blocks to states allows to overcome the necessity to define activities, making the method less heavy for the application and reduces time for the formalization of the DE.

Typically in CRISP-DM, the very last step is the *deployment*. However, the deployment is considered out of scope in this work and therefore the method ends with the workflow.

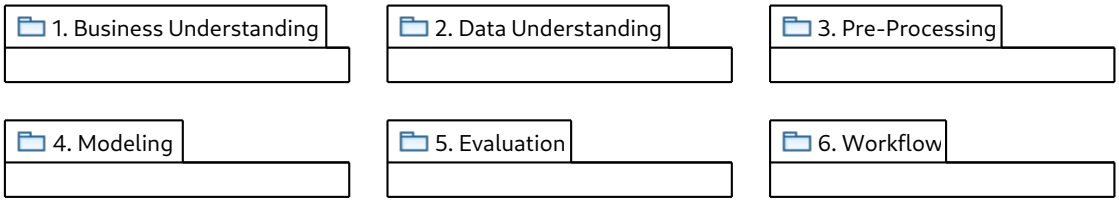


Figure 8.4: The implementation structure aligned with CRISP-DM.

8.1.3 Functional Interpretation

For the purpose of implementing DE functionalities, the functional programming paradigm is used [GM23]. It utilizes higher order functions, invoked on (data-)objects which are returning objects. This allows for a step-by-step decomposition, filtering and transformation of data, without side-effects, i.e., changes to variables, in comparison to the imperative programming paradigm.

This sequence of function invocation aligns well with how UML and other modeling languages implement abstraction-levels to reflect a relevant selection of properties to focus on the aspects of interest [BCW17]. Functions are blackboxes with processing capability that are associated with (data-)artifacts upon which they can be called, and are associated with data-artifacts they produce as output. The abstraction is realized by describing functions or a set of functions with a single stereotype and instances with blocks.

A class in UML is defined among others by attributes, stereotypes, operations (methods), constraints and relationships to other classes. In SysML, a block describes a system or subsystem with a similar definition as a class in UML. A DE task and the respective subtasks can be seen as a system with subsystems. Therefore, each subtask is modeled using blocks, aligned with the syntax described in Section 2.3.3. Particularly, only input values represented as attributes of a block and the relationship to other blocks are modeled. The operations (methods) are defined as stereotypes with abstracted implementations. Attributes defined on the stereotype are mandatory input values for the definition of a DE subtask. The attributes defined on a block itself are optional for documentation or to extend the stereotype with fine-grained details, e.g., *utc* attribute in the *Format_Date2* block in Figure 8.5. The output of a subtask (block) is implicitly defined in the implementation of the code snippet related to a stereotype and not explicitly depicted in the model. The output of a block can be used as input for other blocks, e.g., *CSV_1* block as input for the *Format_Date* block.

Figure 8.5 depicts a few samples of the aforementioned concepts. On top right, a date conversion subtask is modeled as *Format_Date*. The date conversion stereotype has a mandatory attribute to define the format of the output of the conversion. The input for the date conversion is the block *CSV_1*, connected using a part association.

In this sample, the *date* attribute is the only input value matching due to the stereotype *Datetime*. However, if the input is ambiguous because the datetime is stored for instance

as integer or multiple attributes of the connected block are in the correct input format, it is necessary to add additional attributes to the date conversion to select the particular input, e.g., with a new attribute, whose value is the particular input attribute from the connected block.

The block *Format_Date2* inherits from *Format_Date*. Therefore, the input and the attributes are the same except of manual overwritten values, e.g., changes on the output datetime format or the added additional attribute *utc*.

Another example in Figure 8.5 shows the integration of multiple inputs. The *Merge_DF* block consists of two input blocks and the attributes on which the merging function shall be applied are defined using an attribute that consists of two values (*MergeOn*). The *MergeOn* attribute is mandatory and therefore defined on the stereotype.

Although the implicit execution order of the subtasks is defined by the associations and the necessity to compute first inputs, the execution order might be ambiguous, e.g., execute first the *Format_Date* or the *Merge_DF*. As described in Section 2.3.3, structural diagram elements, such as blocks, require the integration of behavioral diagrams to allow the definition of an execution order [BCW17].

To enable the connection of a block with a state in a state diagram, custom stereotypes are applied. The stereotypes for the states consist of a single mandatory attribute. The mandatory attribute references a block with a stereotype that inherits from the root stereotype *DE*.

8.2 Evaluation

This section presents two case studies, i.e., a weather system that predicts weather conditions based on sensor data (details in Section 5.2.2), and an image similarity check that makes it possible to assess whether the actual print of a 3D model with a 3D printer corresponds to the desired output. As a result, the printing process can be stopped prematurely, saving filament and time.

8.2.1 Weather Prediction based on Sensor Data

Figure 8.6 illustrates the composition of the weather system that is split in two parts. On the left side, a local station is equipped with various sensors, delivering a CSV file with sensors for measuring and on the right side, a weather prediction that additionally delivers a CSV file with weather predictions over the internet.

From an SE perspective, the weather system is a CPS and can be configured with various sensors. Figure 8.7 depicts the SysML model of the weather system with a specific configuration aligned with Figure 8.6. Particularly, Figure 8.7 depicts a method aligned with [Wei14] that allows to formalize variations. Additionally, the modeling of the system from an business perspective is the first step of the method. Focus is put on the values of interest, which are the output values of the subsystems, to keep the business

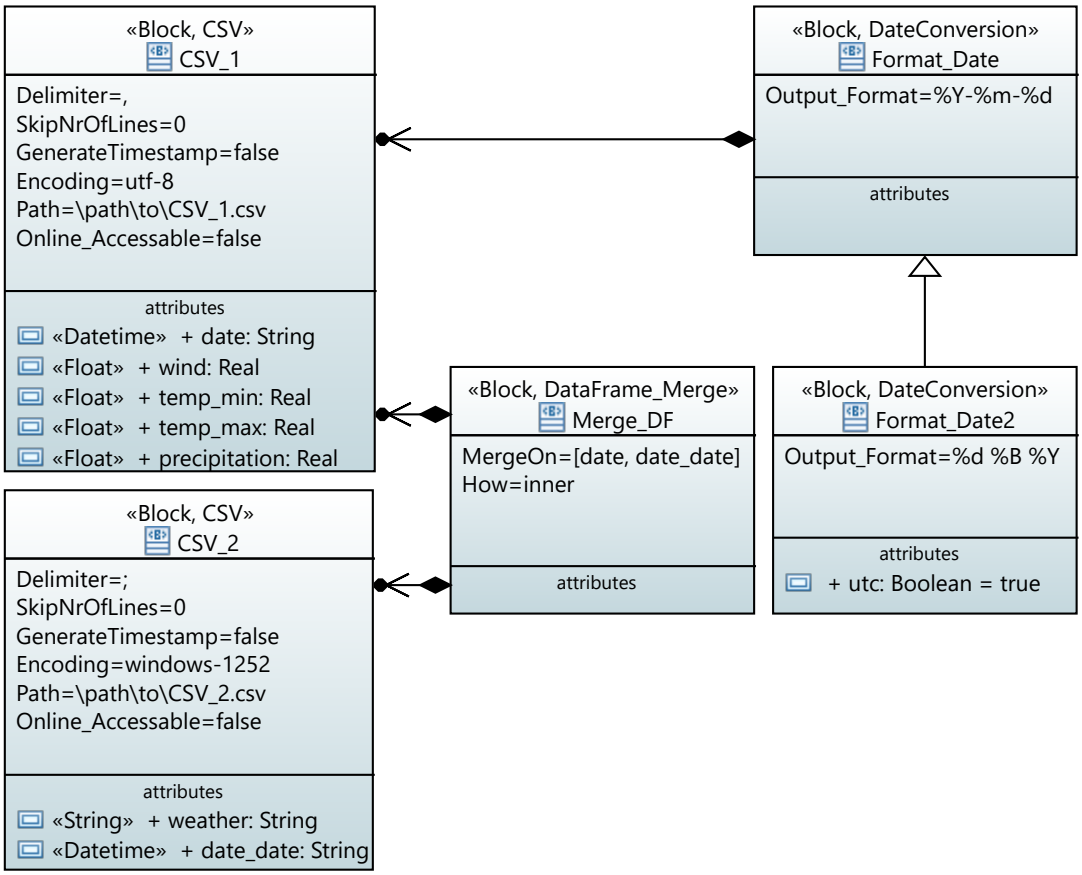


Figure 8.5: DE data pre-processing based on a sample in Section 8.2.

understanding as concise as possible. In the middle of the figure, the core weather system configuration is depicted. The surrounding subsystems are sensors or subsystems, e.g., an API (right side). The attributes of the sensors are output values of each subsystems to align with the CRISP-DM business understanding that aims to get a general idea of the system and from where data originates.

To transform the business understanding in valuable data understanding, connections between the system in the business understanding and output data formats are established. Particularly, a *realization* connection between the CPS and blocks describing the data format using stereotypes inheriting from *DE* are modeled. In the blocks, each attribute has a type representing the actual data type in the data source and a stereotype with a *DE* attribute describing the representation in the DE method, e.g., *CSV_2* attribute *date_date* is of type *String* and is mapped to the stereotype *Datetime* that considers aspects such as the datetime format. Additionally, stereotype attributes are defined such as the *Encoding* or the *Delimiter* to describe the composition of the *CSV* file.

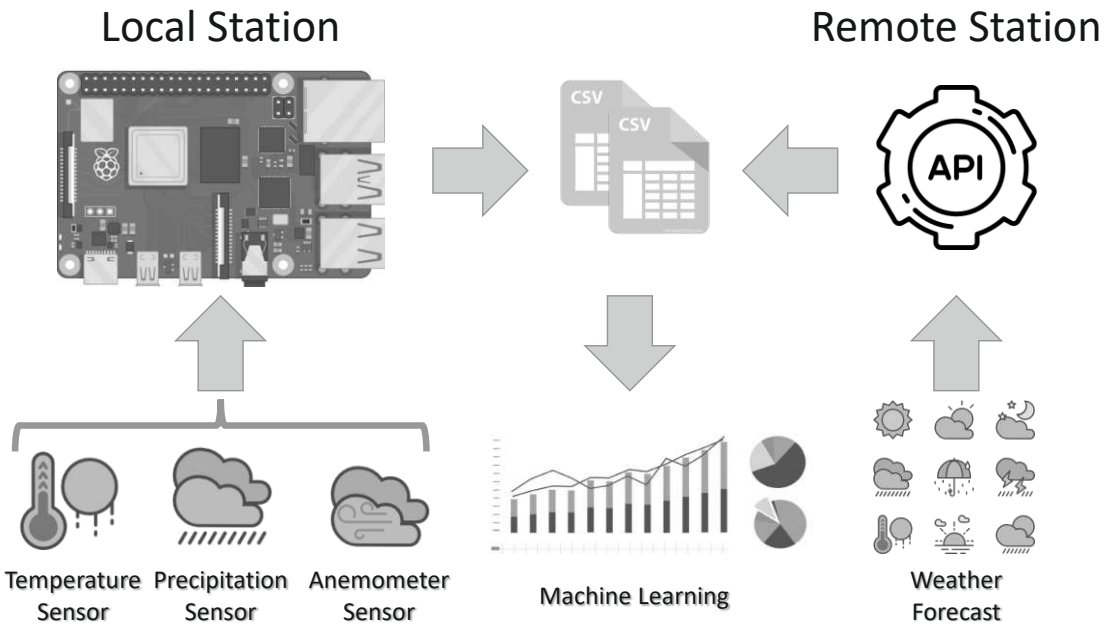


Figure 8.6: Illustration of the weather system use case.

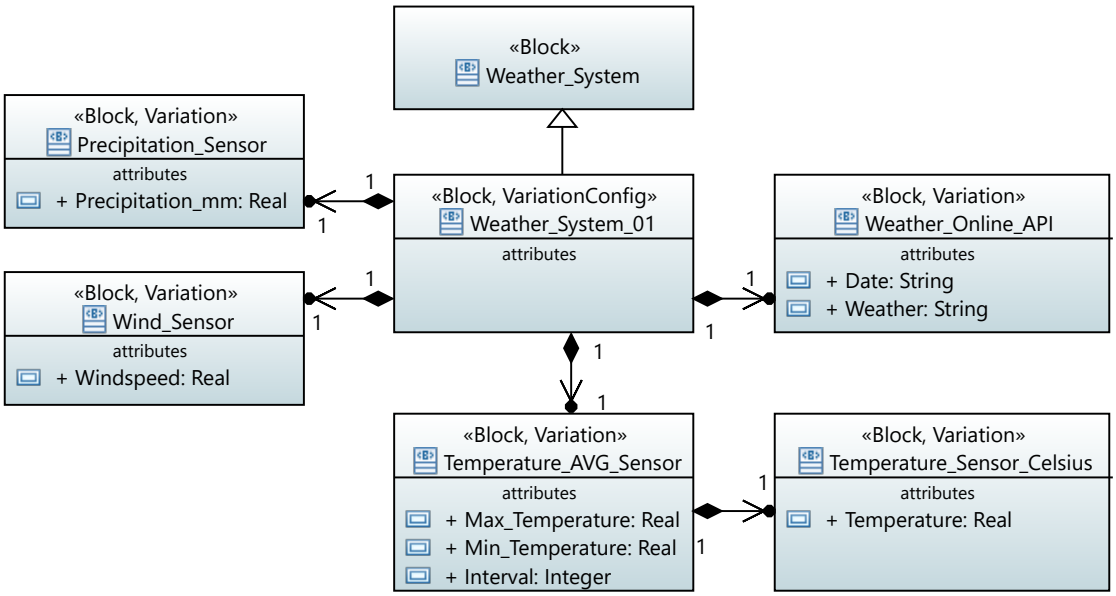


Figure 8.7: Business Understanding of the weather system.

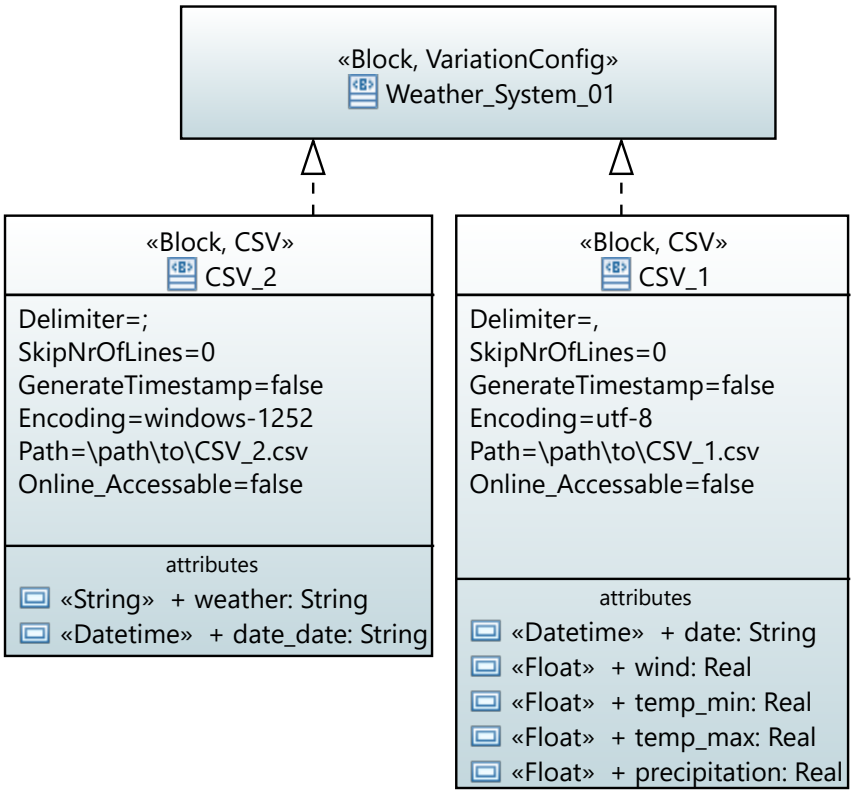


Figure 8.8: Data Understanding of the weather system.

Figure 8.5 depicts a set of subtasks applied to the data sources defined in Figure 8.8. For and explanation of Figure 8.5, please refer to Section 8.1.

Figure 8.9 illustrates the application of a train-test-split and the integration of the split data into two different regression algorithms, which are specified in a mandatory attribute. As of the definition of the stereotypes, no further parameters are mandatory. For the *RandomForestRegressor*, the optional hyper-parameter *max_depth* is defined.

Figure 8.9: Modeling of algorithms.

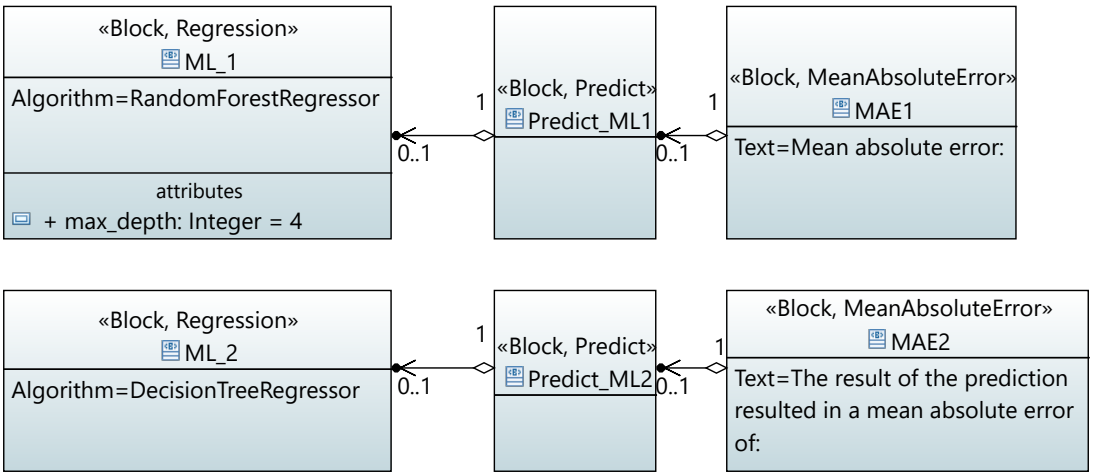


Figure 8.10: Evaluation of the weather prediction.

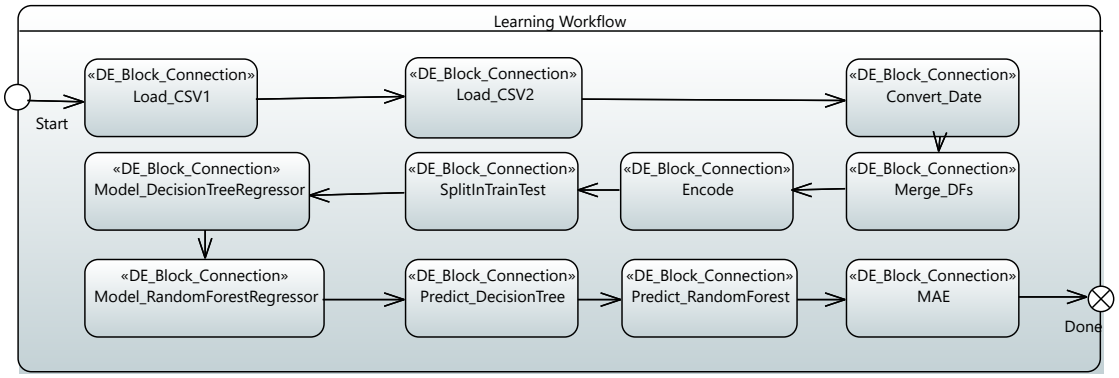


Figure 8.11: Sample integration of the workflow.

Figure 8.10 depicts the prediction and the application of metrics such as mean absolute error (MAE). The mandatory parameter text is a placeholder allowing to add text that shall be implemented with the evaluation result.

The method's final step is integrating the blocks into an execution workflow. Figure 8.11 illustrates the execution order of the algorithm steps. As can be seen, the *Format_Date2* block modeled in Figure 8.5 is not depicted in the workflow, meaning that it is not taken into concern during the implementation and is left out as an artifact unnecessary over the formalization evolution. The state's name is to readily understand the workflow and the blocks connected with the *DDE_Block_Connection* stereotype.

As the scope of this work is to formalize DE tasks and not to improve the executable code or to derive the code automatically, the result of the DE and the implementation itself are not depicted. The automatic code derivation is described in Chapter 9.

8.2.2 3D Printer Success Evaluation during Printing

The purpose of the second use case is to evaluate the application to detect faulty 3D prints during the printing process by comparing the actual status of the printed model with the intended model. This use case illustrates the method's applicability to other data sources, such as image data, and the integration of the method into an executable workflow engine. Additionally, the integration of pre-trained models is depicted by integrating TensorFlow Hub. The idea of image similarity checking is based on an image similarity tutorial[2].

The use case process is described below and illustrated in Figure 8.12. In this respect, CPEE [MR14, MR22] is adopted to orchestrate the application process, as CPEE provides a lightweight and straightforward user interface to orchestrate any application that allows interaction via REST web services. Figure 8.12 shows the workflow of the application, consisting of image generation and printing. The first three process steps define the slicing of a STL file and the generation of the reference images. Particularly, a Python script is called that generates the slices based on a given STL file and stores the generated reference images for later comparison and similarity check. The second part of the process consists of a loop that prints a slice, takes a photo with a camera from the top center of the working area, and calls a similarity script to compare the intended and actual printed model. The image similarity algorithm is defined using the DE formalization method, proposed here. The defined algorithm provides a similarity index compared to a threshold value. If the threshold is exceeded, the printing process is aborted, otherwise, the process steps are repeated.

[2] https://towardsdatascience.com/image-similarity-with-deep-learning-c17d83068f59

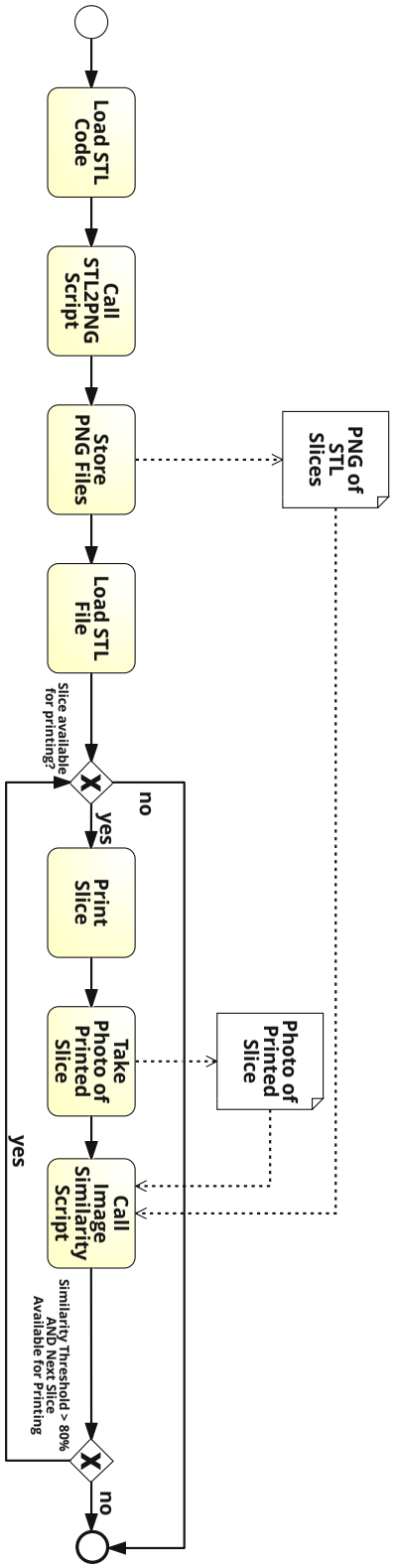


Figure 8.12: Workflow Integrating the formalized DE method to early stop 3D printing.

