JOHANNES KEPLER
UNIVERSITY LINZ

Univ.-Prof. Dipl.-Ing. Mag. Dr. techn. Alexandra Mazak-Huemer

Institute of Business Informatics - Software Engineering

December 2021

Temporal Model-Driven Systems Engineering

Cumulative Habilitation Thesis for
Applied Computer Science
at the Faculty of Social Sciences, Economics
and Business of the Johannes Kepler
University Linz

JOHANNES KEPLER
UNIVERSITY LINZ
Altenbergerstrasse 69
4040 Linz, Austria
www.jku.at
DVR 0093696

Abstract

Due to the paradigm shift towards Industry 4.0, the role of software-intensive systems is becoming more and more important. In particular, the trend towards physical components being controlled by software has led to the *Internet-of-Things (IoT)* and *Cyber-Physical-Systems (CPS)*. As a consequence, companies face highly complex systems that are undergoing a constant change process resulting from shorter innovation cycles and rapidly changing customer needs. It is important that they keep their high-level requirements organized and consistent over multiple revision cycles across the entire life cycle of such a system, i.e., from design over development to implementation and operation.

Modeling is considered as a promising technique to better understand the dependencies within such complex systems. By following the *Model-Driven Engineering (MDE)* paradigm, systems are developed on a higher level of abstraction, and therefore, models are used as an integral part covering requirements, analysis, design, implementation, and verification. Although the term "*model-integrated computing*" has been coined almost twenty years ago, it has to be emphasized that the integration of models in the system life cycle is still mainly concerned with forward engineering, i.e., the development of new systems through generative techniques. Much less effort in MDE is spent on the evolutionary aspects of systems changing over time. For tackling this issue, models must no longer be considered as isolated one-shot system prescriptions, but as evolutionary and reusable descriptions of reality.

The research scope of this cumulative habilitation thesis is explicitly addressing this evolutionary aspect by focusing on *temporal aspects* of models of CPS. It follows a *Model-Driven Systems Engineering (MDSE)* approach by identifying and integrating appropriate concepts, languages, techniques, and tools for the systematic adoption of models throughout the engineering process. Models are continuously revised, often by considering feedback from other resources, until they are released. However, also the feedback after the release, i.e., from the operation, is reflected in the models. In the first part of this cumulative habilitation thesis, we elaborate on the integration of data from heterogeneous sources in order to provide a homogenized meaningful stack of information from the running system to a higher level of abstraction. In the second part, we cover the evolutionary aspects of engineering artefacts, i.e., models. Thereby, the focus is not only to represent the current state to steer the system, but on the representation of the system's history. In the final part, we provide MDE techniques for analyzing runtime data and extracting descriptive models for reasoning about and validating the operation of systems.

Contents

1 Introduction **1**
1.1 Motivation 1
1.2 Background 2
1.3 On the Need for *Temporal Model*-Driven Systems Engineering 5
1.4 Challenges of *Temporal Model*-Driven Systems Engineering 6
1.5 Contributions to *Temporal Model*-Driven Systems Engineering 10
1.6 Summary 31
1.7 Outlook 33
1.8 Bibliography 34

2 From business functions to control functions:
Transforming REA to ISA-95 . . . . **51**
A. Mazak and C. Huemer
Proceedings of the 17th IEEE International Conference on Business Informatics (CBI), IEEE Computer Society, (2015), pp. 33–42.
DOI: 10.1109/CBI.2015.50

3 AutomationML, ISA-95 and Others:
Rendezvous in the OPC UA Universe **63**
B. Wally, C. Huemer, A. Mazak and M. Wimmer
Proceedings of the 14th IEEE International Conference on Automation Science and Engineering (CASE), IEEE, (2018), pp. 1381–1387.
DOI: 10.1109/COASE.2018.8560600

4 Leveraging integration facades for model-based tool interoperability . . **71**
G. Paskaleva, A. Mazak-Huemer, M. Wimmer and T. Bednar
Automation in Construction, Elsevier, 128, (2021), pp. 103689.
DOI: 10.1016/j.autcon.2021.103689

5 A View on Model-Driven Vertical Integration: Alignment of Production Facility Models and Business Models **93**
B. Wally, C. Huemer and A. Mazak
Proceedings of the 13th IEEE International Conference on Automation Science and Engineering (CASE), IEEE, (2017), pp. 1012–1018.
DOI: 10.1109/COASE.2017.8256235

6 Flexible Production Systems: Automated Generation of Operations Plans Based on ISA-95 and PDDL **101**
B. Wally, J. Vyskocil, P. Novák, C. Huemer, R. Sindelár, P. Kadera, A. Mazak-Huemer and M. Wimmer
IEEE Robotics and Automation Letters (RA-L), IEEE, 4, (2019), pp. 4062–4069.
DOI: 10.1109/LRA.2019.2929991

7 Thirteen years of SysML: A systematic mapping study **111**
S. Wolny, A. Mazak, C. Carpella, V. Geist and M. Wimmer
Journal of Software and Systems Modeling, Springer, 19(1), (2020), pp. 111–169.
DOI: 10.1007/s10270-019-00735-y

8 Towards Liquid Models: An Evolutionary Modeling Approach **171**
A. Mazak and M. Wimmer
Proceedings of the 18th IEEE International Conference on Business Informatics (CBI), IEEE Computer Society, (2016), pp. 104–112.
DOI: 10.1109/CBI.2016.20

9 Model-Driven Time-Series Analytics **181**
S. Wolny, A. Mazak, M. Wimmer, R. Konlechner and G. Kappel
International Journal of Conceputal Modeling - Enterprise Modelling and Information Systems Architectures (EMISA), Springer, 13(Special), (2018), pp. 252–261.
DOI: 10.18417/emisa.si.hcm.19

10 Temporal Models on Time Series Databases **193**
A. Mazak, S. Wolny, A. Gómez, J. Cabot, M. Wimmer and G. Kappel
Journal of Object Technology, Springer, 19(3), (2020), pp. 3:1–3:15.
DOI: 10.5381/jot.2020.19.3.a14

11 Execution-Based Model Profiling **209**
A. Mazak, P. Patsuk-Bösch and M. Wimmer
Data-Driven Process Discovery and Analysis – 6th IFIP WG 2.6 International Symposium (SIMPDA), Revised Selected Papers, Vol. 307, LNBIP, Springer, (2016), pp. 37–52.
DOI: 10.1007/978-3-319-74161-1_3

12 Reverse Engineering of Production Processes based on Markov Chains 227
A. Mazak, P. Patsuk-Bösch and M. Wimmer
Proceedings of the 13th IEEE International Conference on Automation Science and Engineering (CASE), IEEE, (2017), pp. 680–686.
DOI: 10.1109/COASE.2017.8256182

13 Automatic Reverse Engineering of Interaction Models from System Logs 235
A. Mazak, S. Wolny and M. Wimmer
Proceedings of the 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, (2019), pp. 57–64.
DOI: 10.1109/ETFA.2019.8869502

14 Model-driven Runtime State Identification 245
S. Wolny, A. Mazak, M. Wimmer and C. Huemer
40 Years SIG EMISA: Digital Ecosystems of the Future: Methods, Techniques and Applications, 39(1), De Gruyter, (2019), pp. 29–44.
Corpus ID: 220055432

15 From AutomationML to AutomationQL: A By-Example Query Language for CPPS Engineering Models 261
A. Mazak and M. Wimmer
Proceedings of the 14th IEEE International Conference on Automation Science and Engineering (CASE), IEEE, (2018), pp. 1394–1399.
DOI: 10.1109/COASE.2018.8560448

1 Introduction

1.1 Motivation

Today, most of the systems we interact with are linked to a globally networked infrastructure in which products, processes, and resources interact with embedded hardware and software far beyond the scope of a single application [24]. For instance, in the automated manufacturing engineering domain, the *Internet of Things (IoT)* creates cyber-phyiscal networks transforming the traditional production to a so-called "smart production" [18, 42, 85]. *Cyber Physical Systems (CPS)* enable the convergence of the physical world and the virtual world in a higher level environment [71]. In CPS, the physical environment is populated by highly interwoven communicating objects, such as networked controllers, sensors, actuators, and other smart devices [129]. Therefore, flexible monitoring and control approaches are needed to adapt the systems' behavior to ever-changing requirements and tasks, unexpected conditions, as well as structural transformations [81]. CPS components are capable of autonomously interacting with each other and with the environment itself. As a result, both, the volume and the level of detail of data generated are highly increasing. Due to this ever growing complexity also resulting from shorter innovation cycles, rapidly changing customer needs, etc., the evolutionary aspect of engineering artifacts that change over time is becoming more and more important [82, 49, 133]. This requires real-time communication, seamless data exchange as well as data stream analysis [131, 134, 135].

In order to deal with such software- and data-intensive systems, *modeling* is considered as a promising technique to understand and simplify reality through abstraction, and thus, *models* are in the center of the engineering process [13]. In a *Model-Driven Engineering (MDE)* approach, models represent the most important artifacts throughout all, most often interdisciplinary activities during an engineering process [13]. There are many kinds of models such as simulation models, prescriptive models, descriptive models, statistical models, planning models, etc. From an MDE perspective, when modeling the digital side of a system (such as the controller of a machine), the software is developed and implemented in a model-driven way [23]. This means that throughout this engineering process, models are the main "driver" and not solely used, e.g., for documentation purposes [13].

On the one hand, a model serves as an abstraction for a specific purpose, as a kind of "blueprint" concentrating on a system's structure as well as desired behavior executed by actuators based on controller commands. Accordingly, such design models represent early snapshots of the system. On the other hand, there are so-called *runtime models* providing real abstractions of systems during operation basing on the data gathered through sensors. There is however a discrepancy between models of design time (development phase) and models of runtime (operation phase) [93, 43]. For reducing the aforementioned complexity in CPS as well as for monitoring systems over time, an a-priori description of a system at hand is not enough [132]. It requires a connection from such an initial model to its runtime counterpart. Accordingly, models should be held "in-the-loop" from design to operation in order to be fully integrated in the life cycle of a system. Thus, the *temporal aspect* of models has to be taken into account in MDE.

The remainder of the introduction chapter of this cumulative habilitation thesis is structured as follows: In Section 1.2, we discuss the foundational concepts that provide the basis for our research work. We briefly introduce MDE principles followed by an overview on the important subtopics metamodeling, modeling, languages as well as temporal languages in general. Section 1.3 elaborates on the need for *Temporal Model-Driven Systems Engineering*. The challenges when developing and implementing this approach are discussed in Section 1.4. These challenges relate to data connectivity and integration issues, the handling of temporal models, and semantic-based data analytics. Section 1.5 reflects the core of the habilitation thesis by elaborating on our contributions which are divided into three main parts: (i) *Model-Driven Connectivity*, (ii) *Temporal Model Management*, and (iii) *Data-Driven Model Analytics*. Since these contributions would not have been possible without appropriate funding, we also outline the research projects enabling our research works. We provide a mapping between the identified challenges and presented contributions. Furthermore, we show which papers contribute to solve these challenges and discuss each contribution in more detail. A summary in Section 1.6 and an outlook in Section 1.7 conclude the introduction chapter.

1.2 Background

1.2.1 Model-Driven Engineering

In *Model-Driven Engineering (MDE)*, the abstraction power of models is used to tackle the complexity of systems [117, 72, 23]. From an abstracted point of view, MDE follows the principle *"everything is a model"* [14]. This means MDE supports system- and software engineers by providing formal models, like a tool box, to achieve simplicity, generality, and integration when modeling a system.

Historically, MDE has been mainly applied in *Software Engineering* [14, 23], but in recent years, MDE has also emerged in *Systems Engineering* [13, 146], such as in the industrial automation domain [64, 136, 119, 12], or in the Architecture, Engineering, Construction (AEC) industry [99, 110]. Furthermore, MDE concepts have been extended by runtime aspects [93, 11, 37, 9, 59, 73]. In MDE, models are the central artifact used as a main "driver" throughout the development process, finally leading to an automated generation of systems [40]. Therefore, MDE models are considered as *first class citizens* [15]. Well distinguished from MDE is *Model-Based Systems Engineering (MBSE)* which is considered as a "softer" version of MDE. According to the International Council on Systems Engineering (INCOSE) MBSE is defined as *"the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases"* [67].

In the following, we give a short overview on the difference between *Model-Based Engineering (MBE)* and MDE according to [23]. In MBE-processes models play an important role although they are not necessarily the key artifacts of the development. This means that they do not "drive" the process like in MDE. In MBE, models are created as a kind of "blueprint" directly handed out to the programmers for having a guideline for manually implemented code. There is no need for an automatic generation by code generators and there is no explicit definition of any platform-specific model. In a nutshell, models have an important role, but are not the central artifacts as in MDE processes, and therefore, MBE is defined as a superset of MDE [12, 23].

In the early phase of system development in MDE so-called *prescriptive models* are used to create and describe the scope of interest in a certain granularity of detail [65]. Thus, by definition, a model never describes reality in its entirety, rather it describes a scope of reality for a certain purpose in a given context [23]. Mayr et al. [88] critically note that models are mainly used as prescriptive documents only, and therefore, aim for a model-centered architecture paradigm. Their intention is to keep models, and in general, any developed artifact synchronized in all phases of development as well as in the operation phase. Therefore, in the later phases of the system life cycle so-called *descriptive models* may be employed to better understand how the system is actually realized and how it is operating in a certain environment [65]. Compared to prescriptive models, such descriptive models are only marginally explored in the field of MDE, and if used at all, they are created manually [93].

The implementation phase deals with the mapping of such prescriptive models to executable systems and consists of three levels [23]: (i) the *modeling level* where the models are defined, (ii) the *realization level* where the solutions are implemented through artifacts that are then employed in the running system, and (iii) the *automation level* where mappings from the modeling to the realization phase are defined. Those mappings enable to automatically transform model elements to code statements which can be executed on a platform [23]. For this step, MDE provides model transformations as one of its key techniques, which are *Model-to-Model (M2M)* and *Model-to-Text (M2T)* transformations [14]. Additionally, other MDE techniques such as model validation, model verification, model simulation, or model execution enable the automation of engineering process steps and support the traceability of engineering artifacts in genarel [23].

1.2.2 Metamodeling and Modeling Languages

The aforementioned levels are currently only used for downstream processes. The possibility of considering upstream processes, by gathering value streams of running systems, is rather neglected in MDE [93]. This means that even if systems are described by means of modeling languages and code generators are used to transform model elements to corresponding code statements, the execution of those statements is typically not represented in the metamodel.

In general, modeling languages are defined by metamodels. *Metamodels* are used to describe the abstract syntax of modeling languages. Based on this, the concrete syntax and semantics are defined to model a domain of interest (e.g., software controller of a machine) [23]. This means that each model created by using a modeling language is an instance of its metamodel, and thus, has to conform to it [15]. Additionally, whenever needed, metamodels can be adapted or extended, e.g., to tackle certain problem spaces of a domain. In addition, metamodeling enables (i) to compare models and reason about differences between them, (ii) to align models to create an integrated representation of a system, and (iii) to translate to other formalisms such as for code generation.

The *Unified Modeling Language (UML)* [106] is one of the best known modeling languages based on the *Meta Object Facility (MOF)* [109] standard. The advantages of UML are platform independence as well as adaption and extension capabilities for users to meet their own requirements for modeling specific interests. UML offers a wide range of views and different types of diagrams to represent the structure and behavior of a system [121]. One example of a metamodeling language that is based on a core subset of UML and MOF is Ecore from the *Eclipse Modeling Framework (EMF)* [45]. Since, Ecore supports the key concepts of using models as input to development and integration tools, it is one of the most widely used languages for code generation and model serialization for data interchange. We use UML and an extended subset of it known as *Systems Modeling Language (SysML)* [125] as well as Ecore for most of our contributions as presented in Section 1.5. Like UML, SysML is an *Object Management Group (OMG)* [105] standard, a so-called *general-purpose modeling language*, providing a graphical modeling language for describing complex systems by considering software as well as hardware parts. It should be briefly mentioned, that, additionally to these general modeling languages, there are *Domain Specific Languages (DSLs)* specialized for a specific application domain, or a specific purpose, with their own notations [23].

While in the early phases model-driven approaches are frequently used to design systems, in the later phases data-driven approaches are used, for instance, to reason on different key performance indicators of systems when operating in an environment [98]. This immediately poses the question how *operational data* can be mapped back to design models to evaluate existing designs and to reason about future re-designs. Thus, the combination of model-driven and data-driven approaches is required. Otherwise it would be difficult to prove whether the design model corresponds to its runtime counterpart. Even though the identification of discrepancies is not straight forward, engineers in industry would benefit if they could treat runtime data in the same way like standard UML, or SysML models.

With the emergence of CPS and other sophisticated runtime monitoring infrastructures, time-series databases [7] are frequently applied to store historical data of systems as well as to provide powerful analysis by dedicated query languages. There is abundant research on temporal extensions for modeling languages to specify the temporal characteristics of the system data (e.g., consider the survey of [62]), but not regarding the temporal dimensions of models themselves. Further works advance these first attempts by extending also the query languages with temporal properties, mainly to enable the validation and verification of temporal properties of the data.

Temporal OCL (TOCL) [151] and *Temporal UML* [28] are two examples of OCL extensions for the evaluation of temporal constraints. Temporal extensions have also been applied to specific types of systems (e.g., adaptive systems [103]) and DSLs (e.g., timed Petri nets [10, 78]). Even TOCL, which can be seen as a generic language, can also be used as a component in other DSLs as described in [100]. In this line, Bousse et al. [22] discuss and apply a pattern to extend modeling languages by events, traces, and further runtime concepts. The extention is used to represent the state of a model's execution. Futhermore, it provides verification support by applying TOCL for defining properties that are, together with the models, mapped to formal domains. Efficiency of these types of temporal inspection queries are also in the focus of [54] as well as [55].

All these approaches are mostly oriented towards the retrieval of specific past states of the model/data, elaborating on the concepts of *valid time* and *transaction time* of (bi)temporal models. Instead, we explicitly focus on the support for complete time-series storage and analysis, which opens the door to more powerful and rich possibilities, like the computation of different key performance indicators for models as part of design exploration or simulation scenarios.

1.3 On the Need for *Temporal Model*-Driven Systems Engineering

Today, evolving models are stored in model repositories, which often rely on existing versioning systems or standard database technologies [16]. These approaches are sufficient for hosting different model versions of an evolving model by storing each version separately as self-contained model instance. Those different versions of a model allow to reason about evolution concerns. However, the time dimension is not explicitly targeted on the model element level, and therefore, is not accessible. In order to support systems engineering processes where models are not only used in the system design phase, but also, in the operation phase, a more explicit representation of time is needed in order to reflect and reason about *temporal aspects*, such as element or state changes over time. Thus, the explicit consideration of temporal aspects on the level of model elements is essential from design to operation and backwards by combining downstream information from the MDE-process with upstream information gathered at runtime. Thus, a novel approach is needed that provides models on time-series databases for enabling runtime queries and for enriching models with historical data from the operation phases. This additionally enables a so-called *data-driven model analytics* [98] in order to analyze the actual runtime behavior of a system compared to the one initially designed. Thereby, the challenge is not only to bundle sensor value streams (e.g., from IoT networks) and aggregate them to a higher logical state level for process-oriented viewpoints, but also to consider uncertainties of the realization precision of sensor measurements during long-running operations. This allows to build a "vertical bridge" from the operation technology layer to the IT layer, where process views are integrated and model-based monitoring as well as analytics may be performed.

1.4 Challenges of *Temporal Model*-Driven Systems Engineering

In this section, we give a brief summary of research works related to the approaches presented in this cumulative thesis and discuss challenges in the scientific communities by specifically focusing on (i) heterogeneity management, (ii) temporal models, (iii) runtime models, and (iv) data analytics. In order to truly integrate models in a system life cycle, dedicated foundational as well as applied research efforts are needed.

1.4.1 Heterogeneity Management

The full integration of models in the system life cycle requires a continuous acquisition of real-time data (e.g., machine data, sensor data, event streams) from various distributed devices. Thereby, the data flow of those sources goes hand in hand with the translation among different data models with different syntax and semantics [25]. Another important challenge when linking data to models is the integration of data available in different formats (XML, native database formats, comma separated file formats, etc.). For this purpose an alignment and mapping of different entities needs to be accomplished. Each system is usually described by a multitude of models. The differences are a result of producing models of different granularity with continuous refinement during the engineering process. Also, the application of different tools in this process indicates differences when modeling a domain of interest [120]. Evidently, the information in all these models is partially overlapping and partially disjoint. It is essential to carry information of existing models forward to activities creating new models. Thus, model transformations bridging syntactical and semantic issues are needed.

Challenge I: Connectivity and Integration

The challenge of providing connectivity and integration can be divided into two subparts: (i) technical integration by means of appropriate interfaces, and (ii) semantic integration [104]. The latter one has to consider data and object models for storing and dealing with historical data, parameters and configurations. There are several standards used for horizontal as well as vertical integration on different layers, especially in the manufacturing domain. Furthermore, it requires approaches at the interface between technical and semantic integration, such as ISO 15926 which is used to store life cycle information as discussed in [80], to mention only one example. Depending on the application context, there is a plethora of industry standards that have to be investigated for the combination and revamping of information from heterogeneous sources into new homogenized representations.

1.4.2 Runtime and Temporal Models

Most approaches for runtime modeling aim for bridging the gap between design time modeling and runtime modeling to enable runtime analysis. Blair et al. [19] show the importance of supporting runtime adaptations to extend the use of MDE. These so-called *Models@run.time* provide a formal basis for dynamic adaptions, analysis, and predictions of a system when operating for supporting dynamic adaptation [102]. Different stakeholders can use those models in various ways for runtime monitoring such as dynamic state monitoring. Hartmann et al. [63] combine the concept of runtime models with reactive programming and peer-to-peer distribution. Reactive programming aims for enabling support for interactive applications, which react on events by focusing on real-time data streams. For this purpose a typical publish/subscribe pattern, well known as observer pattern in software engineering [53], is used. Khare et al. [76] show the application of such an approach in the IoT domain.

Hartmann et al. [63] define runtime models as a stream of model chunks, as it is common in reactive programming. The models are continuously updated during runtime, therefore they grow indefinitely. With their interpretation that every chunk has the data of one model element, they process them piecewise without looking at the total size. In order to prevent the exchange of full runtime models, peer-to-peer distribution is used between nodes to exchange model chunks. In addition, automatic reloading mechanisms are used to respond to events. As the models are distributed, operations like transformations have to be adapted. For this purpose transformations on streams as proposed by Cuadrado and de Lara [38] as well as Dávid et al. [39] can be used.

In order to successfully implement such runtime models and to bridge the gap between design-time modeling and runtime modeling, the temporal aspect of data is necessary to be included in the investigations. Initial work was done in the field of temporal relational databases, as presented in [26], followed by approaches presenting dedicated mappings of design models to different data base technologies on the basis of different data paradigms as discussed in [58]. The research works in this context were continued and extended in the area of temporal data warehouses [60], and have mainly resulted in the explicit support of temporal SQL in many existing relational database systems. Especially, multi-version object-oriented databases allow for revisions to model evolution in time and variants enabling parallel ways of development [29, 115]. In addition, there are research works in the domain of CAD engineering tools and accompanying data stores [115].

These approaches have in common that the temporal aspect is solely a record with a timestamp. Additionally, the data models are residing "behind the walls" of repositories without an appropriate connection to the running system. To keep data synchronized and consistent between the system's design and runtime phase, an appropriate temporal framework is needed. Such a framework must address the hosting of data of runtime monitoring as well as the handling of temporal models that go beyond representing and processing the current state of a system and its components [61, 150, 16]. This enables the shift from monolithic models to evolutionary models populated by timing aspects, where the focus is not only on the current state to steer the system but also on the history of changes.

Challenge II: Handling Temporal Models

To tackle the limitation and open issues described above, it is essential to focus not only on providing a model infrastructure to instantiate models but on the history of changes of those models and their components (e.g., state changes, value changes) during the system's operation. For this purpose a specific temporal model framework is required that enables an automatic monitoring of systems by using temporal models. To further close the gap between design time modeling activities and runtime monitoring activities [59], an explicit mapping of design models to time series databases that can be considered as a special type of temporal databases [20, 116] is needed. This mandates a dedicated profile for extending metamodels with an appropriate annotation mechanism to drive and optimize the generation of model-based time-series database connectors. In addition, an appropriate mapper is needed that allows to inject data to time-series databases from model changes as well as to extract data from the time-series databases by model-based queries, e.g., in OCL [27], for efficient monitoring purposes.

1.4.3 Data Analytics

The term *data analytics* subsumes techniques examining big data sets to uncover hidden patterns, unknown correlations, and other information that is used to make better decisions. The steps that data passes through in most applications are the following: acquiring, cleansing, storing, exploring, learning, and predicting [111]. Mostly, the acquisition of information that derive from a big amount of data gathered from heterogeneous sources is challenging. Since, there are very large data volumes (*Volume*), data arrives very fast in form of data streams (*Velocity*), and data has varying and complex formats, types, and meanings (*Variety*) [5, 79]. Later on, further characteristics of big data, that are not discussed in detail here, were identified: value, veracity, viscosity, variability, volatility, viability, and validity [75]. These characteristics of big data projects put high requirements on data storage (e.g, NoSQL, RDBMS), data processing (e.g., batch, incremental, interactive), orchestration (scheduling, provisioning), and interfaces for offering access points (e.g., SQL, script, graphical) to mention only four of six pillars of the big data analytic ecosystem, as introduced in [74].

A multitude of methods and techniques have emerged to support data analytics, such as: data mining [50], the CRISP-DM process [143], pattern recognition and machine learning [17]), anomaly detection (e.g., in [30]), predictive as well as prescriptive analytics (e.g., [123, 111]), and process mining [127]. At the end the aim is to recognize meaningful data models within big data, e.g., to recover existing patterns and to discover new ones [41].

Challenge III: Semantic Data Analytics

The *model-driven perspective* describes how an intended system should work. The *data-driven perspective* focuses on how data logs can be taken from a system during operation to reason about the actual realization of the system. While the former perspective is lacking concepts to define runtime data such as time-series, the latter one has to correctly interpret the collected data logs. Often, data is stored in various text-based formats (e.g., XML, comma separated file formats), which are difficult to read and process, since the stored information is not obvious. Thus, it is not possible to easily trace runtime processes in the system and between system components. This requires a scalable reverse-engineering approach to provide an object-oriented view on this data and to enable the reverse engineering of runtime information into design models. The challenge is to overcome the gap between both perspectives by (i) monitoring meaningful data from operation, (ii) aligning data logs with the design model for providing a semantic anchoring of data, and (iii) providing meaningful analytics to reason about improvements or fulfillment of given requirements.

1.5 Contributions to *Temporal Model*-Driven Systems Engineering

This section presents in detail how the papers (**Papers 1 to 14**) part of this cumulative habilitation thesis address the challenges discussed in Section 1.4. It elaborates on solutions for considering interoperability aspects as well as evolutionary aspects of engineering artifacts by adding a temporal dimension to models and model elements.

1.5.1 Overview

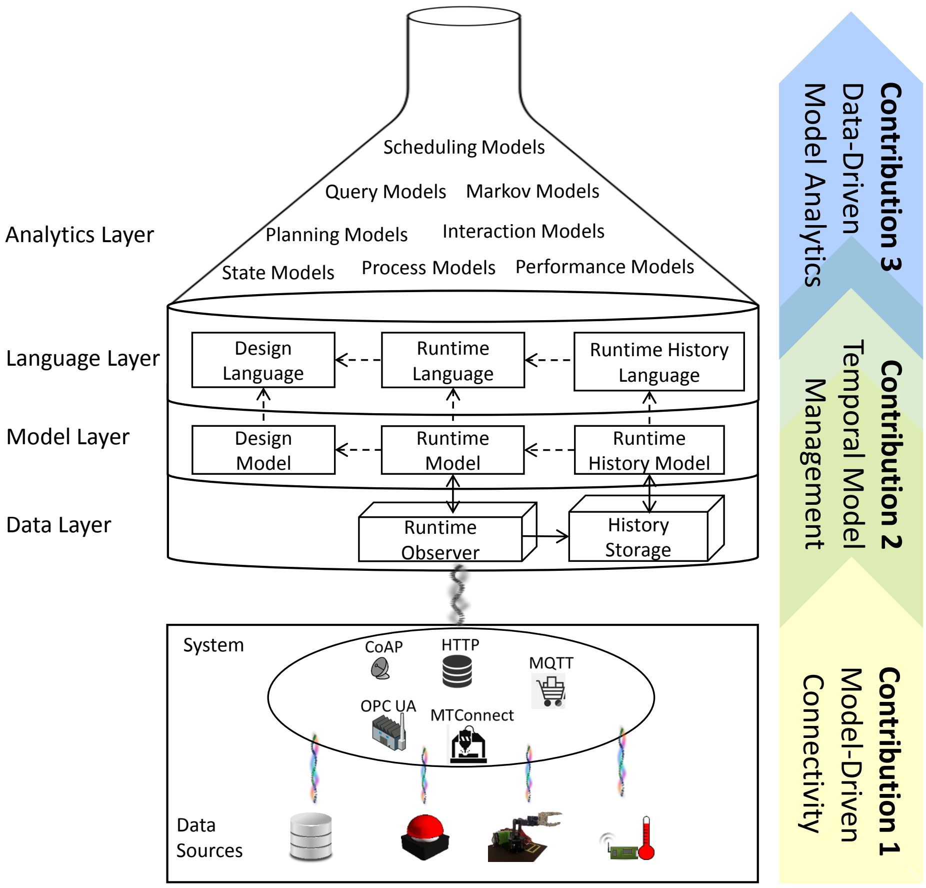


Figure 1.1: Unifying Framework for *Temporal Model*-Driven Systems Engineering.

The contributions are located in three main research areas: (i) *Model-Driven Connectivity*, (ii) *Temporal Model Management*, and (iii) *Data-Driven Model Analytics*. Figure 1.1 illustrates how these contributions are arranged. In particular, the focus is on how to connect runtime environments (i.e., heterogeneous data sources) to a temporal model framework to extract so-called *operation models* from the `Data Layer`, connect them to design models at the `Model Layer`, and to establish (logical) model analytics at the `Analytics Layer`. For this purpose an appropriate language extension on the

`Language Layer` is required based on the operational semantics of the employed language. Thus, the contributions are as follows:

(i) *Model-Driven Connectivity* deals with the combination and revamping of information from heterogeneous sources into new representations and provides a model-driven approach to accumulate various engineering views from embedded systems over network technologies. Thus, it tackles the integration of heterogeneous tools from various engineering domains (i.e., manufacturing, Architecture, Engineering, Construction (AEC) industry) through mappings realized as model transformations of underlying industry standards.

(ii) *Temporal Model Management* provides the capability to handle and store runtime as well as historical data by a unifying framework linking this data to temporal models, and additionally, providing query capabilities. In particular, we enrich models with a temporal component to shift them from static to evolutionary artifacts on the basis of the data gathered in the operation phase of a system. This enables an explicit representation of time on the model level as well as on the model element level.

(iii) *Data-Driven Model Analytics* investigates on the issue of model element changes over time. We apply model-driven approaches to reason about a system's behavior and model element changes based on the descriptive (operational) models obtained from runtime.

These contributions were evaluated by either observational, analytical, experimental, or descriptive methods. For this purpose we built up three lab-sized automation system environments with an increasing complexity: a traffic light system, a self-driving car, and a five axes grip-arm robot. Additionally, we employed test-beds from research partners of the Otto-v.-Guericke University Magdeburg (e.g., as presented in **Paper 11**) and the CTU's Czech Institute of Informatics (e.g., as presented in **Paper 5**).