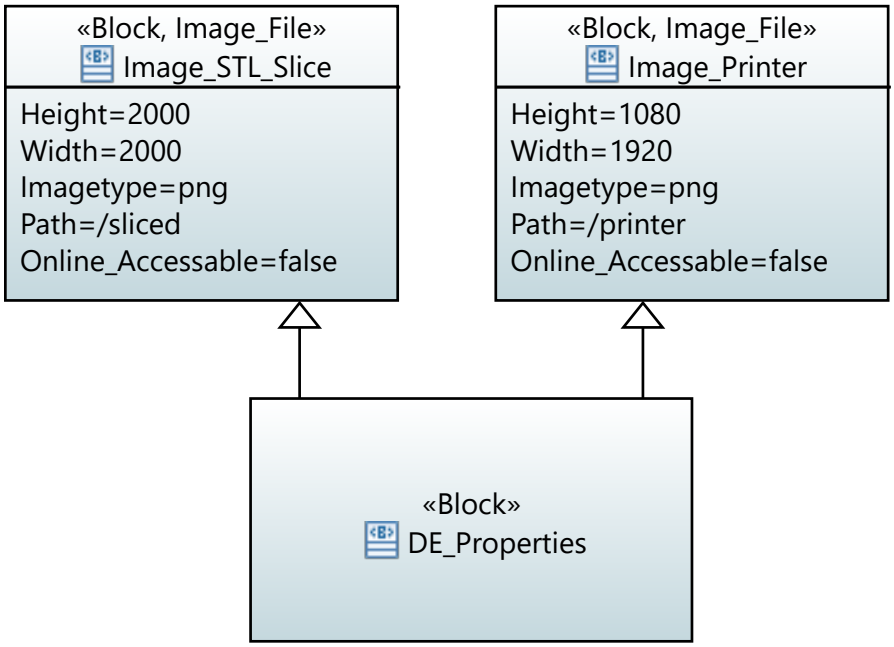


Figure 8.13: Image definition used for the similarity prediction.

The DE model integrated into the printing process is formalized below. Figure 8.13 depicts input data consisting of two images: the image sliced from the STL file and the photo from the 3D printer camera. In contrast to the first use case, the data attributes are not further detailed with stereotypes because the input data do not show any variations, i.e., the format and resolution of the images do not change.

Figure 8.14 depicts the scaling of the images such that they have the same dimension. The conversion parameter L allows comparing the images on a black-and-white basis. Normalization of the pixels and colors between 0 and 1 is also applied. The normalization in the block *Convert_PixelsAndNormalize* should be defined as a new stereotype. In this case, the application of the *CustomCode* stereotype is shown, allowing for the injection of program code, which allows rapid prototyping. However, flaws, such as vulnerability or hijacking of the method might lead to reduced understanding and reproducibility. Additionally, it is not the purpose of the method to use a single block that represents a complex solution that is programmed within the SysML block. As a sample, Figure 8.15 depicts on top the use of the model to represent the programming in a single block. Underneath, the recommended use of the model is illustrated, representing the same functionality. For further discussion on potential issues, see Section 8.4.3.

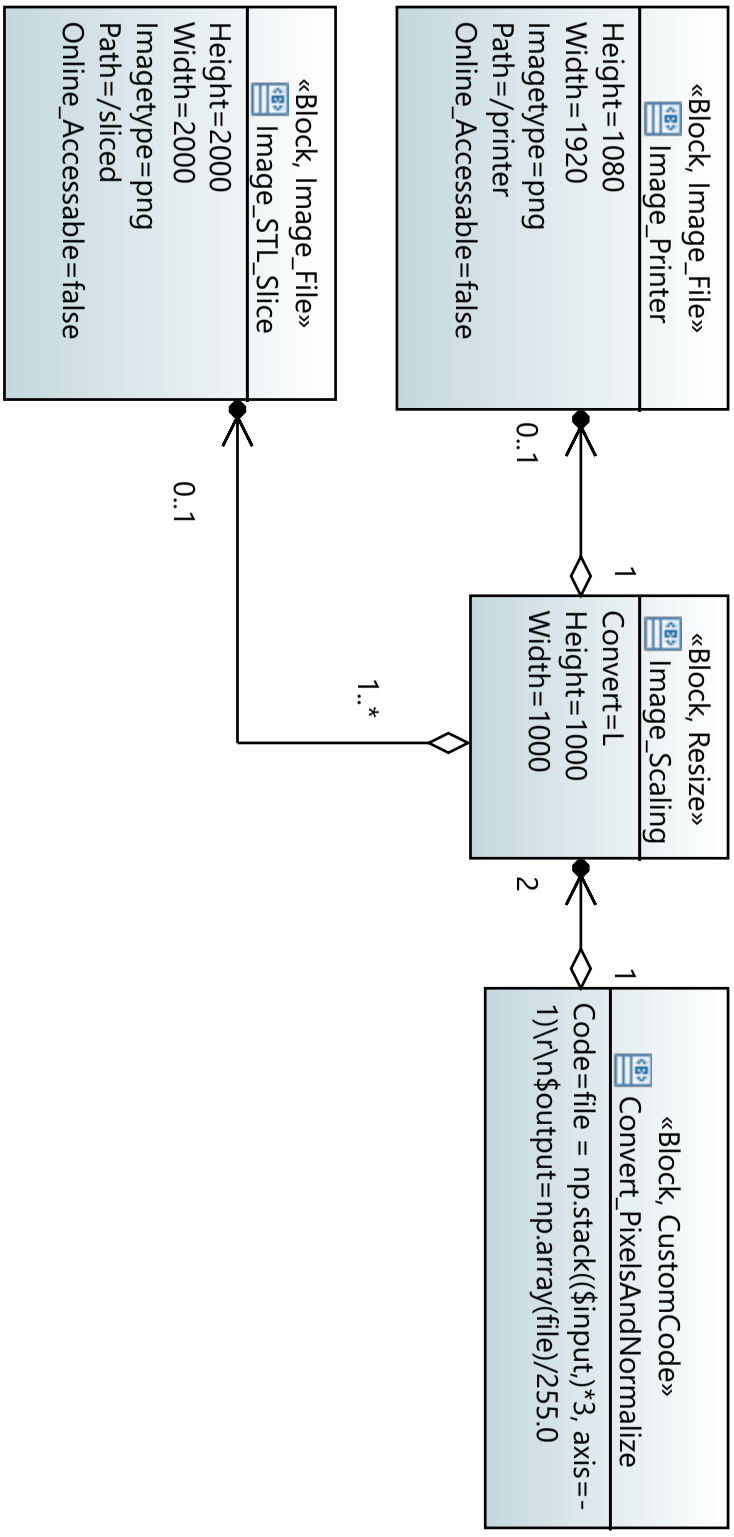


Figure 8.14: Image scaling and normalization used for data preprocessing.

Figure 8.15: On top the wrong application of the method and below correct use.

Furthermore, the two images are fed to the classification algorithm, as illustrated in Figure 8.16. The input value *Model* describes a TensorFlow Hub input, a pre-trained model to classify images. Finally, the result is measured using *cosine* distance metrics. The threshold for canceling the printing is implemented in the workflow and can be adjusted by the user. Finally, Figure 8.17 depicts the execution sequence of the algorithm.

Figure 8.16: Integration of a pre-trained model and prediction with cosine distance to express the similarity of the images.

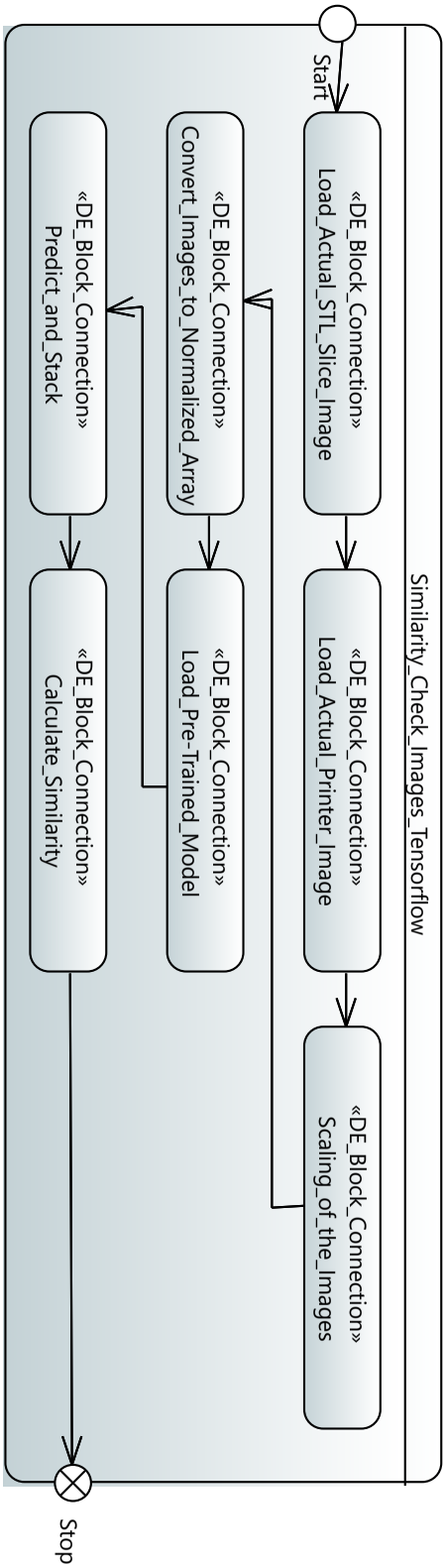


Figure 8.17: The execution workflow of the TensorFlow-based prediction algorithm.

8.3 User Study

This section presents a user study to assess feasibility and applicability of the method. Typical users of the presented method are computer scientists and engineers from various disciplines, depending on the application area. Therefore, this study aims to assess and compare computer scientists' and mechanical engineers' subjective workload and user experience regarding understanding, modifying, and creating DE tasks in a model-based method. Furthermore, the time required for applying changes or creating constructs in SysML is assessed to allow a comparison of the participants based on previous experiences, e.g., programming or modeling prior knowledge. Since the study and the modeling is conducted using the SysML modeling tool Papyrus[3], it is impossible to eliminate distortions due to the usability of the underlying tool, e.g., "How to model a block". Therefore, the study director will provide verbal assistance if a participant requires support due to the tool's usability.

Large sample sizes are necessary to enable quantitative evaluation, which is not applicable due to resource constraints. Therefore, discount usability principles are applied to test only a small group of customers and identify key usability issues by conducting small qualitative user studies with three to five users in a detailed scenario and a think-aloud method [Nie93]. According to [Nie93], a 70% chance to find 80% of the usability issues is given with five users. However, in literature, there are reports that the increase of five participants to ten significantly changes the amount of found issues [Fau03]. In this respect, a total number of 12 users were tested, equally distributed among the two groups, Computer Scientists (CSs) and Mechanical Engineers (MEs).

In the following, the experimental setting is illustrated. Next, an introduction to the evaluation procedure is given, followed by an introduction of the test cases in Section 8.3.3. Finally, the results of the user studies are depicted in Section 8.3.4. A discussion on the implications from the user study is given in Section 8.4.4.

8.3.1 Experimental Setting

The user study was conducted with 12 participants. Each participant has a university degree (B.Sc., M.Sc., or Ph.D.) and received a basic introduction to programming at university. Half of the participants are CSs, and half of the participants MEs. Other engineers can serve as potential users and equally valid test users, as well. However, to obtain a more homogeneous group, engineers are limited to MEs.

Due to the participants' different knowledge in modeling, programming, and data science, a self-assessment of their experience was made at the beginning of the user test. Table 8.1 summarizes the knowledge levels of the participants based on their highest university degree, years of experience, position at the current job, and self-assessment on the three relevant dimensions.

[3] https://www.eclipse.org/papyrus/

Table 8.1: Participants of the user study aligned with self-assessment of experience.

User	Univ. Degree	Years of Experience	Position	Programming Skills	Data Science Skills	UML/ SysML Skills
CS-1	B.Sc.	5	Software Engineer	7	3	6
CS-2	M.Sc.	3	Software Engineer	8	6	7
CS-3	M.Sc.	1	Ph.D. Student	7	6	3
CS-4	M.Sc.	2	Ph.D. Student	6	7	6
CS-5	M.Sc.	1	Ph.D. Student	6	7	8
CS-6	B.Sc.	1	Application Manager	7	4	4
ME-1	M.Sc.	6	Project Manager	4	1	2
ME-2	B.Sc.	11	Project Manager	2	3	1
ME-3	Ph.D.	10	Digital Engineering Manager	6	4	8
ME-4	B.Sc.	2	Simulation Engineer	2	2	1
ME-5	M.Sc.	3	Expert Powertrain	2	1	3
ME-6	M.Sc.	1	Manufacturing Engineer	1	2	1

8.3.2 Evaluation Procedure

The study started with a basic introduction to SysML and an overview of the method introduced in this work, taking approximately 10 minutes and involving the presentation of two predefined BDDs as samples with a focus on the modeling and understanding of a BDD and the application of the introduced stereotypes.

Following this, the users had to perform three tasks, i.e., (1) showing that they understand the purpose of the modeling and the basic idea of the method by describing the modeled methods in Figure 8.5, (2) replacing a *CSV* stereotype with *Text-file* stereotype and redefining the attribute properties of the text file, and (3) adding a new function by inserting and connecting a new block with a particular stereotype to an existing block.

Table 8.2: The three main tasks to be performed by the participants, with subtasks that can be used to assess whether the task has been completed.

Main Task	Subtask
Task 1 Understanding	Identification of input files
	Description of values stored in CSV_2 input file
	Description of attributes of the data stereotype of CSV_2 values
	Identification of stereotype properties, e.g., path of CSV_2 file
Task 2 Changes	Stereotype identified
	Stereotype removed
	Stereotype added
	Stereotype attribute identified
	Stereotype attribute value set
Task 3 Modeling	Block added to view
	Block associated with input
	Stereotype added
	Stereotype attribute value set

Each of the tasks (1) – (3) is subdivided into sub-activities to allow fine-grained evaluation of the tasks and the performance achieved by the participants. The sub-activities are presented with their tasks in Table 8.2.

For each participant, the time taken to perform the tasks is recorded. After each of the three tasks, NASA Task Load Index (NASA-TLX) [Har06, HS88] and the Systems Usability Scale (SUS) [Bro96]) questionnaire are filled out by the users to assess the participants' subjective workload and usability. Before filling out the questionnaire, the users were explicitly told to evaluate the method's usability, not the usability of Papyrus.

8.3.3 Test Cases

Table 8.2 depicts the subtasks to accomplish the tasks of the user study. Therefore, each subtask is assessed by the study leader to determine whether they are completed correctly or not. If a user could not find a specific button due to the usability of Papyrus, but could justify why it is being searched for, e.g., "I need to remove a stereotype and add a new one so that a new function is defined", the task is evaluated as correct.

To achieve reproducibility, the tasks were set exactly with the following wordings:

Task 1 Understanding: Please describe what can be seen in the currently displayed diagram and what function it fulfills. Additionally, please answer the following questions:

a) What are the two input files, and in which format?

b) What values are stored within CSV_2?

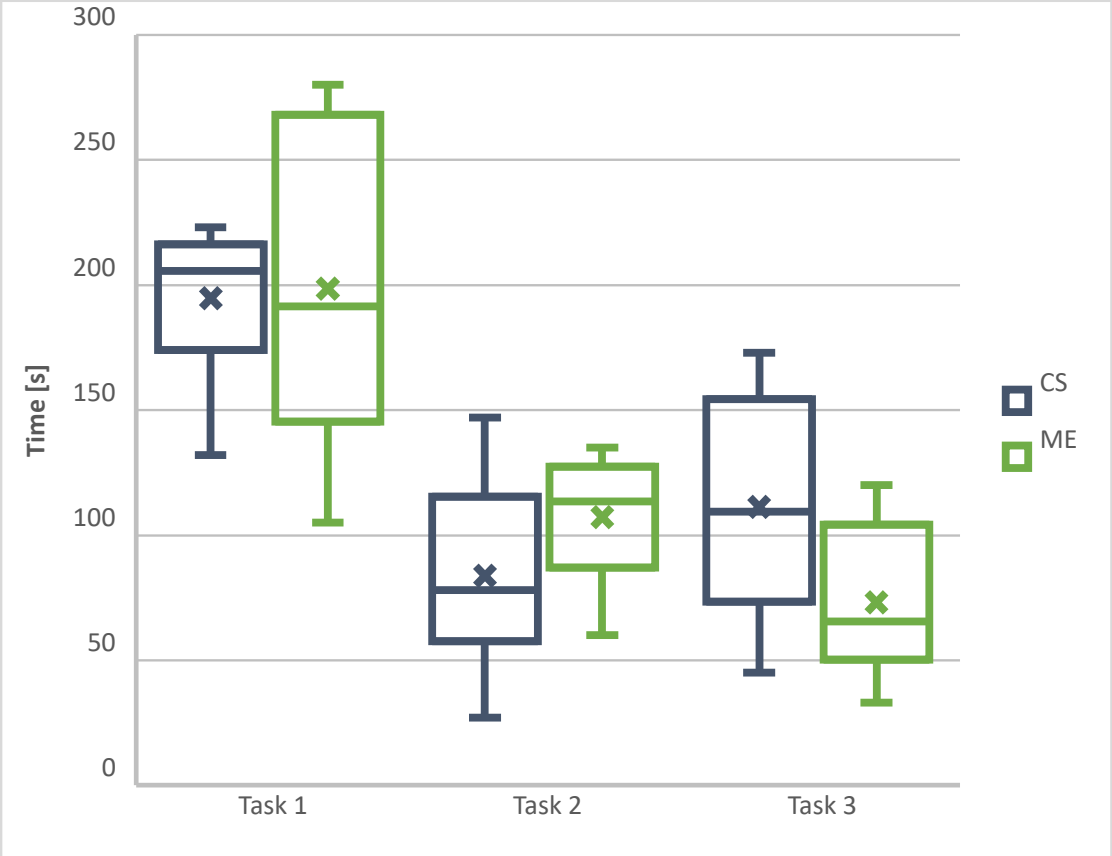


Figure 8.18: The time required by the participants per task and training direction.

c) What is the type of date_date, and how is it represented in the DE model?

d) What are the path and encoding of the two input files?

e) What are the properties of DataFrame_Merge Stereotype?

Task 2 Function Exchange: Behind the here presented *TextFile* function, a *CSV* stereotype is defined. However, the type is incorrect. Please change the file type to ***Text-File***. Additionally, set the encoding to ***UTF-8*** and the path to ***C:/file.txt***.

Task 3 Adding a Function: In the following view, you can see two input files connected to a merge block. Additionally, a normalization of the merge block is required. Please add the function for ***Normalization*** and set the value of the normalization method to ***MaxAbsScalar***.

8.3.4 Survey Results

Figure 8.18 shows boxplots of the required times for the individual tasks grouped per task and education of the participants (CSs or MEs).

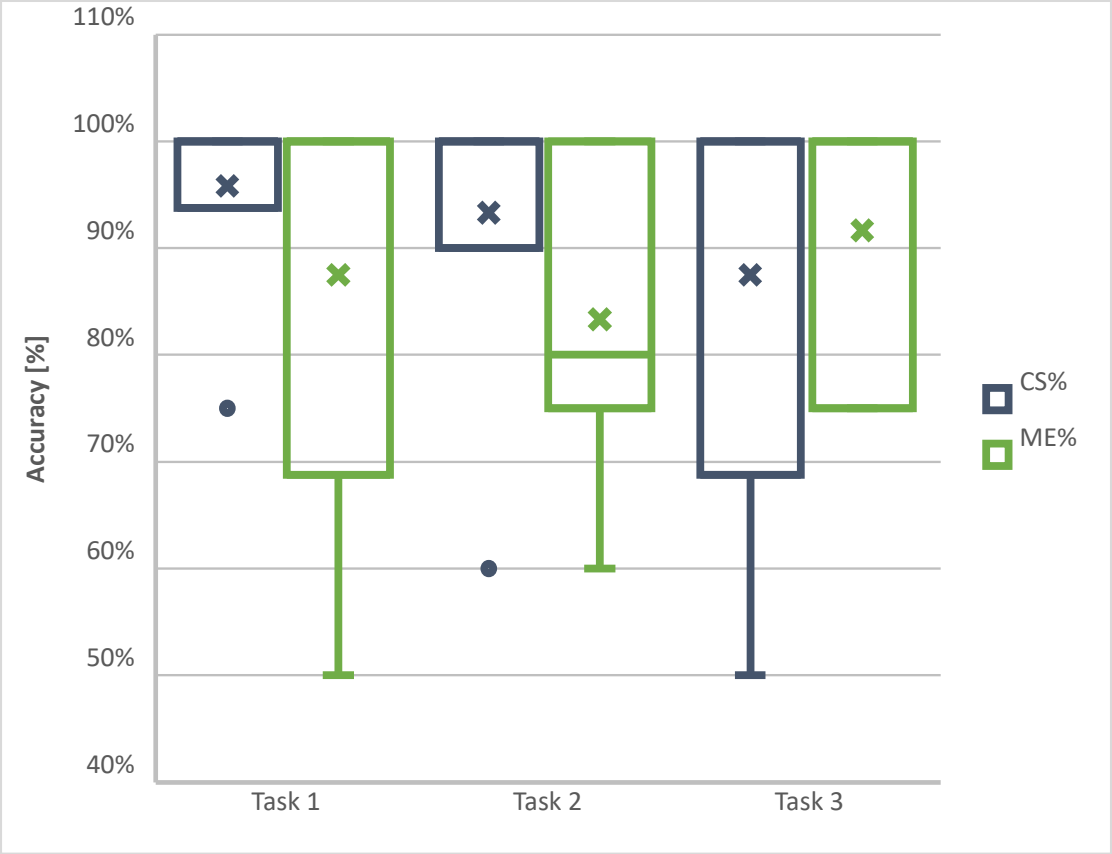


Figure 8.19: The degree of correct performance of the tasks.

For Task 1, the time required is higher than for Task 2 and Task 3, whereas Task 2 and Task 3 shows a comparable average and distribution. One reason for the higher time for Task1 is that the users had to describe a model by speaking which is a more time-consuming task. It was also observed that repetitive tasks made the users faster, which also came as feedback from the participants. Furthermore, the dispersion of Task 1 for MEs is higher compared to CSs. This scatter might be explained because of the varying experience levels of the participants with respect to modeling and data science. However, there was no correlation between the time spent and the correctness of the execution of the sub-activities. Regarding the dispersion of CSs, interestingly, Tasks 2 and 3 vary more than Task 1. This can mainly be explained by the familiarity with the Papyrus modeling environment. Thus, participants with more Papyrus experience had completed the tasks much faster than those who used Papyrus for the first time.

Figure 8.19 shows the result of the individual tasks in terms of correctness in relationships to the subtasks of Table 8.2. CSs perform better for Task 1 and Task 2, which can be explained by the extended prior experience regarding UML of CSs obtained during university education. In Task 3, however, MEs perform better. This can be explained

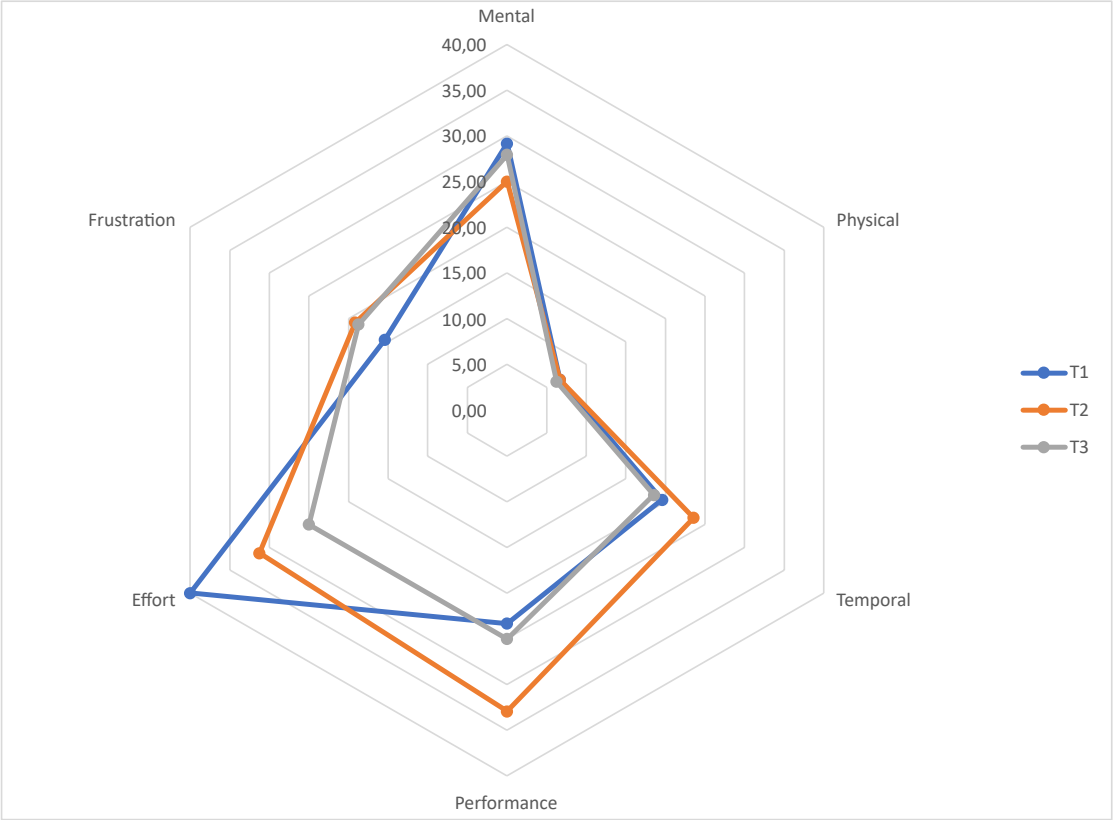


Figure 8.20: Result of the NASA-TLX questionnaire.

by an outlier value for CSs that performs significantly below the average. The overall accuracy of MEs increased with the evolving tasks although the average of Task 2 is lower than for Task 1.

The results of the applied NASA-TLX test to indicate the perceived workload of the participants for the specific tasks are presented in Figure 8.20. The lower the value of a dimension of the NASA-TLX, the lower the perceived workload. Consequently, a low scale value is seen as positive. The *Effort* dimension shows, for example, that with increasing experience or task, the perceived effort decreases. Furthermore, the frustration increases and the performance decreases compared to Task 1. For Task 3, the standard error is larger than for Task 1 and Task 2. Both might be justified due to the increasing complexity of the tasks. However, it is in contrast to the achieved accuracy in Figure 8.19.

The raw overall scores of the tasks are depicted in Table 8.21. According to [HNM88, PBU19], the workload is categorized as 'medium', which is the second best score and ranging from 10 to 29 points. The cumulative results of CSs and MEs shows a decreasing workload among the evolving tasks. For CSs, the workload appears to be higher than for MEs, especially for Task 3. As of the user feedback, no justification can be given on the

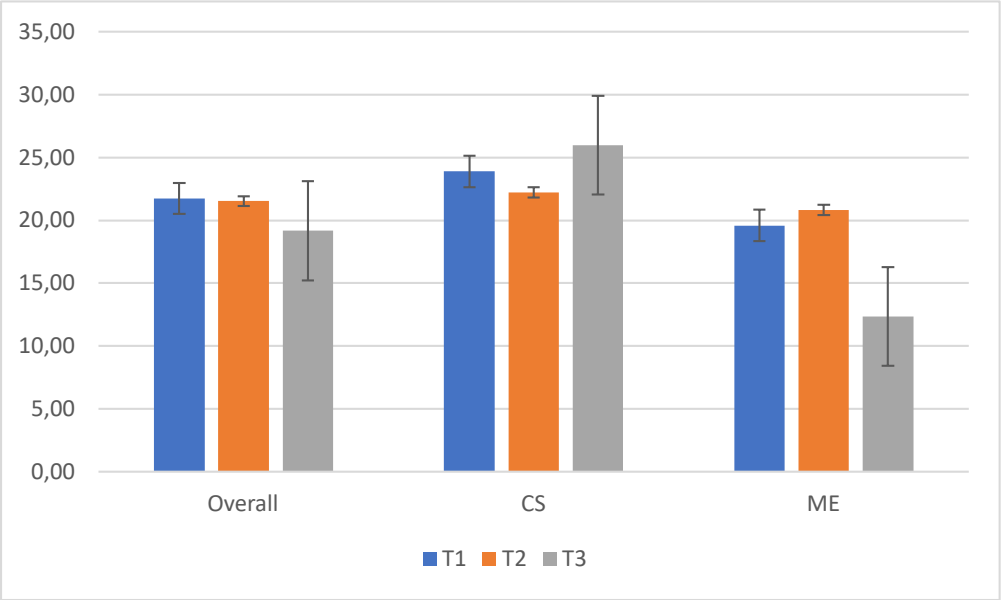


Figure 8.21: NASA-TLX overall score.

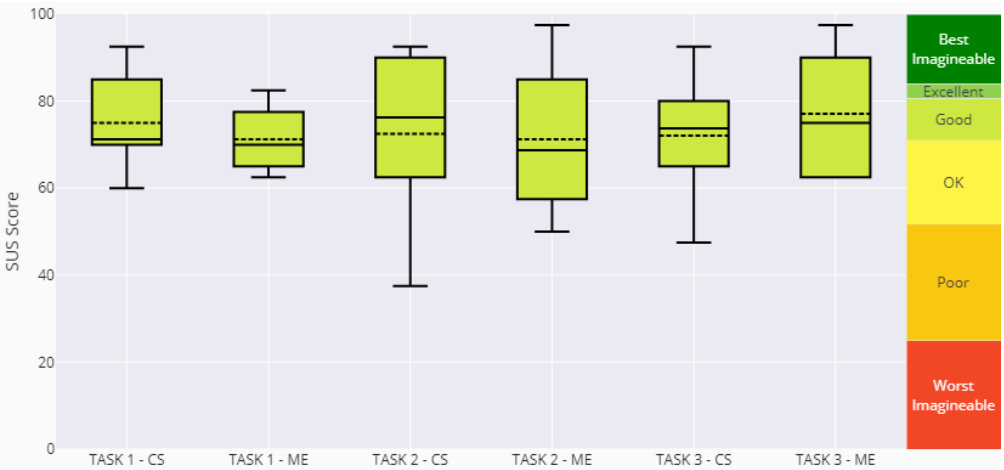


Figure 8.22: Boxplot of the SUS score.

difference between CSs and MEs.

The results of the SUS test with different rating scales are shown in Table 8.3 based on [BBP22].

Table 8.3: SUS analysis results.

Variable	SUS Score (mean)	Percentile	SD	Min	Max	1. Quartile	Median	3. Quartile	Adjective Scale	Quartile Scale	Acceptability Scale
Task1 - CS	75.0	72.77	10.7	60.0	92.5	67.5	71.25	86.875	Good	3rd	Acceptable
Task1 - ME	71.25	60.08	7.03	62.5	82.5	64.375	70.0	78.75	Good	3rd	Marginal
Task2 - CS	72.5	64.38	18.65	37.5	92.5	56.25	76.25	90.625	Good	3rd	Marginal
Task2 - ME	71.25	60.08	16.12	50.0	97.5	55.625	68.75	88.125	Good	3rd	Marginal
Task3 - CS	72.08	62.95	13.8	47.5	92.5	60.625	73.75	83.125	Good	3rd	Marginal
Task3 - ME	77.08	79.24	13.1	62.5	97.5	62.5	75.0	91.875	Good	3rd	Acceptable

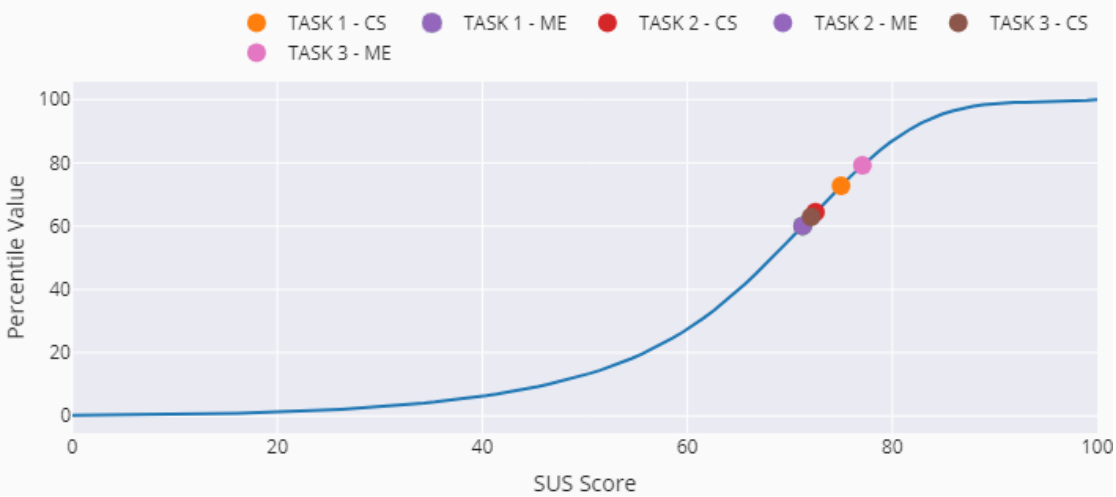


Figure 8.23: Percentile curve of the SUS questionnaire.

Figure 8.22 presents the SUS score as boxplot, prepared with an online tool for analyzing SUS questionnaire results [BBP22].

The *adjective scale* score in the boxplot is aligned with [Jef18], which is based on [BKM08]. The figure highlights that each task achieves the rating good for both CSs and MEs. The standard error of CSs is slightly higher than for MEs, which can also be seen in Table 8.3. The values of quartile scale shown in Table 8.3 are according to [BKM09] and acceptability scale according to [BKM08]. MEs increased the score in Task 3. Task 1 and Task 2 are equal. CSs decreased the score among the tasks. However, the changes in the scores are little and therefore not justifiable.

Figure 8.23 depicts the percentile scale based on [SL16]. Since the percentile score is not uniform or normally distributed, a percentile score was created based on 5000 SUS studies. In this respect, the comparison shows that the tests achieved a percentile between 60 and 79. Task 3 is over performed by MEs with 79. For CSs and MEs the average percentile is 66. Task 1 and Task 2 for MEs have exactly the same value, which is why they are shown as one color in the figure.

8.4 Discussion

This section discusses advantages and potential flaws of the newly introduced method to formalize DE tasks. The structure of the section is as follows: First, the metamodel's extension and the stereotypes' proposed structure are discussed. Next, the benefits and shortcomings of relationship modeling are assessed with a particular focus on the applicability and potential ambiguous interpretation. Next, potential risks of DE and future work are presented. Finally, the implications of the user study are presented and discussed.

8.4.1 Stereotypes and Structure of the Custom Metamodel

The integration of custom stereotypes has been proven beneficial in the literature [KSW04]. In this method, the use of stereotypes to encapsulate and abstract knowledge about DE tasks is beneficial as implementation details are hidden, thus supporting communication between different engineers not necessarily experienced in DE or programming. With structuring the stereotypes using packages, a stereotype organization aligned to the CRISP-DM methodology is given, supporting refinements and extension in a fine-grained, hierarchical manner. Particularly, the definition of blackbox and abstract stereotypes allows the description of various functions without the necessity to specify each DE function in detail. In the custom metamodel, custom *Enumerations* are defined to limit the number of attribute values, which reduces the model's wrong specifications. Another opportunity to reduce the scope of possible selections is to reduce the number of allowed stereotypes, e.g., only inheritance of the abstract stereotype *PreProcessing* can be assigned as a value for a specific attribute. However, the filtering of stereotypes requires specific rules that have not yet been integrated or elaborated. Although various methods are defined using stereotypes, the level of detail might be too little for practical application. *DateConversion*, for example, can be applied to manifold input values and various outputs, e.g., output representation as a string or Coordinated Universal Time (UTC). Adding multiple *DateConversion* stereotypes for each case is possible. Still, with a growing number of stereotypes, the complexity of selecting the correct, unambiguous stereotype increases while the maintainability decreases. Similarly, if too many stereotype attributes have to be set, the complexity and the effort for the application increases. With respect to these uncertainties at the level of detail required for fine-grained definition of DE tasks, industrial case studies have to be conducted to elaborate and validate sufficient degree of detail and additionally define future work.

8.4.2 Interpretability of Functional Specification

Defining an implementation structure that is consistent with the CRISP-DM methodology and ranges from business understanding to the definition of evaluation and workflows promises to be useful because of the integration of a comprehensive and mature method into an MBSE method. In addition, more experienced computer scientists who are familiar with CRISP-DM can draw on experience and the advantages of CRISP-DM. Furthermore, in practice, one-third of data scientists lack business understanding and communication skills [Ana22], which can be supported by CRISP-DM's model-based method. Although there is room for interpretation in the modelling approach, the model can be used as a basis for implementation. A more stringent definition of the modelling regarding output definition, and the potential extension by a model checker makes it even more suitable for widespread use. The modeling further enables reusability by defining building blocks representing a specific concern of interest. Based thereon, the reuse enables the preservation of knowledge and contributes to standardisation in modelling and implementation, which in turn leads to a reduction in costs and risks in the design and maintenance of applications. The use of model transformations on structured modeling

is shown in Chapter 9.

8.4.3 Potential of Model-Driven Machine Learning

The given proposal to describe DE tasks using a model-based method has some benefits but also disadvantages. A core disadvantage is the initial effort to introduce stereotypes and formalize the model. In this respect, traditional programming might be less time consuming and therefore, users might use the *CustomCode* stereotype to inject code. However, it is not the purpose of the method to insert code injection due to vulnerability risks and the reduced documentation and understanding by others. Consequently, future work is required to investigate an extension of the method that allows to generate code from the model but with limitations so that code injections like described in the second use case are not possible. Another disadvantage of the stereotypes is the potential effort for maintenance if interfaces are proprietary or rapidly changing, e.g., due to configuration changes or replacement of machines. Closely related, for huge projects, the complexity of the resulting models might be very high, including potential errors in the model or ambiguous associations, which might be very hard to find and thus lead to additional communication effort. Nevertheless, the shortcoming of a complex ramp-up might also be a benefit in the end due to the possibility of introducing model libraries containing well-defined models, leading to standardized parts that can be reused. Furthermore, the method allows to use the formalization as documentation of the implemented technologies that improve the maintainability and extendability for various engineers. Additionally, with further investigations regarding model validation and model debugging features, errors in the semantics can be found and repaired without actually implementing the DE application. However, to use this efficiently, the integration into advanced model lifecycle management [FNF+14] might be necessary to allow collaborative working.

Due to the non-programming description of DE tasks, the method is promising to increase the communication among various disciplines. In particular, with the integration of the general-purpose language SysML and the intersection of CRISP-DM and MBSE, the heterogeneous communities are broadly supported, which favors the implementation of DE in industrial practice and supports to shift knowledge in enterprises regarding data-driven supporting solutions. The integration of different disciplines and integration in MBSE methods is additionally an advantage over visual workflow modelers for ML, which typically address DE needs exclusively, such as RapidMiner[4], DataIKU[5] or KNIME[6]. Furthermore, the method can be integrated into early product development due to the abstract definition that allows to foresee various data interfaces which might have been forgotten during the development. This potentially leads to increased accuracy of the DE applications and might reduce e.g., failing DE projects, which is a well-known problem in industries [RR22].

[4] https://rapidminer.com/
[5] https://www.dataiku.com/
[6] https://www.knime.com/

Finally, a major advantage of formalized knowledge is the use of machine-readable artifacts (models). The use of model transformations enables the automatic generation of executable code using programming languages such as Python. This promotes the implementation of DE in practice. More details on the generation of code based on the model are given in Chapter 9.

8.4.4 Implications from the User Study

The user study was conducted with two groups that are representative for using the method presented in this work in practice. The results show that the majority of the tasks were successfully accomplished. From a study perspective, the users could perform each task without additional guidance on the modeling method. Still, problems occurred with the user-interface of Papyrus, e.g., expanding a group of elements to select a *block* element for modeling. However, learning effects could be observed with both CSs and MEs.

The assessment of the NASA-TLX showed that the mental demand for each task is comparable. A similar observation can be made for the level of frustration, which is slightly lower for the first task. Contrary to expectations, the participants perceived the effort as decreasing. With regard to the task, the effort for modeling should have been higher than for understanding a model. Nevertheless, it can be implied that both CSs and MEs can use the method in terms of task load without being more strained.

From an usability perspective, the method achieved good results. Users rated especially the consistency of the method as very high. Comparing the method with others using the percentile curve, it achieved a rank over 66.

However, the first positive results could be due to some shortcomings in the study design. In particular, the demand for rating Papyrus might have a larger impact on the study design than expected. The usability feeling of the users is more dedicated to the experience with Papyrus than to the method, although it was said before to focus on the method. In this respect, a paper prototype where users had to move paper snippets on the table might have been more valuable. Furthermore, most of the participants reported their data science knowledge as low and yet were able to explain what happens in a given model or create a model building block themselves. However, modeling their own data science application might not be possible, as the general understanding of data science is too low.

Nevertheless, it can be seen as a result of the study that the modeled knowledge can be used as a communication medium. Therefore, it should also be possible for non-data scientists to perform a plausibility analysis, as they can gain an understanding of the process without understanding programming.

However, this would need to be evaluated in a further study. Similarly, an evaluation of the results with the help of a larger study should be sought.

8.4.5 Implications for Industry

The method to formalize DE tasks using SysML promotes the integration of DE capabilities into MBSE and fosters the understanding of DE and particularly the concept of DDE in practice. Thus, an increased awareness of DE capabilities in a SE-related field is expected. Moreover, with the formalization of knowledge, validation and verification of a DE system is enabled, leading to further integration in complex environments, such as aviation [RBGM21]. By integrating methods from SE and DE, different sources of knowledge are brought together, enabling non-programmers to support the development of DE tasks. Hence, the formalization can be used as requirements and guide for implementing DE capabilities. Furthermore, through the integration of CRISP-DM, a method from the DE community is used, which supports the alignment with existing DE elaboration processes and thus existing processes can be used to a certain extent.

8.4.6 Implications for Research

The method presented here contributes to the consolidation of the different scientific communities of such as DE and engineering. With the integration of the GPML SysML, support for the elaboration of DE integrating methods in the field of MBSE is given. Additionally, with the formalization of knowledge, a contribution is made to the research branch of trustworthy AI. Finally, the accurate documentation of the application potentially promotes the transition of methods from academia to industry.

8.5 Summary

In this chapter, a method for DE task formalization using means of SysML is introduced. Particularly, the metamodel of SysML is extended with stereotypes to reflect functions from the DE domain. To guide the development of DE applications, the CRISP-DM methodology is used as basis for the structure of the models to organize the development with specific viewpoints. The method is evaluated in a case study showing the integration of DE task definition in a CPS as well as in a case study where a workflow engine is integrated for the interruption of a 3D printer task if the aimed result cannot be achieved. Additionally, a user study is performed to collect an overview of the perceived workload using NASA-TLX questionnaire and to check usability of the system using the SUS questionnaire. The findings of the evaluation showed that the entire workflow of a DE solution can be reflected using SysML and hence, guide the implementation of a DE application. Furthermore, the connection between the domain relevant knowledge for the implementation of (mechanical/electrical) engineers and DE experts is shown. With the MBSE integration and the involvement of various stakeholders from different disciplines, an improvement in communication is expected as shown in a user study. The user study implies that non-experts in DE can use the method as medium of communication. However, future work is required to validate the improvement of communication rather than referencing it as benefit [HS21]. Additionally, a case study is necessary to develop a minimum level of detail required to sufficiently define a DE model that can be used for

communication, and thus guide the implementation of the executable code through the formalization of the DE model.

Due to the findings of the method evaluation, the answer to RQ6.1 *"What means of SysML can be used to represent a sequence of Data Engineering statements?"* is that means of metamodel extension allows to define stereotypes that encapsulate a sequence of DE statements that can be specified using BDDs.

Closely related to the answer of RQ6.1 is the answer to RQ6.2 *"What means of SysML can be used to represent the order of execution?"*. To allow sequential execution of the sequence of DE task statements, activity diagrams are extended with a stereotype to associate a state with a block.