The research work on the papers collected in this thesis has been conducted in the context of the following research projects where the applicant has been acting as a scientific leader and/or principal investigator:

- InteGra 4.0 — Horizontal and Vertical Interface Integration 4.0 an Exploratory Study, FFG ICT of the Future, Austrian Research Promotion Agency (FFG) and the Federal Ministry of Science, Research and Economy (BMWFW), [849944], 2015–2016, Principal Investigator;

- DigiTrans 4.0 — Innovationslehrgang zur Digitalen Transformation in der Produktentwicklung und Produktion, FFG Innonvationslehrgänge, Austrian Research Promotion Agency (FFG) and Austrian Federal Ministry for Digital and Economic Affairs (BMDW), [854157], 2016–2018, Principal Investigator;

- CDL-MINT — Christian Doppler Laboratory for Model-Integrated Smart Production, Austrian Federal Ministry for Digital and Economic Affairs (BMDW) and the National Foundation for Research, Technology and Development (CDG), 2017–2020, Scientific Lead of the Module Reactive Model Repositories;

- TransIT — Plattform zur digitalen Transformation im Tief- und Tunnelbau, Austrian Federal Ministry for Education, Science and Research (BMBWF), [BMBWF-11.102/0033-IV/8/2019], 2020–2024, Principal Investigator.

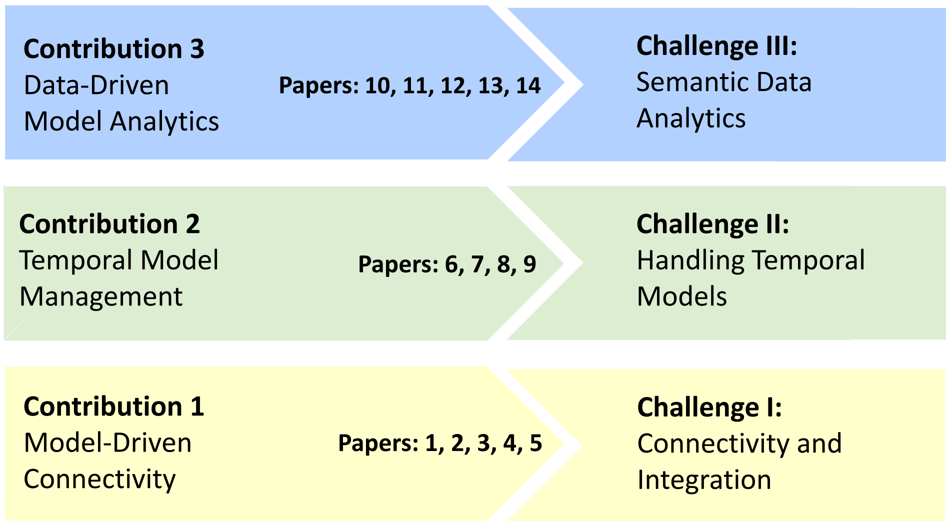


Figure 1.2: Mapping of Contributions to Challenges.

Figure 1.2 gives an overview of the contributions, which we map to the identified challenges. Furthermore, we specify the papers including the contributions to solve the challenges. In the following, we present the list of these papers:

Contribution 1: Model-Driven Connectivity

1. A. Mazak and C. Huemer: *From business functions to control functions: Transforming REA to ISA-95*; Proceedings of the 17th IEEE International Conference on Business Informatics (CBI), IEEE Computer Society, (2015), pp. 33–42 [90].

2. B. Wally, C. Huemer, A. Mazak and M. Wimmer: *AutomationML, ISA-95 and Others: Rendezvous in the OPC UA Universe*; Proceedings of the 13th IEEE International Conference on Automation Science and Engineering (CASE), IEEE, (2017), pp. 1381–1387 [139].

3. G. Paskaleva, A. Mazak-Huemer, M. Wimmer and T. Bednar: *Leveraging integration facades for model-based tool interoperability*; Elsevier Automation in Construction, 128, (2021), 103689 [110].

4. B. Wally, C. Huemer and A. Mazak: *A View on Model-Driven Vertical Integration: Alignment of Production Facility Models and Business Models*; Proceedings of the 13th IEEE Conference on Automation Science and Engineering (CASE), IEEE, (2017), pp. 1012–1018 [137].

5. B. Wally, J. Vyskočil, P. Novák, C. Huemer, R. Sindelár, P. Kadera, A. Mazak-Huemer and M. Wimmer: *Flexible Production Systems: Automated Generation of Operations Plans Based on ISA-95 and PDDL*; IEEE Robotics and Automation Letters, 4, (2019), pp. 4062–4069 [140].

Contribution 2: Temporal Model Management

6. S. Wolny, A. Mazak, C. Carpella, V. Geist and M. Wimmer: *Thirteen years of SysML: A systematic mapping study*; Journal of Software and Systems Modeling, 19(1), (2020), pp. 111–169 [146].

7. A. Mazak and W. Wimmer: *Towards Liquid Models: An Evolutionary Modeling Approach*; Proceedings of the 18th IEEE International Conference on Business Informatics (CBI), IEEE Computer Society, (2016), pp. 104–112 [93].

8. S. Wolny, A. Mazak, M. Wimmer, R. Konlechner and G. Kappel: *Model-Driven Time-Series Analytics*; Enterprise Modelling and Information Systems Architectures - International Journal of Conceptual Modeling, 13(Special), (2018), pp. 252–261 [150].

9. A. Mazak, S. Wolny, A. Gómez, J. Cabot, M. Wimmer and G. Kappel: *Temporal Models on Time Series Databases*; Journal of Object Technology, 19(3), (2020), pp. 3:1–3:15 [96].

Contribution 3: Data-Driven Model Analytics

10. A. Mazak, P. Patsuk-Bösch and M. Wimmer: *Execution-Based Model Profiling*; Data-Driven Process Discovery and Analysis – 6th IFIP WG 2.6 International Symposium (SIMPDA), Revised Selected Papers, Vol. 307, LNBIP, Springer, (2016), pp. 37–52 [95].

11. A. Mazak, P. Patsuk-Bösch and M. Wimmer: *Reverse Engineering of Production Processes based on Markov Chains*; Proceedings of the 13th IEEE International Conference on Automation Science and Engineering (CASE), IEEE, (2017), pp. 680–686 [94].

12. A. Mazak, S. Wolny and M. Wimmer: *Automatic Reverse Engineering of Interaction Models from System Logs*; Proceedings of the 24th IEEE International Conference on Emerging Technologies and Factory Automation, (ETFA), IEEE, (2019), pp. 57–64 [148].

13. S. Wolny, A. Mazak, M. Wimmer and C. Huemer: *Model-driven Runtime State Identification*; 40 Years SIG EMISA: Digital Ecosystems of the Future: Methods, Techniques and Applications, 39(1), De Gruyter, (2019), pp. 29–44 [149].

14. W. Wimmer and A. Mazak: *From AutomationML to AutomationQL: A By-Example Query Language for CPPS Engineering Models*; Proceedings of the 14th IEEE International Conference on Automation Science and Engineering (CASE), IEEE, (2018), pp. 1394–1399 [141].

In the following, each of the listed contributions is presented in more detail. We explain how each challenge is addressed according to the *type* of the contribution. Furthermore, we present the *evaluation* of each of the contributions.

1.5.2 Contribution 1: Model-Driven Connectivity

The contributions of the presented papers in this section deal with the combination and revamping of data from heterogeneous sources into new and harmonized representations for providing meaningful insights. Furthermore, our approaches yield towards a nearly seamless exchange of relevant information, both vertical (from the data sources to the data layer downwards and upwards, cf. Figure 1.1) and horizontal (from one system to another at the same layer). After resolving certain forms of heterogeneity in the handling and integration of data, the data is processed within distributed operational models for further elaboration.

Paper 1: From business functions to control functions: Transforming REA to ISA-95 [90]

In the context of Industry 4.0, a seamless information exchange between information systems on the same layer (i.e., *horizontal integration*) and between information systems on different layers (i.e., *vertical integration*) is a key issue. In 2013, the German working committee for Industry 4.0 pointed out that production systems are to be linked vertically to business processes within decentralized production sites and enterprises, and that they are to be distributed horizontally among suppliers, distributors, and companies. In order to meet these requirements, we aim for an *integrated modeling framework* spanning over the horizontal layer (value networks) and the vertical one (production chains). Thereby, MDE acts as an enabler for managing such an interface integration framework. This means that we do not start from scratch by defining an own all-encompassing modeling language, rather we build up on existing well defined standardized ones.
For *vertical integration* of information flows within the company, we consider the concepts and models of the standard *ISA-95* (ANSI/ISA-95; IEC 62264) [69]. This interna-

tional standard describes information flows between *Enterprise Resource Planning Systems (ERP)* on the enterprise level and the *Manufactoring Execution Systems (MES)* on the control level. The interface for the *horizontal integration* of information flows of different business partners is described by the concepts of the *Resource-Event-Agent business ontology (REA)* (ISO 15944-4) [1]. In the business domain, REA is used to identify the value-adding activities of a company. For realizing integration by means of value networks, we transform REA concepts to ISA-95 models (e.g., material model). This approach is independent of any software solution. This means that companies may use different ERP and MES systems and still collaborate with each other.
Type of Contribution: We present a meta model (cf. Figure 1.1, at the `Language Layer`) for REA and for ISA-95 to align the concepts of both. For this purpose we extend the resource concepts of REA by similar concepts from ISA-95 for providing specialized concepts of resources such as equipment, physical asset, and material. These extensions concern the REA type level. Most importantly, we develop dedicated transformation rules for an automatic mapping to transform a REA model into a ISA-95 one. In particular, we map REA *duality models* to ISA-95 *operation segments*. This enables converting information about input and output of business functions to control functions.
Validation of Contribution: In a first step, we implemented the proposed REA extensions into our REA-DSL tool, which we developed in previous research work [89]. In a second step, we added transformation rules. In this initial paper, we hard coded these rules. In subsequent contributions (see **Paper 2** and **Paper 4**), we worked on an automated generation of these rules. We demonstrated the technical feasibility by mapping from our REA DSL to B2MML (the XML equivalent of ISA-95). The syntactical correctness of the transformation was checked by the proof of valid B2MML instances.

Paper 2: AutomationML, ISA-95 and Others: Rendezvous in the OPC UA Universe [139]

Based on our lessons learned of **Paper 1**, we extend the previously presented approach by exploiting two additional standards, the *Automation Markup Language (AutomationML)* [34] and the *OPC Unified Architecture (UA)* [52]. AutomationML consists of three parts: (i) CAEX, (ii) COLLADA, and (iii) PLCOpen. CAEX (Computer Aided Engineering Exchange) is a data format that has been defined in the scope of IEC 62424 [32]. CAEX is based on XML and enables the metamodeling and modeling of, e.g., the hierarchical architecture of a plant, including involved machines and controllers and their physical and logical connections. AutomationML defines an abstract interface class *ExternalDataConnector* which is used to reference external documents and elements therein. Two use cases of this external data connector have been defined in separate whitepapers: (i) COLLADAInterface specifies how external COLLADA3 documents are referenced [36] and (ii) PLCopenXMLInterface defines how PLCopen4XML documents [2] (which are based on IEC 61131-3 [33]) can be referenced from AutomationML documents [35].

This referencing mechanism plays an important role in the analysis of the presented contribution.

OPC UA is a commonly used communication standard in industrial settings. It is a versatile platform for hosting information from a large variety of domains [86]. Those domains provide only partially overlapping information, since there are different views on a specific entity or different levels of detail needed for describing a specific interest, etc. Emerging from a multi-disciplinary engineering process, these different views can stem from various tools that have been used to deal with that entity, or from different stages in an engineering process, e.g., from requirements engineering over system design and implementation to operations.

The problem that needs to be addressed is that domain-specific information is kept out of the OPC UA standard as strictly as possible [31]. Instead, companion specifications are used to extend OPC UA by introducing meaning (semantics) depending on the domain of interest. Furthermore, numerous engineering tools are used in the design (and runtime) of automated productions systems. The high number of combinations of engineering tools and companion specifications leads to a great interoperability challenge. We argue that a concise but expressive set of OPC UA reference types that allow the persistent instantiation of additional knowledge with respect to relations between OPC UA nodes may help with the semantic alignment of such diversified entities.

Type of Contribution: We provide an OPC UA information model for explicitly linking heterogeneous data, generated from different source domains. This information model provides a concise but expressive set of OPC UA reference types: *RepresentDifferentViews*, *HasRefinement*, *HasVerification*, *HasImplementation* that allow the explication of relations between engineering artifacts. Instances of these reference types are meant to be used between OPC UA nodes of different domains. However, they can also be applied in the context of a single domain, if this domain does not provide corresponding reference types that might be of value. These reference types can be sub-classed to describe more specialized inter-model relations, if required.

Validation of Contribution: We created a set of domain models based on AutomationML, ISA-95, and MTConnect [51] to evaluate our set of OPC UA reference types. For each of these models there exists a mapping to OPC UA, or they are natively designed in OPC UA. As evaluation example, we considered a milling device. We represented this device named *MyMilling* as three different components at an OPC UA server: (i) as an AutomationML internal element, (ii) as an ISA-95 physical asset, and (iii) as a MTConnect device. Using the *RepresentDifferentView* reference type it becomes clear that the different OPC UA nodes are all describing the same physical entity, a milling device, but from different domain specific perspectives. A similar example was given with the entity *MyWorker*. In AutomationML it is a proxy element (usually personnel is not modeled in AutomationML), but modeled in more detail in ISA-95 by the personnel model. This refinement was expressed using a *HasRefinement* relation. The validation of the type set showed that there are use cases in which such explicit links make sense, e.g., when the domain model does not provide a facility to describe the relations as discussed.

Paper 3: Leveraging integration facades for model-based tool interoperability [110]

In the domain of Architecture, Engineering and Construction (AEC) industries, the volume of data to be exchanged among various stakeholders (e.g., building developers, energy suppliers, architects, structural engineers, building physicists, etc.) is rapidly increasing in daily business. Each of these stakeholders possesses specific domain knowledge and has a specific view on the building project. These different perspectives may cause different forms of heterogeneity in the definition and handling of data that hinders an interference-free communication. It is still common practice that IT systems exchange information through extensive interfaces, but can only utilize specific pieces of that information. The situation is further worsened by the problem that many different interfaces introduce dependencies whose management can become complex and hard to achieve. This leads to a drastically rising complexity of the system. In this paper, we focus again on multi-disciplinary engineering processes. We discuss the lack of interoperability among different domain-specific engineering tools, and heterogeneity issues resulting from different perspectives of stakeholders on the same entity as well as on different information granularity needed in various project phases. In particular, we transfer the lessons learned of **Paper 2** to the application field of the AEC industry and extend that previous research work.

For this purpose we explicitly formulate challenges to work on when implementing a road map towards full semantic and pragmatic integration that need to be tackled in any data exchange process, namely *exposing semantics*, *exposing functionality*, *staying up-to-date*, and *making pragmatics explicit*. These challenges address a seamless integration considering the semantics and pragmatics of a single application. Nevertheless, full interoperabiltiy requires an unbroken communication network involving multiple applications. Additionally, the data exchange within or across different phases of a building project is challenging due to different exchange standards in use. Therefore, *Building Information Modeling (BIM)* has become more and more established in recent years [124]. The main motivation of BIM is accommodating heterogeneous nature of the AEC industries and involved domains, and providing seamless data flows within any building or infrastructure project [21].

Type of Contribution: Interoperability in BIM requires integration along, both, the semantic and the pragmatic dimensions. In this contribution, we present and evaluate a modeling framework capable of working with multiple semantic type systems inhabiting the same syntax in the context of multiple applications. In addition, the framework

provides formal methods for modeling the pragmatic aspects of a data exchange and converting them from an implicit assumption into an explicit formal specification. It enables the integration of semantics at runtime and of pragmatics at design time. This combined model of semantics and pragmatics provides a so-called *integration facade* that enables transparency and traceability during interoperability testing.
Validation of Contribution: *Use Case 1: an application simulating particle motion under gravity.* This software simulates the motion of a swarm of particles, each with an initial mass and velocity, under the influence of gravity. The results are represented by an animation on a two-dimensional canvas and as a table containing mass, positions and velocities. *Use Case 2: an application simulating the propagation of sound waves.* This software simulates the propagation of sound waves generated by user-defined sound emitting point or line sources. The resulting interference pattern is calculated numerically over a discrete grid and is displayed as an animation on a canvas in one, two, or three dimensions. With these two use cases we validated the overcome of the addressed *Challenge 1: exposing semantics* and *Challenge 2: exposing functionality*. For *Use Case 3*, we used a spreadsheet for simulating the thermal behavior of a single space. Thereby, we calculated the temperature, humidity, and CO^2 concentration in a single enclosed space over the period of one week during a heat wave in summer. This case demonstrated the steps involved in addressing *Challenge 4: translating between different semantics*.
In all of these three use cases, we made adaptions in the source code and evaluated both the adaption itself and the implementation of the presented domain-specific data exchange workflows. The results show that the prototype of our modeling framework for integration facades provides full integration for any data model, i.e., a semantic and pragmatic consensus. This independence of concerns, allows the semantics to be modelled independently of the notation or representation of information in any available tool, which makes our approach universally applicable to data exchange processes. In addition, we can model pragmatics, both, along the semantic and the representational dimensions. This makes a full integration of semantics and pragmatics feasible into an *integration facade*, which is a prerequisite for producing a so-called “single source of truth” as envisioned for Big Open BIM.

Paper 4: A View on Model-Driven Vertical Integration: Alignment of Production Facility Models and Business Models [137]

With this paper, we continue our work of **Paper 1**. In this **Paper 4**, the application focus is also on Automated Production Systems (aPS) where modern IT systems are required at all levels of the automation hierarchy: from business-related software at the corporate management level, down to the programmable logic controllers at the field level. For a well-designed coupling of systems that are located at different levels, it is necessary to identify, define, and implement clear data conversion mechanisms. This endeavor is also known as *vertical integration* as already explained in **Paper 1** and is further elaborated in **Paper 5**.

In a vertical integration scenario, IT systems of different vendors might be in use and proprietary interfaces need to be defined in order to exchange relevant information from one system to another, an issue we already discussed and worked on in **Paper 3**. In this paper, we present an MDE approach for the co-evolution of models, residing on different levels of the automation hierarchy, based on a generic alignment of corresponding metamodels through appropriate model transformations (cf. Section 1.2.2).

Type of Contribution: The contribution is an MDE-based architectural metamodel for vertical integration of IT systems. We integrate parts of AutomationML, IEC 62264-2, and REA (cf. **Paper 1**) through model transformations techniques, more specific by M2M-transformations. The metamodels of these three industry standards are used for the representation of hierarchy level-specific system properties and the alignment of key concepts in order to provide bridging functions for the transformation between different IT systems. We provide explicit mappings between model elements, and therefore, do not rely solely on the metamodel level (cf. Figure 1.1 at the `Language Layer`). The approach enables that (i) changes in one system are automatically propagated to other systems, if possible, and (ii) the overall architectural model is not modeled explicitly, but to be inferred from multiple domain models. We employ the Epsilon Object Language (EOL) [46] for querying model states, Epsilon Transformation Language (ETL) [47] for M2M-transformations, and EVL [48] for the validation of models.

While the given technique does not provide a *single point of intervention* when it comes to changes in the models, it facilitates the creation of stub models and provides means for cross-model validation. The main contribution of this paper is the model-driven propagation of basic model elements and changes of model elements between models of different hierarchy levels.

Validation of Contribution: We validated the approach by an in-depth domain analysis using an application scenario of a fictitious company which we named *Glulam Ltd.*, specialized in the production of *glu*ed *lam*inated timber. The idea was born from a visit in a real company of the woodworking sector in the course of the InteGra 4.0 project. Even though, we evaluated the application of our approach on a very specific fictitious company, the approach itself and most of the implementation are company as well as domain agnostic. In the evaluation results, we refrained from presenting details about the transformation between AutomationML and ISA-95 models, as the alignment between these two metamodels was demonstrated in more detail in our research work presented in [138] (a paper which is not part of this cumulative habilitation thesis). This means that corresponding transformation rules can be inferred from the mappings defined in that previous paper. The results show that the developed metamodel provides a semantically enriched revision of the XML schema-based original definition of the CAEX format, and so of AutomationML.

The evaluation figures out that the benefit of this approach is that a common understanding of concepts from different domains is accomplished by relating metamodel elements to each other. This approach is agnostic to the kind of business a company is involved. Specific implementations could consider industry-related information in order to better acknowledge peculiarities and conventions.

Paper 5: Flexible Production Systems: Automated Generation of Operations Plans Based on ISA-95 and PDDL [140]

Since, the standard IEC 62264 (see **Paper 1** and **4**) provides a conceptual model for the representation of manufacturing operation management information, in this paper we explore how this information could be exploited for an automated generation of production plans. For this purpose we use the *Planning Domain Definition Language (PDDL)* [57] as encoding format. PDDL provides a standardized and object-oriented way of specifying planning domains and concrete planning problems.
The idea behind the presented approach is that IEC62264 (as already showcased in **Paper 1**) can be used to model (i) the machinery of a production system (the equipment), (ii) the material that is being consumed and produced, (iii) the production processes that are available (the so-called *process segments*), as well as (iv) the relations that arbitrary resources can have to each other. Our approach establishes a conversion rule from IEC 62264 information to PDDL information. In essence, the IEC 62264 process segments are translated into PDDL actions that could be applied by a planning solver in order to progress from an initial state to a goal state. In our presented use case, we automatically compute the sequence of actions that will be necessary to prepare just-in-time-delivery of raw material for a production process that is to be executed later, by ordering a sequence of shuttles in a monorail intra-logistic transportation system.
Type of Contribution: The main contribution is the transformation of IEC 62264 information into a valid PDDL representation. For this purpose we develop a concrete metamodel and appropriate M2T-transformations for PDDL. This provides the transformation of pure IEC 62264 metamodel information to PDDL domain description fragments which we examine by the example of equipment information. By employing this metamodel, classes are made available as PDDL types and metamodel relations are transformed to PDDL predicates with a corresponding name as well as properly typed parameters. The PDDL elements generated in this way serve as the backbone for the remaining elements that are inferred from IEC 62264 model information. For instance, (i) all classes (e.g., equipment classes) are converted into constants, (ii) process segments are converted into actions and their segment requirements are added thereto as parameters, and (iii) all kinds of instances (e.g., equipment) are converted into objects of a corresponding problem definition file.
The so computed production plan is then written back into the production system

model in terms of an operations definition, where each PDDL action call is converted into an operations segment. If no plan can be found, no such operations definition is created, and instead, the user is notified that no plan could be found for the given production problem. The corresponding engineer is then able to explore whether the product is faulty, or the production system is simply not capable of producing the selected kind of product.
Validation of Contribution: We applied our research concepts on the basis of a lab-sized Industry 4.0 testbed at the CTU's Czech Institute of Informatics, Robotics and Cybernetics (CIIRC) [66]. This testbed comprises four robot cells which are connected by a monorail-based intra-logistics system featuring shuttles that may drive along the track and carry material to and from the cells. We created a workflow, starting with the design of a production system model based on the IEC 62264 standard with an initial state description and the formulation of an envisioned goal state. Additionally, we generated a set of PDDL artifacts, one domain file and one to many problem files, corresponding to the number of provided goal state models. These files were handed over to an off-the-shelf PDDL solver that tried to find a sequence of actions leading from the initial state to the goal state (i.e., a so-called production plan). For this case study, we assembled toy trucks from a few sub-components. We could successfully proof (i) that the production system that has been designed is capable of re-sorting the transportation shuttles from a source sequence to a target sequence and (ii) how exactly the re-sorting can be accomplished, in terms of step-by-step directions.

1.5.3 Contribution 2: Temporal Model Management

For the contribution of *Temporal Model Management*, we elaborate the research directions listed below. These research directions have been identified by a systematic mapping study presented in **Paper 6: Thirteen years of SysML: A systematic mapping study** [146]:

Model Life Cycle Support: The results show that there is only limited support when using a modeling language such as SysML in the implementation phase, and very limited support for describing the whole life cycle of a system's model from design to operation and backwards. These limitations are also discussed in **Paper 7** [93], **Paper 8** [150], and **Paper 9** [96]. The SMS concludes that there is a need to exploit and adapt modeling languages (e.g., UML, SysML) for supporting the execution and analysis of systems during runtime as well as to align operational data with design model elements.

Modeling Hybrid Systems: Most of the selected publications in the SMS consider either discrete or continuous challenges when designing systems [8, 70]. This means that very rarely hybrid solutions in systems design are provided [56]. Therefore, further investigations should be undertaken for designing formal semantics for SysML to close the gap when combining discrete and continuous modeling and simulation. A challenge, we discuss in **Paper 11** [94], **Paper 12** [148], and **Paper 13** [149].

Operational Semantics: Currently, there is no support, e.g., to shift property specification and verification tasks up to the model level. There is still a rule-based operational semantics missing to ensure a step-wise, state-based semantics, e.g., to describe a finite execution trace through a sequence of changes. In this context we present solutions in **Paper 3** [110], **Paper 8** [150], **Paper 10** [95], and **Paper 11** [94].

Paper 7: Towards Liquid Models: An Evolutionary Modeling Approach [93]

This paper presents initial concepts towards the extension of MDE by temporal aspects. Many of the ideas contained in this paper reflect our experience gained in the course of the InteGra 4.0 project (cf. Section 1.5.1). In this exploratory study, we conducted in-depth face-to-face interviews with managers as well as engineers of three software companies and nine companies from the areas of steel processing, wood processing, and paper production. These nine companies were a mix of small, medium-sized, and large companies, with varying economical dependencies. The diversity of the companies helped us to get a better understanding of open gaps between a system's development and operation and motivated us to this paper.
Type of Contribution: We present early conceptual results of a unifying architecture for hosting so-called "liquid models". This framework links design models to runtime concerns derived from distributed and heterogeneous systems during operation. We elaborate on proposed technologies for the layers of this architecture and identify research challenges ahead. At the same time the artifact represents a corner stone of the research work of Module 3 in the CDL-MINT project (see Section 1.5.1).
Validation of Contribution: We presented a first draft of an architecture, for stimulating a shift from isolated, one-shot, monolithic system descriptions to evolutionary, reusable artifacts. Based on this, we defined open research issues based on the identified open challenges derived from a comprehensive research of the state-of-the-art of various research fields (e.g., MDE, Process Mining, Software and Systems Engineering, etc.). Additionally, we presented a very first proof-of-concept of methods and techniques to address the challenges.

Paper 8: Model-Driven Time-Series Analytics [150]

In this paper, we move towards a well-defined mix of approaches to better manage the full life cycle of a system by combining prescriptive and descriptive model types. In particular, we introduce a model-driven time-series data analytics architecture for harmonizing model-driven and data-driven approaches. Based on this architecture, we show how data analytics works for modeling languages using standard metamodeling techniques. This means, design-oriented languages are extended for representing runtime states as well as runtime histories, which in turn allow the formulation and computation of runtime properties by employing the *Object Constraint Language (OCL)* [108]. The extensions needed on the metamodel level are non-intrusive and related to existing approaches for specifying the operational semantics of languages. The presented runtime history metamodel fragments are applicable for any modeling language used during design (e.g., UML, SysML) comprising features to be measured and events to be tracked as the current metamodeling languages Ecore and OCL are reused.

Type of Contribution: We present a unifying architecture, as concrete metamodel, for a design language to cope with a model-driven as well as data-driven perspective on systems. This architecture builds on the classical MDE approach by modeling a system downstream in terms of code generators, but at the same time supports an upstream in terms of mapping data back to design models. At the metamodel level (cf. Figure 1.1 at the `Language Layer`), the design language is defined with the help of a metamodeling language, which is in our setting Ecore. Conforming to the design language, the design models are defined at the model level describing the static (i.e., structural) as well as dynamic (i.e., behavioral) aspects of a system to be developed. For the vertical transition from the model to the realization level, we assume the existence of an M2T-transformation code generator, as presented further in **Paper 10**, **Paper 11** as well as **Paper 12**. In a next step, we generate a so-called *runtime observer* from the design model. The runtime observer collects important information from the running system to represent the current state of the system. Those observations should not only be recorded by observing the running system, but should also be representable at the model level. Thus, we extend the design language with a dedicated *runtime language*. This metamodel defines the syntax to represent snapshots of the running system connected to the design model elements. Those snapshots are represented in the so-called *runtime state models* which extend the design models and may be directly updated by the observer during runtime.

In summary, our architecture is used to monitor a system on the model level. In a first step, we map runtime data at the model level for one single point in time. In a next step, we define the runtime history of a system. For reasoning purposes, it is important to have the complete history of value changes as starting point, since a single snapshot is definitely not sufficient for giving useful insights of the operation of a system. Thus, in the time-series database, we store the observations of the running system. Based on those collected observations, the *runtime history models* may be directly updated. These models conform to the *runtime history language*, which is an extension of the runtime language. In the runtime history language, the syntax is defined for representing histories of runtime phenomena of interest, e.g., property values, events, etc. Finally, after

defining those concepts for storing observation histories at the model level, it is also possible to analyze those observations. For this purpose we define *runtime properties* based on OCL by introducing derived properties for the metamodel elements.
Validation of Contribution: Our recurring example of a lab-sized five axis grip-arm robot is used for evaluating the pertaining temporal model management and data-driven model analytics. We validated the concepts of the proposed design modeling language, i.e., extensions for runtime states, runtime histories, and runtime properties, as reusable metamodeling "blueprints". We demonstrated the time series analysis regarding property value changes of the gripper's axis angles during operation in a laboratory-like environment of a sorting system. The implementation and evaluation results can be found on our project page [97].

Paper 9: Temporal Models on Time Series Databases [96]

Building on the temporal architecture presented in **Paper 8**, we extend the approach for enabling partial mappings from metamodels and their instances (i.e., models) to time-series databases. In addition to time-series representations, this special type of temporal databases enables time-series analytics in order to deal with additional activities in engineering technical systems, an emerging trend we surveyed in **Paper 6**. In the presented approach, we allow for model simulation runs which may be analysed by time series analytics as well as for model-based runtime monitoring of systems reporting their changes and states to time series databases. We demonstrate both scenarios in a production system case study and evaluate in particular two mapping strategies with respect to the required data storage and query answering performance.
Type of Contribution: For combining models, especially EMF-based models, with a *Time Series Database (TSDB)*, we aim for a *polyglot* solution where the static information resides in the model as it is already available, e.g., by XMI or other model persistence mechanisms, and only the time-sensitive information is stored in the TSDB. The two storage parts are combined by a ModelAPI that can be accessed and used by various applications. This unifying API abstracts implementation details and allows for a similar way of working with models as it is provided by EMF out-of-the-box. In particular, we reuse as much as possible and only extend those parts which are really required. As a result, the model sticks as close as possible to the EMF standard and the required information for the TSDB is attached in a light-weight manner. As design choice for implementing the unifying ModelAPI with a polyglot, we propose in a first step a so-called *Time-Series (TS) profile*, realized with EMF annotations, for extending existing metamodels by time-series aspects. This TS-profile defines different kinds of stereotypes for annotating classes, structural features as well as operations. Based on the usage of the profile, we present the so-called *Model-to-Time Series Mapper (M2TS-mapper)* and two mapping strategies for this mapper: (i) to store each temporal

property individually (strategy of *single property mappings*), and (ii) to store the whole object with all its associated information (strategy of *complete object mappings*). For instance, the first one enables to map a single property like the temperature of a room in order to continuously log the progression of the property value in the TSDB and to query it. By employing the second one, individual properties do not have to be annotated as temporal features (like in case of single property mappings), but the containing classes, and thus, the associated objects with their properties are stored in the TSDB as measurements. On the basis of the TS-profile and the applied mapping strategy, we provide appropriate query capabilities in a further step. These query capabilities enable the M2TS-mapper not only to inject data to the TSDB, but to extract data from the TSDB by model-based queries. As the derived runtime properties are in essence standard-derived properties, they can be simply reused in standard OCL queries.
Validation of Contribution: For evaluation purposes we validated (i) the *scalability* of database sizes and (ii) the *performance* of runtime queries. For (i), the TSDB size on the basis of model changes showed that both strategies had a linear increase. We recognized that the size of the database for the complete object mapping strategy increased slightly faster than for the single property mapping strategy. This could be explained by the fact that whenever a value of a property changed, the entire object was stored with a new timestamp. Regarding the performance of runtime queries ((ii)), the queries were fast (from 1ms to about 7ms) depending on the entries in the TSDB. However, as the size of the database increased, the queries for `MeanMaxMode` and `GetValueAt` for the single property mapping became slightly slower than in the case of the complete object mapping. This could be explained by the fact that starting from a certain number of entries, it plays a role whether the possible results have to be selected first, or are already selected and only need to be screened. Finally, the hypothesis testing (i.e., Wilcoxon rank-sum testing [130]) showed that there was no significance regarding the difference between the two mapping strategies.