Based on the created stereotypes, the answer to RQ6.3 *"What stereotypes and associated structure need to be defined to enable reuse, extensibility and simplicity?"* is that a hierarchical composition of stereotypes enables the definition of small reusable DE functions. In particular, for each stereotype, a specific behavior must be defined that can be applied to a specific set of supplied information, e.g., date conversion is only applicable to a dataset with a date format and the only behavior of the stereotype is to convert a specific date to another format. Due to the hierarchical structure and the small separated stereotypes, the extension of the metamodel by various new stereotypes is possible. Even if a particular stereotype does not fit into the given hierarchy, it can be added, since DE stereotypes need only be derived from the root stereotype *DE* and the hierarchy only aims to support organizing the stereotypes.

In terms of organization, it was not only the composition of the stereotypes that was considered supportive. Therefore, the answer to RQ6.4 *"What means of graphical modeling can be used to represent and guide the development of Data Engineering tasks?"* is that the integration of a DE methodology such as CRISP-DM is beneficial to separate concerns of interest, e.g., the business understanding that is potentially modeled by engineers, and the data understanding, which is the first CS viewpoint.

CHAPTER 9

Data Engineering Code Generation using Model-Driven Techniques

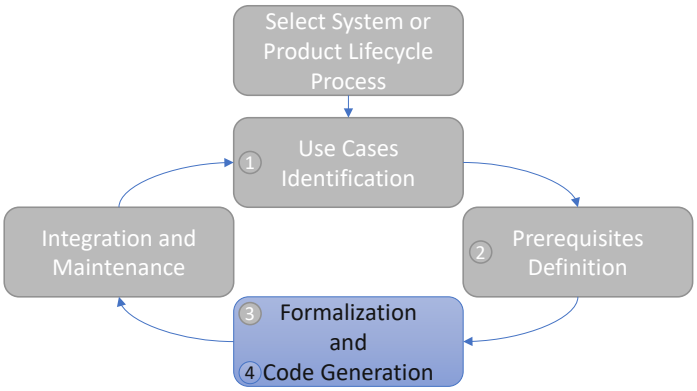
This chapter addresses the research objective ④ in Figure 9.1 related to the automatic code generation based on DE formalization using SysML as introduced in Chapter 8. As the figure shows, several research implications, such as reuse of validated and formalized knowledge to enable generation of code based on SysML formalization of DE tasks, and flexible and maintainable code generation can be deduced.

Based on the formalization of DE, this method aims to facilitate the implementation by leveraging MDE techniques, namely model transformation to automatically generate code for DE.

In this respect, the following general RQ is identified:

> **RQ7** Given a system model that represents data attributes, interfaces, and the formalization of Data Engineering tasks: What model properties can be used to automatically derive an executable Data Engineering model using Model-Driven Engineering techniques?

To address the identified RQ, the elaborated method to generate code based on model transformation is presented in the following. For evaluation purpose, the method is applied to the formalization of a use case facilitating sensor data to predict weather station data as illustrated in Section 5.2.2. The remainder of this chapter is organized as follows: First, a method is introduced, allowing to derive DE code using model transformation techniques. Next, an evaluation based on an open dataset regarding weather prediction is



Objectives	Research Implications	Challenges	Realization
1 Identify use cases for Data-Driven Engineering	• Identify relevant business processes • Identify data attributes and data interfaces • Identify potential in actual processes	• Diversity of processes • Distributed knowledge sources • Ambiguity of communication	Chapter 6
2 Define prerequisites for integrating Data-Driven Engineering	• Define data associations • Define automatic data collection • Integrate desired tool in actual processes	• Diversity of processes • Legacy hardware support • Diverse data level of detail	Chapter 7
3 Formalize Data Engineering tasks using SysML	• Formalize validated knowledge • Reuse of knowledge from MBSE • Enhanced interdisciplinary communication	• Ambiguous modeling • Complex formalization • Level of detail of formalization	Chapter 8
4 Automatic Code Generation for Data Engineering based on SysML	• Reuse validated and formalized knowledge • Enable flexibility and maintainability	• Decomposition of knowledge • Generation of executable code	Chapter 9

Figure 9.1: Overview of research objectives, implications and challenges addressed in Chapter 9.

presented. Finally, the results are discussed, future work is highlighted, and a conclusion is presented.

A selection of text, figures and tables within this chapter is based on the publication in box „Publications 7: Data Engineering Code Generation using Model-Driven Techniques“:

Publications 7: Data Engineering Code Generation using Model-Driven Techniques

[RRRR24] S. Rädler, M. Rupp, E. Rigger, and S. Rinderle-Ma, “Model-Driven Engineering for Machine Learning Code Generation using SysML” March 2024, doi: 10.18420/MODELLIERUNG2024_019.

9.1 Method

In the previous Chapter 8, a method to describe relevant information for implementing a DE task using SysML is introduced [RRMR22]. Particularly, the SysML model represents all information concerning the composition of various relevant systems, their related data interfaces and the formalization of relevant data transformation and DE-related tasks on

a single step (subtask) level. Additionally, the execution order of the DE tasks during the implementation is formalized using state diagrams. Each state of the diagram describes a set of sub-activities, e.g., a sequence of Python functions with a dedicated purpose, such as the transformation of *Datetime* into another format.

To enable code generation from the defined SysML model, the presented method relies on templates, defined as code snippets in a dedicated programming language, such as Python, and a mapping configuration that allows to identify a template based on a stereotype. The purpose of the template-based approach is to enable extendability and maintainability without the necessity to make changes in the model transformation. Hence, the underlying templates can be exchanged to allow the generation of a documentation for the SysML model. Additionally, an exchange of the template can be used to derive code in another programming language, such as JAVA[1], Python[2] or R[3]. Figure 9.2 depicts the generic method to generate DE code based on templates in a flow diagram aligned workflow with a sample model transformation depicted as images on top of the figure.

The transformation applies the following sequential steps:

1. A state diagram is provided as input, referencing each DE subtask formalized using stereotypes and blocks
2. For each of the states, which are provided in ascending order, the DE blocks are identified.
3. Based on the unique stereotype name, a template is selected.
4. Stereotype and block attributes ① are mapped to the template ③ using a mapping configuration ② to generate a code snippet ④ (see Figure 9.2)
5. A file is generated representing the executable code snippets in the correct execution order. In the actual prototype implementation, a Jupyter Notebook[4] is generated.

[1] https://www.java.com/de/
[2] https://www.python.org/
[3] https://www.r-project.org/
[4] https://jupyter.org/

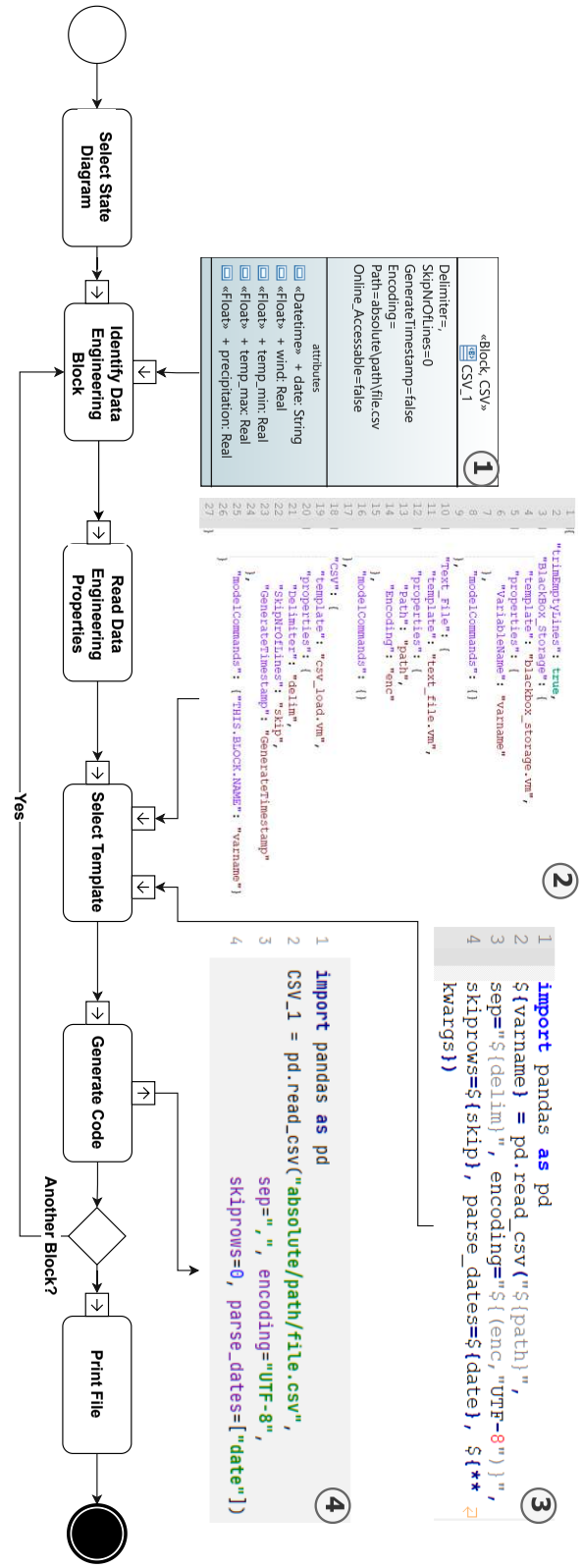


Figure 9.2: A sample model transformation to load a CSV file.

The model transformation in Figure 9.2 is slightly simplified to show the overall process of code generation. Therefore, the step of an intermediate transformation is omitted in the figure and introduced in Section 9.1.1. Next, the composition of the templates with placeholders is discussed. Finally, the mapping configuration is presented, focusing on model commands.

9.1.1 Intermediate Model

The purpose of the intermediate model is to extract information from the SysML model and to merge the state diagram information with the linked blocks. The source metamodel is a SysML profile, and the target metamodel is a custom one, referred to as "block context" in the following. The block context consists of the following parts:

First, a reference to the original block in the SysML model to allow change tracking and to potentially enable synchronizing changes in the generated code with the original model.

Second, a list of rich-text blocks that can be rendered as text before a code block, modeled as so-called *owned comments* in the SysML model. Note that rich text annotations are represented as text block cells in the current implementation. This is a special feature of Jupyter Notebooks and must be considered separately for other environments or programming languages, e.g., by representing the rich text as comments above the generated code.

Third, references to connected block contexts based on the qualified name, a unique identifier for named SysML elements. Due to the uniqueness of the qualified name, it can be used as an identifier for attributes or blocks.

Fourth, a list of block and stereotype attributes with their values. If a value is a primitive type, the value is used. Otherwise, the qualified name is stored and translated to a value during the assignment.

Finally, an integer value is defined in the state diagram stereotype to describe the execution order of the code snippets. The transformation is executed for each block connected to a state and for each adjacent block. Care is taken to prevent the multiple execution of the transformation for the same block by tracing the unique identifiers of a block.

9.1.2 Code Snippet Template Definition

The code snippets templates are defined in textual editors. Particularly, a template consists of formatted plain-text with various placeholders filled with property values from the stereotypes during the code generation. The marker ③ in Figure 9.2 depicts a sample of a template with all possible types of variables, which are:

1. Standard variables highlighted with *${variable name}*. In this case, the attribute is mandatory and has to be set in the model.

2. Optional variables: set with a default value in the variable definition, *${(variable name, default value)}*.

3. Arbitrary other attributes.

Since a function in a code snippet potentially consists of countless attributes, not all attributes must be defined in a stereotype, and it would not make sense due to the complexity for the user. Therefore, additional properties can be added to a block instance without being defined in the stereotype. These additional properties are added to a specific position in the template indicated by an anchor-indicator ***kwargs*. For an additional property to be used for the template function, the name of the additional property must be similar to the parameter name of the corresponding programming language function, but with two asterisks before. For example, if a parameter of a chart printing function in Python calls *X-Axis Name*, the attribute in the block must be named ***X-Axis Name*. The double-asterisks attributes are rendered to the template in the following format *attribute_name* = *attribute_value* without the double-asterisks.

9.1.3 Mapping Configuration

A mapping configuration in ③ in Figure 9.2 illustrates the content of a mapping between a stereotype and a code snippet template using the JSON[5] file format. The definition of the JSON mapping is depicted in Listing 9.1.

The mapping configuration is defined as follows:

First, the mapping allows defining whether empty lines shall be trimmed during the generation of the Jupyter Notebook (Line 2 in Listing 9.1). Second, the definition of constant values allows reusing specific strings as static text, e.g., as a global variable for all templates (Line 3-6 in Listing 9.1). The stereotype mapping (Line 7-18 in Listing 9.1) allows specifying which template to use for a stereotype. The stereotype mapping (Line 10-13 in Listing 9.1) defines the mapping of stereotype properties to template variables.

A command can be defined (Line 14-17 in Listing 9.1) and mapped to a variable by using the following keywords to collect information:

1. **THIS**: the information can be found in the block with the stereotype

2. **CONNECTED[Name="", Nr=0, StereotypeName="", AttributeValue={"AttributeName": ""}, OUTPUT_Name=""]**: the information can be found on an associated block based on a search query, e.g., CONNECTED[Name="CSV_-1"] to get attributes of the "CSV_1" block from the perspective of the block *Format_Date* in Figure 8.5

3. **BLOCK**: the information is stored on the block directly

[5] https://www.json.org/

```
{
  "trimEmptyLines": <true||false>,
  "constants": {
    "<TemplateVariableName>": "<ConstantValue>",
    ...
  },
  "stereotypeMappings": {
    "<StereotypeName>": {
      "template": "<TemplateName>",
      "properties": {
        "<stereotypeAttributeName>": "<TemplateVariableName>",
        ...
      },
      "modelCommands": {
        "<ModelCommandKeywordCombination>": "<
    ↪ TemplateVariableName>",
        ...
      }
    },
  "nameMappings": {
    "<BlockName>": {
      "template": "<TemplateName>",
      "properties": {
        "<PropertyOrStereotypeAttributeName>": "<
    ↪ TemplateVariableName>",
        ...
      },
      "modelCommands": {
        "<ModelCommandKeywordCombination>": "<
    ↪ TemplateVariableName>",
        ...
      }
    }
  }
}
```

Listing 9.1: JSON mapping structure.

4. **STEREOTYPE["StereotypeName"]**: the information is stored on a specifically applied stereotype (blocks can inherit from multiple stereotypes)

5. **NAME**: the information is the name of the block specified by the preceding keywords

6. **ATTRIBUTES**: the information is a list of attributes defined in a specific block

7. **STEREOTYPEofATTRIBUTE["AttributeName"]**: the information is stored in a data stereotype of an attribute, e.g., STEREOTYPEofATTRIBUTE["date"] to get the stereotype *Datetime* of the "date" attribute of the block *CSV_1* in Figure 8.5

8. **OUTPUT**: the information is the last declared variable name of the template, which refers to the block specified by the preceding keywords

The command's syntax consists of at least three keywords, separated by a period. The first keyword is either *THIS* or *CONNECTED* with a selector to choose the correct connected block. The second keyword is either *BLOCK* if the information is directly stored on the block or *STEREOTYPE* with a parameter specified for the stereotype name if it does not belong to the block itself. The third parameter is depicted in the enumeration list of keywords above with the item numbers 5-8. After the last keyword, it is always possible to select a value if the result is a list using square selector [*Nr.*]. After the *ATTRIBUTES* and *STEREOTYPEofATTRIBUTE*, optionally *ATTRIBUTES* or *STEREOTYPEofATTRIBUTE* can be defined again to dig deeper into specific information. The *OUTPUT* value is one of the essential values to connect a code block with the result of a previous one.

If a specific mapping is only applied to a specific block, name mapping can be used (Line 19-31 in Listing 9.1). Name mapping is similar to stereotype mapping, but it specifies the input model block via the block name instead of the stereotype name. The only difference is that properties can also be defined on the block without being defined on the stereotype. Name mappings take precedence over stereotype mappings if both apply for a block.

9.1.4 Composition of Code Snippets

Based on the generated code snippets and the defined execution order of the snippets, an executable file can be generated. The method presented in this work is implemented as an example for Jupyter Notebook. For this, the following steps for composition are conducted:

1. Rich-text information modeled as owned or applied comment is directly converted to a Jupyter rich-text cell.

2. The generated templates are put in a source-code cell. Each block context (intermediate model) from the state machine gets one source code cell and, optionally, one rich-text cell.

The code snippets are analyzed for *"from ... import ..."* or *"import ..."* lines of code to increase the readability and reduce potential errors due to multiple inputs of modules required. These lines are cut out and inserted in the first code cell on top of the Jupyter Notebook file.

After all block contexts are iterated over, the cells are put together as a single file, leading to an executable Jupyter Notebook file. Finally, the syntax is validated using specific tools such as runipy[6], so the execution is ensured. If the syntax is incorrect, the user is notified by the script. Still, the task is evaluated as completed, since also a partial code generation is considered valid, if only parts of the program are formalized. The validation for semantics is considered out of scope due to the high complexity.

9.2 Evaluation

The evaluation of the presented method aims to assess the feasibility and applicability of the method for generating executable DE code. As of [BCW17], two approaches can be followed to implement a model transformation, 1) using current high-level programming languages, APIs, and frameworks or 2) relying on MDE principles and dedicated languages such as ATL Transformation Language (ATL)[7], and Epsilon[7]. This evaluation uses traditional programming paradigms and the well-known high-level programming language JAVA. The JAVA implementation that enables code generation was programmed in a master's thesis that I supervised, and is available online[8]. The concept and methodology of code generation was developed and conceptually evaluated by me as part of this dissertation. The master's thesis extended the conceptual work to include software engineering topics such as extensibility, necessary fallbacks during generation and integration into Jupyter Notebooks.

In the following, a case study used for the evaluation is presented with the used artifacts from an open dataset. Additionally, an excerpt of the generated artifact is presented.

9.2.1 Case Study and Artifacts

The dataset for the evaluation is based on an open dataset[9] for weather prediction based on sensor data from a weather station. The scenario of a weather prediction based on weather station data is suitable for application in the engineering domain because the data comprises multiple sensors with different timestamps and sampling rates. Additionally,

[6] https://pypi.org/project/runipy/
[7] https://www.eclipse.org/atl/
[8] https://github.com/sraedler/MDE_for_ML_Generation/
[9] https://www.kaggle.com/datasets/ananthr1/weather-prediction

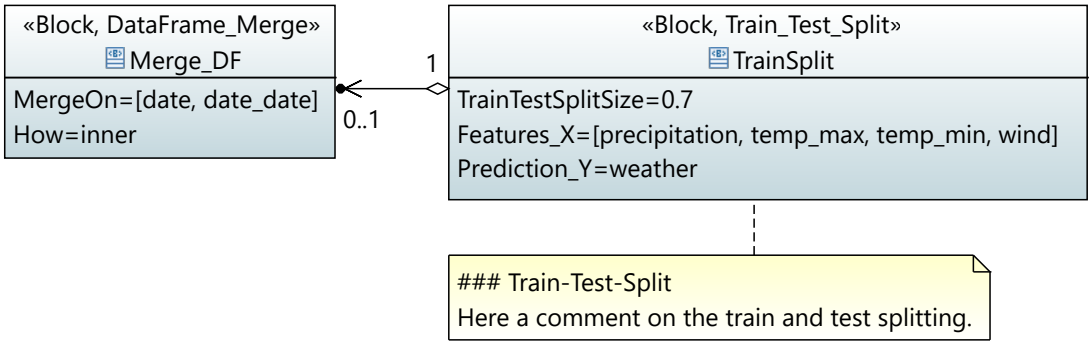


Figure 9.3: Sample input model.

the use of temperature or humidity sensors is also relevant in manufacturing specific components and their resulting quality. The model transformation concept is decoupled from DE in the SE domain and could therefore be evaluated for any DE problem.

9.2.2 Example Transformation

This section depicts the results from the model transformation applied to the model in [RRMR22].

Figure 9.3 to Figure 9.6 depict the four parts of the developed model transformation.

Figure 9.3 depicts two blocks with stereotype properties defined, and a block comment connected to a block, which is further used in the final Jupyter Notebook as Rich-Text Cell. The *TrainSplit* block is defined only by stereotype attributes. Additional attributes for hyper-parameter tuning, etc. are not defined. The composition indicates that the *Merge_DF* block is an input value for the *TrainSplit* function. Therefore, it is accessible through the *modelCommand* functionality defined in Listing 9.1.

To enable the mapping from the input model in Figure 9.3 to the output in Figure 9.6, a mapping configuration as defined in Figure 9.4 and a template as depicted in Figure 9.5 is required. The mapping configuration assigns a stereotype *Train_Test_Split* to a template with a name and, potentially, a path if sub-folders are used in the given structure. Each stereotype property is defined within the template's properties, whereas the left side of the assignment is the original variable in the stereotype and the right side is the placeholder in the template. The mapping defines two *modelCommands*, i.e., the first to get the name of the actual block and the second one to collect the output variable of the first connected block.

Figure 9.5 illustrates a sample code snippet for a DE function, more precisely, a template for the train-test-split. Within each template, necessary imports must be defined, and arranged at the end of the code generation, as defined in Section 9.1.4.

Figure 9.6 depicts the generated code based on the template and the input model attributes. As it can be seen, the formatting is aligned with the template in Figure 9.5.

```
"Train_Test_Split": {
  "template": "train_test_split.vm",
  "properties": {
    "Features_X": "feat_x",
    "Prediction_Y": "pred_y",
    "TrainTestSplitSize": "split"
  },
  "modelCommands": {
    "THIS.BLOCK.NAME": "split_name",
    "CONNECTED[0].BLOCK.OUTPUT": "new_name"
  }
},
```

Figure 9.4: Mapping configuration.

```
from sklearn.model_selection import train_test_split
X=${new_name}[${feat_x}]
y=${new_name}.${pred_y}
${split_name}_train_X, ${split_name}_test_X,\
${split_name}_train_y, ${split_name}_test_y = \
    train_test_split(X, y,random_state = 0, train_size=${split})
```

Figure 9.5: Template for the Train_Test_Split stereotype.

Train-Test-Split

Here a comment on the train and test splitting.

```
from sklearn.model_selection import train_test_split
X=Merge_DF[["precipitation", "temp_max", "temp_min", "wind"]]
y=Merge_DF.weather
TrainSplit_train_X, TrainSplit_test_X, \
TrainSplit_train_y, TrainSplit_test_y = \
    train_test_split(X, y,random_state = 0, train_size=0.7)
```

Figure 9.6: Result of the code generation.

9.3 Discussion

This section discusses the introduced code generation method for DE based on model transformation utilizing a SysML model with stereotypes. First, general advantages and disadvantages are discussed. Next, quality attributes of model transformation are discussed to allow an assessment of code generation. Finally, potential future work is presented.

9.3.1 Advantages and Disadvantages

Using model transformation to decompose formalized DE tasks is beneficial in several ways. First, it reduces the programming effort required for DE due to a substitution with modeling. Therefore, we assume that the effort for rarely available data scientists reduces [RR22] due to the decomposition of a model that partly can be elaborated by non-programmers, see Section 8.3.4. In addition, it allows the formalized knowledge in the model to be validated from an implementation point of view. Particularly, the correctness of the formalization in the SysML model can be validated by checking the sequence of statements in the generated code. Furthermore, generating code based on validated knowledge enables the creation of a proven DE model library, leading to potential standardization of DE implementation within an organization's infrastructure. This potentially favors the creation of DE tasks without profound programming knowledge.

However, the method can be very costly due to the initial effort required to create and validate templates. Still, the resulting templates lead to standardization and can thus be reused in multiple projects, which becomes an advantage in future projects. Another disadvantage of the example implementation is the complexity of the JAVA implementation. Particularly, means of ATL might fit better for attribute and type mappings. Furthermore, the approach leads to an overwhelming number of templates due to the number of algorithms and functions available for specific programming languages. Therefore, the integration of a component-based approach such as Orange Data Mining [10] would be fruitful. Finally, it should be mentioned that the transformation is currently only directional and does not allow changes in the generated code to be synchronized with the model. Thus, model transformation does not yet contribute to an authoritative source of truth in the sense of MBSE.

9.3.2 Quality Attributes of Model Transformation

Quality attributes of model transformation can be distinguished in direct assessment, which is the actual assessment of the model transformation and its properties, and indirect by analyzing the input and output artifacts, e.g., metamodels [van10, van12]. Furthermore, a distinction is made between internal quality, which focuses on development and maintenance, and external quality, which focuses on compliance with requirements and performance[van10, van12]. In the following, direct internal quality attributes are

[10] https://orangedatamining.com/

discussed. Although various metrics are available to assess these quality dimensions, a qualitative discussion is chosen. The metrics are adapted to the transformation language used, which is not applicable here as the implementation is not based on a transformation language, but on a general purpose programming language[vdBN10].

Understandability: The effort required to understand the purpose of the model transformation [vLv08].

From an computer scientist's perspective, the model transformation is easy to understand because a high-level programming language is used for the implementation, which can be adopted by most programmers. In contrast, the use of specific transformation languages such as ATL or Epsilon is less common for -MDE experts and therefore, easier to be adapted. Additionally, it is not trivial to perform model transformations in JAVA, precisely why transformation languages were developed. Furthermore, the overall concept of mapping model artifacts using a configuration in JSON file format is a simple technique with typical concepts known from programming. Still, the integration of different modules to build the whole model transformation is arguable compared to the integration of the mapping within the JAVA implemenation for code generation.

Modifiability: The effort required to adapt a model transformation to provide other or additional functions [vLv08].

The effort for modifications is potentially small because 1) the input metamodel can be adapted, and the concept of mapping attributes to a template is simple 2) the mapping configuration is highly customizable and can be adapted without profound programming experience, and 3) the output templates are small code snippets that can be formulated in any programming language. In addition, any functions can be added from the programming perspective by adding additional templates or stereotypes without touching the JAVA implementation. The mapping already provides *modelCommands*, allowing the collection of specific attributes or related information. Even if more complex extensions are required, such as inserting security-related code to authenticate users, this can be adapted due to the use of the high-level language JAVA.

However, to enable mapping from a stereotype to a template, one needs to become familiar with weaving techniques, which is less easy for MDE experts compared to a standard model transformation language, as MDE experts are already familiar with it.

Reusability: The extent to which parts of a model transformation can be reused by other model transformations [vLv08].

Due to the possibility to exchange the output templates, the transformation can be applied to any textual programming language that enables to be assembled from small code fragments and has abilities for DE capabilities, e.g., data pre-processing or ML algorithms. Similarly, the concept of transformation can be used for any other model-to-code generation that can be broken down into small fragments, as it is simply a mapping mechanism between input stereotype and output template, e.g., automatic documentation generation by exchanging templates.

Modularity: The extent to which a model transformation is systematically separated and structured[vLv08].

Modularity is given in two aspects. First, stereotypes can be arbitrarily organized as long as they inherit from the core *DE* stereotype. Second, output templates can be stored in folders to structure the organization of templates. However, the method does not allow defining a mapping only for a specific subset of functions. Therefore, always a single JSON is required to represent the mapping configuration of a single stereotype. Nevertheless, extending the method to include the ability to parse multiple JSON files for mapping configuration is possible with little effort, allowing for complete separation and modularization of certain aspects of transformation.

Completeness: The extent to which a model transformation is fully derived from the requirements [vLv08].

Completeness is executed concerning two kinds of requirements, functional requirements and non-functional requirements of the model transformation.

The functional requirements for the model transformation can be summarized as the ability to generate executable DE code, which is given as of the first evaluation.

From a non-functional perspective, aspects such as generation performance must be evaluated. Due to the early stage of development, the non-functional requirements are not yet assessed.

Consistency: The extent in which a model transformation is implemented in a uniform manner [vLv08].

Because of the few programming code lines that compile the input with the mapping configuration and templates to executable code, consistency in the sense of [vLv08] is not a major issue of the developed model transformations.

Conciseness: The extent to which a model transformation is free of superfluous elements [vLv08].

Due to the high entanglement of the mapping configuration and the code templates, superfluous elements are barely available. Additionally, the functionality to add arbitrary attributes to the generation using ***kwargs* reduces the number of superfluous elements. Furthermore, elements can be created during the modeling, and unnecessary templates may be defined. However, these expressions are part of the nature of the application rather than a weakness of the model transformation.

9.3.3 Implications for Industry

With the application of model transformation, the modeling of DE tasks is streamlined by using automatic code generation based on a decomposition of the DE formalization. Furthermore, with an increasing application of code snippets used for the code generation, DE standardization and the building of a standard procedure within a company is

fostered. In this respect, the efficiency of DE elaboration and a reduction of costs for the implementation are expected, leading to an increased amount of use cases supported by DE capabilities. Furthermore, the amount of data-driven decisions in MBSE environments is potentially fostered due to an increasing amount of implemented DE capabilities. Finally, automated code generated on the basis of validated knowledge can support the validation of DE applications. In this respect, the method supports industries with a high demand for proven, secure and maintainable applications [RBGM21].

9.3.4 Implications for Research

With the application of model transformation, a contribution to the MDE community is made by generating DE code based on the SysML. Additionally, the scientific research field of MDE is made visible in the DE domain, which potentially leads to further methods and applications of MDE4AI. Furthermore, the benefits of MDE for practitioners will be clarified, possibly leading to a greater awareness of MDE capabilities. Finally, a contribution to the transfer of methods from academia, especially from the field of MDE, to industry is expected.

9.3.5 Future Work

Future work involves implementing improvements and validating the method within user studies to prove its applicability in industrial projects. Furthermore, the systematic backflow of results from DE to the SysML model requires to be implemented to allow to use the yielded results in further Model-Based Engineering (MBE) methods. Similarly, it is beneficial if changes in the Jupyter Notebook can be traced back to the model so that synchronization and an authoritative source of truth[11] can be achieved. With respect to this, the actual transformation traces the model elements, allowing the identification of the origin, and within the Jupyter Notebook, unique block markers can be used to map the changes to the model elements. However, profound changes require a mechanism to generate further blocks or adapt templates.

Additionally, a comparable implementation using MDE languages such as ATL is desired to compare benefits and flaws. Particularly, a use study will be conducted to assess the advantages of the approach in combination with the method to formalize DE using SysML presented in Chapter 8.

9.4 Summary

This chapter presented a model transformation to facilitate DE applications using model-based techniques based on SysML. The goal of the code generation is to streamline the implementation of DE code within a company, enable to decompose formalized knowledge on DE tasks and prove feasibility of DE code generation based on SysML formalization.

[11] https://www.omgwiki.org/MBSE/doku.php?id=mbse:authoritative_source_of_truth

The code generation is enabled by generic templates providing concise code snippets that are mapped using a mapping configuration defined in the JSON file format to stereotypes or specific blocks in the SysML formalization. The generated executable code enables the validation of the formalized DE tasks in the SysML model from an implementation perspective.

Hence, the answer to RQ7 "*Given a system model that represents data attributes, interfaces, and the formalization of Data Engineering tasks: What model properties can be used to automatically derive an executable Data Engineering model using Model-Driven Engineering techniques?*" is that specific task-related templates are created with a relationship to defined stereotypes and attributes. The properties of the stereotypes, as well as the attributes and values of the model instance, are used to generate code snippets, which are organised in an execution order formalised using state diagrams. Additionally, small mutable templates are integrated with placeholders that are used to generate the final implementation code snippet. This also allows the templates to be transformed into other programming languages or adapted for a different purpose, such as generating documentation. By extending stereotypes and adapting or adding new templates, the method can be extended without changes to the code generation.

CHAPTER 10

Summary and Open Issues

The work presented in this thesis aims to support the integration and implementation of DE in engineering domains thus leveraging the advantages of DE in SE. This chapter summarizes the developed method to support the integration of DE in SE from the following perspectives. First, the applied research method based on Blessing and Chakrabarti [BC09] is discussed. Second, the implications of the elaborated method for industry and its contribution to research are highlighted. Finally, the limitations of the work are presented and future research is highlighted to promote the integration of MDE4AI and foster the application of DDE.

Applied Research Method

This work follows the descriptive-prescriptive-descriptive approach proposed in the Design Research Methodology (DRM) [BC09]. The first descriptive study (DS-I) consists of two parts, an SLR and an industry survey. The SLR allowed the identification of research gaps, becoming familiar with MDE4AI approaches, getting to know underlying modeling frameworks (e.g., WebGME, Monticore) and intents, and comparing different methods for the formalization of DE tasks. Within the first iteration of the SLR, few results were found due to the novelty of the research area [BBGW21, BKWZ21]. Still, the few identified publications could support and guide the development of the method in the prescriptive phase (PS-I) in terms of the tools and underlying frameworks used, the respective intentions and the values required to define DE tasks for code generation. To enable an updated SLR, the SLR protocol was re-executed during the course of the work to collect current approaches and compare the contributions of the work. The results were compared with the identified research gaps and have further supported the development of the methods from a scientific perspective.

From the industry's point of view, valuable insights on obstacles hindering the application of DE in practice could be obtained by an online survey. The results influenced the

development of the method in such a way that no further DSML was developed, but an extension of SysML has been developed that allows the integration with MBSE and thus, the integration in the state of the art methods of SE. Remarkably, the extension might be viewed as a DSML but still, it enables the integration in arbitrary SysML related methods.

Based on the results of the first descriptive study, a four-step method is developed in the prescriptive study (PS-I). The elaborated four steps can be grouped into two parts, first, *preconditions* for the formalization and integration of DDE, and second, the actual formalization of DE tasks with subsequent code generation. The first and second part are evaluated in separate case studies suitable for validating feasibility and applicability. Due to the division of the method, each part is evaluated separately. The two sub-methods of the two parts are elaborated and evaluated sequentially due to their internal dependence, whereas the evaluation of the two parts is executed concurrently.

In the second descriptive study (DS-II), the four-step method is evaluated against the use cases. The evaluation of the *preconditioning* steps was carried out in an industrial use case at the pilot factory of the TU Wien. The focus of the use case is to elaborate on existing data and processes and identify shortcomings that could be addressed using DDE capabilities.

The evaluation of DE task formalization with code generation is performed based on an open-source dataset for forecasting weather conditions. The use case demonstrated the feasibility of modeling DE tasks using SysML. Sill, a larger DDE use case is needed to evaluate whether the approach scales for complex tasks. Then, particular attention should be paid to the formalization of DE tasks, because a detailed formalization supports both automatic code generation and manual implementation.

Implication for Industry

The presented method builds upon graphical modeling languages to enable the formalization of relevant DE knowledge for the implementation and integration of existing processes.

A toolbox is elaborated to support domain experts' awareness on the opportunities of DDE in actual applied processes and IT applications. Particularly, the use of EA in combination with participative workshops enables the involvement of relevant stakeholders and supports the representation of various viewpoints in a graphical formalization of actual business processes with IT infrastructure. Based thereon, data attributes and interfaces are elaborated using graphical modeling, which again aims to support the understanding of potentials and opportunities of the involved domain experts. With the application of Value-Stream Mapping (VSM) and target integration modeling, the vision is made more tangible and can thus be discussed with management and other relevant stakeholders. The evaluation results revealed that the graphical formalization of the relevant knowledge enabled maintenance of the model when changes are required, e.g., if the validation of the

knowledge during the workshops shows shortcomings in data acquisition. Additionally, the graphical overview of the knowledge supports communication with relevant stakeholders. Consequently, the methods are favored as a medium for communication with external experts, e.g., to support the implementation of DE tools in the SE context. Furthermore, the toolbox might be used as basis for the integration of DDE capabilities within certain processes by formalizing the requirements of the aimed solution.

For integrating DE tasks into the overall MBSE ecosystem, SysML is chosen. With the integration of a SysML-based method to formalize DE capabilities, a medium for communication, documentation, and maintenance is fostered. In practice, the evaluation results showed that engineers unfamiliar with DE could use the modeling environment, after a short introduction, to understand and change entities of DE tasks. Therefore, the method is promising to allow the involvement of domain experts from the early concept phases until the implementation phase. Furthermore, with the application of stereotypes, standardization and reuse of the modeling is achieved, aiming to increase applicability and performance. Finally, the derivation of code snippets based on DE task formalization enables implementation performance and reuse modeled knoweldge.

Generally, the method promises to favor the transition from academia to industry, as it can be easily applied to existing processes and promotes an understanding of the need for DDE capabilities. This also makes it possible to increase the rate of supported decisions based on DDE, as shown in the impact diagram in Figure 1.2.

Contribution to Research

Based on the findings of the elaborated method and its evaluation using two use cases, the overall research question *"What means are required to support the implementation and integration of Data Engineering in Systems Engineering?"* is answered as follows:

Contribution 1: *Consolidation of knowledge regarding DE task formalization using MDE techniques (MDE4AI).*

An SLR is conducted to systematically collect and analyze literature from the field of MDE with a particular focus on support for the development of AI applications, which is called MDE4AI. From an MDE point of view, the interesting concerns are related to details on a language engineering level, e.g., metamodel, concrete syntax, and model transformation. DE concerns are assessed using typical implementation phases of CRISP-DM to enable comparability on the phases of development. With the collection and comparison of the addressed MDE4AI methods, the identification of commonalities and common research gaps regarding MDE4AI could be derived.

Particularly, the supported phases of CRISP-DM and the shortcomings of existing methods concerning the implementation support of DE could be uncovered. Existing implementation support focuses on the formalization of algorithm application rather than supporting early phases, such as CRISP-DM with its *business understanding* that relates to the understanding of the actual situation and integration with domain knowledge, and

the *data preparation* aiming to prepare datasets such that they are applicable for DE algorithms.

Concerning MDE, weaknesses in the adaptivity of the approaches for generalization and applicability in other use cases were identified. Particularly, less attention is paid to the extensibility of the approaches in terms of integrating algorithms or extending the model transformation.

Contribution 2: *Indication of opportunities of DE capabilities in SE.*

In SE, the literature reveals a demand for AI capabilities, in particular DE capabilities [BBGW21, BKWZ21, DFM+22]. In this regard, the method proposes to use model-based methods such as EA to identify potential use cases in current processes.

To facilitate the definition of DE tasks, means of SysML stereotypes are extended, allowing the description of DE functions and workflows. With SysML being the de facto standard for MBSE, the integration of DE capabilities in a model-based definition of SE is given. Furthermore, MBSE methods, such as the VAMOS method for variation modeling, are integrated, allowing to apply DE tasks on a particular configuration of a system.

Contribution 3: *A method that enables the identification of DDE opportunities and integration into current processes, reflects the needs of different stakeholders and viewpoints in practice and promotes the development of DE capabilities using model-based techniques.*

To favor the model-based development of DDE, means of EA and SysML are used to define current and target processes and allow the identification of shortcomings, potentially solvable using DDE. Particularly, the formalization and documentation of actual business processes and related IT artifacts allow for deriving use cases related to shortcomings and various stakeholders' viewpoints. Furthermore, the application of VSM and FMEA enables quantifying the identified use cases and promote the implementation of potentially most impacting DDE solutions first. Additionally, the target integration of the DDE solution and related data collection mechanisms allow for evaluating a DE application before implementation starts.

In summary, the goals and objectives of the thesis have been achieved and evaluated in case studies. By interpreting DE into MBSE, a step has been made regarding true MDE4AI. Furthermore, methods that can be used in practice have been developed, thus enabling a transition from research to industry.

Limitations and Future Work

This section summarizes the limitations of the developed method and the conducted work. These limitations might be used as starting point for potential future work. Aspects to be improved are the following:

1. The proposed method to support the implementation of DDE use cases focuses on the improvement of shortcomings embedded in existing processes. However, the application on newly introduced processes or during the development of a system is not evaluated. Hence, future research requires to focus on the applicability on early product design integration and the application on less known processes.

2. The group of the first two and the second two sub-methods were each evaluated sequentially in two separated use cases. Hence, it was assumed that the transition from the first two sub-methods to the second two sub-methods can be executed seamless without bigger shortcomings. Therefore, in future work, the entire method must be evaluated based on a single use case to validate that the transition of the sub-method groups is smoothly integrated.

3. The granularity of the modeling methods with respect to process decomposition using EA, and the formalization of DE tasks using SysML requires to be assessed and defined so to reduce unnecessarily complex formalization and effort during the elaboration. Furthermore, the method might be an overkill for small processes or small projects. Hence, future work requires to refine the method with a lightweight version that can be applied with less heavy methods and fewer effort doable for small projects.

4. The integration of DE formalization within the processes of MBSE requires the collaborative working on a single model. Currently, the software used assumes that only one engineer works on a model and collaborative work is not supported. Additionally, changes in the model are not automatically propagated or validated. Hence, inconsistent models might be formalized. Therefore, a method to enable the collaborative working, propagating of changes and automatic validation of the model is required.

5. The validation of the model requires manual work and no model-checker is implemented. Accordingly, the correctness of the generated artifacts is not given and only the Jupyter Notebook checking mechanisms are applied. Therefore, in future work, the correctness of the modeling in terms of the relationships of the blocks, as well as the syntax checking of the generated artifacts, must be investigated.

6. In the current approach, a code snippet must be prepared so that it can be used in the model transformation. This code snippet creation is purely manual and text-editor based, which means that syntactical errors cannot be eliminated and a potentially high effort in the creation can be expected. Accordingly, an editor should be created in future work that supports the generation of code snippets based on existing code and libraries. The syntactical correctness of snippets should also be checked.

7. The developed method was evaluated in two use cases, 1) in the pilot factory of the TU Wien and 2) using a weather station based weather forecast. Unfortunately, the approach was never evaluated in its entirety in a single use case, which would

have clearly demonstrated the consistency of the methods. Furthermore, the first use case was not evaluated in an industrial environment but in a pilot factory. Accordingly, in future work the entire method is to be implemented in a single industrial use case.

List of Figures

List of Tables

Acronyms

ACDP Austrian Center for Digital Production. 6

AI Artificial Intelligence. 7, 10, 13, 14, 17, 29, 31–39, 42, 44, 46, 49–53, 77, 78, 107, 119, 171, 191, 192, 199

AI4MDE Artificial Intelligence for Model-Driven Engineering. 17, 29

ANN Artificial Neural Networks. 38

API Application Programming Interface. 72, 81, 86, 140, 146, 181

ATL ATL Transformation Language. 181, 184, 185, 187

BDD Block Definition Diagram. 21, 86, 87, 99, 100, 117, 128, 136, 143, 160, 172, 196, 197

BPEL Business Process Execution Language. 109

BPMN Business Process Model and Notation. 25, 48, 109–111, 196

CAD Computer-Aided Design. 69, 71, 86, 89, 92, 113, 122, 123, 195, 196

CAM Computer-Aided manufacturing. 92, 122

CNC Computerized Numerical Control. 89

CPEE Cloud Process Execution Engine. 110, 111, 114, 115, 122, 124, 130, 135, 151, 196

CPPS Cyber-Physical Production System. 71, 109

CPS Cyber-Physical System. 4, 38, 48, 49, 71, 72, 109, 110, 145, 146, 171

CRISP-DM Cross Industry Standard Process for Data Mining. 10, 11, 13, 15, 31, 36, 45, 49, 50, 72, 73, 75, 77–79, 137, 139, 140, 143, 144, 146, 168, 169, 171, 172, 191, 196, 197

CS Computer Scientist. 159, 160, 162–167, 170, 172

DDE Data-Driven Engineering. 2–4, 6–8, 10, 11, 23, 55, 66–68, 73, 75, 77–80, 86, 89, 90, 96, 99, 105, 107, 109, 112, 120–122, 130, 132, 135, 136, 171, 189–193, 196, 199

DE Data Engineering. 1–4, 6–8, 10, 11, 13–15, 20, 23, 52, 55, 61, 63–65, 67, 71–73, 75, 77, 85, 86, 89, 96, 104, 107, 109, 112, 119, 120, 122, 127, 130–141, 143, 144, 146, 150–153, 159, 162, 167–175, 181, 182, 184–193, 195–197

DFSS Design For Six Sigma. 23

DL Deep Learning. 14

DM Data Mining. 10, 13, 14

DP-FMEA Design Process Failure Mode and Effect Analysis. 24

DRM Design Research Methodology. 5, 6, 189

DS Data Science. 1, 10, 13, 14, 55–62, 195

DS-I Descriptive Study I. 5, 6, 10, 189

DS-II Descriptive Study II. 6, 11, 190

DSML Domain Specific Modeling Language. 10, 17, 33, 34, 36, 38–42, 45, 137, 190

EA Enterprise Architecture. 10, 11, 13, 24–27, 66, 79–81, 83, 86, 89, 92–96, 103, 105, 110, 112–115, 119, 124, 130, 133–136, 190, 192, 193, 195, 196

EC Exclusion Criteria. 30, 33, 199

EGL Epsilon Generation Language. 43, 44

EGX EGL Co-Ordination Language. 43

EMF Eclipse Modeling Framework. 39–41, 48, 50

EOL Epsilon Object Language. 43

ERP Enterprise-Resource-Planning. 58

ESB Enterprise Service Bus. 110

ETL Extract, Transform, Load. 14

FMEA Failure Mode and Effect Analysis. 24, 77, 85, 103, 112, 192

GD&T Geometric Dimensioning and Tolerancing. 69

GPML General-Purpose Modeling Language. 4, 10, 17, 171

GQM Goal-Question-Metric. 79, 80, 103, 112, 120, 199

GSN Goal Structuring Notation. 38, 44, 48

IBD Internal Block Diagram. 86, 88, 89, 99, 101, 102, 118, 129, 136, 196, 197

IC Inclusion Criteria. 30, 33, 199

IIOT Industrial Internet of Things. 109

INCOSE International Council on Systems Engineering. 1, 19

IOT Internet of Things. 34, 39, 47–50

IT Information Technology. 2, 4, 11, 25, 58, 66, 68, 76, 82, 86, 103, 105, 119, 190, 192, 196