1.5.4 Contribution 3: Data-Driven Model Analytics

During the early phases, MDE approaches are frequently used to design systems, whereas during operation data-driven approaches are used to reason about the system's behavior based on data logs. The main challenge is to establish so-called *trace links* between the initial design model and the operation model for (i) monitoring meaningful data from operation, (ii) aligning data logs with the design model for providing a semantic anchoring of data, as well as (iii) providing meaningful analytics to reason about improvements or the fulfillment of given requirements. Additionally, it must be considered that the data acquisition is taking place during operation time. As a result, the data is not a finite set, but a continuous stream of data.

Paper 10: Execution-based Model Profiling [95]

We present an initial unifying framework to combine MDE with Process Mining (PM) techniques [128, 127] on the basis of previous research work presented in [92]. Thereby, not only the dependencies between different process steps may be uncovered, but also, dependencies between data and process steps are approachable. In this **Paper 10**, we are focusing on *prescriptive* models needed for realizing a system and *descriptive* models used for describing a system as it is actually realized in a runtime environment. For aligning these two kinds of models, we introduce an approach which we call *execution-based model profiling*. Thereby, model profiles are automatically generated from execution logs of running systems. In particular, we define execution-based model profiling as a continuous process to generate observation models during a system's operation and to check whether these models correspond to the initial design model or not. The approach is based on executable modeling languages which provide operational semantics for interpreters. It further provides translational semantics in form of code generators to produce code for a concrete platform to realize a system (see Figure 1.1 `Language Layer` to `Data Layer`). This contribution follows one of the research directions identified by our survey in **Paper 6**.
Type of Contribution: In order to combine the prescriptive perspective with the descriptive one, we introduce a first conceptual unifying framework and an operational language that acts as a logging metamodel at the language layer (cf. Figure 1.1). This observation language determines which runtime changes should be logged (e.g., state changes, attribute value changes) by the proposed stereotype «observe». This stereotype has to be annotated in the design model. Thus, the metamodel defines the syntax and semantics of the data logs we observe from a running system. These execution logs are stored as observation models which conform to our observation language. These models could be then considered at the mining layer (see Figure 1.5.1) as input for any kind of analytical tool, for instance, to check non-functional properties (performance, correctness, appropriateness).
Validation of Contribution: We evaluated the unifying framework by an explanatory case study based on a traffic light system example where we combined MDE techniques with Process Mining (PM) techniques [127]. For the purpose of validating the *transformability* of the approach, the outcomes show that the operational semantics of the modeling language is rich enough to automatically derive observation metamodels from log files. The results of validating the *interoperability* of the observed models show that they fulfill the requirements of general workflow-oriented formats of PM tools. In this paper, we employed the open source tool ProM Lite in Version 1.1 [126]. For runtime verification (*usefulness*), we applied the $\alpha++$ *algorithm* of ProM Lite to derive a Petri net. The generated net corresponded to the initial state machine. Thus, we could demonstrate that the state machine was realized by the code generator as intended at

design time. For the detection of timing inconsistencies (*timeliness*), we filtered the sequence of transitions by an ATL transformation and analyzed it with the performance plug-in of ProM Lite. The inconsistencies between the specification and implementation levels were within a range of milliseconds (40 to 190 milliseconds average delay per transition). In a final step, we demonstrated that the average values of the delays could be propagated back to the design model so that the timing becomes more precise during system execution.

Paper 11: Reverse Engineering of Production Processes based on Markov Chains [94]

New technologies, such as IoT, enable us to continuously observe running systems based on sensor data, which helps us to get a clear picture of the current status of a system based on the gathered data during the execution of processes. Since, such raw data are streams of performed actions, we need mechanisms to transform those streams into so-called *descriptive models* for further analysis. Therefore, we present an automated reverse engineering approach based on Markov chains that combines model-based downstream information derived from prescriptive models with sensor-based upstream information of a lab-sized production line during operation.
Type of Contribution: We present a reverse engineering approach by computing behavioral models from timely observations based on the system components' emission in form of operational logs. These logs reflect the activities happening in a system during operation. In particular, we consider the duration time of operations applied on work items (e.g., resource-specific operation calls). We transform these logs to Markov chains. This enables us to deal with the complexity of runtime information as well as to provide reasoning mechanisms for future adaptations. The approach bases on previous research work in the field of Web engineering [6], where we additionally identified trends for user analysis in this field.
Validation of Contribution: The procedure, languages, and tools have been evaluated based on a lab-sized production system, which is hosted at IAF of the Otto-v.-Guericke University Magdeburg [68]. One main focus was on feasibility and usefulness of the introduced modular transformation chain. Furthermore, we analyzed workload characteristics and bottlenecks when using the IAF plant in certain settings by statistical performance measurings. All artifacts of this evaluation can be found on our project page [3].

Paper 12: Automatic Reverse Engineering of Interaction Models from System Logs [148]

During the execution of software when the system is operating, the executed operations can be traced based on sensor value streams or logging code. This enables to derive the

behavior of the system based on the communication of system components. Usually, such log traces have the form of huge text-based files which are difficult to work with. As a consequence it is not straightforward to fully track and understand the interaction and communication between the components. In this paper, we tackle this challenge by presenting a scalable reverse engineering approach to automatically transform log traces to an appropriate and user-friendly graphical representation. For this purpose we use established methods and techniques from MDE as basis (cf. Section 1.2) to provide an end-to-end traceability from design to runtime and backwards.
Type of Contribution: The artifacts are a so-called *log-metamodel* for enabling an object-oriented representation of system logs and an architecture to automatically reverse engineer interaction models in terms of UML sequence diagrams. This architecture is divided into three parts: (i) the creation of object-oriented interaction models in form of sequence diagrams derived from executed operations, (ii) the alignment between those sequence diagrams and the corresponding prescriptive models (created at design time) by so-called *trace links*, and (iii) the creation of a runtime profile and its display in the corresponding design models. The log-metamodel and the architecture enable an object-oriented view on executed operations and help to trace the inter-object communication among system components. Additionally, the profiled information could be back-propagated to the initial design model for reactivity purposes.
Validation of Contribution: The procedure and prototypical implementation were evaluated on the basis of a case study by employing a self-driving car. We based this case study on the guidelines of Runeson and Höst [113]. The results show that we are able to generate interaction models from system logs based on a T2M-transformation and that the relationships between runtime models and design models could be established by means of generated queries. However, the scaling of modeling tools is still a crucial issue. The strength of our approach is that we can keep the relevant information in a unified modeling language, namely UML. Thus, design tools can be reused with their integrated tooling and there is no need to learn new technologies or languages for analyzing runtime information, which is a benefit for domain experts. For presenting the case study as well as the results, we have provided a project page [145].

Paper 13: Model-driven Runtime State Identification [149]

In this contribution, we combine MDE techniques with *Time-Series Database (TSDB)* (see Section 1.5.3) and *Process Mining (PM)*. Thereby, we take up the challenge to continuously listening to value streams in order to determine whether a state has indeed occurred, i.e., if the specific combinations of variable values have occurred over all streams at the same time. In particular, the realization precision of systems as well as measuring inaccuracies complicate this process as false positives and false negatives may occur when matching state templates to data streams. Based on first ideas pre-

sented in [147], we address this challenge by introducing a novel approach where we automatically generate state realization event queries derived from state machines for an appropriate state identification at runtime. This enables us to continuously observe multiple data streams of distributed sensor devices in order to identify a system's entire state during runtime. The approach enables to automatically transform behavioral models (i.e., state machines) into time-series queries for matching sensor value streams with pre-defined variable values of the design model in order to report identified states from execution. Additionally, the approach provides a recording mechanism, an abstraction part, and a runtime analysis.
Type of Contribution: We present a metamodel named *MD-RISE* which we prototypically implemented. For this purpose we have a number of prerequisites that must be met: (i) the system's workflow must be expressible by means of a state machine, (ii) the different states of the system must be unambiguous that values describing a state are not identical for two different states, (iii) numeric values must be returned by sensors at runtime and must be storable in a TSDB, and (iv) the time stamps must be accessible. Based on this prerequisites we motivate the approach by an example of a five axes grip-arm robot (gripper). The gripper is an automation system consisting of a controller, sensors, and actuators. At design time, we model the structure and behavior of the gripper by using a subset of SysML, namely the block definition diagram and a state (machine) diagram [125]. Additionally, we consider for each property a specified tolerance range (based on expert knowledge) that defines an acceptable deviation of the assigned property values during runtime. Such deviations may occur due to sensor delays, measurement inaccurancies, etc.

Based on the metamodel, we automatically derive a query on the basis of the state machine, a so-called *state realization event query*. This query helps to identify states based on the recorded sensor value streams in the TSDB. For this we use an M2T-transformation (cf. Section 1.2.2) to automatically transform model elements to query statements in form of text strings. The thereby identified states contain information as follows: the actual time in the granularity of seconds (i.e., timestamp) and the recognized state. In a next step, we generate a state-based log model that consists of the information of all identified states, and additionally, a case ID for identifying the corresponding process instance. Such a *case ID* is required when using PM tools in order to distinguish different executions of the same process. We employ this case ID in our approach to identify single runs of the state machine when executed. In a further step, the state-based log model is transformed to an event-based log model by employing an M2M-transformation. Similar to the approach presented in **Paper 10**, we use ProM Lite in Version 1.1.
Validation of Contribution: The transformation, event queries and tool support were evaluated by means of a case study of a laboratory-sized five-axis gripper arm robot. This setting allowed us to analyze the (i) correctness of the identified states, (ii) com-

pleteness of the identified states, and (iii) performance of the queries. The outcomes show a linear increase as well as good precision and recognition values, but depending on the tolerance range as well as distinctness of the states. The case study design followed again the guidelines defined by Runeson and Höst [113]. The study and its results are published on a project page [144].

Paper 14: From AutomationML to AutomationQL: A By-Example Query Language for CPPS Engineering Models [141]

In this paper, we extend the framework of **Paper 10** by a query component. By following the main principles of the *Query By-Example approach (QBE)*, we present a novel AutomationML query approach for querying AutomationML models at the mining layer (see Figure 1.1 on top) by formulating queries by-example as model fragments. In particular, we propose a dedicated query language for AutomationML called *AutomationQL (AQL)*, which is directly derived from AutomationML. Using this query language, queries can be defined in a QBE manner which allows engineers to formulate queries in terms of AutomationML concepts instead of being burdened when switching to an implementation-oriented query language. Thus, the engineers are as close as possible to modeling languages they usually work with.
Type of Contribution: We present AQL which is a graph pattern-based query language based on concepts of AutomationML. AQL supports positive and negative graph patterns as well as the computing of transitive closures to investigate recursive tree structures and to match for element sets. The query results are explicitly represented in a result model which acts as a proxy to the AutomationML base model elements.
Validation of Contribution: We implemented a prototype of AQL in Eclipse, based on the CAEX workbench, which provides tool support for AutomationML in Eclipse [87]. In particular, we specified AQDL and AQRL as Ecore-based metamodels. Using the standard EMF capabilities, we generated tree-based modeling editors for both languages. For executing AQDL queries on AutomationML models, we implemented a prototypical interpreter in Java. The interpreter red the AutomationML models and AQDL models and produced AQRL models as output. For demonstration purposes we employed the *Pick and Place Unit (PPU)* demonstrator [107] hosted at the Institute of Automation and Information Systems at TUM. We instantiated each language feature by defining a specific query (Q1 - Q8) requiring this feature. We figured out that AQL supports positive and negative graph patterns, computing transitive closures to investigate recursive tree structures and to match for element sets. The query results were explicitly represented in a result model. We have provided an open source implementation of our prototype with further description and examples on our project website [142].

1.6 Summary

This cumulative habilitation thesis follows a temporal model- and data-driven approach in the application field of Systems Engineering. Thereby, automation capabilities are provided for reusing design models from the very beginning, which are then during a system's operation or simulation continuously augmented with runtime data. In addition, we provide dedicated services for a continuous planning, model mining, and issue management. Thereby, we reduce time consuming manual tasks for (i) figuring out appropriate model elements to reuse, (ii) checking consistency, (iii) searching for positive and negative modeling patterns, and/or (iv) providing specific views for particular stakeholders. For this purpose, the explicit consideration of *temporal aspects* not only on the model, but also on the level of model elements is essential. We provide a models "in-the-loop" approach from design to operation and backwards by combining downstream information from the MDE-process with upstream information gathered at runtime. This enables a shift from isolated one-shot system prescriptions to evolutionary models populated by timing aspects, where the focus is not only to represent the current state to steer the system, but on the representation of the system's history.

The presented contributions are all following a design science methodology considering the design, implementation, and evaluation of artifacts (i.e., frameworks, models, and methods) in the context of two application fields, the stationary industry (i.e., manufacturing) and the AEC-industry. In the first of the three contribution clusters, we focus on the handling of heterogeneous sources distributed at various layers in the automation pyramid and beyond (cf. Section 1.4.1). For overcoming connectivity and integration shortcomings (cf. Figure 1.1, bottom down), the presented papers (cf. **Papers 1** to **5**) in this section deal with the combination and revamping of data from heterogeneous sources into new and harmonized representations to provide a nearly seamless vertical (from data sources downwards and upwards along the layers) and horizontal (from one system to another at the same layer) exchange of relevant data. The second cluster, *Temporal Model Management* (cf. Section 1.5.3), considers temporal aspects of models. In **Paper 6** we identified this temporal aspect as a missing link. Accordingly, we developed and implemented a unified temporal model framework as presented in the **Papers 7** to **9**. These contributions address challenges regarding hosting runtime monitoring and handling temporal models (cf. Section 1.4.2). For this purpose we provide query facilities for reasoning about various model element aspects over time (e.g., **Paper 9**). In the third cluster, we provide *Data-Driven Model Analytics* (cf. Section 1.5.4) to overcome the challenge presented in Section 1.4.3. We establish so-called *trace links* between the initial design model and the operation model for (i) monitoring meaningful data from operation, (ii) aligning data logs with the design model for providing a semantic anchoring of data, as well as (iii) providing meaningful analytics to reason about improvements or the fulfillment of given requirements (cf. from **Papers 10** to **14**).

All of these contributions were evaluated by observational, analytical, experimental, and descriptive methods by following the guidelines of Runeson and Höst [114]. As main evaluation environment, we built up a lab-sized automation system environment in form of a five axes grip-arm robot, around which we set up various demonstration cases. In addition, we employ a traffic light system and a self-driving car as demonstrators, which we have developed with support of a project partner of the CDL-MINT laboratory. Furthermore, we use lab-sized test-beds from our research partners such as the lab-sized production system IAF of the Otto-v.-Guericke University Magdeburg (as presented in **Paper 11**) and the CTU's Czech Institute of Informatics (as presented in **Paper 5**).

Table 1.1: Meta-information summary of papers included in this thesis

Paper	FFG InteGra 4.0	FFG DigiTrans 4.0	CDG CDL-MINT	BMBWF TransIT	Model-Driven Connectivity (MDC) / Temporal Model Management (TMM) / Data-Driven Model Analytics (DDMA)	Collaborations
1	X	-	-	-	MDC:: Procedure & Notation	Industry: Eisenstraße Niederösterreich, Zukunftsakademie Mostviertel
2	-	X	X	-	MDC:: Technique & Descriptive Model	-
3	-	-	-	X	MDC:: Analytic Model & Notation	Scientific: MUL, TU Wien Industry: buildingSMART International
4	-	X	-	-	MDC:: Procedure & Prototype	-
5	-	-	X	-	MDC:: Technique & Tool	Scientific: CTU Prague, TU Wien
6	-	-	X	-	TMM:: Descriptive Model	Scientific: Software Competence Center Hagenberg (SCCH)
7	X	-	-	-	TMM:: Procedure	-
8	-	-	X	-	TMM:: Technique & Notation & Analytic Model	-
9	-	-	X	-	TMM:: Technique & Notation & Descriptive Model	Scientific: Universitat Oberta de Catalunya
10	-	X	-	-	DDMA:: Notation & Descriptive Model	Industry: Lieber Lieber Software GmbH
11	-	-	X	-	DDMA:: Analytic Model	Scientific: University Magdeburg
12	-	-	X	-	DDMA:: Analytic Model & Prototype	Industry: Lieber Lieber Software GmbH
13	-	-	X	-	DDMA:: Technique & Analytic Model & Prototype	Scientific: Practical Robotics Institute Austria (PRIA)
14	-	-	X	-	DDMA:: Technique & Notation	-

To give a condensed overview, Table 1.1 summarizes the presented papers accumulated for this thesis and their assignments to research projects, general research fields based on the classification of result types as presented in [122], and scientific as well as industrial collaborations.

1.7 Outlook

While having the above mentioned contributions as an important cornerstone for *Temporal Model*-Driven Systems Engineering, there are several other concerns that are not discussed in this habilitation thesis, but are currently investigated by the applicant as presented by a research roadmap.

First of all from a runtime perspective, several challenges have to be tackled to realize an efficient monitoring process of system components, since (i) not all relevant parameters are directly observable, (ii) parameter values keep changing during observation—a fact that is known as *concept drift*, and (iii) the observation has to be performed while the system operates, to name just a few.

Second, an interoperable tool chain is needed starting from engineering, over simulation, to operation and in the reverse direction by back-propagating runtime data and experiences from operations to engineering. Based on the investigated temporal model framework, groundbreaking foundations as well as applied research are required to further detail and realize them in the direction of *Digital Twin Platforms* and *Model-Driven Digital Twin Engineering*. We present a preliminary reference architecture in this direction in [84, 83]. However, using digital twins also poses new requirements on traditional software engineering practices [112].
Although existing digital twin platforms of big cloud providers (e.g., Azure [101], Eclipse [44], or AWS [4]) provide a lot of benefits to practitioners, the creation and maintenance of digital twins still involves a lot of (human) effort. Information about the system must be entered into different tools that comprise the digital twin, and synchronized with the system to send data to the correct digital twin. Even if this is done, every change in the system requires changes at different positions in its digital twin to ensure consistency. Different research works [77, 118] show that MDE techniques can already be used to automate the construction of a digital twin. However, there is still little knowledge about digital twins of evolving systems.

Third, a broader application context has to be considered already starting in the early phase of requirements elicitation by additionally taking non-functional properties into account. This would be useful, e.g., to better estimate the sensitivity of variables based on external factors such as different workloads, product types, etc. We have already started research work in this direction, as presented in [91, 95] or [94].

Fourth, in the AEC industry, a wide variety of different actors with vastly different background and perspectives are involved during a project. This has also lead to a variety of isolated software solutions as well as a slew of different artifacts and types of information with little to no collaborative access. In addition to different tools, internal company standards, best practices, and guidelines complicate collaboration on infrastructure projects. Accordingly, there is a need for a common temporal model framework for hosting, versioning, and querying those various artifacts. While existing versioning management systems such as Git and CSV provide support for storing, retrieving, and keeping track of different versions of source code and documents, they lack support for organizing different types of artifacts, managing dependencies between artifacts, and providing adequate role and permission models for handling complex workflows in an environment with highly diverse stakeholders. Therefore, a collaborative platform for industry domain experts (with little to no computer science background) is needed to store and easily maintain different variants and versions of a multitude of diverse artifacts (models, paper made notes, 2D construction plans, spreadsheets, 3D models, etc.). Furthermore, providing sophisticated query and retrieval mechanisms across the different artifacts (and their respective versions and variants) is key to the digital transformation in the AEC domain.

1.8 Bibliography

[1] I. 15944-4 2015. Information technology - Business operational view - Part 4: Business transaction scenarios - Accounting and economic ontology. `https://www.iso.org/standard/67199.html`. last access: August 27, 2021.

[2] P. T. C. 6. XML Formats for IEC 61131-3. Technical report, PLCopen, 2009.

[3] M. A. Case Study Artefacts for CASE 2017. `https://cdl-mint.se.jku.at/case-study-artefacts-for-case-2017/`, 2017. last access: July 30, 2021.

[4] Amazon Web Services, Inc. AWS IoT Greengrass. `https://aws.amazon.com/greengrass/`. last access: November 05, 2021.

[5] I. Anagnostopoulos, S. Zeadally, and E. Exposito. Handling big data: research challenges and future directions. *The Journal of Supercomputing*, 72(4):1494–1516, 2016.

[6] J. Artner, A. Mazak, and M. Wimmer. Towards Stochastic Performance Models for Web 2.0 Applications. In J. Cabot, R. D. Virgilio, and R. Torlone, editors, *Proc. of the 17th International Conference on Web Engineering (ICWE 2017)*, volume 10360 of *Lecture Notes in Computer Science*, pages 360–369, 2017.

[7] A. Bader, O. Kopp, and M. Falkenthal. Survey and Comparison of Open Source Time Series Databases. In *Workshopband der 17. Fachtagung des GI-Fachbereichs*

Datenbanken und Informationssysteme (DBIS) Datenbanksysteme für Business, Technologie und Web (BTW 2017), volume P-266 of *LNI*, pages 249–268. GI, 2017.

[8] O. Batarseh and L. F. McGinnis. SysML to Discrete-Event Simulation to Analyze Electronic Assembly Systems. In *Proc. of the 2012 Symposium on Theory of Modeling and Simulation - DEVS Integrative M&S Symposium*, TMS/DEVS '12, pages 48:1–48:8. Society for Computer Simulation International, 2012.

[9] N. Bencomo, S. Götz, and H. Song. Models@run.time: a guided tour of the state of the art and research challenges. *Software and Systems Modeling*, 18(5):3049–3082, 2019.

[10] D. F. Bender, B. Combemale, X. Crégut, J. Farines, B. Berthomieu, and F. Vernadat. Ladder Metamodeling and PLC Program Validation through Time Petri Nets. In *Proc. of the 4th European Conference on Model Driven Architecture - Foundations and Applications, ECMDA-FA 2008*, volume 5095 of *Lecture Notes in Computer Science*, pages 121–136. Springer, 2008.

[11] A. Benelallam, T. Hartmann, L. Mouline, F. Fouquet, J. Bourcier, O. Barais, and Y. L. Traon. Raising Time Awareness in Model-Driven Engineering: Vision Paper. In *Proc. of the 20th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS 2017*, pages 181–188. IEEE Computer Society, 2017.

[12] L. Berardinelli, S. Biffl, A. Lüder, E. Mätzler, T. Mayerhofer, M. Wimmer, and S. Wolny. Cross-disciplinary engineering with AutomationML and SysML. *at - Automatisierungstechnik*, 64(4):253–269, 2016.

[13] L. Berardinelli, A. Mazak, O. Alt, M. Wimmer, and G. Kappel. Model-Driven Systems Engineering: Principles and Application in the CPPS Domain. In S. Biffl, A. Lüder, and D. Gerhard, editors, *Multi-Disciplinary Engineering for Cyber-Physical Production Systems, Data Models and Software Solutions for Handling Complex Engineering Projects*, pages 261–299. Springer, 2017.

[14] J. Bezivin. On the unification power of models. *Software & Systems Modeling*, 4(2):171–188, 2005.

[15] J. Bézivin, R. F. Paige, U. Aßmann, B. Rumpe, and D. C. Schmidt. Manifesto - Model Engineering for Complex Systems. *CoRR*, abs/1409.6591, 2014.

[16] R. Bill, A. Mazak, M. Wimmer, and B. Vogel-Heuser. On the Need for Temporal Model Repositories. In *Software Technologies: Applications and Foundations - STAF 2017 Collocated Workshops, Revised Selected Papers*, volume 10748 of *Lecture Notes in Computer Science*, pages 136–145. Springer, 2017.

[17] C. M. Bishop. *Pattern recognition and machine learning, 5th Edition*. Information science and statistics. Springer, 2007.

[18] BITKOM, VDMA, and ZVEI. Umsetzungsstrategie Industrie 4.0. Ergebnisbericht der plattform industrie 4.0, Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (BITKOM), Verband Deutscher Maschinen- und Anlagenbau e. V. (VDMA), Zentralverband Elektrotechnik- und Elektronikindustrie e. V. (ZVEI), 2015.

[19] G. Blair, N. Bencomo, and R. France. Models@ run.time. *Computer*, 42(10):22–27, 2009.

[20] M. H. Böhlen, A. Dignös, J. Gamper, and C. S. Jensen. Database Technology for Processing Temporal Data. In *Proc. of the 25th International Symposium on Temporal Representation and Reasoning, TIME 2018*, volume 120 of *LIPIcs*, pages 2:1–2:7. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

[21] A. Borrmann, M. König, C. Koch, and J. Beetz. *Building Information Modeling - Technologische Grundlagen und industrielle Praxis*. Springer, 2015.

[22] E. Bousse, T. Mayerhofer, B. Combemale, and B. Baudry. Advanced and efficient execution trace management for executable domain-specific modeling languages. *Software and Systems Modeling*, 18(1):385–421, 2019.

[23] M. Brambilla, J. Cabot, and M. Wimmer. *Model-Driven Software Engineering in Practice: Second Edition*. Morgan & Claypool, 2017.

[24] M. Broy and A. Schmidt. Challenges in Engineering Cyber-Physical Systems. *Computer*, 47(2):70–72, 2014.

[25] H. Brunelière, E. Burger, J. Cabot, and M. Wimmer. A feature-based survey of model view approaches. *Software and Systems Modeling*, 18(3):1931–1952, 2019.

[26] F. Budinsky, D. Steinberg, E. Merks, R. Ellersick, and T. J. Grose. *Eclipse Modeling Framework*. Addison-Wesley, 2004.

[27] J. Cabot and M. Gogolla. Object Constraint Language (OCL): A Definitive Guide. In *Formal Methods for Model-Driven Engineering - 12th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM 2012*, volume 7320 of *Lecture Notes in Computer Science*, pages 58–90. Springer, 2012.

[28] J. Cabot, A. Olivé, and E. Teniente. Representing Temporal Information in UML. In *Proc. of the 6th International Conference on Modeling Languages and Applications, «UML» 2003 - The Unified Modeling Language*, volume 2863 of *Lecture Notes in Computer Science*, pages 44–59. Springer, 2003.

[29] W. Cellary, G. Vossen, and G. Jomier. Multiversion object constellations: A new approach to support a designer's database work. *Engineering with Computers*, 10(4):230–244, 1994.

[30] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):15:1–15:58, 2009.

[31] I. E. Commission. OPC Unified Architecture - Part 1: Overview and Concepts. `https://webstore.iec.ch/publication/7172`. last access: August 03, 2021.

[32] I. E. Commission. IEC 62424: Representation of process control engineering - Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools. `https://global.ihs.com/doc_detail.cfm?item_s_key=00510851`, 2009. last access: October 29, 2021.

[33] I. E. Commission. IEC 61131-3: Programmable controllers - Part 3: Programming languages. `https://global.ihs.com/doc_detail.cfm?item_s_key=00145671`, 2013. last access: October 29, 2021.

[34] A. consortium. AutomationML. `https://www.automationml.org/`. last access: October 19, 2021.

[35] A. consortium. Whitepaper Part 4 - AutomationML Logic. `https://www.automationml.org/about-automationml/specifications/`, 2010. last access: October 31, 2021.

[36] A. consortium. Whitepaper Part 3 - Geometry and Kinematics. `https://www.automationml.org/about-automationml/specifications/`, 2015. last access: October 31, 2021.

[37] J. G. Cruz, A. Sadovykh, D. Truscan, H. Brunelière, P. Pierini, and L. L. Muniz. MegaM@Rt2 EU Project: Open Source Tools for Mega-Modelling at Runtime of CPSs. In *Proc. of the 16th IFIP WG 2.13 International Conference on Open Source Systems, OSS 2020*, volume 582 of *IFIP Advances in Information and Communication Technology*, pages 183–189. Springer, 2020.

[38] J. S. Cuadrado and J. de Lara. Streaming Model Transformations: Scenarios, Challenges and Initial Solutions. In *Proc. of the 6th International Conference on Theory and Practice of Model Transformations (ICMT)*, ICMT, pages 1–16. Springer, 2013.

[39] I. Dávid, I. Ráth, and D. Varró. Foundations for Streaming Model Transformations by Complex Event Processing. *Software & Systems Modeling*, 17(1):135–162, 2018.

[40] J. de Lara, E. Guerra, and J. S. Cuadrado. Model-driven engineering with domain-specific meta-modelling languages. *Software and System Modeling*, 14(1):429–459, 2015.

[41] P. M. Domingos and G. Hulten. Catching up with the Data: Research Issues in Mining Data Streams. In *Proc. of the 6th International Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD)*. cs.cornell.edu, 2001.

[42] R. Drath. Industrie 4.0–eine Einführung. *Open Automation*, 3:1–6, 2014.

[43] M. Eck, N. Sidorova, and W. Aalst. Discovering and Exploring State-Based Models for Multi-perspective Processes. In *Business Process Management (BPM)*, pages 142–157. Springer International Publishing, 2016.

[44] Eclipse. ditto. `https://www.eclipse.org/ditto/`. last access: November 05, 2021.

[45] Eclipse Foundation. Eclipse Modeling Framework (EMF). `https://www.eclipse.org/modeling/emf/`. last access: July 29, 2021.

[46] I. Eclipse Foundation. The Epsilon Object Language (EOL). `https://www.eclipse.org/epsilon/doc/eol/`. last access: August 28, 2021.

[47] I. Eclipse Foundation. The Epsilon Transformation Language (ETL). `https://www.eclipse.org/epsilon/doc/etl/`. last access: August 28, 2021.

[48] I. Eclipse Foundation. The Epsilon Validation Language (EVL). `https://www.eclipse.org/epsilon/doc/evl/`. last access: August 28, 2021.

[49] A. Fay, B. Vogel-Heuser, T. Frank, K. Eckert, T. Hadlich, and C. Diedrich. Enhancing a model-based engineering approach for distributed manufacturing automation systems with characteristics and design patterns. *Journal of Systems and Software*, 101:221–235, 2015.

[50] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From Data Mining to Knowledge Discovery: An Overview. In *Advances in Knowledge Discovery and Data Mining*, pages 1–34. American Association for Artificial Intelligence (AAAI), 1996.

[51] O. Foundation. MTConnect - Semantic data and contextual models for factory floor devices. `https://opcfoundation.org/markets-collaboration/mtconnect/`. last access: August 26, 2021.

[52] O. Foundation. Unified Architecture - OPC Foundation. `https://opcfoundation.org`. last access: August 19, 2021.

[53] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Pearson International, 1995.

[54] A. García-Domínguez, N. Bencomo, and L. H. G. Paucar. Reflecting on the past and the present with temporal graph-based models. In *Proc. of MODELS 2018 Workshops: ModComp, MRT, OCL, FlexMDE, EXE, COMMitMDE, MDETools, GEMOC, MORSE, MDE4IoT, MDEbug, MoDeVVa, ME, MULTI, HuFaMo, AM-MoRe, PAINS co-located with ACM/IEEE 21st International Conference on Model Driven Engineering Languages and Systems (MODELS 2018)*, volume 2245 of *CEUR Workshop Proceedings*, pages 46–55. CEUR-WS.org, 2018.