JSON JavaScript Object Notation. 67, 178, 185, 188

KMF Kevoree Modeling Framework. 40, 44

LSP Language Service Protocol. 40

MAE Mean Absolute Error. 47

MBD Model-Based Definition. 69

MBE Model-Based Engineering. 187

MBSE Model-Based Systems Engineering. 1–4, 6, 7, 10, 11, 20–23, 55, 67, 73, 137, 168, 169, 171, 184, 187, 190–193

MDE Model-Driven Engineering. 1, 4, 10, 13, 16, 17, 29, 31, 34–39, 44, 49, 50, 52, 61, 71, 132, 173, 181, 185, 187, 191, 192, 199

MDE4AI Model-Driven Engineering for Artificial Intelligence. 10, 17, 29, 31–33, 50–52, 187, 189, 191, 192

ME Mechanical Engineer. 159, 160, 162–167, 170

ML Machine Learning. 1, 3, 8, 10, 13, 14, 23, 36, 38–40, 44–46, 140, 143, 169, 185

MLP Multi-Layer Perception. 36

MPS Meta Programming System. 38, 41, 44, 48

MSB Manufacturing Service Bus. 110

MSE Mean Squared Error. 47

NASA-TLX NASA Task Load Index. 161, 164, 165, 170, 171, 197

NN Neuronal Networks. 36, 47

pdVSM Product Development Value Stream Mapping. 24, 79, 83, 84, 94, 95, 105, 196

PLC Programmable Logic Controller. 109

PLM Product Lifecycle Management. 1, 11, 55, 62, 72, 78, 79, 86, 99, 101, 105, 109, 127, 135

PS-I Prescriptive Study I. 5, 6, 11, 189, 190

REST Representational State Transfer. 110, 136, 151

RG Research Goal. 6, 29–31

ROI Return of Investment. 61

RQ Research Question. 6, 7, 30–32, 37, 55, 56, 63, 64, 75, 76, 107–109, 137–139, 173

SE Systems Engineering. 1–3, 6, 7, 10, 19–21, 23, 55, 63, 65, 66, 143, 145, 171, 182, 189–192, 195

SFC Sequential Function Charts. 109

SIPOC Supplier-Input-Process-Output-Customer. 23, 24, 78–82, 90–92, 103, 105, 196

SLR Systematic Literature Review. 7, 10, 29–31, 42, 51, 52, 137, 189, 191, 195, 199

SoI System of Interest. 63, 66, 67

SUS Systems Usability Scale. 161, 165–167, 171, 197, 199

SysML Systems Modeling Language. 4, 10, 11, 13, 17, 20–22, 35, 44, 46, 50, 66, 67, 79, 86–89, 99, 100, 102, 104, 105, 112, 115, 117, 118, 124, 128–130, 133, 136–140, 144, 145, 153, 159, 160, 169, 171–175, 177, 184, 187, 188, 190–193, 196, 197

UML Unified Modeling Language. 17, 21, 34, 39–41, 144, 163

UTC Coordinated Universal Time. 168

VAMOS Variant Modeling with SysML. 18, 19, 22, 143, 192, 195

VSM Value-Stream Mapping. 11, 24, 25, 103, 190, 192, 199

W-FMEA Waste Failure Mode and Effect Analysis. 24, 79, 85, 89, 96, 105

WebGME Web-based Generic Modeling Environment. 35, 39–41, 44, 48

XES eXtensible Event Stream. 110

Overview of generative AI tools used

During the writing of this thesis, the tools DeepL and Grammarly were used to achieve improved grammar and spelling.
Grammarly is an AI-based tool for identifying possible grammatical or spelling improvements.
DeepL is a translation program that helps with the translation or reformulation of certain text passages.

Bibliography

[ACMA20] S. Alsheibani, Y. Cheung, D. C. Messom, and M. Alhosni. Winning AI Strategy: Six-Steps to Create Value from Artificial Intelligence. *AMCIS 2020 Proceedings*, Aug. 2020. URL https://aisel.aisnet.org/amcis2020/adv_info_systems_research/adv_info_systems_research/1.

[AE12] T. F. Abdelmaguid and T. M. El-Hossainy. Optimal cutting parameters for turning operations with costs of quality and tool wear compensation. In *Proc. 2012 Int. Conf. Ind. Eng. Oper. Manag. Istanbul, Turkey, July 3*, volume 6, pages 924–932, 2012.

[AFG+21] D. Ameller, X. Franch, C. Gomez, S. Martinez-Fernandez, J. Araujo, S. Biffl, J. Cabot, V. Cortellessa, D. M. Fernandez, A. Moreira, H. Muccini, A. Vallecillo, M. Wimmer, V. Amaral, W. Bohm, H. Bruneliere, L. Burgueno, M. Goulao, S. Teufl, and L. Berardinelli. Dealing with Non-Functional Requirements in Model-Driven Development: A Survey. *IEEE Transactions on Software Engineering*, 47(4):818–835, Apr. 2021. doi:10.1109/TSE.2019.2904476.

[AH17] V. P. Andelfinger and T. Haenisch, editors. *Industrie 4.0: Wie cyber-physische Systeme die Arbeitswelt verändern.* Gabler Verlag, 2017. doi:10.1007/978-3-658-15557-5.

[Al-20] I. Al-Azzoni. Model Driven Approach for Neural Networks. In *2020 International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*, pages 87–94, 2020. doi:10.1109/IDSTA50958.2020.9264067.

[Ana22] Anaconda. State of Data Science Report 2022. Technical report, Anaconda, 2022. URL https://www.anaconda.com/state-of-data-science-report-2022.

[Archi19] The Open Group. ArchiMate® 3.1 Specification, 2019. URL https://pubs.opengroup.org/architecture/archimate3-doc/.

[Archi22] The Open Group Architecture Forum and The Open Group ArchiMate Forum. How to Use the ArchiMate® Modeling Language to Support

the TOGAF® Standard, Apr. 2022. URL https://publications.opengroup.org/g21e.

[Archi23] The Open Group. ArchiMate® 3.2 Specification - Relation to TOGAF, Jan. 2023. URL https://pubs.opengroup.org/architecture/archimate3-doc/ch-Relationship-to-Other-Standards-Specifications-and-Guidance-Documents.html#sec-The-TOGAF-Framework.

[Årz96] K.-E. Årzén. Grafchart: A Graphical Language for Sequential Supervisory Control Applications. *IFAC Proceedings Volumes*, 29(1):4831–4836, June 1996. doi:10.1016/S1474-6670(17)58445-1.

[AS08] A. Azevedo and M. F. Santos. KDD, SEMMA and CRISP-DM: A Parallel Overview. *IADIS European Conference Data Mining*, pages 182–185, 2008. URL http://hdl.handle.net/10400.22/136.

[ASX+20] R. Akkiraju, V. Sinha, A. Xu, J. Mahmud, P. Gundecha, Z. Liu, X. Liu, and J. Schumacher. Characterizing Machine Learning Processes: A Maturity Framework. In D. Fahland, C. Ghidini, J. Becker, and M. Dumas, editors, *Business Process Management*, pages 17–31, Cham, 2020. Springer International Publishing. doi:10.1007/978-3-030-58666-9_2.

[AVDS+17] G. Acampora, A. Vitiello, B. Di Stefano, W. van der Aalst, C. Gunther, and E. Verbeek. IEEE 1849: The XES Standard: The Second IEEE Standard Sponsored by IEEE Computational Intelligence Society [Society Briefs]. *IEEE Computational Intelligence Magazine*, 12(2):4–8, May 2017. doi:10.1109/MCI.2017.2670420.

[AZ13] A. Albers and C. Zingel. Challenges of Model-Based Systems Engineering: A Study towards Unified Term Understanding and the State of Usage of SysML. In M. Abramovici and R. Stark, editors, *Smart Product Engineering*, Lecture Notes in Production Engineering, pages 83–92, Berlin, Heidelberg, 2013. Springer. doi:10.1007/978-3-642-30817-8_9.

[Bak12] J. Baker. The Technology–Organization–Environment Framework. In Y. K. Dwivedi, M. R. Wade, and S. L. Schneberger, editors, *Information Systems Theory*, volume 28, pages 231–245. Springer New York, New York, NY, 2012. doi:10.1007/978-1-4419-6108-2_12.

[BB22] A. Beckhaus and H. Bertsch. How Traceability becomes the Enabler for Manufacturing Analytics, Mar. 2022. URL https://www.knime.com/blog/traceability-in-manufacturing-analytics.

[BBGW21] L. Burgueño, A. Burdusel, S. Gérard, and M. Wimmer. MDE Intelligence 2019: 1st Workshop on Artificial Intelligence and Model-Driven Engineering. In *Proceedings of the 22nd International Conference on Model Driven*

Engineering Languages and Systems, MODELS '19, pages 168–169, Munich, Germany, 2021. IEEE Press. doi:10.1109/MODELS-C.2019.00028.

[BBK+19] 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)*, pages 1607–1612, 2019. doi:10.1109/BigData47090.2019.9006518.

[BBP22] J. Blattgerste, J. Behrends, and T. Pfeiffer. A Web-Based Analysis Toolkit for the System Usability Scale. In *Proceedings of the 15th International Conference on PErvasive Technologies Related to Assistive Environments*, PETRA '22, pages 237–246, New York, NY, USA, July 2022. Association for Computing Machinery. doi:10.1145/3529190.3529216.

[BC09] L. T. M. Blessing and A. Chakrabarti. *DRM, a Design Research Methodology*. Springer, Dordrecht ; London, 2009. URL https://doi.org/10.1007/978-1-84882-587-1.

[BCF+19] L. Burgueño, F. Ciccozzi, M. Famelis, G. Kappel, L. Lambers, S. Mosser, R. F. Paige, A. Pierantonio, A. Rensink, R. Salay, G. Taentzer, A. Vallecillo, and M. Wimmer. Contents for a Model-Based Software Engineering Body of Knowledge. *Software and Systems Modeling*, 18(6):3193–3205, Dec. 2019. doi:10.1007/s10270-019-00746-9.

[BCJS92] D. Bell, L. Cox, S. Jackson, and P. Schaefer. Using causal reasoning for automated failure modes and effects analysis (FMEA). In *Annual Reliability and Maintainability Symposium 1992 Proceedings*, pages 343–353, Jan. 1992. doi:10.1109/ARMS.1992.187847.

[BCR94] V. R. Basili, G. Caldiera, and H. D. Rombach. The Goal Question Metric Approach. In *Encyclopedia of Software Engineering*, pages 528–532. Wiley Online Library, 1994.

[BCW17] M. Brambilla, J. Cabot, and M. Wimmer. *Model-Driven Software Engineering in Practice*. Synthesis Lectures on Software Engineering. Morgan and Claypool Life Sciences, San Rafael, Calif., 2 edition, Mar. 2017. doi:10.1007/978-3-031-02549-5.

[BCWZ22] L. Burgueño, J. Cabot, M. Wimmer, and S. Zschaler. Guest editorial to the theme section on AI-enhanced model-driven engineering. *Software and Systems Modeling*, 21(3):963–965, June 2022. doi:10.1007/s10270-022-00988-0.

[BKM08] A. Bangor, P. T. Kortum, and J. T. Miller. An Empirical Evaluation of the System Usability Scale. *International Journal of Human-Computer Interaction*, 24(6):574–594, July 2008. doi:10.1080/10447310802205776.

[BKM09] A. Bangor, P. Kortum, and J. Miller. Determining what individual SUS scores mean: Adding an adjective rating scale. *Journal of usability studies*, 4(3):114–123, 2009. URL https://uxpajournal.org/de/determining-what-individual-sus-scores-mean-adding-an-adjective-rating-scale/.

[BKWZ21] L. Burgueño, M. Kessentini, M. Wimmer, and S. Zschaler. MDE Intelligence 2021: 3rd Workshop on Artificial Intelligence and Model-Driven Engineering. *2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pages 148–149, 2021. doi:10.1109/MODELS-C53483.2021.00026.

[BNS03] P. Bernus, L. Nemes, and G. Schmidt, editors. *Handbook on Enterprise Architecture*. Springer, Berlin, Heidelberg, 2003. doi:10.1007/978-3-540-24744-9.

[BOF+14] 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. Technical report, INCOSE, San Diego, California, 2014. URL https://www.incose.org/docs/default-source/se-vision-2025/se-vision-2025/incose-se-vision-2025.pdf.

[BPR21] M. Brunnbauer, G. Piller, and F. Rothlauf. Idea-AI: Developing a Method for the Systematic Identification of AI Use Cases. In *AMCIS 2021 Proceedings*, Aug. 2021. URL https://aisel.aisnet.org/amcis2021/art_intel_sem_tech_intelligent_systems/art_intel_sem_tech_intelligent_systems/17.

[BPR22] M. Brunnbauer, G. Piller, and F. Rothlauf. Top-Down or Explorative? A Case Study on the Identification of AI Use Cases. In *PACIS 2022 Proceedings*, July 2022. URL https://aisel.aisnet.org/pacis2022/161.

[Bre14] D. Breuker. Towards Model-Driven Engineering for Big Data Analytics – An Exploratory Analysis of Domain-Specific Languages for Machine Learning. In *2014 47th Hawaii International Conference on System Sciences*, pages 758–767, Waikoloa, HI, Jan. 2014. IEEE. doi:10.1109/HICSS.2014.101.

[BRJ17] R. Bakelaar, E. Roubtsova, and S. Joosten. A Framework for Visualization of Changes of Enterprise Architecture. In B. Shishkov, editor, *Business Modeling and Software Design*, Lecture Notes in Business Information Processing, pages 140–160, Cham, 2017. Springer International Publishing. doi:10.1007/978-3-319-57222-2_7.

[Bro96] J. Brooke. SUS: A 'Quick and Dirty' Usability Scale. *Usability Evaluation In Industry*, pages 207–212, June 1996. doi:10.1201/9781498710411-35.

[BVR21] S. Bitrus, I. Velkavrh, and E. Rigger. Applying an Adapted Data Mining Methodology (DMME) to a Tribological Optimisation Problem. In P. Haber, T. Lampoltshammer, M. Mayr, and K. Plankensteiner, editors, *Data Science – Analytics and Applications*, pages 38–43, Wiesbaden, 2021. Springer Fachmedien. doi:10.1007/978-3-658-32182-6_7.

[Cao17] L. Cao. Data Science: A Comprehensive Overview. *ACM Computing Surveys*, 50(3):43:1–43:42, June 2017. doi:10.1145/3076253.

[CCK+00] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, and R. Wirth. Step-by-step data mining guide. *SPSS inc.*, 1.0:76, 2000. URL https://www.semanticscholar.org/paper/CRISP-DM-1.0%3A-Step-by-step-data-mining-guide-Chapman-Clinton/54bad20bbc7938991bf34f86dde0babfbd2d5a72.

[CGY+12] N. Chungoora, G. Gunendran, R. Young, Z. Usman, N. Anjum, C. Palmer, J. Harding, K. Case, and A. Cutting-Decelle. Extending product lifecycle management for manufacturing knowledge sharing. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 226:2047–2063, Dec. 2012. doi:10.1177/0954405412461741.

[CHR+20] B. Camburn, Y. He, S. Raviselvam, J. Luo, and K. Wood. Machine Learning-Based Design Concept Evaluation. *Journal of Mechanical Design*, 142(3), Jan. 2020. doi:10.1115/1.4045126.

[CI06] L. P. Chao and K. Ishii. Design Process Error Proofing: Failure Modes and Effects Analysis of the Design Process. *Journal of Mechanical Design*, 129(5):491–501, July 2006. doi:10.1115/1.2712216.

[CKCR03] S. Crosby, A. Kundu, R. Curran, and S. Raghunathan. Fabrication and Assembly Cost Drivers for Aircraft Manufacturing. In *AIAA's 3rd Annual Aviation Technology, Integration, and Operations (ATIO) Forum*, Aviation Technology, Integration, and Operations (ATIO) Conferences. American Institute of Aeronautics and Astronautics, Nov. 2003. doi:10.2514/6.2003-6827.

[CNdST+13] V. Carvalho, J. Nardi, M. d. G. da Silva Teixeira, R. Guizzardi, and G. Guizzardi. Towards a Semantic Alignment of the ArchiMate Motivation Extension and the Goal-Question-Metric Approach. In *CEUR Workshop Proceedings*, volume 1041, Sept. 2013.

[CNYM12] L. Chung, B. A. Nixon, E. Yu, and J. Mylopoulos. *Non-Functional Requirements in Software Engineering*. Springer Science & Business Media, Dec. 2012.

[Cre14] J. W. Creswell. *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*. SAGE Publications, Thousand Oaks, 4th edition, 2014.

[CWS+18] B. Chen, J. Wan, L. Shu, P. Li, M. Mukherjee, and B. Yin. Smart Factory of Industry 4.0: Key Technologies, Application Case, and Challenges. *IEEE Access*, 6:6505–6519, 2018. doi:10.1109/ACCESS.2017.2783682.

[DB21] A. Dogan and D. Birant. Machine learning and data mining in manufacturing. *Expert Systems with Applications*, 166:114060, Mar. 2021. doi:10.1016/j.eswa.2020.114060.

[dC14] R. V. B. de Souza and L. C. R. Carpinetti. A FMEA-based approach to prioritize waste reduction in lean implementation. *International Journal of Quality & Reliability Management*, 31(4):346–366, Apr. 2014. doi:10.1108/IJQRM-05-2012-0058.

[DFM+22] C. Davey, S. Friedenthal, S. Matthews, D. Nichols, P. Nielsen, C. Oster, T. Riethle, G. Roedler, P. Schreinemakers, E. Sparks, and H. Stoewer. Systems Engineering Vision 2035 – Engineering Solutions for a Better World. Technical report, INCOSE, San Diego, California, 2022. URL https://www.incose.org/docs/default-source/se-vision/incose-se-vision-2035.pdf.

[DIN03] DIN Deutsches Institut fur Normung e. V. IEC 61131-3: Speicherprogrammierbare Steuerungen-Teil 3: Programmiersprachen. *Berlin: Beuth*, 2003.

[DSL21] Q. Demlehner, D. Schoemer, and S. Laumer. How can artificial intelligence enhance car manufacturing? A Delphi study-based identification and assessment of general use cases. *International Journal of Information Management*, 58:102317, June 2021. doi:10.1016/j.ijinfomgt.2021.102317.

[ED19] F. Emmert-Streib and M. Dehmer. Defining Data Science by a Data-Driven Quantification of the Community. *Machine Learning and Knowledge Extraction*, 1(1):235–251, Mar. 2019. doi:10.3390/make1010015.

[EKL07] K. Ehrlenspiel, A. Kiewert, and U. Lindemann. *Cost-Efficient Design*. Springer-Verlag, Berlin Heidelberg, 2007. URL https://www.springer.com/de/book/9783540346470.

[Eri05] C. A. Ericson. *Hazard Analysis Techniques for System Safety*. John Wiley & Sons, Inc., Hoboken, NJ, USA, July 2005. doi:10.1002/0471739421.

[Est07] J. A. Estefan. Survey of model-based systems engineering (MBSE) methodologies. *Incose MBSE Focus Group*, 25:1–70, 2007. URL https://edisciplinas.usp.br/pluginfile.php/5348231/mod_resource/content/1/MBSE_Methodology_Survey_RevB.pdf.

[Eur20] European Commission. Joint Research Centre. *AI Watch: Defining Artificial Intelligence : Towards an Operational Definition and Taxonomy of Artificial Intelligence.* Publications Office, LU, 2020. URL https://data.europa.eu/doi/10.2760/382730.

[EW22] J. A. Estefan and T. Weilkiens. MBSE Methodologies. In A. M. Madni, N. Augustine, and M. Sievers, editors, *Handbook of Model-Based Systems Engineering*, pages 1–40. Springer International Publishing, Cham, 2022. doi:10.1007/978-3-030-27486-3_12-1.

[Fau03] L. Faulkner. Beyond the five-user assumption: Benefits of increased sample sizes in usability testing. *Behavior Research Methods, Instruments, & Computers*, 35(3):379–383, Aug. 2003. doi:10.3758/BF03195514.

[FMS19] T. Fountaine, B. McCarthy, and T. Saleh. Building the AI-Powered Organization. *Harvard Business Review*, July 2019. URL https://hbr.org/2019/07/building-the-ai-powered-organization.

[FNF+14] 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*, 24(1):207–229, 2014. doi:10.1002/j.2334-5837.2014.tb03145.x.

[Fow10] M. Fowler. *Domain Specific Languages.* Addison-Wesley Professional, 1st edition, 2010.

[FPS96] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From Data Mining to Knowledge Discovery in Databases. *AI Magazine*, 17(3):37–37, Mar. 1996. doi:10.1609/aimag.v17i3.1230.

[FZZ+20] Y. Feng, Y. Zhao, H. Zheng, Z. Li, and J. Tan. Data-driven product design toward intelligent manufacturing: A review. *International Journal of Advanced Robotic Systems*, 17(2), Mar. 2020. doi:10.1177/1729881420911257.

[GBC16] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* Adaptive Computation and Machine Learning. The MIT Press, Cambridge, Massachusetts, 2016.

[Ger17] D. Gerhard. Product Lifecycle Management Challenges of CPPS. In S. Biffl, A. Lueder, and D. Gerhard, editors, *Multi-Disciplinary Engineering for Cyber-Physical Production Systems: Data Models and Software Solutions for Handling Complex Engineering Projects*, pages 89–110. Springer International Publishing, Cham, 2017. doi:10.1007/978-3-319-56345-9_4.

[GET19] D. Gurdur, J. El-khoury, and M. Torngren. Digitalizing Swedish industry: What is next? *Computers in Industry*, 105:153–163, Feb. 2019. doi:10.1016/j.compind.2018.12.011.

[GGC22] J. Giner-Miguelez, A. Gómez, and J. Cabot. DescribeML: A tool for describing machine learning datasets. In *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, pages 22–26, Montreal Quebec Canada, Oct. 2022. ACM. doi:10.1145/3550356.3559087.

[Gil16] A. Gilchrist. *Introduction to the Industrial Internet*, pages 1–12. Apress, Berkeley, CA, 2016. doi:10.1007/978-1-4842-2047-4_1.