[55] A. García-Domínguez, N. Bencomo, J. M. P. Ullauri, and L. H. G. Paucar. Querying and Annotating Model Histories with Time-Aware Patterns. In *Proc. of the 22nd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, MODELS 2019*, pages 194–204. IEEE, 2019.

[56] J.-M. Gauthier, F. Bouquet, A. Hammad, and F. Peureux. A SysML Formal Framework to Combine Discrete and Continuous Simulation for Testing. In *Proc. of the 17th International Conference on Formal Engineering Methods, ICFEM 2015, Paris, France, November 3-5*, volume 9407 of *Lecture Notes in Computer Science*, pages 134–152. Springer, 2015.

[57] M. Ghallab, A. Howe, C. Knoblock, D. McDermott, A. Ram, M. Veloso, D. Weld, and D. Wilkins. PDDL - The Planning Domain Definition Language Version 1.2. Technical report, AIPS-98 Planning Competition Committee, 1998.

[58] M. Gogolla. Tales of ER and RE Syntax and Semantics. In *Transformation Techniques in Software Engineering*, volume 05161 of *Dagstuhl Seminar Proceedings*. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2005.

[59] M. Gogolla, N. Desai, and K. Doan. Developing User and Recording Interfaces for Design Time and Runtime Models. In *STAF 2019 Co-Located Events Joint Proceedings: 1st Junior Researcher Community Event, 2nd International Workshop on Model-Driven Engineering for Design-Runtime Interaction in Complex Systems, and 1st Research Project Showcase Workshop co-located with Software Technologies: Applications and Foundations (STAF 2019)*, volume 2405 of *CEUR Workshop Proceedings*, pages 39–48. CEUR-WS.org, 2019.

[60] M. Golfarelli and S. Rizzi. Temporal Data Warehousing: Approaches and Techniques. In *Integrations of Data Warehousing, Data Mining and Database Technologies: Innovative Approaches*, pages 1–18. IGI Global, 2011.

[61] A. Gómez, J. Cabot, and M. Wimmer. TemporalEMF: A Temporal Metamodeling Framework. In *Proc. of the 37th International Conference on Conceptual Modeling, ER 2018*, volume 11157 of *Lecture Notes in Computer Science*, pages 365–381. Springer, 2018.

[62] H. Gregersen and C. S. Jensen. Temporal Entity-Relationship Models - A Survey. *IEEE Trans. Knowl. Data Eng.*, 11(3):464–497, 1999.

[63] T. Hartmann, A. Moawad, F. Fouquet, G. Nain, J. Klein, and Y. L. Traon. Stream my Models: Reactive Peer-to-Peer Distributed Models@run.time. In *Proc. of the 18th International Conference on Model Driven Engineering Languages and Systems (MoDELS)*. ACM/IEEE, 2015.

[64] I. Hegny, M. Wenger, and A. Zoitl. IEC 61499 based simulation framework for model-driven production systems development. In *Proc. of the IEEE Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8. IEEE, 2010.

[65] R. Heldal, P. Pelliccione, U. Eliasson, J. Lantz, J. Derehag, and J. Whittle. Descriptive vs prescriptive models in industry. In *Proc. of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems (MODELS)*, pages 216–226. ACM, 2016.

[66] C. T. U. in Prague. Czech Institute of Informatics, Robotics and Cybernetics. `https://www.cvut.cz/en/czech-institute-of-informatics-robotics-and-cybernetics`. last access: August 19, 2021.

[67] Incose. *INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*. Wiley, 4th edition, 2015.

[68] M. S. Institute of Ergonomics and A. at Otto-v. Guericke University Magdeburg. Equipment Center for Distributed Systems. http://www.iaf-bg.ovgu.de/en/technische_ausstattung_cvs.html, 2016. last access: August 08, 2021.

[69] International Society of Automation (ISA). ISA95, Enterprise-Control System Integration. `https://www.isa.org/standards-and-publications/isa-standards/isa-standards-committees/isa95`. last access: August 25, 2021.

[70] T. Johnson, A. Kerzhner, C. Paredis, and R. Burkhart. Integrating models and simulations of continuous dynamics into SysML. *Journal of Computing and Information Science in Engineering*, 12(1):135–145, 2012.

[71] H. Kagermann, W. Wahlster, and J. Helbig. Recommendations for Implementing the Strategic Initiative INDUSTRIE 4.0 – Securing the Future of German Manufacturing Industry. Final report of the industrie 4.0 working group, Forschungsunion im Stifterverband für die Deutsche Wirtschaft e. V., 2013.

[72] D. Karagiannis, H. C. Mayr, and J. Mylopoulos, editors. *Domain-Specific Conceptual Modeling, Concepts, Methods and Tools*. Springer, 2016.

[73] A. Kästner, M. Gogolla, K. Doan, and N. Desai. Sketching a Model-Based Technique for Integrated Design and Run Time Description - Short Paper - Tool Demonstration. In *Software Technologies: Applications and Foundations - STAF 2018 Collocated Workshops, Revised Selected Papers*, volume 11176 of *Lecture Notes in Computer Science*, pages 529–535. Springer, 2018.

[74] S. Khalifa, Y. Elshater, K. Sundaravarathan, A. B. Bhat, P. Martin, F. Imam, D. Rope, M. McRoberts, and C. Statchuk. The Six Pillars for Building Big Data Analytics Ecosystems. *ACM Comput. Surv.*, 49(2):33:1–33:36, 2016.

[75] N. Khan, M. Alsaqer, H. Shah, G. Badshah, A. A. Abbasi, and S. Salehian. The 10 Vs, Issues and Challenges of Big Data. In *Proc. of the 2018 International Conference on Big Data and Education, ICBDE 2018, Honolulu, HI, USA, March 09-11, 2018*, pages 52–56. ACM, 2018.

[76] S. Khare, K. An, A. S. Gokhale, S. Tambe, and A. Meena. Reactive Stream Processing for Data-centric Publish/Subscribe. In *Proc. of the 9th ACM International Conference on Distributed Event-Based Systems (DEBS)*, pages 234–245. ACM DL, 2015.

[77] J. C. Kirchhof, J. Michael, B. Rumpe, S. Varga, and A. Wortmann. Model-driven digital twin construction: synthesizing the integration of cyber-physical systems with their information systems. In *MoDELS '20: ACM/IEEE 23rd International Conference on Model Driven Engineering Languages and Systems*, pages 90–101. ACM, 2020.

[78] T. Kosar, S. Bohra, and M. Mernik. Domain-Specific Languages: A Systematic Mapping Study. *Information and Software Technology*, 71:77–91, 2016.

[79] D. Laney. 3D Data Management: Controlling Data Volume, Velocity, and Variety. Technical report, META Group, 2001.

[80] D. Leal. ISO 15926 "Life Cycle Data for Process Plant": an Overview. *Oil & Gas Science and Technology-revue De L Institut Francais Du Petrole - OIL GAS SCI TECHNOL*, 60:629–637, 07 2005.

[81] E. A. Lee. Cyber Physical Systems: Design Challenges. In *11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing (ISORC)*, pages 363–369. IEEE, 2008.

[82] C. Legat, D. Schütz, and B. Vogel-Heuser. Automatic generation of field control strategies for supporting (re-)engineering of manufacturing systems. *Journal of Intelligent Manufacturing*, 25(5):1101–1111, 2014.

[83] D. Lehner, S. Sint, M. Vierhauser, W. Narzt, and M. Wimmer. AML4DT: A Model-Driven Framework for Developing and Maintaining Digital Twins with AutomationML. In *Proc. of the 26th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2021*. IEEE, 2021.

[84] D. Lehner, S. Wolny, A. Mazak-Huemer, and M. Wimmer. Towards a Reference Architecture for Leveraging Model Repositories for Digital Twins. In *Proc. of the 25th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2020, Vienna, Austria, September 8-11, 2020*, pages 1077–1080. IEEE, 2020.

[85] P. K. R. Maddikunta, Q.-V. Pham, P. B, N. Deepa, K. Dev, T. R. Gadekallu, R. Ruby, and M. Liyanage. Industry 5.0: A survey on enabling technologies and potential applications. *Journal of Industrial Information Integration*, page 100257, 2021.

[86] W. Mahnke, Leitner, Stefan-Helmut, and M. Damm. *OPC Unified Architecture*. Springer, 2009.

[87] T. Mayerhofer, M. Wimmer, L. Berardinelli, and R. Drath. A Model-Driven Engineering Workbench for CAEX Supporting Language Customization and Evolution. *IEEE Transactions on Industrial Informatics*, 14(6):2770–2779, 2018.

[88] H. C. Mayr, J. Michael, S. Ranasinghe, V. A. Shekhovtsov, and C. Steinberger. *Model Centered Architecture*, pages 85–104. Springer, 2017.

[89] D. Mayrhofer and C. Huemer. Extending the REA-DSL by the Planning Layer of the REA Ontology. In M. Bajec and J. Eder, editors, *Proc. of the International Workshops on Advanced Information Systems Engineering - CAiSE 2012 , Gdańsk, Poland, June 25-26, 2012*, volume 112 of *Lecture Notes in Business Information Processing*, pages 543–554. Springer, 2012.

[90] A. Mazak and C. Huemer. From business functions to control functions: Transforming REA to ISA-95. In *Proc. of the 17th IEEE International Conference on Business Informatics, CBI 2015, Lisbon, Portugal, July 13-16, 2015 - Volume 1*, pages 33–42. IEEE Computer Society, 2015.

[91] A. Mazak, A. Lüder, S. Wolny, M. Wimmer, D. Winkler, K. Kirchheim, R. Rosendahl, H. Bayanifar, and S. Biffl. Model-based generation of run-time data collection systems exploiting AutomationML. *at - Automatisierungstechnik*, 66(10):819–833, 2018.

[92] A. Mazak and M. Wimmer. On Marrying Model-driven Engineering and Process Mining: A Case Study in Execution-based Model Profiling. In P. Ceravolo, C. Guetl, and S. Rinderle-Ma, editors, *Pro. of the 6th International Symposium on*

Data-driven Process Discovery and Analysis (SIMPDA 2016), Graz, Austria, December 15-16, 2016, volume 1757 of *CEUR Workshop Proceedings*, pages 78–88. CEUR-WS.org, 2016.

[93] A. Mazak and M. Wimmer. Towards Liquid Models: An Evolutionary Modeling Approach. In E. Kornyshova, G. Poels, C. Huemer, I. Wattiau, F. Matthes, and J. L. C. Sanz, editors, *Proc. of the 18th IEEE International Conference on Business Informatics, CBI 2016, 29th August - 1st September 2016, Paris, France, Volume 1 - Conference Papers*, pages 104–112. IEEE Computer Society, 2016.

[94] A. Mazak, M. Wimmer, and P. Patsuk-Boesch. Reverse engineering of production processes based on Markov chains. In *Proc. of the 13th IEEE Conference on Automation Science and Engineering, CASE 2017, Xi'an, China, August 20-23, 2017*, pages 680–686. IEEE, 2017.

[95] A. Mazak, M. Wimmer, and P. Patsuk-Bösch. Execution-Based Model Profiling. In P. Ceravolo, C. Guetl, and S. Rinderle-Ma, editors, *Data-Driven Process Discovery and Analysis - 6th IFIP WG 2.6 International Symposium, SIMPDA 2016, Graz, Austria, December 15-16, 2016, Revised Selected Papers*, volume 307 of *Lecture Notes in Business Information Processing*, pages 37–52. Springer, 2016.

[96] A. Mazak, S. Wolny, A. Gómez, J. Cabot, M. Wimmer, and G. Kappel. Temporal Models on Time Series Databases. *J. Object Technol.*, 19(3):3:1–3:15, 2020.

[97] A. Mazak, S. Wolny, and R. Konlechner. Case Study Artefacts for EMISA 2017. `https://cdl-mint.se.jku.at/case-study-artefacts-for-emisa-2017/`, 2017. last access: October 30, 2021.

[98] A. Mazak, S. Wolny, and M. Wimmer. On the Need for Data-Based Model-Driven Engineering. In S. Biffl, M. Eckhart, A. Lüder, and E. R. Weippl, editors, *Security and Quality in Cyber-Physical Systems Engineering, With Forewords by Robert M. Lee and Tom Gilb*, pages 103–127. Springer, 2019.

[99] A. Mazak-Huemer, R. Galler, R. Wenighofer, M. Vierhauser, and C. Huemer. BIM-basierte digitale Transformation im Untertagebau anhand von zwei anwendungsorientierten Forschungsprojekten. *BHM Berg- und Hüttenmännische Monatshefte*, 165:658–665, 2020.

[100] B. Meyers, R. Deshayes, L. Lucio, E. Syriani, H. Vangheluwe, and M. Wimmer. ProMoBox: A Framework for Generating Domain-Specific Property Languages. In *Proc. of the 7th International Conference on Software Language Engineering, SLE 2014*, volume 8706 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2014.

[101] Microsoft Corporation. Azure Digital Twins. `https://azure.microsoft.com/services/digital-twins/`. last access: October 30, 2021.

[102] B. Morin, O. Barais, J. M. Jezequel, F. Fleurey, and A. Solberg. Models@ Run.time to Support Dynamic Adaptation. *Computer*, 42(10):44–51, 2009.

[103] L. Mouline, A. Benelallam, F. Fouquet, J. Bourcier, and O. Barais. A Temporal Model for Interactive Diagnosis of Adaptive Systems. In *Proc. of the IEEE International Conference on Autonomic Computing, ICAC 2018*, pages 175–180. IEEE Computer Society, 2018.

[104] P. Novák, E. Serral, R. Mordinyi, and R. Sindelár. Integrating heterogeneous engineering knowledge and tools for efficient industrial simulation model support. *Adv. Eng. Informatics*, 29(3):575–590, 2015.

[105] Object Management Group. Standards Development Organization. `https://www.omg.org/`. last access: August 20, 2021.

[106] Object Management Group (OMG). Unified Modeling Language Version 2.5.1. `https://www.omg.org/spec/UML/About-UML/`, 2017. last access: July 05, 2021.

[107] I. of Automation and I. S. at TUM. Pick and Place Unit (PPU) – Demonstrator for Evolution in Industrial Plant Automation. `https://www.mec.ed.tum.de/ais/forschung/demonstratoren/ppu/`. last access: November 16, 2021.

[108] OMG. Object Constraint Language (OCL). Version 2.4. `https://www.omg.org/spec/OCL/2.4/About-OCL/`, 2014. last access: July 13, 2021.

[109] OMG. Meta Object Facility (MOF). Version 2.5.1. `https://www.omg.org/mof/`, 2016. last access: July 12, 2021.

[110] G. Paskaleva, A. Mazak-Huemer, M. Wimmer, and T. Bednar. Leveraging Integration Facades for Model-based Tool Interoperability. *Automation in Construction*, 128:103689, 2021.

[111] T. B. Pedersen. Managing Big Multidimensional Data - A Journey From Acquisition to Prescriptive Analytics. In J. Bernardino, C. Quix, and J. Filipe, editors, *Proc. of the 6th International Conference on Data Science, Technology and Applications, DATA 2017, Madrid, Spain, July 24-26, 2017.*, page 5. SciTePress, 2017.

[112] L. F. Rivera, H. A. Müller, N. M. Villegas, G. Tamura, and M. Jiménez. On the Engineering of IoT-Intensive Digital Twin Software Systems. In *ICSE '20: 42nd International Conference on Software Engineering, Workshops*, pages 631–638. ACM, 2020.

[113] P. Runeson and M. Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131–164, 2009.

[114] P. Runeson and M. Höst. Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering. *Empirical Software Engineering*, 14:131–164, 2009.

[115] J. Rykowski and W. Cellary. Using Multiversion Object-Oriented Databases in CAD/CIM Systems. In *Proc. of the 7th International Conference on Database and Expert Systems Applications (DEXA)*, pages 1–10, 1996.

[116] D. Schmidt, A. K. Dittrich, W. Dreyer, and R. W. Marti. Time Series, A Neglected Issue in Temporal Database Research? In *Proc. of the International Workshop on Temporal Databases, Recent Advances in Temporal Databases*, Workshops in Computing, pages 214–232. Springer, 1995.

[117] D. C. Schmidt. Guest Editor's Introduction: Model-Driven Engineering. *IEEE Computer Society Press*, 39(2):25–31, 2006.

[118] G. N. Schroeder, C. Steinmetz, R. N. Rodrigues, R. V. B. Henriques, A. Rettberg, and C. E. Pereira. A Methodology for Digital Twin Modeling and Deployment for Industry 4.0. *Proceedings of the IEEE*, 109(4):556–567, 2020.

[119] D. Schütz, C. Legat, and B. Vogel-Heuser. MDE of manufacturing automation software - Integrating SysML and standard development tools. In *Proc. of the 12th IEEE International Conference on Industrial Informatics (INDIN)*, pages 267–273. IEEE, 2014.

[120] Á. M. Segura, J. de Lara, P. Neubauer, and M. Wimmer. Automated modelling assistance by integrating heterogeneous information sources. *Comput. Lang. Syst. Struct.*, 53:90–120, 2018.

[121] M. Seidl, M. Scholz, C. Huemer, and G. Kappel. *UMLClassroom: An Introduction to Object-Oriented Modeling*. Springer Publishing Company, Incorporated, 2015.

[122] M. Shaw. Writing Good Software Engineering Research Papers. In *Proceedings of the 25th International Conference on Software Engineering (ICSE)*, pages 726–736. IEEE Computer Society, 2003.

[123] G. Shmueli and O. R. Koppius. Predictive Analytics in Information Systems Research. *MIS Quarterly*, 35(3):553–572, 2011.

[124] A. Standards. Building Information Modeling (BIM). `https://www.austrian-standards.at/en/topics/construction-real-estate/building-information-modelling-bim`. last access: August 26, 2021.

[125] SysML.org. Systems Modeling Language (SysML). `https://sysml.org/`. last access: August 25, 2021.

[126] W. van der Aalst, P. van den Brand, M. de Leoni, B. van Dongen, C. Günther, B. Hompes, M. Leemans, S. Leemans, X. Lu, F. Mannhardt, E. Verbeek, and M. Westergaard. The Process Mining Toolkit - ProM Lite 1.1. `https://www.promtools.org/doku.php?id=promlite11`, 2016. last access: August 21, 2021.

[127] W. M. P. van der Aalst, A. Adriansyah, A. K. A. de Medeiros, F. Arcieri, T. Baier, T. Blickle, R. P. J. C. Bose, P. van den Brand, R. Brandtjen, J. C. A. M. Buijs, A. Burattin, J. Carmona, M. Castellanos, J. Claes, J. Cook, N. Costantini, F. Curbera, E. Damiani, M. de Leoni, P. Delias, B. F. van Dongen, M. Dumas, S. Dustdar, D. Fahland, D. R. Ferreira, W. Gaaloul, F. van Geffen, S. Goel, C. W. Günther, A. Guzzo, P. Harmon, A. H. M. ter Hofstede, J. Hoogland, J. E. Ingvaldsen, K. Kato, R. Kuhn, A. Kumar, M. L. Rosa, F. M. Maggi, D. Malerba, R. S. Mans, A. Manuel, M. McCreesh, P. Mello, J. Mendling, M. Montali, H. R. M. Nezhad, M. zur Muehlen, J. Munoz-Gama, L. Pontieri, J. Ribeiro, A. Rozinat, H. S. Pérez, R. S. Pérez, M. Sepúlveda, J. Sinur, P. Soffer, M. Song, A. Sperduti, G. Stilo, C. Stoel, K. D. Swenson, M. Talamo, W. Tan, C. Turner, J. Vanthienen, G. Varvaressos, E. Verbeek, M. Verdonk, R. Vigo, J. Wang, B. Weber, M. Weidlich, T. Weijters, L. Wen, M. Westergaard, and M. T. Wynn. Process Mining Manifesto. In *Proc. of the Business Process Management Workshops (BPM)*, pages 169–194. Springer, 2011.

[128] W. M. P. van der Aalst, T. Weijters, and L. Maruster. Workflow Mining: Discovering Process Models from Event Logs. *Trans. Knowl. Data Eng.*, 16(9):1128–1142, 2004.

[129] H. Vangheluwe, V. Amaral, H. Giese, J. F. Broenink, B. Schätz, A. Norta, P. Carreira, I. Lukovic, T. Mayerhofer, M. Wimmer, and A. Vallecillo. MPM4CPS: Multi-Paradigm Modelling for Cyber-Physical Systems. In C. Dubois, F. Parisi-Presicce, D. S. Kolovos, and N. Matragkas, editors, *Joint Proc. of the Doctoral Symposium and Projects Showcase Held as Part of STAF 2016 co-located with Software Technologies: Applications and Foundations (STAF 2016), Vienna, Austria, July 4-7, 2016*, volume 1675 of *CEUR Workshop Proceedings*, pages 40–47. CEUR-WS.org, 2016.

[130] W. N. Venables and B. D. Ripley. *Modern applied statistics with S*. Statistics and Computing. Springer, 4 edition, 2002.

[131] B. Vogel-Heuser and S. Biffl. Cross-discipline modeling and its contribution to automation. *at - Automatisierungstechnik*, 64(3):165–167, 2016.

[132] B. Vogel-Heuser, A. Fay, I. Schaefer, and M. Tichy. Evolution of software in automated production systems: Challenges and research directions. *Journal of Systems and Software*, 110:54–84, 2015.

[133] B. Vogel-Heuser, J. Folmer, and C. Legat. Anforderungen an die Softwareevolution in der Automatisierung des Maschinen- und Anlagenbaus. *at - Automatisierungstechnik*, 62(3):163–174, 2014.

[134] B. Vogel-Heuser and D. Hess. Guest Editorial Industry 4.0-Prerequisites and Visions. *IEEE Trans. Automation Science and Engineering*, 13(2):411–413, 2016.

[135] B. Vogel-Heuser and S. Kowalewski. Cyber-physische Systeme. *at - Automatisierungstechnik*, 61(10):667–668, 2013.

[136] V. Vyatkin. Software Engineering in Industrial Automation: State-of-the-Art. *IEEE Transactions on Industrial Informatics*, 9(3):1234–1249, 2013.

[137] B. Wally, C. Huemer, and A. Mazak. A View on Model-Driven Vertical Integration: Alignment of Production Facility Models and Business Models. In *Proc. of the 13th IEEE Conference on Automation Science and Engineering, CASE 2017, Xi'an, China, August 20-23, 2017*, pages 1012–1018. IEEE, 2017.

[138] B. Wally, C. Huemer, and A. Mazak. Entwining plant engineering data and ERP information: Vertical integration with AutomationML. In *Proc. of the 3rd International Conference on Control, Automation and Robotics (ICCAR)*, pages 356–364. IEEE, 2017.

[139] B. Wally, C. Huemer, A. Mazak, and M. Wimmer. AutomationML, ISA-95 and Others: Rendezvous in the OPC UA Universe. In *Proc. of the 14th IEEE International Conference on Automation Science and Engineering, CASE 2018, Munich, Germany, August 20-24, 2018*, pages 1381–1387. IEEE, 2018.

[140] B. Wally, J. Vyskocil, P. Novák, C. Huemer, R. Sindelár, P. Kadera, A. Mazak, and M. Wimmer. Flexible Production Systems: Automated Generation of Operations Plans Based on ISA-95 and PDDL. *Robotics Autom. Lett. (RA-L)*, 4:4062–4069, 2019.

[141] M. Wimmer and A. Mazak. From AutomationML to AutomationQL: A By-Example Query Language for CPPS Engineering Models. In *Proc. of the 14th IEEE International Conference on Automation Science and Engineering, CASE 2018, Munich, Germany, August 20-24, 2018*, pages 1394–1399. IEEE, 2018.

[142] M. Wimmer and A. Mazak. Prototype Artefacts for CASE 2018. `https://cdl-mint.se.jku.at/%20prototype-artefacts-for-case-2018/`, 2018. last access: November 13, 2021.

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

[144] S. Wolny and A. Mazak. Case Study Artefacts for EMISA 2019. `https://cdl-mint.se.jku.at/case-study-artefacts-for-emisa-2019/`, 2019. last access: July 30, 2021.

[145] S. Wolny and A. Mazak. Case Study Artefacts for ETFA 2019. `https://cdl-mint.se.jku.at/case-study-artefacts-for-etfa-2019/`, 2019. last access: November 11, 2021.

[146] S. Wolny, A. Mazak, C. Carpella, V. Geist, and M. Wimmer. Thirteen Years of SysML: A Systematic Mapping Study. *Softw. Syst. Model.*, 19(1):111–169, 2020.

[147] S. Wolny, A. Mazak, R. Konlechner, and M. Wimmer. Towards Continuous Behavior Mining. In P. Ceravolo, M. van Keulen, and K. Stoffel, editors, *Proc. of the 7th International Symposium on Data-driven Process Discovery and Analysis (SIMPDA 2017), Neuchâtel, Switzerland, December 6-8, 2017*, volume 2016 of *CEUR Workshop Proceedings*, pages 149–150. CEUR-WS.org, 2017.

[148] S. Wolny, A. Mazak, and M. Wimmer. Automatic Reverse Engineering of Interaction Models from System Logs. In *Proc. of the 24th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2019, Zaragoza, Spain, September 10-13, 2019*, pages 57–64. IEEE, 2019.

[149] S. Wolny, A. Mazak, M. Wimmer, and C. Huemer. Model-driven Runtime State Identification. *EMISA Forum*, 39(1):29–44, 2019.

[150] S. Wolny, A. Mazak, M. Wimmer, R. Konlechner, and G. Kappel. Model-Driven Time-Series Analytics. *Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model.*, 13(Special):252–261, 2018.

[151] P. Ziemann and M. Gogolla. OCL Extended with Temporal Logic. In *Proc. of the 5th International Andrei Ershov Memorial Conference on Perspectives of Systems Informatics, PSI 2003, Revised Papers*, volume 2890 of *Lecture Notes in Computer Science*, pages 351–357. Springer, 2003.

2 From business functions to control functions: Transforming REA to ISA-95

A. Mazak and C. Huemer;
Proceedings of the 17th IEEE International Conference on Business Informatics (CBI),
IEEE Computer Society, (2015), pp. 33–42.
DOI: 10.1109/CBI.2015.50

2015 IEEE 17th Conference on Business Informatics

From business functions to control functions: Transforming REA to ISA-95

Alexandra Mazak
TU Vienna
Austria, Vienna
Email: mazak@big.tuwien.ac.at

Christian Huemer
TU Vienna
Austria, Vienna
Email: huemer@big.tuwien.ac.at

***Abstract*—In the context of smart factories, a seamless information exchange between information systems on the same layer (horizontal integration) and between information systems on different layers (vertical integration) is a key issue. For this purpose we aim for an integrated modeling framework spanning over production chains and value networks. In building this framework, we first concentrate on the layers realizing the business functions and the manufacturing control functions. Thereby, we build up on the Resource Event Agent (REA) business ontology (ISO/IEC 15944-4) to describe external activities requiring horizontal integration with business partners and internal activities serving as a hook for vertical integration within a manufacturing enterprise. Furthermore, we base our framework on the ISA-95 industry standard (ANSI/ISA-95; IEC 62264) to describe the vertical integration within an enterprise. In this paper, we demonstrate how information given in REA models is transformed to corresponding ISA-95 skeletons. In other words, we show how a model describing the main business functions of an enterprise is used to derive essential concepts relevant to the manufacturing execution system.**

I. INTRODUCTION

The German working committee for Industrie 4.0[1] has identified among others the following research issues [1]:

- horizontal integration through value networks
- vertical integration of networked manufacturing systems
- end-to-end digital integration of engineering across the entire value chain

Industrie 4.0 use case scenarios relating, e.g., to networked manufacturing, self-organizing adaptive logistics, and customer-integrated engineering will require business models that will primarily be implemented by what could be a highly dynamic network of businesses rather than by a single company (e.g., to link products of a manufacturing company with appropriate services provided by another company) [1]. On the one hand—for realizing a *horizontal integration through value networks*—we need appropriate language constructs to describe business relationships between companies also taking their different business views into account. On the other hand—to enable a seamless *vertical integration of networked manufacturing systems*—we need a fundamental understanding of activities and information flows within manufacturing companies. However, information flows between the horizontal layer (business partner networks) and the vertical layer (from an ERP system to a Manufacturing Execution System) are very limited or not even possible at all [1]. IT systems still tend not to cross company or factory boundaries. The German initiative for Industrie 4.0 points out that the use of information technology in this context has largely failed to reflect the existence of manufacturing networks. One problem is, among others, that value chains (from customer requirements to production and distribution) tend to be relatively static since they often have been created over many years.

From a technical as well as an economic perspective an *end-to-end digital integration* will be a key issue to realize smart factories. This integration will enable all parts of a manufacturing company (enterprise level, shop floor control level, and shop floor level) to be connected to each other through a global information system with customers, suppliers, and other external participating parties. The potential of an end-to-end integration is huge. For example, this will allow in future to individual, customer-specific criteria to be included in the design, configuration, ordering, planning, manufacture and operation phase. This will enable last-minute changes to be incorporated and very low production volumes (batch size of 1). The realization of this ambitious goal requires appropriate interfaces for integrating the individual subsystems [2]. However, it is still common practice that IT systems exchange information through extensive interfaces, but can only utilize specific pieces of that information. The situation is further worsened by the problem that many different interfaces introduce dependencies whose management can become complex and hard to achieve. Thus, the system complexity will rise drastically.

There is still a lack of appropriate concepts for interface integration by which different operational layers can be connected for communication. However, to provide a universal infrastructure for a seamless information exchange is crucial for a successful implementation of the Industrie 4.0 initiative. Modeling can act as an enabler for managing this integration. Models are representations of real and hypothetical scenarios that only include those aspects that are relevant to the issue under consideration. The working group of the German initiative points to the fact that *"the use of models constitutes an important strategy in the digital world and is of central importance in the context of Industrie 4.0"* [1]. For this purpose appropriate language constructs are required to formally describe the increasing functionality, increasing

[1]Please note, that the approach introduced in this paper is aligned with the German initiative "Industrie 4.0", and therefore, we do not translate it to the English term "Industry".

978-1-4673-7340-1/15 $31.00 © 2015 IEEE
DOI 10.1109/CBI.2015.50

product customization, dynamic delivery requirements, and the rapidly changing forms of cooperation between different companies in order to provide end-to-end transparency.

II. Approach

The approach presented in this paper is based in its orientation on the recommendations of the German working committee for Industrie 4.0 which was released in 2013. Amongst other things, the working committee points out that production systems are to be linked vertically with business processes within decentralized production sites and enterprises, and that they are to be distributed horizontally among suppliers, distributors and customers. In order to meet these requirements, we aim for an *integrated modeling framework* spanning over the horizontal layer (value networks) and the vertical layer (production chains). For this purpose we do not intend to start from scratch by defining our own all-encompassing modeling language. In contrary, we want to build up on existing well-accepted modeling languages.

The German working group defines the *vertical integration* as *"the integration of the various IT systems at the different hierarchical levels (e.g., the actuator and sensor, control, production management, manufacturing and execution and corporate planning levels in order to deliver end-to-end solution"*, [1]. We consider the concepts and models of the industry standard *ISA-95* (ANSI/ISA-95; IEC 62264) [3], [4] as appropriate to model the vertical integration of information flows between the different levels within an enterprise. ISA-95 is an international standard released by the *International Society of Automation* for developing an automated interface between *Enterprise Resource Planning Systems (ERP)* on the enterprise level and *Manufacturing Excecution Systems (MES)* on the shop floor (control) level. Based upon this standard, which consists of five parts, the standard IEC 62264 was established.

In analogy, we apply concepts of the *Resource-Event-Agent business ontology (REA)* (ISO 15944-4) [5] which allows describing the interfaces between the systems of different business partners as a horizontal integration of information flows. The German working group defines the *horizontal integration* as referring to *"the integration of the various IT systems used in the different stages of the manufacturing and business planning processes that involve an exchange of materials, energy and information both within a company (e.g. inbound logistics, production, outbound logistics, marketing) and between several different companies (value networks)"*, [1]. In a business environment, REA is used to identify the value adding activities of the company. In general, value adding activities are either *transformations* of resources by producing something or *transfers* of resources by exchanging something with an external party. In other words, REA is able to provide the binding clue between the internal production processes requiring vertical integration and the external trading activities requiring horizontal integration.

In our approach, we elaborate on a seamless integration of the horizontal and vertical layers which implies that necessary information must flow between these layers. For realizing a vertical as well as a horizontal integration through value networks appropriate language constructs are needed to describe interface integration within the company between different kinds of IT systems (ERP, MES) at different levels and between multiple enterprises and various participating parties (vendors, sub-contractors, customers). For this purpose we transform REA concepts to ISA-95 concepts. Our approach is independent of any software solution. In fact, companies may use different ERP and MES systems and still have to collaborate with each other.

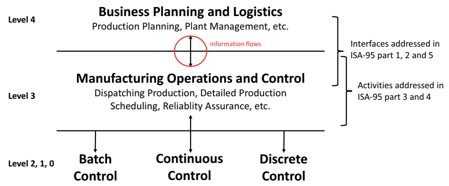


Fig. 1. ISA-95: Functional Hierarchy model [IEC 62264-1]

Following our aim of an integrated modeling framework by transforming concepts of REA to concepts and models of ISA-95, we concentrate on these two standards in Section III on related work. Section IV presents the REA meta model and its core concepts. In Section V, we present the ISA-95 meta model. Section VI provides the core of our paper describing the transformation rules from REA to ISA-95. This transformation is illustrated by examples in Section VII and Section VII-B. We close the paper with a summary of our contribution in Section VIII.

III. Related Work

A. Industry Standard ISA-95

The ISA-95 standard has been developed for global manufacturers, i.e., a production company with decentralized, networked production plants. This standard fosters a universal communication within a manufacturing company (headquarters and distributed industrial premises). ISA-95 can be applied in all industries, and in all sorts of production processes like batch processes, continuous processes, and repetitive processes. ISA-95 was specifically developed for creating interfaces between the enterprise domain with its ERP system at Level 4 and the shop floor control domain with its MES at Level 3 and lower (Levels 2, 1, 0). It offers a fundamental understanding of activities and information flows within a manufacturing company. The standard describes hierarchy models which are based on the *Purdue Enterprise Reference Architecture (PERA)* for Computer Integrated Manufacturing (CIM) [6].

Figure 1 shows in a simplified manner the different levels of the *functional hierarchy model*. In addition, the equipment (e.g., site, area, process cell, production line, storage zone) are usually organized in a hierarchical fashion. The red cycle in Figure 1 shows the *enterprise-control interface* between Level 4 and Level 3. Between these levels the standard points to 31 information flows, as outlined in Figure 2. The wide dotted line of this *functional enterprise control model* illustrates the boundary of the enterprise-control interface. Everything that lies outside the dotted lines belongs to Level 4, and everything that lies inside the dotted lines belongs to Level 3. The

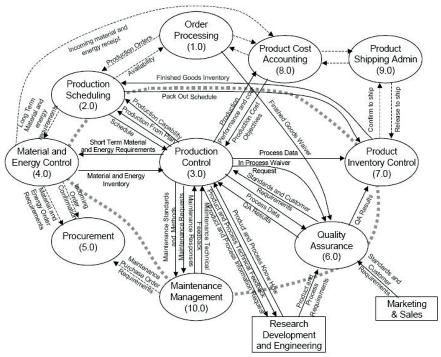


Fig. 2. ISA-95: Functional Enterprise-control model [IEC 62264-1]

labeled lines indicate the 31 information flows of importance to manufacturing control. The model contains 12 functions [3].

ISA-95 describes step by step the tasks of each of these functions. The functions shown in rectangles (e.g., research, development and engineering, marketing, sales) are external entities and as such they are not described in the functional enterprise control model. These entities are components outside the boundaries of this model that send data and receive data from the functions. The basic data to be exchanged in this model are information flows which are defined by ISA-95 for the sectors *personnel, material, equipment, physical asset* and *process segment*. The process segment is a logical group of equipment, physical asset, personnel, material required to carry out a specific part of a process (e.g, mixing, sawing, etc.). These sectors are defined as *object models* in ISA-95 which constitute basic building blocks with which the information flows of the functional hierarchy model are constructed (cf. Figure 1). In order to standardize the 31 information flows between Level 4 and Level 3 ISA-95 groups them into four categories: (i) production capability information, (ii) production definition information, (iii) production schedule information, and (iv) production performance information [3].

B. Resource-Event-Agent Business Ontology

The Resource-Event-Agent business ontology (REA) was developed by William McCarthy [7] for the application-independent description of *economic phenomena* (i.e., exchanges which can either be transfers or transformations of resources). The acronym REA stands for the three main concepts of the ontology *Resource*, *Event*, and *Agent*. Agents are persons, companies, or organizational units capable of having control over resources, who/which participate in an *economic exchange*. Resources are transferred or transformed during an economic exchange. Resources can be goods, material, rights, labor, equipment, physical assets or services which agents have control of and which should be monitored and controlled in a business environment. An event is considered as a class of phenomena reflecting exchanges of resources. REA has its roots in the accounting discipline and is based on strong concepts of the literature in economic theory [8]. Additionally, REA focuses on IT implementation issues and follows a conceptual modeling approach [9]. This makes it a good choice for being used in a business model-driven engineering approach. Moreover, the REA business ontology is a wide accepted language in the academic world to design enterprise information systems. For instance, in the ISO/IEC 15944-4 Open-edi standard [5]—which addresses business communications between enterprises—REA is used as an ontological framework for specifying concepts and relationships involved in business transactions and scenarios. REA initially focuses on concepts of economic exchanges of the present and the past.

IV. The REA Meta Model

In this section, we elaborate on the REA meta model. Thereby, we build up on previous work [10], [11], [12]. In these papers we developed a domain specific language (DSL) for the REA ontology called REA-DSL. The REA-DSL provides a formal definition of the REA language concepts by means of Object Management Group's (OMG) meta-modeling architecture called Meta-Object Facility (MOF) [13]. MOF comes with a meta-meta model (M3 layer) that allows us to define the REA concepts as a meta-model (M2 layer). In this section, we introduce the existing REA concepts by means of meta-models and also show some additional extensions required for this work.

REA consists of three different layers concerning entrepreneurial logic and details at a different level of granularity. The three layers from top down are:

1) value chain specification layer
2) duality specification layer
3) task specification layer

In the following subsections, we explain the meta-models of these REA layers.

A. REA Value Chain

A business model defines how a company creates value. It specifies a competitive strategy by looking at those activties that create value for the company. A seminal work in this respect has been Michael E. Porter's book "Competitive Advantage" [14] in which he first introduces the concept of the value chain. A *value chain* is a set of activities that an organization carries out to create value. Porter proposes the concept of a value chain to examine all of a company's activities, and see how these are connected.

The *REA value chain* is based on Porter's definition. It is built by a number of value activities. A *value activity* takes some resources as input and creates some resources as output. From an economic perspective it is important that the output is considered to be of higher value than the input. On a high level of abstraction there are two ways to create additional value by an activity: firstly, one may use and/or consume some input resources in order to produce some output (e.g., a finished good),—this is called a *transformation* in REA. Secondly, in a trading relationship with external business partners one may receive resources (e.g., material, equipment, transport service, etc.) and give resources (e.g., cash) in return,—this is called a *transfer* in REA.

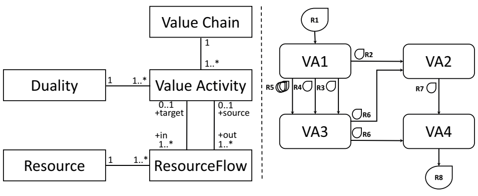


Fig. 3. REA: Value Chain Meta Model and its Instantiation

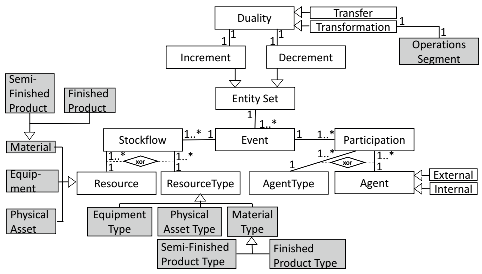


Fig. 4. REA: Duality Meta-Model

Furthermore, REA is built on the economic principle that any output by one value activity serves as input to another value activity. It follows that it is the resources which connect the different value activities. Thus, a REA value chain contains a number of value activities and specifies the resource flows amongst them—nothing more, nothing else [15]. More details are available on the second layer—the duality specification layer—where we find duality models for each of the value activities (cf. Figure 4).

The left hand side of Figure 3 presents the meta-model of the REA value chain. A value chain includes one to many value activities that are depicted by rectangles with rounded corners (cf. right hand side of Figure 3). A value activity is used only once in one distinctive value chain. A value activity points to exactly one duality (described in the next subsection). A *duality* is usually the basis of one value activity, but may be referred to by multiple value activities.

Resource flows tie the value activities together. A resource flow is a directed association that usually starts from a source value activity and ends at a target value activity (cf. right hand side of Figure 3). When analyzing a whole company, there is in theory no final output and no input that is not based on an output of another value activity. For the purpose of a partial analysis, we permit resource flows that have either no source value activity or no target value activity. It follows that a value activity has at least one, but up to many *outgoing* resource flows. Similarly, a value activity has at least one, but up to many *ingoing* resource flows. Each resource flow points to exactly one resource. This resource is depicted by the symbol of a drop next to the directed arc of the information flow. A resource may be included in many resource flows. The right hand side of Figure 3 shows an abstract example model of a value chain which is a valid instance of the meta-model on the left hand side.

B. REA Duality

In the previous subsection, we learned that value activities receive some input resources to create output recources of higher value. Each value activity is further detailed by a duality on the second REA layer. A *duality* is a core economic principle that says that it is impossible to get something for nothing ("there is no free lunch"). Accordingly, a duality consists of two parts: the *decrement entity set* covers events executed by some agents leading to a decrease of some resources. It is compensated by the *increment entity set* that covers events executed by some agents leading to an increment of some (other) resources. By definition the increment in resources is considered of higher value than the decrement in resources. Again, the duality concept applies to transfers (exchanges with external agents) and transformations (value creation inside the enterprise).

Figure 4 shows the meta-model for a duality. The meta classes with white background describe the existing REA concepts, the ones with gray background represent our proposed extensions described further below. A *duality* has two specializations: a *transfer* and a *transformation*. Independent of the specialization a duality is composed of exactly one *increment entity set* and one *decrement entity set*. Both are specializations of the general entity set. Each entity set is represented in a specific swimlane (cf. Figure 5). According to the REA meta-model, an entity set covers at least one but up to multiple events. An *event*—depicted as a hexagon—is specific to the entity set it belongs to (cf. Figure 5). Following the principles of duality, all events in the decrement entity set (give/consume/use) are counterbalanced by the events in the corresponding increment entity set (take/produce) of the same duality (cf. Figure 4).

The relationship between an event and a resource is described by the concept of *stockflow* [15]. A stockflow is represented as a directed arc between exactly one event (hexagon) and one resource (drop) (cf. Figure 5). In the increment set the direction of the arc goes from the resource to the event, in the decrement set in the reverse direction. An event will affect most of the time one resource only, but it may affect multiple ones. Thus, an event may have one up to many stockflows connected. A resource usually is affected by many different events (in different entity sets of different duality models). At a minimum a resource is affected by one event—otherwise it would not be worth considering the resource at all. Consequently, a resource is connected to one up to many stockflows.

In REA, resources can be goods, material, rights, labor, equipment, physical assets, or services. REA does not make any particular differentiation and all of these resources are denoted by the icon of a drop (cf. Figure 5). Due to its dedicated focus on the production domain, ISA-95 differentiates between *material*, *equipment*, and *physical asset* as special kinds of resources. When aiming for an integrated approach the differentiation of these special resources should be reflected in the REA ontology as well. Accordingly, we define *material*, *equipment*, and *physical asset* as specializations of the REA

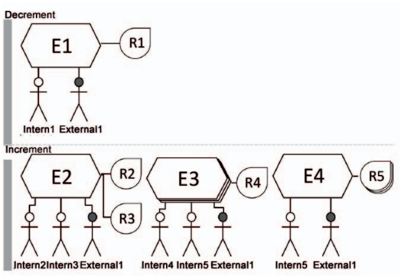


Fig. 5. Duality Example

resource (see classes with gray background in Figure 4). In addition, we define specializations for the corresponding typification concepts, i.e., *material type*, *equipment type*, and *physical asset type* are defined as specializations of the REA resource type. We also define dedicated icons for them. A material is denoted by a cuboid, an equipment by a white gear wheel, and a physical asset by a gray gear wheel. All of these specializations may also be used whenever a resource is expected in REA, i.e., as part of a duality and as classifiers assigned to resource flows. In addition, we introduce two specializations of the (non-abstract) material concept, i.e., *semi-finished product* presented by a white cube and *finished product* presented by a gray cube.

An event involves *agents* depicted as stickfigures. We distinguish between *external agents* (denoted with black heads), e.g., trading partners outside the company, and *internal agents* (denoted with white heads), who are accountable inside the company (cf. Figure 5). The involvement of agents in events is denoted by the concept of *participation*. A participation is an undirected association that connects exactly one event with one agent. An event is associated to at least one, but up to many agents. Hence, an event has one to many participation associations. An agent participates in at least one, but up to many events (in the same, but also in different entity sets of the same or different dualities). Thus, an agent has one to many participations connected. In addition, there are further constraints assigned to the meta-model to handle specifics of transfers. In case of a transfer, each event must be assigned to exactly one outside agent and, in addition, to at least one inside agent [16]. All events of the same transfer (both in the decrement and the increment entity set) must involve one and the same outside agent. Additionally, REA provides concepts for the *typification* of resources and agents [17]. *Resource types* and *agent types* display a small `T` in their icon. It should be noted that due to space limitations, we do not elaborate on the details of event series, resource series and agent series, which are denoted by a staple of hexagons/drops/stickfigures. The interested reader is referred to the paper of Sonnenberg et al. [10]. Figure 5 shows an abstract example model of the REA concept duality which is a valid instance of the meta-model presented in Figure 4.

C. REA task specification layer

In the first two subsections, we elaborated on the top two layers of REA (value chain specification layer and duality specification layer). One may expect that we do the same for the third layer—the *task specification layer*—describing the process to transform the input to the output as defined in the layers above. However, the REA literature does not concentrate on the task specification layer, instead it suggests to use activity diagrams or state machines to describe the task specification layer. REA does not provide any language concepts for linking identified tasks with agents, resources, etc. Accordingly, one may consider either extending the REA ontology for this purpose or specifying transformations to another language. In the context of the production domain, we are confident that ISA-95 is a perfect candidate langauge for the latter case. Accordingly, we propose that each REA duality model points to exactly one ISA-95 *operations segment* (see upper right corner of Figure 4). The relevant ISA-95 meta models with respect to an operations definition are described in the following section.

V. The ISA-95 Meta Model

Production operations are defined by the ISA-95 *operations definition model* that is depicted in key parts in Figure 6. An *operations definition* represents the resources required to perform a specified operation. The operations definition references a *work definition*, which defines the information used to instruct a manufacturing operation (i.e., how to perform the operation) [18]. An operations definition is associated to one to many *operations segments*. *Operations segments* may be recursively structured. An *operations segment* encapsulates the information needed to quantify a segment for a specific operation. It corresponds to one to many *process segments* [18]. Process segments are the smallest elements of manufacturing activities that are visible to business processes.

An operations segment provides a logical grouping of personnel resources, equipment resources, physical asset resources, and material required to perform a specific operations segment. Consequently, it includes different kinds of resource specifications: *personnel specifications*, *material specifications*, *equipment specifications*, and *physical asset specifications* [4]. These resource specifications identify the

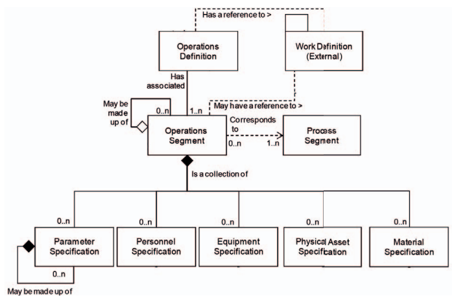


Fig. 6. ISA-95: Part of the Operations Definition Information Model [IEC 62264-2]

resource types and/or concrete resources, their quantity and the unit of measure of the quantity needed to perform an operations segment. For instance, to perform a frame production, we need—amongst other things—a specified quantity of a certain material (e.g., carbon crossbar) or material type (e.g., crossbar). In addition, the operations segment may include one to many *parameter specifications* containing the names and types of values that may be sent to the manufacturing execution systems at Level 3 to parametrize an operation.

Each of the above mentioned resource specifications within an operations segment are references to corresponding ISA-95 models [4]. The *material model* defines the actual materials, material definitions, and information about classes of material definitions. Material information includes the inventory of raw, finished, intermediate materials, and consumables [18]. The role-based *equipment model* contains information about specific equipment, the equipment class, and their particular properties. Role-based means that the equipment model is used to construct hierarchy models used in manufacturing scenarios (enterprise, site, area, work center, work units, process cells, etc) [18]. Due to this role-based view the equipment model is related to the *physical asset model* [4]. This model contains information about the physical piece within the manufacturing enterprise, i.e., a specific equipment. The *personnel model* contains information about specific personnel (class Person), classes of personnel (class Personnel Class) as well as their properties [4].

Accordingly, the schemata of these resource models are very similar and we do not detail all of them due to space limitations. We pick the material model as a typical representative of the resource models and present it in Figure 7. A *material class* may be defined as containing an assembly of material classes and as part of an assembly of material classes. A material class is a grouping of material definitions for an operations definition. A material class may define zero or more *material class properties*. Material class properties may contain nested material class properties. These properties often list the nominal, or standard values for the material (e.g., pH factor, material strength). A material property does not have to match material class properties. A *material definition* shall belong to zero or more material classes. Similar to material class, a material definition may be defined as containing an assembly of material definitions and as part of an assembly of material definitions. For a detailed description of the ISA-95 resource models, we refer the interested reader to the standard[4].

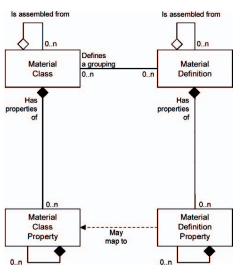


Fig. 7. ISA-95: Material Model [IEC 62264-2]

VI. Transformation Rules from REA to ISA-95

In the previous two sections, we described the REA meta model and the relevant parts of the ISA-95 meta model. In our integrated modeling framework, we intend to use REA for the purpose of modeling the main business functions of an enterprise. These business functions reside on Level 4 of the functional hierarchy model as depicted in Figure 1. In REA, one may distinguish business functions that require the exchange of resources with business partners, i.e., REA-transfers, and business functions that require the transformation of resources and are executed within the enterprise, i.e., REA-transformations. In an industry context, the latter ones are typically the production processes. Evidently, information about business functions describing transformations should be passed to the control functions at Level 3 of the functional hierarchy model. Accordingly, the relevant information in REA models has to be transformed to ISA-95 in order to realize the upper part of our intended integrated modeling framework. In this section, we describe the corresponding transformation rules which are depicted in Figure 8.

Each REA *duality* describing a *transformation* is transformed to an ISA-95 *operations segment* (A). The *name* of the *duality* becomes the *operations segment* ID (A1). Alternatively, one may decide to use logical, system generated identifiers, in which case a REA *duality ID* would map to the *operations segment* ID and the *name* of the *duality* to the *operations segment description*. In this paper, we have opted for "'readable"' IDs, also for other concepts described further below. The REA duality also links to a corresponding process definition which is carried forward to the *process segment* ID referenced by the operations segment (A2). It should be noted that each REA duality model leads to exactly one operations segment. In case that this operations segment is not fine granualar enough for control functions, one may re-work the operations segment in ISA-95 to create nested operations segments within it.

In the next steps, we have to transform the input resources for operations segments, which are personnel (B), equipment (C), physical assets (D), and materials (E). In REA the input side is described within the decrement entity set. Accordingly, calculating the input requires to access all events within the decrement entity set of a duality. The input then corresponds to the REA agents connected by *participation* associations to these events and the REA resources connected by *stockflow* associations.

It follows that each *agent* or *agent type* connected to a *decrement event* leads to a *personnel specification* within the *operations segment* (B). In the case of an *agent type* its *name* is mapped to the *personnel class ID* (B1). Whereas the *name* of a specific *agent* maps to the *person ID* of the *personnel specification* (B2). The *participation* association between an *event* and an *agent* has by default an attribute *quantity*, i.e. the number of *agent* (*types*) involved. This *quantity* is mapped to the *personnel specification quantity* (B3).

An *equipment* or *equipment type* connceted to a *decrement event* results in an *equipment specification* as part of the *operations segment* (C). In case of an *equipment type* its *name* maps to the *equipment class ID* (C1). Whereas the *name* of a specific

A	Duality $\Rightarrow$ Operations Segment
A1	Duality.Name $\Rightarrow$ OperationsSegment.ID
A2	Duality.ProcessDefinition $\Rightarrow$ OperationsSegment->ProcessSegment.ID
B	Duality->Decrement->Event->Participation->Agent/AgentType $\Rightarrow$ OperationsSegment->PersonnelSpecification
B1	Duality->Decrement->Event->Participation->AgentType.Name $\Rightarrow$ OperationsSegment->PersonnelSpecification. PersonnelClassID
B2	Duality->Decrement->Event->Participation->Agent.Name $\Rightarrow$ OperationsSegment->PersonnelSpecification. PersonID
B3	Duality->Decrement->Event->Participation->Quantity $\Rightarrow$ OperationsSegment->PersonnelSpecification.Quantity
C	Duality->Decrement->Event->Stockflow->Equipment/EquipmentType $\Rightarrow$ OperationsSegment->EquipmentSpecification
C1	Duality->Decrement->Event->Stockflow->EquipmentType.Name $\Rightarrow$ OperationsSegment->EquipmentSpecification.EquipmentClassID
C2	Duality->Decrement->Event->Stockflow->Equipment.Name $\Rightarrow$ OperationsSegment->EquipmentSpecification.EquipmentID
C3	Duality->Decrement->Event->Stockflow->Quantity $\Rightarrow$ OperationsSegment->EquipmentSpecification.Quantity
D	Duality->Decrement->Event->Stockflow->PhysicalAsset/PhysicalAssetType $\Rightarrow$ OperationsSegment->PhysicalAssetSpecification
D1	Duality->Decrement->Event->Stockflow->PhysicalAssetType.Name $\Rightarrow$ OperationsSegment->PhysicalAssetSpecification.PhysicalAssetClassID
D2	Duality->Decrement->Event->Stockflow->PhysicalAsset.Name $\Rightarrow$ OperationsSegment->PhysicalAssetSpecification.PhysicalAssetID
D3	Duality->Decrement->Event->Stockflow->Quantity $\Rightarrow$ OperationsSegment->PhysicalAssetSpecification.Quantity
E	Duality->Decrement->Event->Stockflow->Material/MaterialType $\Rightarrow$ OperationsSegment->MaterialSpecification
E1	Duality->Decrement->Event->Stockflow->MaterialType.Name $\Rightarrow$ OperationsSegment->MaterialSpecification.MaterialClassID
E2	Duality->Decrement->Event->Stockflow->Material.Name $\Rightarrow$ OperationsSegment->MaterialSpecification.MaterialDefinitionID
E3	Duality->Decrement->Event->Stockflow->Quantity $\Rightarrow$ OperationsSegment->MaterialSpecification.Quantity
E4	Duality->Decrement->Event->Stockflow->UnitOfMeasure $\Rightarrow$ OperationsSegment->MaterialSpecification.UnitOfMeasure
E5	Duality->Decrement->Event->Stockflow->Material/MaterialType $\Rightarrow$ OperationsSegment->MaterialSpecification->MaterialUse = ”Consumed”
F	Duality->Increment->Event->Stockflow->Material/MaterialType $\Rightarrow$ OperationsSegment->MaterialSpecification
F1	Duality->Increment->Event->Stockflow->MaterialType.Name $\Rightarrow$ OperationsSegment->MaterialSpecification.MaterialClassID
F2	Duality->Increment->Event->Stockflow->Material.Name $\Rightarrow$ OperationsSegment->MaterialSpecification.MaterialDefinitionID
F3	Duality->Increment->Event->Stockflow->Quantity $\Rightarrow$ OperationsSegment->MaterialSpecification.Quantity
F4	Duality->Increment->Event->Stockflow->UnitOfMeasure $\Rightarrow$ OperationsSegment->MaterialSpecification.UnitOfMeasure
F5	Duality->Increment->Event->Stockflow->Material/MaterialType $\Rightarrow$ OperationsSegment->MaterialSpecification->MaterialUse = ”Produced”

Fig. 8. Transformation Rules for ISA-95 Operations Segments

equipment transforms to the *equipment ID* of the *equipment specification* (C2). The *stockflow* association between an *event* and an *equipment* has by default an attribute *quantity* which is mapped to the *equipment specification quantity* (C3). The transformation rules for physical assets (D) are mirroring the ones for equipment (C).

Also the transformation rules for input *materials* (E) are similar to the ones for *equipment* (C) and *physical assets* (D). Evidently, the *quantity* of *materials* is not always a number of pieces. Consequently, there is an additional transformation rule mapping the *unit of measure* of the *quantity* of a *stockflow* to the *unit of measure* of the *material specification* (E4). However, most important is the fact that a *material* connected to a *decrement event* is considered as an input and thus the attribute *material use* of *material specification* is set to the value `consumed` (E5).

The transformation rules B - E describe the input side. The transformation rules for the output side are the ones in section F. The output of an *operations segment* is by definition the produced *material* or *material type* (including the specializations *semi-finished goods* and *finished goods*). In REA, the output are *materials* or *material types* connected via *stockflow* associations to *events* that reside in the *increment* partition. Accordingly, the transformation rules for output *materials* (F) are the same as for input *materials* (E) except for the fact that they apply to the *increment* side and not to the *decrement* side. In addition, the *material use* attribute is set to `produced` (F5). Furthermore, it is worth mentioning that *semi-finished goods* and *finished goods* are specializations of materials, and thus the transformation rules in sections E and F apply as well.

The transformation rules described above are used to map REA duality models to ISA-95 operations segments. In ISA-95, operations segments are not stand-alone items, but are always part of an operations definition. At first sight, one might assume that a REA value chain maps to a single operations definition and all duality models in the value chain become part of this operations definition. However, such an approach is too naive in practice. Our practical experience has shown that usually some duality models are grouped into one operations definition, but it always requires a human decision on this grouping. Accordingly, the transformation of a value chain to an operations definition is always a semi-automatic process requiring feedback from the modeler.

In this paper, we concentrated on the transformation of duality models to operations segments (of operations definition items), because they have a high significance for our approach. Nevertheless, it is important to note that REA also offers concepts to model the attributes of resources (equipment, physical assets, and material) and of agents as well as of their typification. The underlying meta model (cf. [11]) is conceptually very similar to corresponding ISA-95 resource models. Consequently, the transformation is rather straightforward and we do not further elaborate on them due to space limitations.

VII. REA TO ISA-95 TRANSFORMATION EXAMPLE

A. The REA model of Maxi Bike

The business model of Maxi Bike is to produce and sell bicycles. Figure 9 presents Maxi Bike's value chain, which is an instantiation of the value chain meta model depicted on the left hand side of Figure 3. Keeping the example simple and easy to follow, we only present a partial analysis and do not show value activities for acquiring equipment, physical assets, raw materials and labor. The *value chain* covers five *value activities*: `Purchase`, `Transport`, and `Sale`

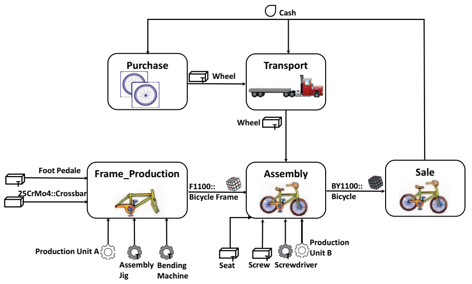


Fig. 9. REA Value Chain of Maxi Bike

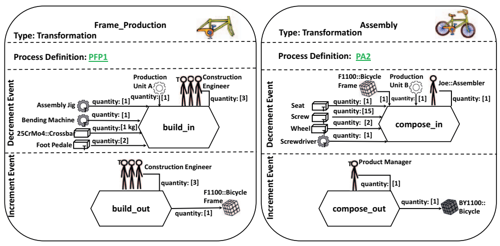


Fig. 10. Frame_Production and Assembly - Duality Models

are REA-transfers requiring horizontal integration, whereas `Production` and `Assembly` are REA-transformations requiring vertical integration. The value chain shows the flow of resources (materials/equipments/physical assets) amongst them. In `Purchase` the *resource* `Cash` is used to get the *material type* `Wheel`. `Wheel` and again `Cash` are used in `Transport` to receive the wheels at the right location (`Production Unit B`).

The *ingoing* resource flows of the *value activity* `Frame_Production` are the *material type* `Foot Pedale`, the *material* `25CrMo4::Crossbar`, the *equipment* `Production Unit A` and the *physical asset types* `Assembly Jig` and `Bending Machine`. The *outgoing* resource flow of this *value activity* is the *specialized material* `F1100::Bicycle Frame` which is a *semi-finished product*. The *value activity* `Assembly` has as *ingoing* resource flows the *material types* `Seat`, `Screw`, `Wheel` and as semi-finished product `F1100::Bicycle Frame`. The other *ingoing* resource flows are the *physical asset type* `Screwdriver` and the equipment `Production Unit B`. These resources are transformed (i.e., used and consumed) to produce the specialized *material* `BY1100::Bicycle` which is the *finished product*. In the *value activity* `Sale` the `BY1100::Bicycle` is turned into `Cash` which is used as input for the other *value activities* mentioned above.

Each of the five *value activities* presented in Figure 9 must be refined by a *duality* model. Due to space limitations we only show the *duality model* for `Frame_Production` and `Assembly` (cf. Figure 10). The left hand side of Figure 10 shows the duality `Frame_Production` which is of the REA type *Transformation*. The `build_in` *decrement* event is performed by the *agent type* `Construction Engineer`. In order to build the *semi-finished product* `F1100::Bicycle Frame`, a *quantity* of `3` engineers is needed. The frame production is carried out in `Production Unit A` and leads to a decrease of a *quantity* of `1 kg` of the input resource *material* `25CrMo4::Crossbar`. To accomplish the frame the *material type* `Foot Pedale` decreases by a *quantity* of `2`. Additionally, the *physical asset types* `Assemble Jig` and `Bending Machine`, each of which with a *quantity* of `1`, are used. In the *increment* event `build_out` the produced good is the *semi-finished product* `F1100::Bicycle Frame` with a *quantity* of `1`, received by `one` *agent type* who has to be a `Construction Engineer`.