[GM23] M. Gabbrielli and S. Martini. Functional Programming Paradigm. In M. Gabbrielli and S. Martini, editors, *Programming Languages: Principles and Paradigms*, pages 335–368. Springer International Publishing, Cham, 2023. doi:10.1007/978-3-031-34144-1_11.

[GMM+22] J. Grueger, L. Malburg, J. Mangler, Y. Bertrand, S. Rinderle-Ma, R. Bergmann, and E. S. Asensio. SensorStream: An XES Extension for Enriching Event Logs with IoT-Sensor Data, June 2022, 2206.11392. doi:10.48550/arXiv.2206.11392.

[GP11] D. Greefhorst and E. Proper. The Role of Enterprise Architecture. In D. Greefhorst and E. Proper, editors, *Architecture Principles: The Cornerstones of Enterprise Architecture*, The Enterprise Engineering Series, pages 7–29. Springer, Berlin, Heidelberg, 2011. doi:10.1007/978-3-642-20279-7_2.

[Hag21] J. A. Hagen. Saving IS From Another Troubled Marriage: Diagnosing and Bridging the Gap Between Data Science and Other Business Functions. *PACIS 2021 Proceedings*, July 2021. URL https://aisel.aisnet.org/pacis2021/203.

[Har06] S. G. Hart. Nasa-Task Load Index (NASA-TLX); 20 Years Later. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 50(9):904–908, Oct. 2006. doi:10.1177/154193120605000909.

[HdFV19] R. Haberfellner, O. de Weck, E. Fricke, and S. Voessner. *Systems Engineering: Fundamentals and Applications*. Springer International Publishing, Cham, 2019. doi:10.1007/978-3-030-13431-0.

[HGH+20] J. Huang, A. Gheorghe, H. Handley, P. Pazos, A. Pinto, S. Kovacic, A. Collins, C. Keating, A. Sousa-Poza, G. Rabadi, R. Unal, T. Cotter, R. Landaeta, and C. Daniels. Towards digital engineering: The advent of digital systems engineering. *International Journal of System of Systems Engineering*, 10(3):234–261, Jan. 2020. doi:10.1504/IJSSE.2020.109737.

[HGZ19] K. He, M. Gao, and Z. Zhao. Soft Computing Techniques for Surface Roughness Prediction in Hard Turning: A Literature Review. *IEEE Access*, 7:89556–89569, 2019. doi:10.1109/ACCESS.2019.2926509.

[Hig19] High-Level Expert Group on AI. Ethics guidelines for trustworthy AI | Shaping Europe's digital future, Apr. 2019. URL https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai.

[Hig20] High-Level Expert Group on AI. Assessment List for Trustworthy Artificial Intelligence (ALTAI) for self-assessment | Shaping Europe's digital future, July 2020. URL https://digital-strategy.ec.europa.eu/en/library/assessment-list-trustworthy-artificial-intelligence-altai-self-assessment.

[HJP+20] P. Hofmann, J. Joehnk, Project Group Business & Information Systems Engineering of the Fraunhofer FIT, University of Bayreuth, Bayreuth, Germany, D. Protschky, University of Bayreuth, Bayreuth, Germany, N. Urbach, and FIM Research Center, University of Bayreuth, Bayreuth, Germany. Developing Purposeful AI Use Cases – A Structured Method and Its Application in Project Management. In *WI2020 Zentrale Tracks*, pages 33–49. GITO Verlag, Mar. 2020. doi:10.30844/wi_2020_a3-hofmann.

[HMFl17] T. Hartmann, A. Moawad, F. Fouquet, and Y. le Traon. The next Evolution of MDE: A Seamless Integration of Machine Learning into Domain Modeling. In *Proceedings of the ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems*, MODELS '17, page 180, Austin, Texas, 2017. IEEE Press. doi:10.1109/MODELS.2017.32.

[HMR+19] C. Hartsell, N. Mahadevan, S. Ramakrishna, A. Dubey, T. Bapty, T. Johnson, X. Koutsoukos, J. Sztipanovits, and G. Karsai. Model-Based Design for CPS with Learning-Enabled Components. In *Proceedings of the Workshop on Design Automation for CPS and IoT*, DESTION '19, pages 1–9, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3313151.3313166.

[HMS+19] T. Hartmann, A. Moawad, C. Schockaert, F. Fouquet, and Y. Le Traon. Meta-Modelling Meta-Learning. In *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pages 300–305, Sept. 2019. doi:10.1109/MODELS.2019.00014.

[HMU+20] J. Hehn, D. Mendez, F. Uebernickel, W. Brenner, and M. Broy. On Integrating Design Thinking for Human-Centered Requirements Engineering. *IEEE Software*, 37(2):25–31, Mar. 2020. doi:10.1109/MS.2019.2957715.

[HNM88] P. A. HANCOCK and E. NAJMEDIN MESHKATI. Human mental workload. *Human mental workload*, 52:XVI–382 p, 1988.

[HP13] J. Holt and S. Perry. *SysML for Systems Engineering: A Model-Based Approach*. Computing and Networks. Institution of Engineering and Technology, 2013.

[HPH22] J. Hagen, J.-A. Pély, and T. Hess. Collaborative mechanisms for big data analytics projects: Building bridges over troubled waters. *ECIS 2022 Research Papers*, June 2022. URL https://aisel.aisnet.org/ecis2022_rp/19.

[HPv09] F. Hermans, M. Pinzger, and A. van Deursen. Domain-Specific Languages in Practice: A User Study on the Success Factors. In A. Schuerr and B. Selic, editors, *Model Driven Engineering Languages and Systems*, Lecture Notes in Computer Science, pages 423–437, Berlin, Heidelberg, 2009. Springer. doi:10.1007/978-3-642-04425-0_33.

[HR97] P. Hines and N. Rich. The seven value stream mapping tools. *International Journal of Operations & Production Management*, 17:46–64, Jan. 1997. doi:10.1108/01443579710157989.

[HRT+19] M. Hennig, G. Reisinger, T. Trautner, P. Hold, D. Gerhard, and A. Mazak. TU Wien Pilot Factory Industry 4.0. *Procedia Manufacturing*, 31:200–205, 2019. doi:10.1016/j.promfg.2019.03.032.

[HS88] S. G. Hart and L. E. Staveland. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In P. A. Hancock and N. Meshkati, editors, *Advances in Psychology*, volume 52 of *Human Mental Workload*, pages 139–183. North-Holland, Jan. 1988. doi:10.1016/S0166-4115(08)62386-9.

[HS19] T. Huldt and I. Stenius. State-of-practice survey of model-based systems engineering. *Systems Engineering*, 22(2):134–145, Mar. 2019. doi:10.1002/sys.21466.

[HS21] K. Henderson and A. Salado. Value and benefits of model-based systems engineering (MBSE): Evidence from the literature. *Systems Engineering*, 24(1):51–66, Jan. 2021. doi:10.1002/sys.21566.

[HSM+19] M. Hesenius, N. Schwenzfeier, O. Meyer, W. Koop, and V. Gruhn. Towards a Software Engineering Process for Developing Data-Driven Applications. In *2019 IEEE/ACM 7th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering (RAISE)*, pages 35–41, Montreal, QC, Canada, May 2019. IEEE. doi:10.1109/RAISE.2019.00014.

[HZZ18] C. L. He, W. J. Zong, and J. J. Zhang. Influencing factors and theoretical modeling methods of surface roughness in turning process: State-of-the-art. *International Journal of Machine Tools and Manufacture*, 129:15–26, June 2018. doi:10.1016/j.ijmachtools.2018.02.001.

[ICM+20] A. Iung, J. Carbonell, L. Marchezan, E. Rodrigues, M. Bernardino, F. P. Basso, and B. Medeiros. Systematic mapping study on domain-specific

language development tools. *Empirical Software Engineering*, 25(5):4205–4249, Sept. 2020. doi:10.1007/s10664-020-09872-1.

[IGBD15] P. Iwanek, J. Gausemeier, M. Bansmann, and R. Dumitrescu. Integration of intelligent features by model-based systems engineering. In *Proceedings of 18th ISERD International Conference*, Tokyo, Japan, Nov. 2015.

[IJ07] M.-E. Iacob and H. Jonkers. Quantitative Analysis of Service-Oriented Architectures:. *International Journal of Enterprise Information Systems*, 3(1):42–60, Jan. 2007. doi:10.4018/jeis.2007010103.

[Jef18] S. Jeff. 5 Ways to Interpret a SUS Score – MeasuringU, Sept. 2018. URL https://measuringu.com/interpret-sus-score/.

[JHWL21] S. Jiang, J. Hu, K. L. Wood, and J. Luo. Data-Driven Design-By-Analogy: State-of-the-Art and Future Directions. *Journal of Mechanical Design*, 144(2), Sept. 2021. doi:10.1115/1.4051681.

[JSD+22] R. Jolak, M. Savary-Leblanc, M. Dalibor, J. Vincur, R. Hebig, X. L. Pallec, M. Chaudron, S. Gérard, I. Polasek, and A. Wortmann. The influence of software design representation on the design communication of teams with diverse personalities. In *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems*, pages 255–265, Montreal Quebec Canada, Oct. 2022. ACM. doi:10.1145/3550355.3552398.

[KB13] B. Kitchenham and P. Brereton. A systematic review of systematic review process research in software engineering. *Information and Software Technology*, 55(12):2049–2075, Dec. 2013. doi:10.1016/j.infsof.2013.07.010.

[KB19] B. Kruse and M. Blackburn. Collaborating with OpenMBEE as an Authoritative Source of Truth Environment. *Procedia Computer Science*, 153:277–284, 2019. doi:10.1016/j.procs.2019.05.080.

[KBAK96] J. R. Koza, F. H. Bennett, D. Andre, and M. A. Keane. Automated Design of Both the Topology and Sizing of Analog Electrical Circuits Using Genetic Programming. In J. S. Gero and F. Sudweeks, editors, *Artificial Intelligence in Design '96*, pages 151–170. Springer Netherlands, Dordrecht, 1996. doi:10.1007/978-94-009-0279-4_9.

[KBC+19] N. Kahani, M. Bagherzadeh, J. R. Cordy, J. Dingel, and D. Varró. Survey and classification of model transformation tools. *Software & Systems Modeling*, 18(4):2361–2397, Aug. 2019. doi:10.1007/s10270-018-0665-6.

[KC07] B. A. Kitchenham and S. Charters. Guidelines for performing Systematic Literature Reviews in Software Engineering. Technical Report EBSE 2007-001, Keele University, July 2007. URL https://www.elsevier.com/__data/promis_misc/525444systematicreviewsguide.pdf.

[KCCR02] A. Kundu, R. Curran, S. Crosby, and S. Ragunathan. Rapid Cost Modelling at the Conceptual Stage of Aircraft Design. In *AIAA's Aircraft Technology, Integration, and Operations (ATIO) 2002 Technical Forum*, Los Angeles, California, Oct. 2002. American Institute of Aeronautics and Astronautics. doi:10.2514/6.2002-5853.

[KD18] V. Kotu and B. Deshpande. *Data Science: Concepts and Practice*. Morgan Kaufmann Publishers, 2 edition, 2018.

[KK16] N. Kass and J. Kolozs. Getting Started with MBSE in Product Development. *INCOSE International Symposium*, 26(1):526–541, July 2016. doi:10.1002/j.2334-5837.2016.00176.x.

[KMS19] K. Koseler, K. McGraw, and M. Stephan. Realization of a Machine Learning Domain Specific Modeling Language: A Baseball Analytics Case Study. In *Proceedings of the 7th International Conference on Model-Driven Engineering and Software Development*, MODELSWARD 2019, pages 13–24, Setubal, PRT, Feb. 2019. SCITEPRESS - Science and Technology Publications, Lda. doi:10.5220/0007245800130024.

[KOM+10] T. Kosar, N. Oliveira, M. Mernik, V. Pereira, M. Crepinsek, C. Da, and R. Henriques. Comparing general-purpose and domain-specific languages: An empirical study. *Computer Science and Information Systems*, 7(2):247–264, 2010. doi:10.2298/CSIS1002247K.

[KPRS19] 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)*, pages 126–133, San Diego, CA, USA, Nov. 2019. IEEE. doi:10.1109/ASEW.2019.00042.

[KSFB20] A. Kossiakoff, S. J. Seymour, D. A. Flanigan, and S. M. Biemer. *Systems Engineering Principles and Practice*. Wiley, 1 edition, July 2020. doi:10.1002/9781119516699.

[KSW04] 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.*, pages 14–23, Bari, Italy, 2004. IEEE. doi:10.1109/WPC.2004.1311043.

[KURD22] D. Kaur, S. Uslu, K. J. Rittichier, and A. Durresi. Trustworthy Artificial Intelligence: A Review. *ACM Computing Surveys*, 55(2):39:1–39:38, Jan. 2022. doi:10.1145/3491209.

[LAD+16] L. Lúcio, M. Amrani, J. Dingel, L. Lambers, R. Salay, G. M. K. Selim, E. Syriani, and M. Wimmer. Model transformation intents and their properties. *Software & Systems Modeling*, 15(3):647–684, July 2016. doi:10.1007/s10270-014-0429-x.

[Lan09] M. Lankhorst. *Enterprise Architecture at Work*. Springer, Berlin, Heidelberg, 2009. doi:10.1007/978-3-642-01310-2.

[LC17] C. Liu and X. Chen. Data-driven design paradigm in engineering problems. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 231(8):1522–1534, June 2017. doi:10.1177/0954410016653502.

[LFB20] H. A. Long, D. P. French, and J. M. Brooks. Optimising the value of the critical appraisal skills programme (CASP) tool for quality appraisal in qualitative evidence synthesis. *Research Methods in Medicine & Health Sciences*, 1(1):31–42, Sept. 2020. doi:10.1177/2632084320947559.

[LO20] T. Lins and R. A. R. Oliveira. Cyber-physical production systems retrofitting in context of industry 4.0. *Computers & Industrial Engineering*, 139:106193, Jan. 2020. doi:10.1016/j.cie.2019.106193.

[Mad18] A. M. Madni. *Transdisciplinary Systems Engineering*. Springer International Publishing, Cham, 2018. doi:10.1007/978-3-319-62184-5.

[MCBG22] A. Moin, M. Challenger, A. Badii, and S. Guennemann. A model-driven approach to machine learning and software modeling for the IoT: Generating full source code for smart Internet of Things (IoT) services and cyber-physical systems (CPS). *Software and Systems Modeling*, 21(3):987–1014, June 2022. doi:10.1007/s10270-021-00967-x.

[MCC22] S. Morales, R. Clarisó, and J. Cabot. Towards a DSL for AI Engineering Process Modeling. *Product-Focused Software Process Improvement*, 13709:53–60, 2022. doi:10.1007/978-3-031-21388-5_4.

[MCF+21] F. Martinez-Plumed, L. Contreras-Ochando, C. Ferri, J. Hernandez-Orallo, M. Kull, N. Lachiche, M. J. Ramirez-Quintana, and P. Flach. CRISP-DM Twenty Years Later: From Data Mining Processes to Data Science Trajectories. *IEEE Transactions on Knowledge and Data Engineering*, 33(8):3048–3061, Aug. 2021. doi:10.1109/TKDE.2019.2962680.

[McM05] H. L. McManus. Product Development Value Stream Mapping (PDVSM) Manual Release 1.0. Sept. 2005. URL https://dspace.mit.edu/handle/1721.1/81908.

[MCO16] R. Mills, K. Chudoba, and D. Olsen. IS Programs Responding to Industry Demands for Data Scientists: A Comparison Between 2011-2016. *Journal of Information Systems Education*, 27(2):131–140, Jan. 2016. URL https://aisel.aisnet.org/jise/vol27/iss2/6.

[MH17] S. Miller and D. Hughes. The quant crunch: How the demand for data science skills is disrupting the job market. Technical report, Burning Glass

Technologies, Boston, 2017. URL http://burning-glass.com/wp-content/uploads/The_Quant_Crunch.pdf.

[MHS05] M. Mernik, J. Heering, and A. M. Sloane. When and how to develop domain-specific languages. *ACM Computing Surveys*, 37(4):316–344, Dec. 2005. doi:10.1145/1118890.1118892.

[Min13] J. Minguez. Der Manufacturing Service Bus. In E. Westkaemper, D. Spath, C. Constantinescu, and J. Lentes, editors, *Digitale Produktion*, pages 271–289. Springer, Berlin, Heidelberg, 2013. doi:10.1007/978-3-642-20259-9_22.

[MJSR13] R. Meran, A. John, C. Staudter, and O. Roenpage. *Six Sigma+Lean Toolset: Mindset zur erfolgreichen Umsetzung von Verbesserungsprojekten*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. doi:10.1007/978-3-642-39945-9.

[MLC11] H. Martinez Leon and J. Cross. Lean Product Development Research: Current State and Future Directions. *Engineering Management Journal*, 23:29–51, Apr. 2011. doi:10.1080/10429247.2011.11431885.

[Möl16] D. P. F. Möller. Systems and Software Engineering. In D. P. Möller, editor, *Guide to Computing Fundamentals in Cyber-Physical Systems: Concepts, Design Methods, and Applications*, pages 235–305. Springer International Publishing, Cham, 2016. doi:10.1007/978-3-319-25178-3_6.

[Mon14] L. Monostori. Cyber-physical Production Systems: Roots, Expectations and R&D Challenges. *Procedia CIRP*, 17:9–13, 2014. doi:10.1016/j.procir.2014.03.115.

[MP03] I. McKeown and G. Philip. Business transformation, information technology and competitive strategies: Learning to fly. *International Journal of Information Management*, 23(1):3–24, Feb. 2003. doi:10.1016/S0268-4012(02)00065-8.

[MP19] A. Madni and S. Purohit. Economic Analysis of Model-Based Systems Engineering. *Systems*, 7(1):12, Feb. 2019. doi:10.3390/systems7010012.

[MPN21] S. Meacham, V. Pech, and D. Nauck. AdaptiveSystems: An Integrated Framework for Adaptive Systems Design and Development Using MPS JetBrains Domain-Specific Modeling Environment. *IEEE Access*, 9:127973–127984, 2021. doi:10.1109/ACCESS.2021.3111229.

[MPRE19] J. Mangler, F. Pauker, S. Rinderle-Ma, and M. Ehrendorfer. Centurio.work - Industry 4.0 integration assessment and evolution. In J. vom Brocke, J. Mendling, and M. Rosemann, editors, *Proceedings of the Industry Forum at BPM 2019 Co-Located with 17th International Conference on Business Process Management (BPM 2019), Vienna, Austria, September 1-6, 2019*, volume 2428 of *CEUR Workshop Proceedings*, pages

106–117. CEUR-WS.org, 2019. URL https://ceur-ws.org/Vol-2428/paper10.pdf.

[MR14] J. Mangler and S. Rinderle-Ma. CPEE - Cloud Process Execution Engine. In L. Limonad and B. Weber, editors, *Proceedings of the BPM Demo Sessions 2014 Co-located with the 12th International Conference on Business Process Management (BPM 2014), Eindhoven, The Netherlands, September 10, 2014*, volume 1295 of *CEUR Workshop Proceedings*, page 51. CEUR-WS.org, 2014. URL http://ceur-ws.org/Vol-1295/paper22.pdf.

[MR22] J. Mangler and S. Rinderle-Ma. Cloud Process Execution Engine: Architecture and Interfaces, Sept. 2022, 2208.12214. doi:10.48550/arXiv.2208.12214.

[MRC+22] F. Melchor, R. Rodriguez-Echeverria, J. M. Conejero, Á. E. Prieto, and J. D. Gutiérrez. A Model-Driven Approach for Systematic Reproducibility and Replicability of Data Science Projects. In X. Franch, G. Poels, F. Gailly, and M. Snoeck, editors, *Advanced Information Systems Engineering*, Lecture Notes in Computer Science, pages 147–163, Cham, 2022. Springer International Publishing. doi:10.1007/978-3-031-07472-1_9.

[MRT18] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. Adaptive Computation and Machine Learning. The MIT Press, Cambridge, Massachusetts, second edition, 2018. URL https://dl.acm.org/doi/book/10.5555/3360093.

[MS18] A. M. Madni and M. Sievers. Model-based systems engineering: Motivation, current status, and research opportunities. *Systems Engineering*, 21(3):172–190, 2018. doi:10.1002/sys.21438.

[MSMF09] O. Marbán, J. Segovia, E. Menasalvas, and C. Fernández-Baizán. Toward data mining engineering: A software engineering approach. *Information Systems*, 34(1):87–107, Mar. 2009. doi:10.1016/j.is.2008.04.003.

[Mul13] G. Muller. Systems Engineering Research Methods. *Procedia Computer Science*, 16:1092–1101, Jan. 2013. doi:10.1016/j.procs.2013.01.115.

[MV18] J. M. Mueller and K.-I. Voigt. Sustainable Industrial Value Creation in SMEs: A Comparison between Industry 4.0 and Made in China 2025. *International Journal of Precision Engineering and Manufacturing-Green Technology*, 5(5):659–670, 2018. doi:10.1007/s40684-018-0056-z.

[MYK+09] B. Mutlu, F. Yamaoka, T. Kanda, H. Ishiguro, and N. Hagita. Nonverbal leakage in robots: Communication of intentions through seemingly unintentional behavior. In *Proceedings of the 4th ACM/IEEE International Conference on Human Robot Interaction - HRI '09*, page 69, La Jolla, California, USA, 2009. ACM Press. doi:10.1145/1514095.1514110.

[NAR+17] F. Nikpay, R. B. Ahmad, B. D. Rouhani, M. N. Mahrin, and S. Shamshirband. An effective Enterprise Architecture Implementation Methodology. *Information Systems and e-Business Management*, 15(4):927–962, Nov. 2017. doi:10.1007/s10257-016-0336-5.

[Nie93] J. Nielsen. *Usability Engineering*. Academic Press, Boston, 1993.

[OB13] P. Olsen and M. Borit. How to define traceability. *Trends in Food Science & Technology*, 29(2):142–150, Feb. 2013. doi:10.1016/j.tifs.2012.10.003.

[OD05] F. J. O'Donnell and A. H. Duffy. A Formalism for Design Performance Measurement and Management. In *Design Performance*, pages 55–87. Springer, London, 2005. doi:10.1007/1-84628-147-4_5.

[OMG19] OMG. OMG Systems Modeling Language (OMG SysML™, Version 1.6), 2019. URL http://www.omg.org/spec/SysML/1.6/PDF/.

[OMG24] The Object Management Group. OMG SysML Home | OMG Systems Modeling Language, 2024. URL https://omgsysml.org/.

[Ope23] OpenAI. GPT-4 Technical Report, Mar. 2023, 2303.08774. doi:10.48550/arXiv.2303.08774.