The right hand side of Figure 10 depicts the *duality* model `Assembly` which is again a *Transformation*. The `compose_in` *decrement* event is performed by the *inside agent* `Joe::Assembler` and leads to a decrease of the input resources by consuming the *semi-finished product* `F1100::Bicycle Frame` and the *material types* `Wheel` with a *quantity* of `2`, `Screw` with a *quantity* of `15` and `Seat` with a *quantity* of `1`. In addition, the *physical asset type* `Screwdriver` (*quantity* `1`) and the *equipment* `Production Unit B` are used. This *decrement* event is compensated by the *increment* event `compose_out`, which produces the specialized *material* `BY1100::Bicycle` as final product received by the *agent type* `Product Manager`. These example models do not specify any process details on how to produce the bicycle frame or assemble the bicycle. They only provide links to the *Process Definitions* `PFP1` and `PA2`.

B. Mapping the REA Duality Models to B2MML

In contrary to the REA-DSL, ISA-95 does not come with any dedicated graphical language as a concrete syntax to represent ISA-95 compliant models. Accordingly, one may only use a corresponding object diagram as an abstract syntax. However, the Business To Manufacturing Markup Language (B2MML) [19] is an XML implementation of ISA-95. In other words, B2MML defines XML schemas that are exact equivalents of the ISA-95 meta model. Accordingly, one may use a B2MML XML file that is valid with respect to the B2MML schema to show a valid instance of the ISA-95 standard. This is our choice for illustrating the example.

In the following, we demonstrate the mapping of the two duality models `Frame_Production` and `Assembly` as depicted in Figure 10 to B2MML. This mapping uses the transformation rules of Figure 8. The resulting B2MML file is listed in Figure 11. For easier readability, we do not use closing XML tags, but use indent style instead. This B2MML file lists an operations definition with two *operations segments* (`Frame_Production` and `Assembly`), which are both one to one mappings of the REA duality models `Frame_Production` and `Assembly`. All the information in green font is a result of applying our transformation rules. It should be noted that the grouping of the two duality models or operations segements, respectively, has been done manually and, consequently, the instances in black font have to be created manually.

```
<OperationsDefinitionInformation>
  <ID> BY1100-ODI
  <Description> Bicycle BY1100 Production
  <OperationsType>Transformation
  <PublishedDate> 2015-03-27
  <OperationsDefinition>
    <ID> BY1100-OD
    <Version> V1
    <Description> BY1100 Bicycle Operations Definition
    <WorkDefinition> WBY1100
    <OperationsSegment>
      <ID> Frame_Production
      <ProcessSegmentID> PFP1
      <PersonnelSpecification>
        <PersonnelClassID> Construction Engineer
        <Quantity> 3
      <EquipmentSpecification>
        <EquipmentID> Production Unit A
      <PhysicalAssetSpecification>
        <PhysicalAssetClassID> Assembly Jig
        <Quantity> 1
      <PhysicalAssetSpecification>
        <PhysicalAssetClassID> Bending Machine
        <Quantity> 1
      <MaterialSpecification>
        <MaterialClassID> Crossbar
        <MaterialDefinitionID> 25CrMo4
        <MaterialUse> Consumed
        <Quantity> 1
        <UnitOfMeasure> kg
      <MaterialSpecification>
        <MaterialClassID> Foot Pedale
        <MaterialUse> Consumed
        <Quantity> 2
      <MaterialSpecification>
        <MaterialClassID> Bicycle Frame
        <MaterialDefinitionID> F1100
        <MaterialUse> Produced
        <Quantity> 1
    <OperationsSegment>
      <ID> Assembly
      <ProcessSegmentID> PA2
      <PersonnelSpecification>
        <PersonID> Joe::Assembler
      <EquipmentSpecification>
        <EquipmentID> Production Unit B
      <PhysicalAssetSpecification>
        <PhysicalAssetClassID> Screwdriver
        <Quantity> 1
      <MaterialSpecification>
        <MaterialClassID> Bicycle Frame
        <MaterialDefinitionID> F1100
        <MaterialUse> Consumed
        <Quantity> 1
      <MaterialSpecification>
        <MaterialClass> Seat
        <MaterialUse> Consumed
        <Quantity> 1
      <MaterialSpecification>
        <MaterialClass> Screw
        <MaterialUse> Consumed
        <Quantity> 15
      <MaterialSpecification>
        <MaterialClass> Wheel
        <MaterialUse> Consumed
        <Quantity> 2
      <MaterialSpecification>
        <MaterialClassID> Bicycle
        <MaterialDefinitionID> BY1100
        <MaterialUse> Produced
        <Quantity> 1
```


Fig. 11. B2MML example: Maxi Bike

The first *operations segment* presented in the B2MML example in Figure 11 with the ID `Frame_Production` is a one to one mapping to the REA *duality model* `Frame_Production`. This *operations segment* contains a *process segment ID* `PFP1` that corresponds to the link specified in the REA *duality* model. The *personnel specification* of the *operations segment* contains the *personnel class ID* `Construction Engineer` which is results from the REA agent type `Construction Engineer` who participates in the *decrement* event `build_in` attributed by a *quantity* of `3`. The *equipment specification* with its ID `Production Unit A` and the *physical assets specifications* `Assembly Jig` and `Bending Machine` are mapped according to the equipment and physical asset types connected to the `build_in` *decrement* event. Each of them has a *quantity* of `1`.

The *decrement* event `build_in` expects a *material type* `Foot Pedale` and a *material* `25CrMo4` which is of *material type* `Crossbar`. Accordingly, we have two *material specifications*. The first one is for the *material class* ID `Crossbar` and the exact *material definition* ID `25CrMo4`, whereas the second one only mentions the *material class* ID `Foot Pedale` without any more detailed *material definition*. These *material specifications* are considered as input resources and thus the attribute *material use*, of both of them, is set to the value `Consumed`. The *material class* ID `Bicycle Frame` with the *material definition* ID `F1100` has the status `Produced` with a quantity of `1`, which is a mapping result of the *increment* event `build_out`. The transformation of the duality `Assembly` to the second *operations segment* is done in the exactly same manner, and thus, is not described in further detail.

VIII. Conclusion

It is our overall goal to develop a universal model-driven approach towards the horizontal and vertical integration in the context of smart factories. For this purpose we strive for an *integrated modeling framework* based on existing modeling approaches. Thereby, we built up on the REA business ontology to identify, both, activities requiring horizontal integration with business partners and activities serving as hooks into the internal systems requiring vertical integration. The latter activities have then to be further detailed by means of the ISA-95 standard. Accordingly, it is of crucial importance to transform concepts of REA to concepts of ISA-95.

First of all, this requires an alignment of concepts that appear to be similar in REA and ISA-95. In this respect, we have extended the resource concept in REA by similar concepts from ISA-95. In particular, we introduce specializations of the concept resource, namely equipment, physical asset, and material. Evidently, these extensions also apply to the REA type level.

Most importantly, we have developed dedicated transformation rules for the purpose of transforming a REA model into an ISA-95 one. In particular, we map REA duality models to ISA-95 operations segments. Thereby, we are able to convert information about the input and output of business functions to the control functions. Nevertheless, it is important to note that later on this information needs to be further detailed on the shop floor control level.

For the evaluation of our approach, we have first implemented the proposed REA extensions into our REA DSL tool. In a next step, we added the transformation rules to our tool. For the moment these rules have been hard coded, but it is planned to use a dedicated transformation language in the future. Accordingly, we demonstrated the technical feasibility of our approach by mapping from REA-DSL to B2MML (the XML equivalent of ISA-95). The syntactical correctness of the transformation has been checked by the proof of valid B2MML XML instances. More extensive case studies are planned for the future, once the overall modeling framework spanning over all hierarchical layers has been realized.

REFERENCES

[1] Industrie 4.0 Working Group, "Recommendations for implementing the strategic initiative INDUSTRIE 4.0," Final report, April 2013.

[2] G. Spur, *Optionen zukünftiger industrieller Produktionssysteme (in German)*. Akademie Verlag, 1997.

[3] *Enterprise system integration – Part 1: Models and terminology*, IEC ISO/IEC 62 264-1, 2012.

[4] *Enterprise-control system – Part 2: Objects and attributes for enterprise-control system integration*, IEC ISO/IEC 62 264-2, 2013.

[5] *Open-edi Part 4: Business transaction scenarios – Accounting and economic ontology*, ISO/IEC ISO 15 944-4, 2007.

[6] T. J. Williams, "The Purdue Enterprise Referene Architecture - A Technical Guide for CIM Planning and Implementation," Institute for Interdisciplinary Engineering Studies, Purdue University, Tech. Rep., 1992.

[7] W. E. McCarthy, "The REA accounting model: A generalized framework for accounting systems in a shared data environment," *The Accounting Review*, vol. 57, no. 3, pp. 554–578, July 1982.

[8] C. Y. Yu, *The Structure of Accounting Theory*. The University Press of Florida, 1976.

[9] P. Chen, "The Entity-Relationship Model - Towards a Unified view of Data," *ACM Transactions on Database Systems*, 1976.

[10] C. Sonnenberg, C. Huemer, B. Hofreiter, D. Mayrhofer, and A. Braccini, "The REA-DSL: A Domain Specific Modeling Language for Business Models," in *Proceedings of the 23rd International Conference on Advanced Information Systems Engineering (CAiSE 2011)*. Springer LNCS 6741, 2011, pp. 252–266.

[11] D. Mayrhofer and C. Huemer, "REA-DSL: Business Model Driven Data-Engineering," in *Proceedings of the 14th IEEE International Conference on Commerce and Enterprise Computing (CEC 2012)*. IEEE CS, 2012, pp. 9–16.

[12] ——, "Extending the REA-DSL by the Planning Layer of the REA Ontology," in *Advanced Information Systems Engineering Workshops (CAiSE 2012)*. Springer LNBIP 112, 2012, pp. 543–554.

[13] *Ontology Definition Metamodel (ODM)*, 1st ed., Third Revised Submission to OMG/RFP, OMG Object Management Group, May 2009, available at http://www.omg.org/spec/ODM/1.0/ (checked online July-11-2011).

[14] M. E. Porter, *Competitive Advantage: Creating and Sustaining Superior Performance*. The Free Press, 1985.

[15] G. L. Geerts and W. E. McCarthy, "Modeling business enterprise as value-added process hierarchies with resource-event-agent object templates," in *In Business Object Design and Implementation*. Springer Verlag, 1997, pp. 94–113.

[16] ——. (2005, March) The Ontological Foundation of REA Enterprise Information Systems. [Online]. Available: http://www.msu.edu/user/mccarth4/tulane.doc

[17] ——, "Type-Level Specification in REA ESystems Information Systems." American Accounting Association, August 2005.

[18] B. Scholten, *The Road to Integration - A Guide to Applying the ISA-95 Standard in Manufacturing*. ISA - Instrumentation, Systems, and Automation Society, 2007.

[19] Business To Manufacturing Markup Language (B2MML). World Batch Forum (WBF). [Online]. Available: http://isa-95.com/b2mml/

3 AutomationML, ISA-95 and Others: Rendezvous in the OPC UA Universe

B. Wally, C. Huemer, A. Mazak and M. Wimmer;
Proceedings of the 14th IEEE International Conference on Automation Science and Engineering (CASE), IEEE, (2018), pp. 1381–1387.
DOI: 10.1109/COASE.2018.8560600

2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)
Munich, Germany, August 20-24, 2018

AutomationML, ISA-95 and Others: Rendezvous in the OPC UA Universe

Bernhard Wally[1], Christian Huemer[2], Alexandra Mazak[1] and Manuel Wimmer[1]

***Abstract*— OPC Unified Architecture (UA) is a powerful and versatile platform for hosting information from a large variety of domains. In some cases, the domain-specific information models provide overlapping information, such as (i) different views on a specific entity or (ii) different levels of detail of a single entity. Emerging from a multi-disciplinary engineering process, these different views can stem from various tools that have been used to deal with that entity, or from different stages in an engineering process, e.g., from requirements engineering over system design and implementation to operations. In this work, we provide a concise but expressive set of OPC UA reference types that unobtrusively allow the persistent instantiation of additional knowledge with respect to relations between OPC UA nodes. We will show the application of these reference types on the basis of a rendezvous of AutomationML and ISA-95 in an OPC UA server.**

I. Introduction

OPC UA is seen as the future of communication in industrial settings, and has been designed in a very flexible way in order to be able to adapt to future demands and developments [1]. As such, domain-specific information has been kept out of the standard as strictly as possible [2]. Instead companion specifications are to be developed for introducing meaning (semantics) to the syntax provided by OPC UA. This is also a necessity in order to foster interoperability of tools in industrial automation. Interestingly, due to the flexibility of OPC UA, an OPC UA server, which is the host for OPC UA information models, can be "polluted" with overlapping and partly redundant (possibly contradicting) information from a variety of sources and domains.

In this work we will examine what kind of overlapping information is typically available in OPC UA servers by investigating on a set of OPC UA companion specifications and how this overlapping information can be aligned in a way that IT systems can deal with this kind of information more confidently.

The paper is structured as follows: after some relevant and necessary background information (Sec. II), we will discuss related work (Sec. III) before we explicate and evaluate our approach (Sec. IV). Following a critical discussion (Sec. V) we conclude and provide further research directions (Sec. VI).

[1]Bernhard Wally, Alexandra Mazak and Manuel Wimmer are with the Christian Doppler Laboratory for Model-Integrated Smart Production, Institute of Information Systems Engineering, TU Wien, 1040 Vienna, Austria, {wally, mazak, wimmer}@big.tuwien.ac.at

[2]Christian Huemer is with the Business Informatics Group, Institute of Information Systems Engineering, TU Wien, 1040 Vienna, Austria, huemer@big.tuwien.ac.at

II. Background

A. OPC UA

OPC Unified Architecture (UA) is a series of standards brought forward by the OPC Foundation; it defines a modern, object- and service-oriented communication stack and modeling paradigm for industrial automation [2]. Its versatility allows the mapping of various domain-specific models into OPC UA equivalent information models, such as AutomationML and ISA-95 models. OPC UA implements a type-object pattern [3], [4], [5] which enables domain modeling (i) at runtime and (ii) decoupled from the OPC UA core standard.

B. AutomationML

The Automation Markup Language (AutomationML) is based on Computer Aided Engineering Exchange (CAEX), which is a data format that has been defined in the scope of IEC 62424 and provides structures (i) for information exchange between *piping and instrumentation diagram* tools and *process control engineering* related *computer aided engineering* tools, as well as (ii) for the representation of process control engineering requests in piping and instrumentation diagrams [6]. CAEX is based on XML and enables the metamodeling and modeling of, e.g., the hierarchical architecture of a plant, including involved machines and controllers and their physical and logical connections.

AutomationML is standardized as IEC 62714 and defines sets of role classes and interface classes with certain restrictions regarding their application [7], [8]. AutomationML defines an abstract interface class `ExternalDataConnector` which is used to reference external documents and elements therein. Two use cases of this external data connector have been defined so far in separate whitepapers: (i) `COLLADAInterface` specifies how external COLLADA[3] documents are referenced [9] and (ii) `PLCopenXMLInterface` defines how PLCopen[4] XML documents (which are based on IEC 61131-3 [11]) can be referenced from AutomationML documents [12]. This referencing mechanism plays an important role in the analysis presented later.

[3]COLLADA—Collaborative Design Activity: an XML based exchange format for 3D assets (cf. https://www.khronos.org/collada/).

[4]PLCopen is a vendor- and product-independent association active in industrial control (cf. http://www.plcopen.org/). PLCopen XML is a data exchange format for the storage of programmable logic controller (PLC) program information according to IEC 61131-3 [10].

978-1-5386-3593-3/18/$31.00 ©2018 IEEE

C. ISA-95

ISA-95 (IEC 62264) is a series of standards that addresses the integration of the enterprise domain with the manufacturing and control domains. It defines a set of object models for the exchanging of information between these domains—it provides a standard terminology and set of concepts for system integration [13]. The relevant part of ISA-95 for this work is part 2, specified in IEC 62264-2:2013 [14], which defines common objects and attributes such as personnel, equipment, material, process segments, schedules and performance data. An XML serialization of ISA-95 has been defined in [15] under the name "business to manufacturing markup language" (B2MML).

III. Related Work

One of the earliest OPC UA companion specifications was about the modeling of ISA-95 data: in [16] an initial OPC UA information model is provided that defines the base entities from which domain-specific OPC UA nodes must be derived in order to be valid ISA-95 instances. This specification is one of the foundations for our analysis.

Similarly, in [17] a first draft for the modeling of AutomationML information in OPC UA was presented, that was later standardized in [18]. There, the rules for transforming AutomationML entities into OPC UA entities are defined alongside an initial OPC UA information model that is to be used as a base for own AutomationML models. The standardized specification is another important input to our analysis presented in Sec. IV.

An AutomationML application recommendation is prepared in [19], where the alignment of AutomationML and ISA-95 is described, i.e., how to encode ISA-95 related information in AutomationML documents or how to refer to external ISA-95 information stored in separate files. This application recommendation is the third main pillar for the analysis given in this work.

In [20] it is shown that vertical integration can go beyond the integration of just two domains. There, besides AutomationML and ISA-95, even an accounting/business metamodel is brought into the vertical integration chain: the Resource-Event-Agent business model language [21]. The approach presented in this work further strengthens the knowledge-base of manufacturing enterprises and could be part of the enabling technologies for reaching from the business model down to the shop floor.

An initial discussion of the alignment of AutomationML and ISA-95 is given in [22], and a bit more in-depth in [23]. However, the approach that has been presented in [23] imposes workarounds for specific instance constellations (e.g., multi-classification of equipment), that could be solved in a more elegant way by changing some of the metamodel mappings that have been defined there. Specifically, equipment classes should be better modeled in terms of AutomationML role classes to overcome the single-classification restriction given by the utilization of system unit classes. Such improvements have been already incorporated in [19].

A production data transformation scheme is brought up in [24], targeting faster data processing for, e.g., decision support systems. There, ID based lookup of entities from a storage system is employed for data retrieval. The mechanism presented in this work would enable a different lookup mechanism based on explicit linking of related entities.

Aligning enterprise information systems (EIS) with production systems is described in [25], using a service oriented approach. Given, that entities of both the business and the production domain are available in OPC UA, this is another use-case for the approach presented in this work. A related idea is superficially presented in [26]: using formalism from ISA-95 in order to refine coarser-grained concepts from an EIS, such as those emerging from REA (cf. also [27] for the applicability of REA as an EIS). Also there, explicit linking of entities within a dedicated communication protocol would help superordinate systems in their orchestration tasks.

The work presented here can be related to the topic of model composition, which is tackled in [28] and that defines an abstract model-weaving metamodel enabling the specification of relations such as "rename", "override" and "merge". For us, this approach is one step too generic, as we would like to explicitly name the kind of inter-model relations, and make this set of relations fixed (especially from a semantics point of view).

Explicit coupling of corresponding elements in models of different domains is presented in [29]. It makes use of a dedicated "linking" metamodel that is used to describe the kind of relationship between corresponding elements. This approach seems to be very promising, as it does not impose changes to the domain meta-models. It could serve as a base for specifying similar behavior in the realm of OPC UA information models. In fact, we are building on the findings discussed there.

A different approach was taken in [30], where AutomationML was transformed into a Resource Description Framework (RDF) representation, which is a generic ontology modeling language. Linking of semantically related entities is realized in this ontological domain, and standard ontology-tools can be used for querying and browsing the loaded AutomationML model(s). It might be worthwhile to use their toolset for the reasoning about inter- and intra-modal relations and then convert them to OPC UA references in a live information model, as we do it in the approach presented in our work.

Inter-model dependencies in production system engineering comprising software, electrics and mechanics are presented in [31]. There, the importance of understanding models of different domains as different views on the same underlying physical aspects is made clear. Their approach was based on the systems modeling language (SysML) and leveraged internal block diagrams and their ports and flows concepts for modeling different engineering views. In our example, SysML would be yet another domain model that might be translated into OPC UA at one point and that would be linked to other domain models by the references defined in our work.

A slightly more complex approach is presented in [32], describing an implementation concept for the “administration shell” of Industry 4.0 components [33]. There, multiple OPC UA servers are employed and they interact with each other through dedicated OPC UA clients. The approach presented in our work would enable a semantic linking of components in different OPC UA servers, since OPC UA references are allowed to point to nodes that are deployed in another OPC UA server [1]. This semantic knowledge could in turn be used for the (auto-)configuration of OPC UA clients dealing with inter-server communication.

IV. Rendezvous in the OPC UA Universe

A. Specific Problem Domain

Overlapping domain-specific models not only exist in the case of AutomationML and ISA-95, but for example with AutomationML and PLCopen. In [34], an information model definition is given that enables the transformation of PLCopen information into an OPC UA information model. Additionally, in [12], a recommendation for the inclusion of PLCopen information into AutomationML documents is given, based on PLCopen XML [10]. So also in this case, two different, yet interwoven domains can be both converted into OPC UA information models by separate means, i.e., there exist rules for transforming each of the domain models into OPC UA, but each of these rule-sets is not necessarily taking the other rule-sets into account.

In [35], a typical plant engineering process is depicted showing the different phases in design, planning and operation. In a fully digitized workflow it is very likely that some of the modeled artifacts are available in an OPC UA server, many of which originating from different tools following different metamodels, implementing diverse domain knowledge. Deploying such artifacts in parallel with each other requires some kind of methods to formulate their relationship to each other (e.g., how about the relation between a physical production machinery and its digital twin?).

It is infeasible to make all OPC UA mapping standards (e.g., [36], [37], [38]) aware of each other and harmonize their information models as far as possible, because (i) the amount of bilateral reconciliations increases dramatically with the addition of another standard and (ii) the standards would have to be edited quite often which would render them inconvenient standards (which are usually supposed to be rather stable). Fig. 1 shows an excerpt of the domain-specific base information models for AutomationML and ISA-95 within OPC UA. It can be observed that the domain-specific object types form two disjoint sub-trees. The consequence is that user-defined object types of an application specific information model can not be modeled as sub-type of both domains. This is due to OPC UA modeling rules that allow only single-inheritance for object type hierarchies.

Instead we believe that this problem would be better addressed at a higher level of abstraction, and it could follow the approach given in [29] by making inter-model-relationships explicit. This could be realized by a separate information model or it could be integrated into the OPC UA standard information model as a means for modeling such relations.

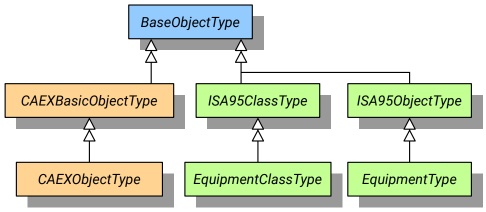


Fig. 1. (Left, orange) AutomationML and (right, green) ISA-95 base information models for OPC UA. `BaseObjectType` on the top (blue) is the root object type defined by the OPC UA standard information model.

To make our point more clear, let’s consider the example of AutomationML and ISA-95 again: it would be very elegant to be able to provide the functionality depicted in Fig. 2. The mapping relations resemble transformation rules for the conversion of information from one domain into information of another domain: (i) function f describes how to transform AutomationML models into the OPC UA space, (ii) function h describes the transformation from ISA-95 to OPC UA, and (iii) function g provides a transformation from ISA-95 to AutomationML. It would be beneficial for a number of use cases if the function composition $f \circ g = h$ would hold, i.e., ISA-95 models that are mapped directly into the OPC UA space via function h have an identical representation there as if they would first be transformed into AutomationML (via g) and only then be mapped into the OPC UA space (via f).

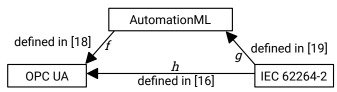


Fig. 2. Composition of transformation functions. Due to the mapping definitions defined in [18], [19] and [16] it follows: $f \circ g \neq h$.

However, it is not possible to provide this function composition, as we can easily show with an example. Consider an AutomationML internal element that implements the ISA-95 role class `Equipment`. It would be instantiated as an OPC UA object with a type definition pointing to a sub-type of the OPC UA object type `AutomationMLBaseRole` following [18]. An equipment instance of ISA-95 would in turn be instantiated as an OPC UA object with a type definition pointing to the OPC UA object type `EquipmentType` following [16]. Since an OPC UA object is allowed only one type definition, it is not possible to create an OPC UA object type that is a sub-type of both `AutomationMLBaseRole` and `EquipmentType`. Consequently, the transformation rules will create two disjoint object type sub-hierarchies that can be connected with each other using OPC UA references.

A first approach is given in [18], where an OPC UA reference type `HasAMLUAReference` is defined. However, its use is tailored to specific inter-model-relationships that are typical for AutomationML documents. This reference type is a directly usable sub-type of the standard OPC UA reference type `NonHierarchicalReferences`. It is used to refer from an OPC UA node that stems from an AutomationML model to an OPC UA model of another domain. Its main use case is in the automatic conversion of `ExternalDataConnectors` that may occur within AutomationML documents and that refer to external data that is stored in separate non-AutomationML files. Fig. 3 depicts an AutomationML document that refers via a specialization of the external data connector, the `B2mmlReference`, to a B2MML document containing ISA-95 information. A transformation of this AutomationML document to OPC UA would generate (at the bottom of the information model) a set of AutomationML nodes, and (at the top of the information model) a set of ISA-95 nodes. The `B2mmlReference` would be transformed as a `HasAMLUAReference` reference pointing from the AutomationML node to the corresponding ISA-95 node.

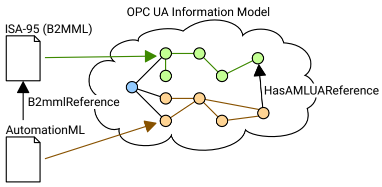


Fig. 3. OPC UA nodes of different domains deployed in a OPC UA server, and an explicit relation between them.

The semantics of `HasAMLUAReference` is simple: there is a relation from an AutomationML node to a non-AutomationML node [18]. This loose definition imposes rather weak application constraints that allow for great flexibility of this reference type, but at the same time provide only little additional knowledge about the kind of relation between nodes that participate in such a relation. Also, it is on one side restricted to the AutomationML domain. Therefore, we propose a set of reference types that allow the modeling of more precise semantics in OPC UA servers that contain overlapping multi-domain nodes.

B. Explicit References

In order to relate nodes of different domains to each other, various relation come into one's mind: *equivalence*, *refinement*, *part-of*, *specialization-of*, etc. Inter- and intra-model relations in the context of requirements engineering have been materialized in [39]. As the intra-model relations should be normally supported by the corresponding domain model, we omit them and find the following four inter-model relations: *refines*, *satisfies*, *tracedFrom* and *verifies* (*tracedFrom* is a very generic relation that has no clear semantics and is left out in our work). In [29], the following relations have been defined: *refines*, *equivalent-to* and *satisfies*. In their work, they have been using these relations for dealing with inconsistencies between different domain models. This is not the focus of this work, where we want to establish relations between potentially consistent models, however, as we will see the set of references is very similar; they same is valid with the previously mentioned relations defined in [39]. Finally, the following relations are identified (cf. Fig. 4):

RepresentsDifferentView This symmetric relation expresses that two nodes represent the same (logical or physical) entity, but from a different engineering point of view. It is a sub-type of `NonHierarchicalReferences`. Application scenarios for this reference type include: (i) a node representing a static design model related to a node representing a live object that is continuously updated at runtime, (ii) a node representing a robot from a plant planning perspective related to the same robot from a "behavior-teaching" perspective, (iii) a node representing a product from a sales perspective related to the same product from a production perspective, (iv) a digital twin related to its physical counterpart[1].

HasRefinement This asymmetric relation expresses that two nodes represent the same (logical or physical) entity, but in different levels of detail. It is a sub-type of `RepresentsDifferentView`. Its inverse relation is named `IsRefinementOf`. An application scenario for this reference type is a node representing a person modeled in AutomationML related to a node representing the same person, but modeled in ISA-95 with much more details, e.g., corresponding personnel classes and their qualification test specification, the qualification test results of this person, etc.

HasVerification This asymmetric relation expresses that one nodes verifies another one, such as (i) a test case verifying a requirement or (ii) operations data verifying the accuracy of a simulation model. It is a sub-type of `NonHierarchicalReferences`. Its inverse relation is named `IsVerificationOf`.

HasImplementation This asymmetric relation expresses that one entity represents an implementation of another entity. It is a sub-type of `NonHierarchicalReferences`. Its inverse relation is named `IsImplementationOf`. An application scenario could be the implementation of PLC code running on a PLC, i.e., a node representing an abstract declaration of PLC code modeled in AutomationML and a node representing the actual definition of that code modeled in PLCopen. Another scenario could be the relation between a node representing a requirement and one representing an engineering artifact that fulfills this requirement (which would be an instantiation of the "satisfies" relation from [39]).

[1]The term "physical counterpart" might be a bit misleading when talking about OPC UA information models. However, imagine an application in which a digital twin is deployed that simulates the physical operations and that can be used to detect anomalies, e.g., based on differences between the predicted/simulated values and the actual values (that stem from the physical counterpart).

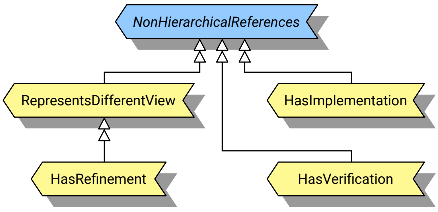


Fig. 4. Reference types for describing inter- and intra-model relations.

C. Evaluation

We are evaluating our set of reference types on the following set of domain models: AutomationML, ISA-95 and MTConnect [40], as depicted in Fig. 5. AutomationML is typically used to model the equipment used in manufacturing environments, such as the machinery/robots/tools, controllers and communication links. ISA-95 provides facilities for modeling the same kind of information, but it is not as flexible in its modeling as AutomationML; but it allows modeling explicit domain knowledge such as processes, and production steps, as well as runtime data such as schedules, performance data and capability/capacity information. MTConnect is a communication protocol for the shop floor based on HTTP and XML. It is trimmed for communication with machine tools and provides a rich vocabulary for this kind of equipment.

For each of these domain models there exists a mapping to OPC UA, or they are natively designed in OPC UA. Typically, an MTConnect-aware machine tool such as a milling device would be modeled in an OPC UA information model three times: (i) from the MTConnect perspective, (ii) as an AutomationML system unit class and (iii) as an ISA-95 physical asset class. Usually, modeling conventions such as naming policies or specific attributes and their values would be needed in order to allow a reasoner to find out that a specific node has corresponding nodes hanging out in the same OPC UA space. However, by using explicit links, this knowledge can be *persisted* and used by other tools which are not introduced to the modeling conventions in use.

Fig. 5 depicts a milling device named `MyMilling` that is represented three times in the OPC UA server: as an AutomationML internal element, as an ISA-95 physical asset and as an MTConnect device. Using the `RepresentsDifferentView` reference type, it can be made clear, that the different OPC UA nodes are really all describing the same physical entity, but from their domain specific perspective. A similar example is given with the entity `MyWorker` that is modeled in AutomationML only as a stub element (usually personnel is not modeled in AutomationML), but modeled in more detail in ISA-95. This refinement is expressed using a `HasRefinement` relation.

It turns out that there are use cases in which such explicit links make sense even within a single domain, e.g., if the domain model does not provide a facility to describe the relations discussed in this work. Consider a manufacturing company that requires a very specific machine "MyMachine" in order to produce their products. Imagine, that this machine is not readily available but needs to be tailor-made and that this manufacturer has its own workshop in which it can produce such machines. Given that ISA-95 is used to model this machine, it will need to be instantiated two times: (i) when it is produced by the internal workshop, it is an entity of the *material* information model, but (ii) as soon as it is used for producing the products of the company, it becomes an entity of the *physical asset* information model, implementing a specific equipment role. ISA-95 does not provide modeling support for this circumstance, but our `RepresentsDifferentView` reference type can be used to implement this relation (cf. Fig. 6).

V. Critical Discussion

It is possible (and likely) that the presented set of OPC UA reference types is not enough for specific application scenarios. In such a case new reference types need to be introduced that are related to the reference types presented in this work, or they could be independent from them. In the use cases that we have explored in the evaluation, the relations that have been defined here were sufficient to model general inter- and intra-model relations that provide additional knowledge that proofs useful for OPC UA client applications.

The example given in Fig. 5 exemplifies the usefulness of our approach: the reference type `HasAMLUAReference` that is defined in [18] relates two nodes with each other, on the basis of an `ExternalDataConnector`. However, due to the modeling constraints imposed by [41], an AutomationML entity needs to create a child internal element representing a document that in turn defines an external interface that resembles the relation to the external entity. The result of this regulation is, that the `HasAMLUAReference` relates the *document* entity with the external entity. In some cases (e.g., in the cases described in [19]) the external reference depicts a correspondence of the *parent* internal element of the document with the external entity. This correspondence can be modeled in terms of a `RepresentsDifferentView` reference between the structurally and semantically better matching OPC UA nodes.

According to this, we believe that the proposed reference types may provide additional information to an observer by resembling semantically enriched hyperlinks between corresponding OPC UA nodes.

VI. Conclusion

We have presented an OPC UA node set comprising domain-agnostic reference types that can be used to model inter- and intra-model relations. The goal of our work is to make explicit some rather generic types of information in order to reach a more complete picture of the model(s) under observation. In the context of OPC UA, models of multiple engineering domains can emerge in the melting pot that is an OPC UA server. It is very useful to provide an understanding of close relations between entities in order to

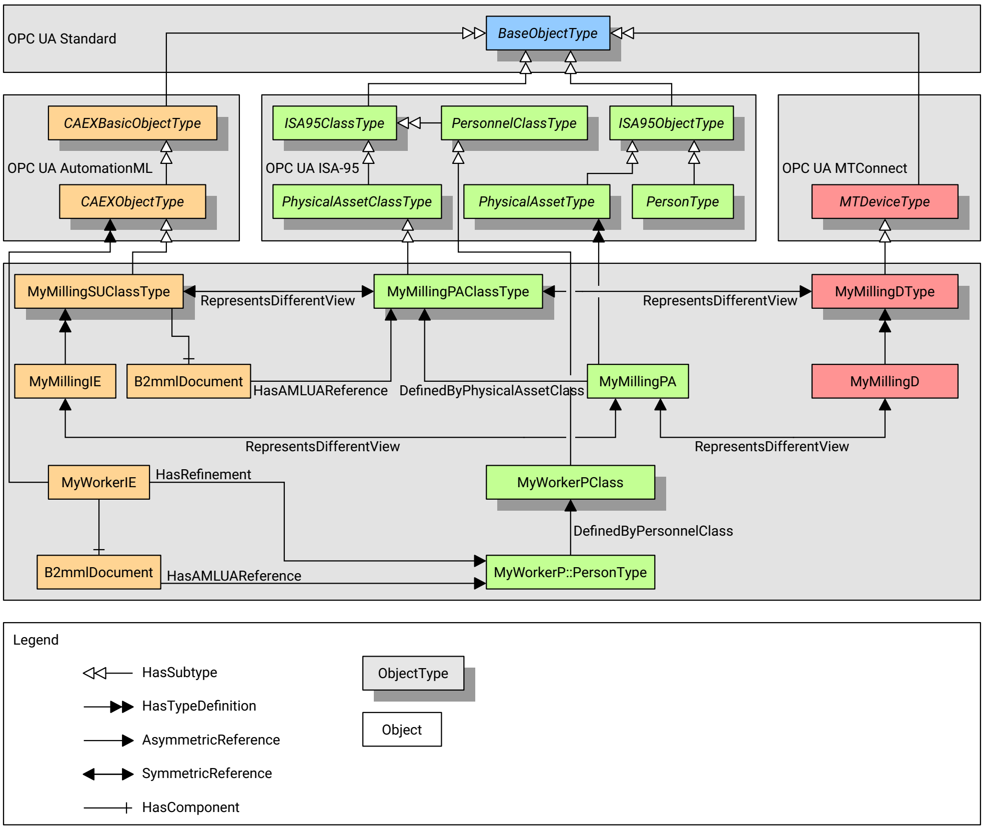


Fig. 5. OPC UA information model of a milling device (`MyMilling*`) and a worker, seen from 3 (or 2 in the case of `MyWorker`) different views. Entities are suffixed with an abbreviation of the corresponding metamodel class (`IE`: internal element, `SU`: system unit, `PA`: physical asset, `P`: person, `D`: device). Some standard relations have been left out to make the diagram less cluttered.

allow automated tools or tools that require a human-in-the-loop to, e.g., unerringly select the correct view of an entity to execute an action, generate a report or display data in a user interface: the `HasImplementation` reference type could, e.g., be used to navigate between planning/design artifacts and implementation/operations artifacts.

Future work will further investigate application scenarios for the reference types defined in this work and strive for either extending or stabilizing the set of relations. One specific topic of interest is the linking of shop floor entities to business objects in order to foster vertical tool integration. Another stream of research might look into the interplay of multiple OPC UA servers.

Acknowledgment

The financial support by the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development is gratefully acknowledged.

References

[1] W. Mahnke, S.-H. Leitner, and M. Damm, *OPC Unified Architecture*. Springer-Verlag Berlin Heidelberg, 2009.
[2] International Electrotechnical Commission (IEC), *OPC Unified Architecture—Part 1: Overview and Concepts*, Std., Rev. 1.0, 2010, IEC 62541-1:2010.
[3] P. Coad, "Object-oriented patterns," *Communications of the ACM*, vol. 35, no. 9, pp. 152–159, 1992.
[4] R. Johnson and B. Woolf, "Type object," in *Pattern Languages of Program Design 3*. Addison-Wesley Longman Publishing Co., Inc., 1997, ch. Type Object, pp. 47–65.
[5] Object Management Group (OMG), *OMG Unified Modeling Language (OMG UML)*, Specification, Rev. 2.5, 2015.
[6] International Electrotechnical Commission (IEC), *Representation of process control engineering—Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools*, International Standard, Rev. 1.0, 2008, IEC 62424:2008.

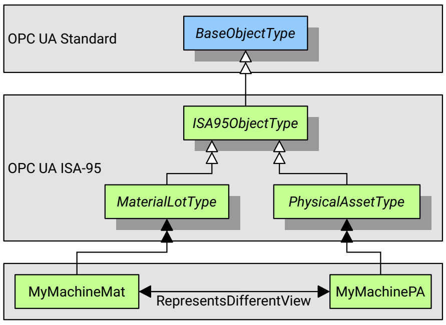


Fig. 6. OPC UA information model of a specific machine "MyMachinePA" (a physical asset required in the production process) that was internally produced as material lot "MyMachineMat".

[7] International Electrotechnical Commission (IEC), *Engineering data exchange format for use in industrial automation systems engineering—Automation markup language—Part 1: Architecture and general requirements*, International Standard, Rev. 1.0, 2014, IEC 62714-1:2014.

[8] International Electrotechnical Commission (IEC), *Engineering data exchange format for use in industrial automation systems engineering—Automation markup language—Part 2: Role class libraries*, International Standard, Rev. 1.0, 2015, IEC 62714-2:2015.

[9] AutomationML consortium, *AutomationML Whitepaper Part 3—Geometry and Kinematics*, Whitepaper, Rev. 2.0, 2015, AML Part 3:2015.

[10] PLCopen Technical Committee 6, *XML Formats for IEC 61131-3*, Technical Paper, Rev. 2.01, 2009, PLCopen XML:2009.

[11] International Electrotechnical Commission (IEC), *Programmable controllers—Part 3: Programming languages*, Std., Rev. 3.0, 2013, IEC 61131-3:2013.

[12] AutomationML consortium, *AutomationML Whitepaper Part 4—AutomationML Logic Description*, Whitepaper, Rev. 2.0, 2010, AML Part 4:2010.

[13] International Electrotechnical Commission (IEC), *Enterprise-control system integration—Part 1: Models and terminology*, International Standard, Rev. 2.0, 2013, IEC 62264-1:2013.

[14] International Electrotechnical Commission (IEC), *Enterprise-control system integration—Part 2: Objects and attributes for enterprise-control system integration*, International Standard, Rev. 2.0, 2013, IEC 62264-2:2013.

[15] Manufacturing Enterprise Solutions Association (MESA) International, *Business To Manufacturing Markup Language*, Standards and Tools, Rev. V0600, 2013.

[16] OPC Foundation, *OPC Unified Architecture for ISA-95 Common Object Model*, Companion Specification, Rev. 1.00, 2013.

[17] R. Henßen and M. Schleipen, "Interoperability between OPC UA and AutomationML," in *Proceedings of the 8th International Conference on Digital Enterprise Technology (DET 2014)*, 2014.

[18] Deutsches Institut für Normung (DIN), *Combining OPC Unified Architecture and Automation Markup Language*, DIN SPEC, 2016, DIN 16592:2016-12.

[19] B. Wally, *Provisioning for MES and ERP—Support for IEC 62264-2 and B2MML*, Application Recommendation, Rev. 2.0.0, 2018. [Online]. Available: https://www.automationml.org/

[20] B. Wally, C. Huemer, and A. Mazak, "A view on model-driven vertical integration: Alignment of production facility models and business models," in *Proceedings of the 13th IEEE International Conference on Automation Science and Engineering (CASE 2017)*, 2017.

[21] W. E. McCarthy, "The REA accounting model: A generalized framework for accounting systems in a shared data environment," *The Accounting Review*, vol. 57, no. 3, pp. 554–578, 1982.

[22] B. Wally, M. Schleipen, N. Schmidt, N. D'Agostino, R. Henßen, and Y. Hua, "AutomationML auf höheren Automatisierungsebenen," in *Proceedings of AUTOMATION 2017*, 2017.

[23] B. Wally, C. Huemer, and A. Mazak, "Entwining plant engineering data and ERP information: Vertical integration with AutomationML and ISA-95," in *Proceedings of the 3rd IEEE International Conference on Control, Automation and Robotics (ICCAR 2017)*, 2017.

[24] B. Suryajaya, C. C. Chen, M. H. Hung, Y. Y. Liu, J. X. Liu, and Y. C. Lin, "A fast large-size production data transformation scheme for supporting smart manufacturing in semiconductor industry," in *Proceedings of the 13th IEEE International Conference on Automation Science and Engineering (CASE 2017)*, 2017, pp. 275–281.

[25] B. Wally, C. Huemer, and A. Mazak, "Aligning business services with production services: The case of REA and ISA-95," in *Proceedings of the 10th IEEE International Conference on Service Oriented Computing and Applications (SOCA 2017)*, 2017.

[26] B. Wally, C. Huemer, and A. Mazak, "ISA-95 based task specification layer for REA in production environments," in *Proceedings of the 11th International Workshop on Value Modeling and Business Ontologies (VMBO 2017)*, 2017.

[27] B. Wally, A. Mazak, B. Kratzwald, and C. Huemer, "Model-driven retail information system based on REA business ontology and Retail-H," in *Proceedings of the 17th IEEE Conference on Business Informatics (CBI 2015)*, vol. 1, 2015, pp. 116–124.

[28] J. Bézivin, S. Bouzitouna, M. D. Del Fabro, M.-P. Gervais, F. Jouault, D. Kolovos, I. Kurtev, and R. F. Paige, "A canonical scheme for model composition," in *Proceedings of the 2nd European Conference on Model Driven Architectur—Foundations and Applications (ECMDA-FA 2006)*, ser. LNCS, no. 4066. Springer-Verlag Berlin Heidelberg, 2006, pp. 346–360.

[29] S. Feldmann, M. Wimmer, K. Kernschmidt, and B. Vogel-Heuser, "A comprehensive approach for managing inter-model inconsistencies in automated production systems engineering," in *Proceedings of the 12th IEEE International Conference on Automation Science and Engineering (CASE 2016)*, 2016, pp. 1120–1127.

[30] M. Sabou, F. Ekaputra, O. Kovalenko, and S. Biffl, "Supporting the engineering of cyber-physical production systems with the AutomationML analyzer," in *Proceedings of the 1st International Workshop on Cyber-Physical Production Systems (CPPS 2016)*. IEEE, 2016.

[31] K. Kernschmidt and B. Vogel-Heuser, "An interdisciplinary SysML based modeling approach for analyzing change influences in production plants to support the engineering," in *Proceedings of the 9th IEEE International Conference on Automation Science and Engineering (CASE 2013)*. IEEE, 2013, pp. 1113–1118.

[32] A. Lüder, M. Schleipen, N. Schmidt, J. Pfrommer, and R. Henßen, "One step towards an industry 4.0 component," in *Proceedings of the 13th IEEE Conference on Automation Science and Engineering (CASE 2017)*, 2017, pp. 1268–1273.

[33] Deutsches Institut für Normung (DIN), *Referenzarchitekturmodell Industrie 4.0 (RAMI4.0)*, DIN Standard, 2016, DIN 91345:2016-04.

[34] PLCopen and OPC Foundation, *OPC UA Information Model for IEC-61131-3*, Companion Specification, Rev. 1.00, 2010, PLCopen-OPC:2010.

[35] S. Biffl, E. Mätzler, M. Wimmer, A. Lüder, and N. Schmidt, "Linking and versioning support for AutomationML: A model-driven engineering perspective," in *Proceedings of the 13th IEEE International Conference on Industrial Informatics (INDIN 2015)*, 2015, pp. 499–506.

[36] OPC Foundation and CC-Link Partner Association, *OPC Unified Architecture for Control & Communication System Profile (for Machine)*, Companion Specification, Rev. 1.00, 2017, OPC-CSP+:2017.

[37] OPC Foundation and Ethernet POWERLINK Standardization Group, *OPC Unified Architecture POWERLINK*, Companion Specification, Rev. 1.00, 2017, OPC-POWERLINK:2017.

[38] German Machine Tool Builders' Association (VDW) and OPC Foundation, *OPC UA Information Model for CNC Systems*, Companion Specification, Rev. 1.00, 2017, OPC-CNC:2017.

[39] Object Management Group (OMG), *OMG Systems Modeling Language*, Std., Rev. 1.5, 2017.

[40] MTConnect Institute and OPC Foundation, *MTConnect OPC UA*, Companion Specification, Rev. 1.02, 2013, OPC-MTConnect:2013.

[41] AutomationML consortium, *AutomationML Best Practice Recommendation—External Data Reference*, Best Practice Recommendation, Rev. 1.0.0, 2016, AML BPR-EDR:2016.

4 Leveraging integration facades for model-based tool interoperability

G. Paskaleva, A. Mazak-Huemer, M. Wimmer and T. Bednar;
Journal Automation in Construction, Elsevier, 128, (2021), pp. 103689.
DOI: 10.1016/j.autcon.2021.103689

Automation in Construction 128 (2021) 103689

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

Leveraging integration facades for model-based tool interoperability

Galina Paskaleva [a,*,1], Alexandra Mazak-Huemer [a,2], Manuel Wimmer [b,3], Thomas Bednar [c,4]

[a] Subsurface Engineering - Digital Transformation in Tunnelling, Montanuniversität Leoben, Erzherzog-Johann Strasse 3, 8700 Leoben, Austria
[b] Institute of Business Informatics - Software Engineering, Johannes Kepler Universität Linz, Altenberger Straße 69, Science Park 3, 4040 Linz, Austria
[c] Research Unit of Building Physics, Institute of Material Technology, Building Physics and Building Ecology, Faculty of Civil Engineering, TU Wien, Karlsplatz 13/207/2, 1040 Vienna, Austria

ARTICLE INFO

Keywords:
MDE
Data exchange
Big Open BIM
Semantic integration
Pragmatic integration
Heterogeneity

ABSTRACT

Data exchange and management methods are of paramount importance in areas as complex as the Architecture, Engineering and Construction industries and Facility Management. For example, Big Open BIM requires seamless information flow among an arbitrary number of applications. The backbone of such information flow is a robust integration, whose tasks include overcoming technological as well as semantic and pragmatic gaps and conflicts both within and between data models. In this work, we introduce a method for integrating the pragmatics at design-time and the semantics of independent applications at run-time into so-called "integration facades". We utilize Model-driven Engineering for the automatic discovery of functionalities and data models, and for finding a user-guided consensus. We present a case study involving the domains of architecture, building physics and structural engineering for evaluating our approach in object-oriented as well as data-oriented programming environments. The results produce, for each scenario, a single integration facade that acts as a single source of truth in the data exchange process.

1. Introduction

Today, the integration of distributed, autonomous and heterogeneous data sources across application boundaries is gaining importance due to the increasing networking of organizations and companies in many application fields. This leads to the situation that the information flow goes hand in hand with the translation among different data models with different syntax and semantics. A current example for this evolution is the domain of Architecture, Engineering and Construction (AEC) as well as Facility Management (FM) industries. In these industries' daily business, the volume of data to be exchanged among various stakeholders (e.g., building developers, energy suppliers, architects, structural engineers, building physicists, etc.) is rapidly increasing. Each of these stakeholders possesses specific domain knowledge and has a specific view of the project. These different perspectives may cause different forms of heterogeneity in the definition and handling of data, which hinders an interference-free communication within a building project.

This heterogeneity can be syntactic, semantic, pragmatic or a difference in the level of detail, i.e., granularity. *Syntactic heterogeneity* is caused by working with different data models in different tools. One example of *semantic heterogeneity* is the concept of homogeneous unit in the description of the building ground in different norms and regulations. The Austrian guideline[5] defines it on the basis of homogeneous substrate, the Swiss norm[6] - on the basis of similar structural behavior. *Pragmatic heterogeneity* occurs, for instance, when in the absence of a clear regulation, one architect considers slabs with inclination of more than 15% as walls and another with inclination of more than 25%. An example of *heterogeneity based on diverging levels of detail* can be found in the following scenario. The architecture domain relies on geometry not

* Corresponding author.
E-mail addresses: galina.paskaleva@tuwien.ac.at (G. Paskaleva), alexandra.mazak-huemer@unileoben.ac.at (A. Mazak-Huemer), manuel.wimmer@jku.at (M. Wimmer), thomas.bednar@tuwien.ac.at (T. Bednar).
1 www.bph.tuwien.ac.at/no_cache/team/assistentinnen/staffmembers/staff/detail/paskaleva/.
2 www.subsurface.at/.
3 www.se.jku.at/manuel-wimmer/.
4 www.bph.tuwien.ac.at/team/professorinnen/staff/show/person/bednar/.
5 https://www.oegg.at/upload/download/downloads/geotech_rili_08_2.1.pdf (last accessed 2021-02-26).
6 https://www.beuth.de/de/norm/sn-531199/246688888 (last accessed 2021-02-26).

https://doi.org/10.1016/j.autcon.2021.103689
Received 21 January 2020; Received in revised form 28 February 2021; Accepted 26 March 2021
Available online 5 May 2021

only as a representation of a building project, but also as an information carrier, whereas the building physics domain is concerned with thermal and hygric flux through the construction of a building. For this reason, building physicists consider points of interest (e.g., the joints between walls) in much greater detail than architects do.

In addition to data heterogeneities, the data exchange within or across different phases of a project is challenging, since there are different exchange standards used as well as various requirements that must be considered.

To overcome these obstacles, *Building Information Modeling (BIM)* has become more and more established in recent years. The main motivation behind BIM is to accommodate the heterogeneous nature of the AEC industries and involved domains, and to provide seamless data flows within any building or infrastructure project. To put it in a nutshell, BIM aims to support the generation and management of digital representations of physical and functional characteristics of a built structure throughout all phases. The ultimate goal of this development is *Big Open BIM*, a method for loss- and distortion-free, possibly real-time, data exchange across technological spaces and domains [7].

Preliminaries to BIM: The realization of BIM is not limited to a single data exchange standard. In fact, there have been multiple attempts to develop a suitable standard for it. For example, the proprietary Drawing Exchange Format (DXF)[7] standard was widely used in the 1980s and 1990s, and is currently still in use. Furthermore, the open Standard for the Exchange of Product model data (STEP)[8] was developed in the 1980s and, subsequently, became an ISO standard (ISO 10303).[9] The Collaborative Design Activity (COLLADA)[10] standard includes the exchange of geometric constraints and animations and is used not just in the AEC industries, but also in the automated production industry as part of the industry standard Automation ML.[11] The Construction Operations Building Information Exchange (COBie) [11] was developed by the US military in 2007 and aims to store construction related data during all phases of a building's life cycle. However, the currently most widely used BIM data exchange standard is the Industry Foundation Classes (IFC) standard [6]. The aim of IFC is to support Big Open BIM by providing a seamless communication among all stakeholders within the AEC domain. Therefore, it offers very abstract (e.g., *IfcObjectDefinition*) as well as very specific (e.g., *IfcBoiler*) concepts to describe a domain of interest.

A deeper look at IFC: Currently, IFC provides semantic types for the following domain groups: Building Controls, Plumbing and Fire Protection, Structural Elements, Structural Analysis, Heating Ventilation Air-Conditioning (HVAC), Electrical, Architecture, and Construction Management [6]. Building physics, which we mentioned above, is not regarded as a domain, but as a *resource*. It is partially covered by module *8.10 Material Resource*. Other sub-domains of building physics, e.g., acoustics, have not been integrated yet. Attempts to extend the standard or to define appropriate property and quantity sets for energy simulation tools have already been made, for example, by Chen et al. in [8] and by Bracht et al. in [3]. Similarly, domains involving underground facilities, such as tunnels, are yet to be fully developed [2,29]. Even the domains with good coverage cannot be regarded as semantically complete, because new technologies and methods enter the industry at a pace the IFC development cycle cannot keep up with. For example, the HVAC semantic types do not contain elements for thermal mass activation as a method of heating, e.g., for floor heating. These examples outline only some of the major challenges even a widely adopted and rigorously developed BIM standard, such as IFC, has to face on the road to Big Open BIM.

BIM Tools: There are software tools developed specifically for a particular domain (e.g., C.A.T.S.,[12] DDS-CAD,[13] Autodesk AutoCAD Mechanical[14] and Electrical[15] for the building services engineering field, Dlubal RFEM,[16] Tekla Structures[17] and AXISVM[18] for the structural engineering field), or for multiple domains (e.g., Autodesk REVIT,[19] Neme-tschek AllPlan[20] and ArchiCAD[21]). Each of these listed tools has its own, in most cases, closed data model. It is to be noted that these are only some of the most widely used software tools. In addition to them, there are many more, most of them dedicated to performing only a small subset of tasks within a single domain or project. However, in order to be "BIM-ready", each and every software tool needs to implement some form of BIM.

1.1. Problem statement

Due to its very active development, wide usage and continuing efforts to integrate more and more domains, IFC presents a particularly suitable example for exploring the still existing technical challenges and missing links towards a realization of Big Open BIM via data exchange standards that honor the multiple types of heterogeneity we mentioned above. In the following, we outline three technical challenges and some practical examples as an illustration of their practical implications (see also Table 1):

Table 1
Problem statement outline: types of heterogeneity that require integration.

No	Heterogeneity	Features	Issues	Integration implications
1	Semantic	Semantic discrepancy between domains or regulations	Can contain explicit and implicit, formal and informal specifications	The implicit and informal parts can lead to errors
2	Pragmatic	Implicit assumptions	Cannot be included in an integration specification	Lead to errors
3	Syntactic	Heterogeneity between standards, seldom within the same standard	Need for consistency checks	Easiest to automate, least potential to produce errors
4	Semantic modeling	Interlocking of syntax and semantics due to deficits in the tools for semantic modeling	The semantics becomes syntax-dependent	A change of syntax during data exchange can lead to a hidden change in semantics

[7] https://www.autodesk.com/techpubs/autocad/acad2000/dxf/ (last accessed 2020-11-13).
[8] https://www.steptools.com/stds/step/ (last accessed 2020-11-13).
[9] https://www.iso.org/standard/55257.html (last accessed 2020-11-13).
[10] https://www.khronos.org/collada/ (last accessed 2020-11-13).
[11] https://www.automationml.org/o.red.c/home.html (last accessed 2020-11-13).
[12] http://www.cats-software.com (last accessed 2020-11-13).
[13] https://www.dds-cad.net/ (last accessed 2020-11-13).
[14] https://www.autodesk.com/products/autocad/included-toolsets/autocad-mechanical (last accessed 2020-11-13).
[15] https://www.autodesk.com/products/autocad/included-toolsets/autocad-electrical (last accessed 2020-11-13).
[16] https://www.dlubal.com/en/products/rfem-fea-software/what-is-rfem (last accessed 2020-11-13).
[17] https://www.tekla.com/products/tekla-structures (last accessed 2020-11-13).
[18] https://axisvm.eu/index.html (last accessed 2020-11-13).
[19] https://www.autodesk.com/products/revit/overview (last accessed 2020-11-13).
[20] https://www.allplan.com/en/ (last accessed 2020-11-13).
[21] https://www.graphisoft.com/ (last accessed 2020-11-13).

(1) ***Explicit heterogeneity:*** Standards serving multiple domains are complex and rich, both syntactically and semantically. Both the syntax and the semantics are formally specified and, therefore, explicit. Inconsistencies in the syntax within the same standard are easily detected by various programming and validating tools. However, the presence of more than one syntax, as in IFC (the EXPRESS modeling language and XSD) can produce syntactic heterogeneity as well. The semantic specification, on the other hand, can contain, in addition to the formal definitions, informal ones in a natural language. For example, the specification of *IfcWallStandardCase* in the IFC specification [6] contains both a formal definition in the EXPRESS modeling language and an informal definition of the requirements for its geometric representation. Such definitions give room for interpretation, which can produce implicit heterogeneity between separate implementations of the same standard.

(2) ***Implicit heterogeneity:*** Even when working with a consistently implemented data exchange standard, there are still communication difficulties among domain experts due to inconsistencies in the modeling styles, or to different interpretations of the same concept within the standard [16,20,25,27]. This is the pragmatic heterogeneity. It is, by definition, implicit, i.e., without formal specification as it occurs in the mind of the user, and cannot be addressed in a purely automated manner.

(3) ***Interlocking of concerns:*** A feature common to both large and small standards is the interlocking of the syntax and the semantics. This involves the re-purposing of syntactic constructs to express semantics. For example, dynamic typing is often implemented as an object-type pattern that employs a syntactic referencing relationship to express semantic instantiating. This makes the semantics syntax-dependent and can be a hidden error source. We will take an in-depth look into this phenomenon in Section 3.3.

There are multiple practical implications arising from the three challenges listed above.

(a) *Implementation overhead:* The manual translation to and from a standard as semantically complex as IFC is challenging. In practice, for small pieces of software, developed from scratch to perform a single dedicated task (e.g., a thermal flow simulation tool in the building physics domain), there are simply not sufficient resources (e.g., finances, person hours) for such implementations. This leads to a large number of small tools offering very similar, often state-of-the-art, functionalities that are never used beyond the limits of one project. This results in the loss of domain expert knowledge for the AEC communities.

(b) *No real-time feedback:* Communication via data exchange standards often involves serialization of large amounts of data, which does not allow real-time feedback. The delay between user action and observable results is not measured in milliseconds, but in minutes or hours. For instance, this makes the fine-tuning of a building simulation extremely tedious and time-consuming.

(c) *No single source of truth:* The implicit heterogeneity we described above produces divergence in semantics between implementations and between models. Therefore, instead of a single "source of truth" there are multiple competing ones. This invariably results in translation errors when transferring information from one implementation or model to another [7,20], even within the same software family, e.g., exchanging walls with wall modifiers as tested by us.

To sum up, the practical implications listed above illustrate that reaching semantic as well as pragmatic consensus is a challenge. For better understanding of the issues involved as well as their interdependency and for motivating our approach as well, we present a practical example of the modeling of a wall in the following section, which also accompanies us as a running example throughout the paper. In Section 2.2.1 we will once again return to these issues and formulate three separate technical challenges, for which we will present our solution in the subsequent sections.

1.2. Motivating example

Let us consider the process of exchanging information about a wall between an architect and a structural engineer. Both model the same object, but consider different aspects of it. For the architect (and the building physicist) the wall is a layered construction with thermal, hygric, fire safety and other properties. For the structural engineer the wall is a structural member with a structural behavior within a system. In the IFC standard, there are elements that aid each of them in their modeling task, *IfcWallStandardCase* and *IfcStructuralSurfaceMember*, respectively. However, there is no formal mechanism for establishing that both concepts can describe different aspects of the same object, i.e., no possibility for effective integration. This is the semantic heterogeneity. In practice, each domain expert typically works in her own tool and the information exchange is relegated to a *BIM Collaboration Format (BCF)* file[22] referencing their respective IFC models.This is a topic also addressed by a data-driven method in the work of [28], which concentrates exclusively on the domains of architecture and structural engineering. The pragmatic heterogeneity occurs due to the assumptions of both stakeholders based on their respective point of view. For example, the structural engineer may view the two semantic concepts as complementary categories of the same object, whereas the architect may regard the one concept as an additional representation of the other. Since these points of view are not formally expressed, the difference is hidden and may give the impression of consensus where there isn't one.

In summary, this example involves both semantic and pragmatic heterogeneity. We will examine different solutions to the integration challenge in Section 3.3.

1.3. Contribution

Interoperability in BIM requires integration along both the semantic and the pragmatic dimensions. In this work, we present and evaluate a modeling framework capable of working with multiple semantic type systems inhabiting the same syntax in the context of multiple applications. In addition, the framework provides tools for modeling the pragmatic aspects of a data exchange and converting them from an implicit assumption into an explicit formal specification. The combined model of the semantics and pragmatics of a data model provides an *integration facade* that enables transparency and traceability during interoperability.

The paper is structured as follows. In Section 2, we review the related work and discuss open challenges we face. Section 3 presents our framework step by step by following the workflow for building an integration facade. In this section we describe relevant methods and techniques we employ for the realization of the framework as a prototypical implementation. Section 4 describes the design, evaluation, and results of our conducted case study on the basis of three practical cases. Section 5 concludes this work and outlines future work.

2. Related work

Much of the work on data exchange standards concentrates on bridging legacy, domain or technological gaps. There are several strategies for the implementation of such exchange that have been explored in the past, as shown in the following subsection.

[22] https://technical.buildingsmart.org/standards/bcf/ (last accessed 2021-02-26).

2.1. Data exchange strategies

Here we differentiate between data exchange strategies based on the type of "bridge" offered to cross a certain communication gap. An overview of those strategies is given in Table 2.

2.1.1. Data exchange via common syntax

Data exchange can take place via syntactic containers. Widely used formats employing this strategy include the Co-mma-separated Values (CSV), Extensible Markup Language (XML), the XML Metadata Interchange XMI and the Java Script Object Notation (JSON)–see the entry *common syntax* in Table 2. These formats are often used as storage for arbitrary data, because they offer no semantics of their own. Therefore, there is no interference with the semantics of the stored data. This makes these formats applicable for all domains. Moreover, transformations between them requires establishing only syntactic correspondences, not semantic ones. This transformation task can be automated easily. However, for processing the stored information correctly, each of the interacting applications must have implemented not only the same semantics but, in most cases, also the same serialization routine, e.g., a mandatory ordering of the information.

An example for the strategy of data exchange via common syntax is demonstrated in the Epsilon project [31], in the context of integrating legacy models into new technologies. In the course of this project, Paige et al. [23] implement interoperability between a proprietary modeling tool and an open source model management suite by providing a layer of communication drivers between a Java-based execution engine and a Component Object Model (COM) interface. This layer contains a dedicated driver for each persistence format (e.g., EMF,[23] XML, spreadsheets, etc.). In essence, it demonstrates a technique for loading and manipulating the same semantics by extracting its artifacts from different syntactic containers. Building upon this, the authors use a similar approach in another study, on the integration of Google Spreadsheets coupled with XML-based configuration models. In this study, they bridge the gap between the object-oriented and the table-row-column-oriented (or data-oriented) paradigms via a syntactic translation approach [13]. They do not need to consider the semantics of these formats (since there is none) in order to perform the translation.