[PAA+15] A. Pyster, R. Adcock, M. Ardis, R. Cloutier, D. Henry, L. Laird, H. B. Lawson, M. Pennotti, K. Sullivan, and J. Wade. Exploring the Relationship between Systems Engineering and Software Engineering. *Procedia Computer Science*, 44:708–717, 2015. doi:10.1016/j.procs.2015.03.016.

[Pat02] M. Q. Patton. Two Decades of Developments in Qualitative Inquiry: A Personal, Experiential Perspective. *Qualitative Social Work*, 1(3):261–283, Sept. 2002. doi:10.1177/1473325002001003636.

[PB13] G. Pahl and W. Beitz. *Pahl/Beitz Konstruktionslehre Methoden und Anwendung erfolgreicher Produktentwicklung*. Imprint: Springer Vieweg, Berlin, Heidelberg, 2013.

[PBBA23] P. P. Senna, A. C. Barros, J. Bonnin Roca, and A. Azevedo. Development of a digital maturity model for Industry 4.0 based on the technology-organization-environment framework. *Computers & Industrial Engineering*, 185:109645, Nov. 2023. doi:10.1016/j.cie.2023.109645.

[PBU19] A. D. Prabaswari, C. Basumerda, and B. W. Utomo. The Mental Workload Analysis of Staff in Study Program of Private Educational Organization. *IOP Conference Series: Materials Science and Engineering*, 528(1):012018, May 2019. doi:10.1088/1757-899X/528/1/012018.

[PF13] F. Provost and T. Fawcett. Data Science and its Relationship to Big Data and Data-Driven Decision Making. *Big Data*, 1(1):51–59, Mar. 2013. doi:10.1089/big.2013.1508.

[PM18] F. Pauker and J. Mangler. Centurio.work - Higher Productivity Through Intelligent Connectivity. In *Wiener Produktionstechnik Kongress*, volume 4. new academic press og, 2018.

[PMRE21] F. Pauker, J. Mangler, S. Rinderle-Ma, and M. Ehrendorfer. Industry 4.0 Integration Assessment and Evolution at EVVA GmbH: Process-Driven Automation Through centurio.work. In J. vom Brocke, J. Mendling, and M. Rosemann, editors, *Business Process Management Cases Vol. 2: Digital Transformation - Strategy, Processes and Execution*, pages 81–91. Springer, Berlin, Heidelberg, 2021. doi:10.1007/978-3-662-63047-1_7.

[Pow16] D. J. Power. Data science: Supporting decision-making. *Journal of Decision Systems*, 25(4):345–356, Oct. 2016. doi:10.1080/12460125.2016.1171610.

[PP14] T. Parsana and M. Patel. A Case Study: A Process FMEA Tool to Enhance Quality and Efficiency of Manufacturing Industry. *Bonfring International Journal of Industrial Engineering and Management Science*, 4(3):145–152, Aug. 2014. doi:10.9756/BIJIEMS.10350.

[PPW+21] D. Piorkowski, S. Park, A. Y. Wang, D. Wang, M. Muller, and F. Portnoy. How AI Developers Overcome Communication Challenges in a Multidisciplinary Team: A Case Study. *Proceedings of the ACM on Human-Computer Interaction*, 5(CSCW1):1–25, Apr. 2021. doi:10.1145/3449205.

[Pri96] C. Price. Effortless incremental design FMEA. In *Proceedings of 1996 Annual Reliability and Maintainability Symposium*, pages 43–47, Jan. 1996. doi:10.1109/RAMS.1996.500640.

[PVK15] K. Petersen, S. Vakkalanka, and L. Kuzniarz. Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 64:1–18, Aug. 2015. doi:10.1016/j.infsof.2015.03.007.

[RBGM21] A. K. Raz, E. P. Blasch, C. Guariniello, and Z. T. Mian. An Overview of Systems Engineering Challenges for Designing AI-Enabled Aerospace Systems. In *AIAA Scitech 2021 Forum*, VIRTUAL EVENT, Jan. 2021. American Institute of Aeronautics and Astronautics. doi:10.2514/6.2021-0564.

[RBW+23] S. Raedler, L. Berardinelli, K. Winter, A. Rahimi, and S. Rinderle-Ma. Model-Driven Engineering for Artificial Intelligence – A Systematic Literature Review, July 2023, 2307.04599. doi:10.48550/arXiv.2307.04599.

[RBW+24] S. Raedler, L. Berardinelli, K. Winter, A. Rahimi, and S. Rinderle-Ma. Bridging MDE and AI: A systematic review of domain-specific languages and model-driven practices in AI software systems engineering. *Software and Systems Modeling*, Sept. 2024. doi:10.1007/s10270-024-01211-y.

[RDS15] A. Rodrigues Da Silva. Model-driven engineering: A survey supported by the unified conceptual model. *Computer Languages, Systems & Structures*, 43:139–155, Oct. 2015. doi:10.1016/j.cl.2015.06.001.

[RFB12] A. L. Ramos, J. V. Ferreira, and J. Barceló. Model-Based Systems Engineering: An Emerging Approach for Modern Systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(1):101–111, Jan. 2012. doi:10.1109/TSMCC.2011.2106495.

[RGJ21] B. Ries, N. Guelfi, and B. Jahic. An MDE Method for Improving Deep Learning Dataset Requirements Engineering using Alloy and UML. In S. Hammoudi, L. F. Pires, E. Seidewitz, and R. Soley, editors, *Proceedings of the 9th International Conference on Model-Driven Engineering and Software Development, MODELSWARD 2021, Online Streaming, February 8-10, 2021*, pages 41–52. SCITEPRESS, 2021. doi:10.5220/0010216600410052.

[RH18] B. Rumpe and K. Hölldobler, editors. *MontiCore 5 Language Workbench*. Number Band 32 in Aachener Informatik-Berichte, Software-Engineering. Shaker Verlag GmbH, Aachen, 2018. URL https://doi.org/10.2370/9783844057133.

[RH22] J. Reis and M. L. Housley. *Fundamentals of Data Engineering: Plan and Build Robust Data Systems*. O'Reilly Media, Sebastopol, CA, 2022.

[RKK+20] S. Ransbotham, S. Khodabandeh, D. Kiron, F. Candelon, M. Chu, and B. LaFountain. Expanding AI's Impact With Organizational Learning. *MIT Sloan Management Review*, Oct. 2020. URL https://sloanreview.mit.edu/projects/expanding-ais-impact-with-organizational-learning/.

[RKR15] L. Ramanan, M. Kumar, and K. Ramanakumar. Knowledge Gap and its Impact on Product and Process Quality. *Applied Mechanics and Materials*, 813–814:1176–1182, Nov. 2015. doi:10.4028/www.scientific.net/AMM.813-814.1176.

[RMNN13] B. D. Rouhani, M. N. Mahrin, F. Nikpay, and P. Nikfard. A Comparison Enterprise Architecture Implementation Methodologies. In *2013 International Conference on Informatics and Creative Multimedia*, pages 1–6, Kuala Lumpur, Malaysia, Sept. 2013. IEEE. doi:10.1109/ICICM.2013.9.

[RMR22] S. Raedler, J. Mangler, and E. Rigger. Requirements for Manufacturing Data Collection to Enable Data-Driven Design. In *Procedia CIRP*, volume

112 of *15th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 14-16 July 2021*, pages 232–237, Gulf of Naples, Jan. 2022. doi:10.1016/j.procir.2022.09.077.

[RMR23] S. Raedler, J. Mangler, and S. Rinderle-Ma. Model-Driven Engineering Method to Support the Formalization of Machine Learning using SysML, July 2023, 2307.04495. doi:10.48550/arXiv.2307.04495.

[RN21] S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Series in Artificial Intelligence. Pearson, Hoboken, NJ, fourth edition edition, 2021.

[RR20] S. Raedler and E. Rigger. Participative Method to Identify Data-Driven Design Use Cases. In *Product Lifecycle Management Enabling Smart X*, volume 594, pages 680–694, Cham, 2020. Springer International Publishing. doi:10.1007/978-3-030-62807-9_54.

[RR22] S. Raedler and E. Rigger. A Survey on the Challenges Hindering the Application of Data Science, Digital Twins and Design Automation in Engineering Practice. In *Proceedings of the Design Society*, volume 2, pages 1699–1708. Cambridge University Press, May 2022. doi:10.1017/pds.2022.172.

[RRMR22] S. Raedler, E. Rigger, J. Mangler, and S. Rinderle-Ma. Integration of Machine Learning Task Definition in Model-Based Systems Engineering using SysML. In *2022 IEEE 20th International Conference on Industrial Informatics (INDIN)*, pages 546–551, Perth, Australia, July 2022. IEEE. doi:10.1109/INDIN51773.2022.9976107.

[RRRR24] S. Raedler, M. Rupp, E. Rigger, and S. Rinderle-Ma. Model-Driven Engineering for Machine Learning Code Generation using SysML. In *Modellierung 2024*, pages 197–212, Potsdam, Mar. 2024. Gesellschaft für Informatik eV. doi:10.18420/MODELLIERUNG2024_019.

[RSS22] E. Rigger, K. Shea, and T. Stanković. Method for identification and integration of design automation tasks in industrial contexts. *Advanced Engineering Informatics*, 52:101558, Apr. 2022. doi:10.1016/j.aei.2022.101558.

[RV18] E. Rigger and T. Vosgien. Design Automation State of Practice - Potential and Opportunities. In *DS 92: Proceedings of the DESIGN 2018 15th International Design Conference*, pages 441–452, 2018. doi:10.21278/idc.2018.0537.

[RVSS19] E. Rigger, T. Vosgien, K. Shea, and T. Stankovic. A top-down method for the derivation of metrics for the assessment of design automation potential. *Journal of Engineering Design*, pages 1–31, Oct. 2019. doi:10.1080/09544828.2019.1670786.

[RZH+17] S. P. Ruemler, K. E. Zimmerman, N. W. Hartman, T. Hedberg, and A. Barnard Feeny. Promoting Model-Based Definition to Establish a Complete Product Definition. *Journal of Manufacturing Science and Engineering*, 139(5):051008, May 2017. doi:10.1115/1.4034625.

[Sal21] J. S. Saltz. CRISP-DM for Data Science: Strengths, Weaknesses and Potential Next Steps. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 2337–2344, Orlando, FL, USA, Dec. 2021. IEEE. doi:10.1109/BigData52589.2021.9671634.

[Sar21] I. H. Sarker. Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*, 2(3):160, Mar. 2021. doi:10.1007/s42979-021-00592-x.

[SCL+21] L. Shen, X. Chen, R. Liu, H. Wang, and G. Ji. Domain-Specific Language Techniques for Visual Computing: A Comprehensive Study. *Archives of Computational Methods in Engineering*, 28(4):3113–3134, June 2021. doi:10.1007/s11831-020-09492-4.

[SFB21] T. Sturm, M. Fecho, and P. Buxmann. To Use or Not to Use Artificial Intelligence? A Framework for the Ideation and Evaluation of Problems to Be Solved with Artificial Intelligence. In *Proceedings of the 54th Hawaii International Conference on System Sciences*, Jan. 2021. URL http://hdl.handle.net/10125/70634.

[Shi21] D. Shin. The effects of explainability and causability on perception, trust, and acceptance: Implications for explainable AI. *International Journal of Human-Computer Studies*, 146:102551, Feb. 2021. doi:10.1016/j.ijhcs.2020.102551.

[SHS+18] D. Schel, C. Henkel, D. Stock, O. Meyer, G. Rauhoeft, P. Einberger, M. Stoehr, M. Daxer, and J. Seidelmann. Manufacturing Service Bus: An Implementation. In *Procedia CIRP*, volume 67, Mar. 2018. doi:10.1016/j.procir.2017.12.196.

[SL16] J. Sauro and J. R. Lewis. *Quantifying the User Experience: Practical Statistics for User Research*. Morgan Kaufmann, Amsterdam Boston Heidelberg, 2 edition, July 2016.

[SLC18] L. N. Sanchez-Pinto, Y. Luo, and M. M. Churpek. Big Data and Data Science in Critical Care. *Chest*, 154(5):1239–1248, Nov. 2018. doi:10.1016/j.chest.2018.04.037.

[SLR17] K. Sun, Y. Li, and U. Roy. A PLM-based data analytics approach for improving product development lead time in an engineer-to-order manufacturing firm. *Mathematical Modelling of Engineering Problems*, 4(2):69–74, June 2017. doi:10.18280/mmep.040201.

[SMM+19] H. Sillitto, J. Martin, D. McKinney, R. Griego, D. Dori, D. Krob, P. Godfrey, E. Arnold, and S. Jackson. Systems engineering and system definitions. Technical report, International Council on Systems Engineering, San Diego, CA, USA, 2019. URL https://www.incose.org/docs/default-source/default-document-library/final_-se-definition.pdf.

[SN12] M. Sony and S. Naik. Six Sigma, organizational learning and innovation: An integration and empirical examination. *International Journal of Quality & Reliability Management*, 29(8):915–936, Jan. 2012. doi:10.1108/02656711211258535.

[SS18] P. P. Shinde and S. Shah. A Review of Machine Learning and Deep Learning Applications. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, pages 1–6, Pune, India, Aug. 2018. IEEE. doi:10.1109/ICCUBEA.2018.8697857.

[SSHK15] M. Seidl, M. Scholz, C. Huemer, and G. Kappel. *UML@Classroom: An Introduction to Object-Oriented Modeling*. Springer, 2015. doi:10.1007/978-3-319-12742-2.

[ST06] S. H. Spewak and M. Tiemann. Updating the enterprise architecture planning model. In *Journal of Enterprise Architecture*, volume 2, pages 11–19, 2006.

[ST15] TR. Sreeram and A. Thondiyath. Combining Lean and Six Sigma in the context of Systems Engineering design. *International Journal of Lean Six Sigma*, 6(4):290–312, Jan. 2015. doi:10.1108/IJLSS-07-2014-0022.

[ST18] J. Suryadevara and S. Tiwari. Adopting MBSE in Construction Equipment Industry: An Experience Report. In *2018 25th Asia-Pacific Software Engineering Conference (APSEC)*, pages 512–521, Dec. 2018. doi:10.1109/APSEC.2018.00066.

[Sta73] H. Stachowiak. *Allgemeine Modelltheorie*. Springer, Wien New York, 1973.

[Sta03] D. H. Stamatis. *Failure Mode and Effect Analysis: FMEA from Theory to Execution*. Quality Press, Jan. 2003.

[Sta21] Stack Overflow. Stack Overflow Developer Survey 2021, May 2021. URL https://insights.stackoverflow.com/survey/2021/.

[SWZ20] I. Someh, B. Wixom, and A. Zutavern. Overcoming Organizational Obstacles to Artificial Intelligence Value Creation: Propositions for Research. In *Proceedings of the 53rd Hawaii International Conference on System Sciences*, pages 5809–5818, Jan. 2020. URL http://hdl.handle.net/10125/64454.

[SYS20] R. Sothilingam, E. Yu, and A. Senderovich. Towards Higher Maturity for Machine Learning: A Conceptual Modelling Approach. *The iJournal: Graduate Student Journal of the Faculty of Information*, 5(1):80–97, Jan. 2020. doi:10.33137/ijournal.v5i1.33476.

[Tag10] S. Taghizadegan. *Essentials of Lean Six Sigma*. Elsevier, July 2010.

[Tan11] J. M. Tanur. Margin of Error. In M. Lovric, editor, *International Encyclopedia of Statistical Science*, pages 765–935. Springer, Berlin, Heidelberg, 2011. doi:10.1007/978-3-642-04898-2_34.

[TCQ+18] F. Tao, J. Cheng, Q. Qi, M. Zhang, H. Zhang, and F. Sui. Digital twin-driven product design, manufacturing and service with big data. *The International Journal of Advanced Manufacturing Technology*, 94(9):3563–3576, Feb. 2018. doi:10.1007/s00170-017-0233-1.

[TDS18] R. Tsui, D. Davis, and J. Sahlin. Digital Engineering Models of Complex Systems using Model-Based Systems Engineering (MBSE) from Enterprise Architecture (EA) to Systems of Systems (SoS) Architectures & Systems Development Life Cycle (SDLC). *INCOSE International Symposium*, 28(1):760–776, July 2018. doi:10.1002/j.2334-5837.2018.00514.x.

[TIK+21] H. Takeuchi, R. Iga, K. Katayama, H. Mitsuyama, R. Motegi, and A. Uematsu. Identification of Business Goals of AI Service System based on GQM+Strategies. In *2021 10th International Congress on Advanced Applied Informatics (IIAI-AAI)*, pages 768–771, July 2021. doi:10.1109/IIAI-AAI53430.2021.00135.

[TINI21] H. Takeuchi, Y. Ito, R. Nishiyama, and T. Isomura. Modeling of Machine Learning Projects Using ArchiMate. In A. Zimmermann, R. J. Howlett, L. C. Jain, and R. Schmidt, editors, *Human Centred Intelligent Systems*, volume 244, pages 222–231, Singapore, 2021. Springer Singapore. doi:10.1007/978-981-16-3264-8_21.

[TOGAF18] T. O. Group. *The TOGAF® Standard, Version 9.2*. Van Haren Publishing, Zaltbommel, 11th ed edition, Apr. 2018.

[TSOO+20] J. Trauer, S. Schweigert-Recksiek, L. Onuma Okamoto, K. Spreitzer, M. Moertl, and M. Zimmermann. Data-Driven Engineering – Definitions and Insights from an Industrial Case Study for a New Approach in Technical Product Development. In *Balancing Innovation and Operation*. The Design Society, 2020. doi:10.35199/NORDDESIGN2020.46.

[Tsy04] A. Tsymbal. The problem of concept drift: Definitions and related work. *Computer Science Department, Trinity College Dublin*, 106(2):58, 2004. URL https://www.scss.tcd.ie/publications/tech-reports/reports.04/TCD-CS-2004-15.pdf.

[van10] M. F. van Amstel. The right tool for the right job : Assessing model transformation quality. *Proceedings of the 34th Annual IEEE Computer Software and Applications Conference (COMPSAC, Seoul, Korea, July 19-123, 2010)*, pages 69–74, 2010. doi:10.1109/COMPSACW.2010.22.

[van12] M. F. van Amstel. *Assessing and Improving the Quality of Model Transformations*. PhD thesis, Technische Universiteit Eindhoven, Eindhoven, 2012. doi:10.6100/IR719526.

[vdBN10] M. F. van Amstel, v. den Brand, M.G.J., and H. Nguyen. Metrics for model transformations. *BENEVOL 2010 (9th Belgian-Netherlands Software Evolution Seminar, Lille, France, December 16, 2010. Proceedings of Short Papers)*, pages 1–5, 2010. URL https://research.tue.nl/files/2877294/Metis245998.pdf.

[VGM+17] F. Veit, J. Geyer-Klingeberg, J. Madrzak, M. Haug, and J. Thomson. The Proactive Insights Engine: Process Mining meets Machine Learning and Artificial Intelligence. In *15th International Conference on Business Process Management (BPM)*, Barcelona, Spain, 2017. URL https://ceur-ws.org/Vol-1920/BPM_2017_paper_192.pdf.

[VGZS20] A. de la Vega, D. García-Saiz, M. Zorrilla, and P. Sánchez. Lavoisier: A DSL for increasing the level of abstraction of data selection and formatting in data mining. *Journal of Computer Languages*, 60:100987, Oct. 2020. doi:10.1016/j.cola.2020.100987.

[VKPV22] D. Vlah, A. Kastrin, J. Povh, and N. Vukašinović. Data-driven engineering design: A systematic review using scientometric approach. *Advanced Engineering Informatics*, 54, Oct. 2022. doi:10.1016/j.aei.2022.101774.

[vLv08] M. F. van Amstel, C. Lange, and M. G. van den Brand. Metrics for analyzing the quality of model transformations. *Proceedings 12th ECOOP Workshop on Quantitative Approaches on Object Oriented Software Engineering (QAOOSE08, Paphos, Cyprus, July 8, 2008 (co-located with ECOOP 2008))*, pages 41–51, 2008. URL https://research.tue.nl/files/3034080/600644033736967.pdf.

[WC18] D. C. Wynn and P. J. Clarkson. Process models in design and development. *Research in Engineering Design*, 29(2):161–202, Apr. 2018. doi:10.1007/s00163-017-0262-7.

[Wei14] T. Weilkiens. *Variant Modeling with SysML*. Leanpub, July 2014. URL https://leanpub.com/vamos.

[WH00] 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.

[WI23] D. D. Walden and International Council on Systems Engineering, editors. *INCOSE Systems Engineering Handbook*. John Wiley & Sons Ltd, Hoboken, NJ, fifth edition edition, 2023.

[WM08] S. A. White and D. Miers. *BPMN Modeling and Reference Guide: Understanding and Using BPMN ; Develop Rigorous yet Understandable Graphical Representations of Business Processes*. Future Strategies Inc, Lighthouse Point, Fla, 2008.

[WRH+12] C. Wohlin, P. Runeson, M. Hoest, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering*. Springer Science & Business Media, June 2012.

[WSS22] J. Westenberger, K. Schuler, and D. Schlegel. Failure of AI projects: Understanding the critical factors. *Procedia Computer Science*, 196:69–76, Jan. 2022. doi:10.1016/j.procs.2021.11.074.

[WTO15] L. Wang, M. Toerngren, and M. Onori. Current status and advancement of cyber-physical systems in manufacturing. *Journal of Manufacturing Systems*, 37:517–527, Oct. 2015. doi:10.1016/j.jmsy.2015.04.008.

[WVMB19] J. Wade, D. Verma, T. McDermott, and B. Boehm. The SERC 5-Year Technical Plan: Designing the Future of Systems Engineering Research. In E. Bonjour, D. Krob, L. Palladino, and F. Stephan, editors, *Complex Systems Design & Management*, Cham, 2019. Springer International Publishing. doi:10.1007/978-3-030-04209-7_27.

[WWIT16] T. Wuest, D. Weimer, C. Irgens, and K.-D. Thoben. Machine learning in manufacturing: Advantages, challenges, and applications. *Production & Manufacturing Research*, 4(1):23–45, Jan. 2016. doi:10.1080/21693277.2016.1192517.

[YE09] K. Yang and B. S. El-Haik. *Design for Six Sigma: A Roadmap for Product Development*. McGraw-Hill, New York, NY., 2. ed edition, 2009.

[Zac87] J. A. Zachman. A framework for information systems architecture. *IBM Systems Journal*, 26(3):276–292, 1987. doi:10.1147/sj.263.0276.

[Zd20] J. Zucker and M. d'Leeuwen. Arbiter: A Domain-Specific Language for Ethical Machine Learning. In A. N. Markham, J. Powles, T. Walsh, and A. L. Washington, editors, *AIES '20: AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, February 7-8, 2020*, pages 421–425. ACM, 2020. doi:10.1145/3375627.3375858.