2.1.2. Data exchange via common semantics

Implementing not only (*i*) the same semantics, but also (*ii*) the same serialization routine, becomes impractical for the extensive semantics typical for the AEC industries. In order to avoid (*ii*), the data exchange format has to incorporate semantic information in addition to the syntax. An example for this approach is the transfer of geographic data in the domain of urban design. The GeoJSON format builds on the purely syntactical JSON format by adding semantics-carrying keywords to its syntactic structures, e.g., *Feature*, *Point*, *Polygon*, etc. The DXF format, which transfers only geometric information, encodes the geometric semantics in context dependent numerical keys. For example, the key 0 indicates the beginning of an object definition. The key 10 in the context of a line indicates the x-coordinate of its first point in the context of a circle. This means it indicates the x-coordinate of its center. The IFC format also belongs to this category (see entry *common semantics* in Table 2). We already discussed some of its features in Section 1. It has two major advantages over the standards operating on pure syntax: It contains expert knowledge and is maintained by domain experts. This results in a certain level of complexity, which has practical implications. Its textual serialization, specified in ISO 10303-21,[24] contains a keyword for each entity, type, property or quantity set. Thus, for version IFC4 and above there exist over 1500 such keywords. An excerpt of this definition is shown below:

```
simple_entity_instance = entity_instance_name '='
simple_record ';'
simple_record = keyword '(' [ parameter_list ] ')'
entity_instance_name = '#' digit { digit }
```

A major challenge when exchanging data via common semantics is the translation between the common semantics and the internal data model of each application. Even when the data exchange standard is open, this translation remains hidden for the user and this may cause unintended results. For example, if the data exchange standard does not know a specific concept, it cannot transfer it. As an illustration of this drawback let us consider the following scenarios: First, the domain of urban design incorporates concepts for many plants. The IFC standard, however, does not. Therefore, each plant defined in an urban development tool is translated to *IfcProxy* in IFC4, which acts as a placeholder for unknown entities [5]. This loss of information cannot be remedied, even if the data exchange takes place between tools whose internal semantics incorporate the concept of plants. Second, a similar problem occurs when the data exchange standard's level of detail in the description of a concept differs from that of the internal data model of one of the interacting applications. For example, IFC4 differentiates between the material layers in a wall construction. Tools for calculating thermal flux within a wall construction, however, also differentiate between layers within the same material layer. If the translation from the tool to the standard uses the maximum flux value over all layers within a material layer, it will produce different results from a translation that takes the average flux value over all layers within a material layer.

There are even more complex translation problems, when no clear correspondence between data model elements can be established. For example, if the data exchange standard knows only the triangular mesh as the geometric representation of a surface, a Computer Aided Design (CAD) tool that works with Non-Uniform Rational B-Spline (NURBS) surfaces will have to perform triangulation before passing the surface information to the data exchange standard mesh object. Depending on the parameters and constraints of the triangulation, the resulting meshes of the same surface can differ so much between exports that they cannot be reliably identified as representing the same surface [10].

Ontologies: Operating on semantics brings us to the topic of the application of an ontology in the data exchange. IFC itself is based on an ontology, the buildingSmart Data Dictionary.[25] This approach was chosen by Akanbi et al. to address the problem of proprietary data formats resulting in errors and missing data during information exchange for automated cost estimation when employing different BIM software tools (compare to the second limiting factor under entry *common semantics* in Table 2). In [1], the authors present a novel data-driven method for automated quantity takeoff (QTO), since current methods do not address QTO from BIM artifacts created by different tools. For this purpose, they employ a data-driven reverse engineering algorithm to generate QTO algorithms for any building component by covering the variety of BIM representations via a mapping to IFC. The authors state that their approach "is neither an algorithm nor a software but a method for developing interoperable QTO algorithms". They argue that their approach is more robust than (traditional) approaches built on proprietary data formats, and therefore, has a higher level of support for interoperability.

In [25], the authors present an ontology-based approach to support change management as well as traceability of changes throughout all design stages. The authors argue that there are so many approaches presenting standardization methods for different file formats and exchanges, but still the document centric approaches result in parsing, interpretation, and serialization problems. Therefore, their approach builds upon

[23] https://www.eclipse.org/modeling/emf/ (last accessed 2020-11-13).
[24] https://www.iso.org/standard/63141.html (last accessed 2020-11-13).
[25] https://www.buildingsmart.org/users/services/buildingsmart-data-dictionary/ (last accessed 2020-11-13).

Table 2
Comparison of data exchange approaches.

No	Method	Examples	Features	Advantages	Limiting factors
1	Common syntax	CSV[a], XML[b], XMI[c], JSON[d]	Domain-independent, do not contain a specialized semantics. Instead, they offer a structure: a tree, a sequence of containers, etc.	(1) Can store any data and do not interfere with the data semantics. (2) Applicable to all domains. (3) Translations btw. such standards can be easily automated.	(1) The correct interpretation of the data requires that each tool has a separate implementation of a common semantics or the exactly same serialization routine. (2) This can become impractical for large domain-specific standards.
2	Common semantics	GeoJSON[e], DXF, IFC, ontology and description logic	Domain-specific, contain (parts of) the domain's semantics, e.g., a wall, a structural element, etc.	(1) Contain expert knowledge. (2) Can be maintained and extended by domain experts according to the requirements	(1) Depending on the level of detail included in the standard, it can become too large for efficient maintenance. (2) The implementation of the standard involves a translation between the standard and the software's data model, which can be challenging. (3) Missing semantic concepts require workarounds.
3	Copy of reality	digital shadow, digital twin	Multiple domain-specific standards are involved, e.g., architecture, geology, building physics, building automation, etc.	(1) Allows for completely automated information flow btw. real-time and its digital representation. (2) Enables automated as well as continuously decision-making. (3) Provides cross-domain traceability.	(1) Does not allow for abstraction and the digital shadow or twin can become exceedingly large (Tera- or Petabytes in size). (2) The communication between the different domain-specific standards requires a dedicated structure for both hardware and software middleware. (3) Requires a dedicated communications standard.
4	Common knowledge	calculation and simulation methods	Can be based on any of the standards listed above.	(1) Contain operational expert knowledge. (2) Can be used for validation and compliance checking.	(1) Operate on their (often implicit) bespoke data models and are inaccessible for most applications. (2) Implemented in different formal languages and on different platforms.
5	Common data	data base, any domain-independent or domain-specific standard	Can contain any type of monitoring or simulation data, often time-stamped, can be used for automatic semantic classification	(1) Supplies information about the actual state of the corresponding physical object, which can be used by a digital shadow or twin. (2) Arbitrarily fine-grained (values per hour or per millisecond). (3) Data model is rarely complex. (4) Can be reverse-engineered to give insight into an unfamiliar system.	1) Can become very large (Tera- or Petabytes in size). (2) Its structure can be completely arbitrary, and therefore, harder to integrate with any other data exchange standard or model. (3) Can be one-sided and skew the performance of an algorithm based on it.

[a] https://tools.ietf.org/html/rfc4180 (last accessed 2020-11-13).
[b] https://www.w3.org/TR/xmlschema11-1/ (last accessed 2020-11-13).
[c] https://www.omg.org/spec/XMI/ (used as a standard persistence format for the Universal Modeling Language (UML) since 2000 - see https://www.omg.org/spec/UML/About-UML/) (last accessed 2020-11-13).
[d] https://www.ecma-international.org/publications/standards/Ecma-404.htm (last accessed 2020-11-13).
[e] https://tools.ietf.org/html/rfc7946 (last accessed 2020-11-13).

semantic web technologies and ontology engineering. Thereby, they shift the problem of exchanging artifacts to the management of knowledge graphs (ontologies). The authors are convinced that semantic web technologies, such as ontologies, reasoning, and SPARQL[26] queries efficiently manage interrelated project information. Their approach parses building information from IFC files, ontologies from the IFC schema, and rule-sets describing implicit knowledge. In the evaluation they successfully show that calculations as well as inferences computed in different tools by different users can be explicitly described in an interoperable manner. For this purpose, the authors make use of the latest developments of ifcOWL.[27]

When choosing an ontology-based approach for semantic integration, assuming that the domain of interest can be specified in sufficient detail, semantic heterogeneity could be overcome by logical inference. This requires a special class of logics, the so-called description logics. For this purpose a sufficiently detailed ontology has to be present, so that the elements of the export schemata of the involved data sources could be described by logical formulas. Additionally, it has to be possible to represent a global query as a formula covering the concepts that are formally well-defined in the ontology. The data sources relevant for the answer can then be determined by automatic inference. However, this means that such an ontology-based approach for semantic integration is essentially based on local-as-view inclusion relationships, since concepts of data sources are described by a global ontology [30]. In [21] the authors describe a filter implemented as multiple ontologies, one common ontology over all domains and multiple domain-specific ones. This setup allows for a flexible interoperability model.

In [14] five criteria for ontology design can be found: clarity, coherence, extensibility, minimal encoding bias (i.e., clear separation between syntax and semantics), and minimal ontological commitment (i.e., finding the minimal set of terms essential for knowledge communication). This kind of approach is certainly promising and we plan to examine it and compare it to the one we present in our future work.

2.1.3. Data exchange via a copy of reality

The typical BIM model is a proper model according to Kühne's definition introduced in [19]: it is based on an original that already exists or is going to exists (e.g., a building), it is an abstraction of the

[26] https://www.w3.org/TR/rdf-sparql-query/ (last accessed 2020-11-13).
[27] https://technical.buildingsmart.org/standards/ifc/ifc-formats/ifcowl/ (last accessed 2020-11-13).

original (e.g., many details are omitted), and it can represent it under certain circumstances (e.g., during the planning process). However, there are developments in the AEC industries aiming at producing *digital twins* of buildings [17]. From a BIM perspective, a digital twin is a BIM model without any abstractions, i.e., a copy of reality and this implies that each software that handles the digital twin has to implement the "semantics of everything".

An example of this development are the numerous attempts to expand the IFC data model by adding more and more detail in various domains [2,9,29]. Many attempts disregard the International Framework for Dictionaries (IFD) mechanism for defining taxonomies for IFC, and consequently, without knowledge of each other, run the risk of producing redundancies and conflicting concepts [7]. As presented in the works of ([2,29], there are not just multiple ways to define a tunnel and its infrastructure, but also multiple possible docking points for new elements in the existing IFC data model. The newest version of IFC, IFC4.3 [6], already contains the basis for infrastructure elements, e.g., *IfcFacility*, *IfcBridge*, in preparation for the definition of, e.g., tunnels. In addition, there is potential for conflict between domains. The data model presented in [2] defines a new subtype of *IfcGeometricRepresentationItem*, the so-called *ProceduralModel*. This model is used to serve as representation of a *TunnelElement*. Another domain that relies heavily on procedural geometry is urban design. This would require a visible and traceable method for establishing a semantic relationship between the concepts of procedural geometry in underground facilities as well as in urban areas.

The developments described above demonstrate the complexity resulting from overlapping semantic and pragmatic heterogeneity.

2.1.4. Data exchange as knowledge communication

The interest in interoperability in the AEC industries includes not just data but also methods for its manipulation. For example, the calculation of the energy efficiency of a building design involves thermal simulation algorithms. New algorithms are routinely developed in the course of research projects and have the potential to bring significant technological advancement, if they could be applied in the industry. However, most of these algorithms operate on their own bespoken data models, and therefore, are inaccessible for other applications (see Section 1.1, practical implication (a)).

Aside from creating a dedicated application, there are multiple approaches to algorithm development in the context of the AEC industries. One of them is the definition of prototypes in applications such as MatLab[28] and Modelica.[29] However, such prototypes are generally only accessible via an Application Programming Interface (API) that only few proprietary software integrate. An additional approach is the use of spreadsheets and small code snippets, e.g., in Microsoft Excel™ and Visual Basic for Applications (VBA), respectively. The drawback of this approach, as well as the previous one, is that both offer no semantic support. A cell in a spreadsheet is a container that holds arbitrary information, which can result in misinterpretation and, ultimately, misapplications of the algorithm. For example, a cell whose upper neighbor holds the text "adjacent zone" could contain either the name of a neighboring thermal zone in a building simulation, or its identifier. Moreover, it gives no indication how the neighbor relationship between zones is to be determined. This example demonstrates that a single cell can potentially lead to multiple interpretation errors due to both semantic and pragmatic heterogeneity. In an additional example, the building automation domain uses the Building Automation Networks (BACnet) protocol[30] to define interoperability between sensors and actuators as well as the control algorithms of components [18]. VDI 3813[31] and VDI 3814[32] define the elements of the standard BACnet workflow. However, BACnet has yet to be adopted as part of BIM, and even if it were fully integrated, it would encounter the same problems we described in Section 1.1.

2.1.5. Data-driven data exchange

This method makes use of the vast amounts of data in the AEC domains as basis for analysis, pattern detection and, ultimately, development of algorithms for automated semantic similarity detection.

In [28] the authors present a data-driven IFC based approach for identifying the same object when transferring information between the architecture and structural engineering domains. The identification process concentrates on features common to both domains, the geometry and the material of the object. In essence, the authors present a method for automated detection and integration of pragmatic heterogeneity.

Petrova et al. couple an ontology persisted as a graph database with numerical data, e.g., geometric, from simulations or monitoring, into a framework that links the graph to the data [24], but does not integrate it into the ontology. In this way, different ontologies can operate on the same data and effectively use it as a communication medium. The drawback of this method is the potential for significant unaddressed pragmatic heterogeneity as the link between an ontological node and the data can be interpreted in a number of different ways.

2.2. Road map: towards full semantic and pragmatic integration

So far, we have listed technological challenges and solutions stemming from different types of heterogeneity. In this subsection we will consolidate our findings and formulate the challenges that need to be addressed to achieve full semantic and pragmatic integration into a *facade* that contains all prerequisites for interoperability not only in BIM but also in any other data exchange context.

2.2.1. Challenges

Full interoperability requires uninterrupted multi-directional information flow through an arbitrary number of applications. The full integration of the underlying data models, or standards, functions as the backbone of this process. In Section 2.1 we examined various data exchange strategies relying on different types of integration. Thereby we established that one vital prerequisite for exchanging not just data, but information, is the semantic and pragmatic consensus.

In an environment as diverse and complex as that of the AEC industries, there is a large volume of explicit domain knowledge distributed over multiple data models. After all, each software tool employed in the AEC industries considers at least one part of the semantics of one or multiple domains.

Considering the issues of both explicit and implicit heterogeneity as well as the interlocking of concerns we discussed in Section 1.1, we can formulate the following challenges on the road to "robust" integration, which we address in this work:

Challenge 1: *Exposing the semantics*. The semantics of a software typically resides within its type system, or data model. In order to interact with a software via another software, we need to discover its semantics, not manually but automatically. Furthermore, we need to make that semantics available in real-time. This requires the ability

[28] https://www.mathworks.com/products/matlab.html (last accessed 2020-11-13).
[29] https://www.modelica.org/ (last accessed 2020-11-13).
[30] http://www.bacnet.org/ (last accessed 2020-11-13).
[31] https://www.vdi.de/richtlinien/details/vdi-3812-blatt-1-automationsfunktionen-fuer-wohngebaeude (last accessed 2020-11-13).
[32] https://www.vdi.de/richtlinien/unsere-richtlinien-highlights/vdi-3814 (last accessed 2020-11-13).

to produce artifacts conforming to it, e.g., valid instances of the semantic types, containing user input on demand.

Challenge 2: *Exposing the functionality*. In addition to discover the semantics of a software, we need access to its functionality, e.g., the methods that trigger all relevant algorithms. This requires the ability to call said functionality with all necessary data with as little additional effort as possible.

Challenge 3: *Staying up-to-date*. Most standards and software in use undergo updates for various reasons, often including their semantics. For example, a change in an algorithm may require a change in the data model. We need to be able to react to such updates as soon as possible. Update cycles of several years, e.g., typical of the IFC standard, place a massive burden on the validity as well as maintenance of already established information flows.

Challenge 4: *Making the pragmatics explicit*. The pragmatics of a standard, concept or software resides in an implicit from in the mind of each user. We need to provide a mechanism for exposing and discussing the pragmatics, in order to reach consensus.

The challenges listed above address a seamless interaction considering the semantics and pragmatics of a single application. Nevertheless, full interoperability requires an unbroken communication network involving multiple applications. This, in turn, requires effective interaction among semantics, which may have partial or full overlap, and between pragmatics that may be contradictory. This brings us to additional challenges concerning (*i*) *translation between semantics*, (*ii*) *translation operations* in cases where there is no one-to-one correspondence between semantic concepts, and (*iii*) *automation of the translation process*. These challenges we will address in our future work.

3. Approach

Our approach hinges on the ability to model and manipulate semantic information in a manner that in no way interferes with it. This means that the syntactic container holding such information does not include any additional semantics, but instead, allows depicting structure in an abstract way, similar to XML or JSON. In addition, unlike XML or JSON, the container is modifiable in real time. This makes model-driven engineering a fitting technique for realising our approach.

3.1. Model-driven engineering: preliminaries

Based on Martin Fowler's classification[33] models are used as: [(1)].

(1) *sketches* for communication purposes, where only partial views of an artifact are specified;
(2) *blueprints* to provide a complete and detailed specification of an artifact;
(3) *programs* instead of code for the development of an artifact.

This means that during development, a team uses models in several different ways as abstraction of reality, which makes the design of an artifact (e.g., software) a model-driven process. Therefore, models are crucial for understanding and sharing knowledge about a domain of interest. *Model-driven Engineering* (MDE) transforms models into so-called "first-class citizens" in the field of software engineering [4]. The purpose of MDE in the software engineering domain ranges from communication between different stakeholders to the executability of the developed software.

According to [4] there are two main concepts in MDE: *models* and *transformations*. The latter is used for employing manipulation operations upon models. The notation for expressing both concepts is known as modeling language. There are different layers of abstraction in MDE. From a bottom-up perspective there are: (*i*) M0, which contains run-time instances of the defined model elements of the next higher layer M1; (*ii*) M1, which describes the domain of interest by a domain model and defines the language describing the semantics of that domain; (*iii*) M2, which defines a modeling language (e.g., UML,[34] SysML,[35] or a domain specific language) for specifying domain models in M1; and the layer (*iv*) M3, which defines a so called meta-language for specifying a metamodel such as MOF.[36] It is essential to note, that the concept of the metamodel in the MDE domain differs considerably from the homonymous concept in the AEC domains, where it is sometimes used to describe correspondences or mappings (see [3]). In the MDE domain, a model is an instance of some more abstract model, or metamodel. This is the reason why we could define an infinite number of model levels. However, a model has to *conform to* its metamodel, which means that all its elements can be expressed as instances of the corresponding elements of the metamodel, as shown in Fig. 1 and implemented in our framework presented in Section 3.2. In a nutshell, a modeling language is a tool that supports engineers in specifying models, be it in graphical or textual representation.

Transformations are used for mapping between models specified at any level. For example, in model-driven software engineering, transformations are used for the automatic transformation of model elements (M1) to corresponding code statements (also M1), which could be executed at a platform and produce run-time instances corresponding to the model run-time instances (M0). Generally, transformations are defined at a model level higher than the level they are applied on [4]. Transformation rules could be defined manually from scratch, or by defining specific mapping rules by means of a model transformation language such as the Atlas Transformation Language[37] (ATL), which is the most widely used rule-based transformation language, both in academia and in industry. ATL contains a mixture of declarative as well as imperative constructs, is uni-directional (i.e., transformation from language A to language B), and transforms *read-only* input or source models into *write-only* output or target models [4].

In the context of this work, we employ the described MDE techniques for extracting a representation of the semantics of any of the involved applications, and for producing valid instances of the types necessary for executing a function call within it. Thereby, MDE enables us to make the sources of semantic conflicts between applications explicit, and to provide a reproducible path towards their resolution. In addition, it provides tools for expressing and negotiating pragmatics. In our case study presented in Section 4, we examine the feasibility of a direct information exchange between applications via our MDE-based framework. We forego of

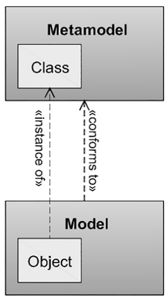


Fig. 1. Relationship between Metamodel (M2) and Model (M1) [4].

[33] https://martinfowler.com/bliki/UmlMode.html (last accessed 2020-11-13).

[34] https://www.uml.org/ (last accessed 2020-11-13).
[35] https://sysml.org/ (last accessed 2020-11-13).
[36] https://www.omg.org/mof/ (last accessed 2020-11-13).
[37] https://www.eclipse.org/atl/ (last accessed 2020-11-13).

employing serialization or using shared memory. Instead, we compile a minimal set of preconditions, coupled with an estimation of the required effort, that need to be met by the involved software to ensure that all input- and output-relevant data structures are exposed as public types, and that there is a suitable *entry point*. Finally, we evaluate the implications of the presented framework for the data exchange between multiple domains, such as building physics and structural engineering.

3.2. The modeling framework for integration facades

Our approach can be demonstrated by a workflow incorporating our central modeling framework that allows us to overcome the challenges we formulated in Section 2. In this section, we give a broad overview of the workflow as well as the framework and take a deeper look at the framework's modeling language. We will elaborate on other aspects of our approach step by step in the course of evaluating our case study results in Section 4.

3.2.1. The workflow

Fig. 2(b) shows the structure of our modeling framework on the left, and the type structure of an application written in an object-oriented programming language on the right, as an example. The two structures are similar. Both require a language for describing the structure. In the target application, the instances of the language build a type system. In most cases this is the semantics of the application. The language provides the syntactic containers to hold the semantics. For example, the C# programming language allows us to create classes. The *class* as a syntactic container resides in the C# language (see the box *Language* in the top right corner of Fig. 2(b)). Class *Person*, on the other hand, is a syntactic instance of *class* and carries semantics representing the concept *person* in the real world. Class *Person* would occupy box *Types* in Fig. 2(b). The same relationship between syntactic containers and semantics holds true in the modeling framework. However, in it, the instances of the language build a model. The *Component* as a syntactic container resides in the language of the modeling framework (see the box *Language* in the top left corner of Fig. 2(b) and the top element in Fig. 2(a)). Metamodel *Person*, in analogy to class *Person*, is a syntactic instance of *Component* and also carries semantics representing the concept *person* in the real world. Metamodel *Person* would occupy the box *Metamodel* in Fig. 2(b).

As we elaborated in **Challenge 1**: *Exposing the semantics*, in order to be able to work with the semantics of any application, it needs to be discovered first. Steps 1, 2 and 3 of the workflow depict the discovery process. Fig. 2(b) also shows the required interfaces. In order to access the target application's type system without the help of the source code, we need a reflection mechanism or a dedicated API. This is what the first step accomplishes - retrieving information about types, e.g., **T1** (see steps 1 and 3 in Fig. 2(b)). In the next step, using a syntax mapping interface integrated into the modeling framework, or its API, we choose the appropriate language elements and build a model of the discovered type system, e.g., the metamodel **MM1**. Finally, in step 3, we establish a link between each type and its corresponding metamodel. However, this is not yet sufficient to overcome **Challenge 1**.

The run-time data resides not in the type system of the target application, but in its instances. For this reason, the modeling framework enables the construction of a model of each instance, which can receive user input. This is made possible by the syntactic relationship *instance-of* implemented by the language of the modeling framework (see relationship *OfType* between *Typed Component* and *Component* in Fig. 2(a)). The models of the instances of the target application are constructed as syntactic instances of the metamodel corresponding to its type system (see the box *Model* in Fig. 2(b)). This enables step 4, in which the user edits the model of each instance. Step 5 uses the established association between each type of the target application and a metamodel to produce models of the run-time instances of the target application's type system. Those models can inject their information into the target application (step 6), interact with instances produced only in the target application (step 7) and supply data for function calls on the target application (step 8), which result in valid output (step 9). This allows us to overcome **Challenge 1**.

At this point, let us take a deeper look into the language of the modeling framework. The relevant excerpt is depicted in Fig. 2(a). Its core consists of two classes, *Component* and *Parameter*. A component can contain an arbitrary number of parameters and other components. It can also reference other components, be an instance of or the representation of another one. In particular, the syntactic relationships our modeling framework can accommodate are the following:

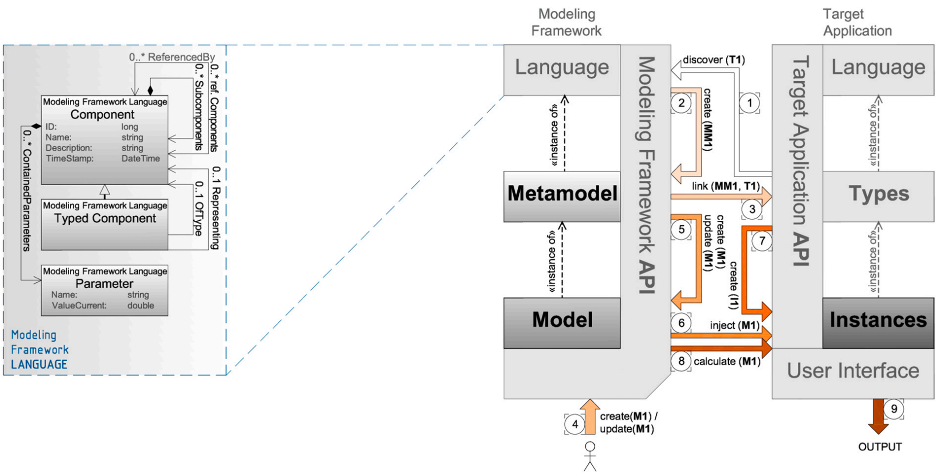
(1) **Association** (unidirectional or bidirectional). This relationship allows coupling, or referencing, between model elements without restrictions. See *ref. Components* and *ReferencedBy* in Fig. 2(a).
(2) **Containment**. This relationship ensures that one element is completely contained in exactly one other element or not contained in any element at all. See *Subcomponents* and *ContainedParameters* in Fig. 2(a).
(3) **Instance-of**. This relationship allows one element to be declared as the type of another. See relationship *OfType* between the classes *Component* and *Typed Component* in Fig. 2(a).
(4) **Representation-of**. This relationship allows one element to be declared as the representation of another. See relationship *Representing* between the classes *Component* and *Typed Component* in Fig. 2(a).

This language allows us to build a model corresponding to any semantic type. *Component* can model reference-type elements (e.g., a class, which is addressed by reference or pointer), *Parameter* can model value-type elements (e.g., an attribute of type integer, which is addressed by value). Both *Component* and *Parameter* as well as each of the above listed syntactic relationships can act as a carrier for arbitrary semantics. In fact, the differentiation between those four kinds of relationships is motivated mainly by convenience. What we truly need is simply the concept of two syntactic elements having a relationship. The specification of its type can be left entirely to the semantics of its content. For example, in this case, composition relationships between classes are expressed as one component being a sub-component of the other, i.e., a true part of it. Association and aggregation relationships, on the other hand, are expressed as one component referencing another. Bidirectional associations require each component to reference the other. The relationship modeling the one between a class and its instances is contained in the *OfType* relationship container. Finally, the relationship modeling the one between a real-world concept and its representation in a model is contained in the *Representing* relationship container. An example of the application of the language of our framework to model class *Person* can be found on our project website.[38]

We will examine the modeling of classes and their instances in more detail in Section 4. Now we turn to the methods of a class and the functionality they implement. Overcoming **Challenge 2**: *Exposing the functionality* depends entirely on the target application and the methods of its types. Unless we misappropriate the reflection mechanism to call private methods, we are dependent on its public ones. Moreover, we need to be able to call them with objects, or instances, as input parameters, to enable real-time interaction. This is step 8 of **Workflow 1**. We will demonstrate two approaches to overcoming this challenge in Section 4, when we discuss specific use cases and programming styles.

Challenge 3: *Staying up-to-date*, on the other hand, is met by steps 1 to 3 already. As the discovery process is fully automated, any change in the semantics of the target application on the right translates into an

[38] https://cdl-mint.se.jku.at/artefacts-for-semantic-integration-for-big-open-bim/##class-Person.



(a) (b)

Fig. 2. (a) The language of the modeling framework and (b) Workflow 1, which answers Challenges 1 and 3.

adapted metamodel on the left. This, in turn, enables the subsequent steps to proceed with updated instances. Only step 4 may require more attention from the users, as they would have to comprehend the update and adapt to it.

The modeling framework offers all tools to meet **Challenge 4**: *Make the pragmatics explicit* since we can enrich the model of the type system of the target application by additional model elements and relationships to add the pragmatics to the semantic model and produce a true *integration facade* for the target application.

Addressing challenges 1 to 4 will allow us to perform a loss- and distortion-free translation considering both semantics and pragmatics, even in cases where there is no one-to-one correspondence between concepts. We will examine the workflows involved in translation between multiple source and target applications and present their evaluation in our future work.

3.3. Integration facades

Our approach, as described in the previous sections, provides *integration facades* for data models that include semantics and pragmatics. It is, therefore, necessary to differentiate between semantics, syntax and representation. These three dimensions are generally handled within the same data model, as exemplified by IFC (see Section 1). A separation can significantly simplify the integration process. The pragmatics, on the other hand, consists of implicit assumptions without a formal expression. We will take a closer look after separating the three explicit dimensions. In Fig. 3 (a) we do just that: semantics are depicted along the *ontological axis*, syntax - along the *linguistic* axis, as it is the formal language of the data model, and the representation of an object of the real world is depicted along the *conceptual axis*. The latter one is where we cross over from abstract modeling into the real world. In Fig. 3 (b) we show the typical case when we model a real-world concept, e.g., that of a wall construction. This concepts is represented by a model element, e.g., *Ontological Type Wall Construction*. This type can be instantiated as an object in the model, in this case *Object wc01* to represent one specific object in the real world, e.g., one specific wall construction that we can interact with in the real world.

However, real world objects are often part of the extension of more than one concept [19]. In our case, the same physical object can be regarded simultaneously as a wall construction and as a vertical slab, as shown in Fig. 3(c). In other words, it has multiple aspects, or it can be viewed from different points of view as discussed in our motivating example (see Section 1.2). On the other hand, the typical object-oriented modeling language does not allow the instantiating of multiple concept representations into the same modeling object. Fig. 3 (d) demonstrates that the *instance-of* relationship is defined only between one class and one object. Therefore, the typical modeling scenario involves modeling each concept by a separate ontological type, instantiating of each type into a separate ontological instance and modeling a connection between them by applying various software patterns. In this case, objects **wc01** and **vs01** can be used as dynamic types in an object-type pattern, either to assume the role of a dynamic type for the other or to provide multiple dynamic typing for a third object.

In either case, such workaround misuses syntactic tools (the syntactic association relationship) to model semantics (the semantic instance-of relationship). Therefore, a change in the syntax in the process of data exchange can inadvertently cause a hidden change in semantics that can be very difficult to trace and needs handling on a case-to-case basis, depending on the syntax used by the data models of interacting tools. For example, if, during data exchange, we transition from a syntax that allows dynamic associations to a syntax that allows only association queries on demand, the semantic information stored in that association will not always be present to restrict the instance behavior as a true static type would. This would completely remove type safety from the translated data model.

The modeling language we use in our approach offers several additional modeling constructs that avoid this type of intermixing of syntax and semantics. Fig. 4(a) shows an attempt to add a semantic connection to object **wc01** and **vs01** by adding an additional semantic abstraction level along the ontological axis. We declare that *Ontological Type Wall Construction* is an instance of *Ontological Type Layered Design*, *Ontological Type Vertical Slab* - an instance of *Ontological Type Structural Design*, and that both are specialisations (or sub-types) of the *Ontological Type Design* (see Fig. 4(b)). This would allow to establish that **wc01** and **vs01** have a common type and, in that type's definition, that they represent aspects of the same real-world physical object. Since the traditional object-oriented paradigm allows only one level of ontological instantiating, between a class and an object, we need a different construct to allow

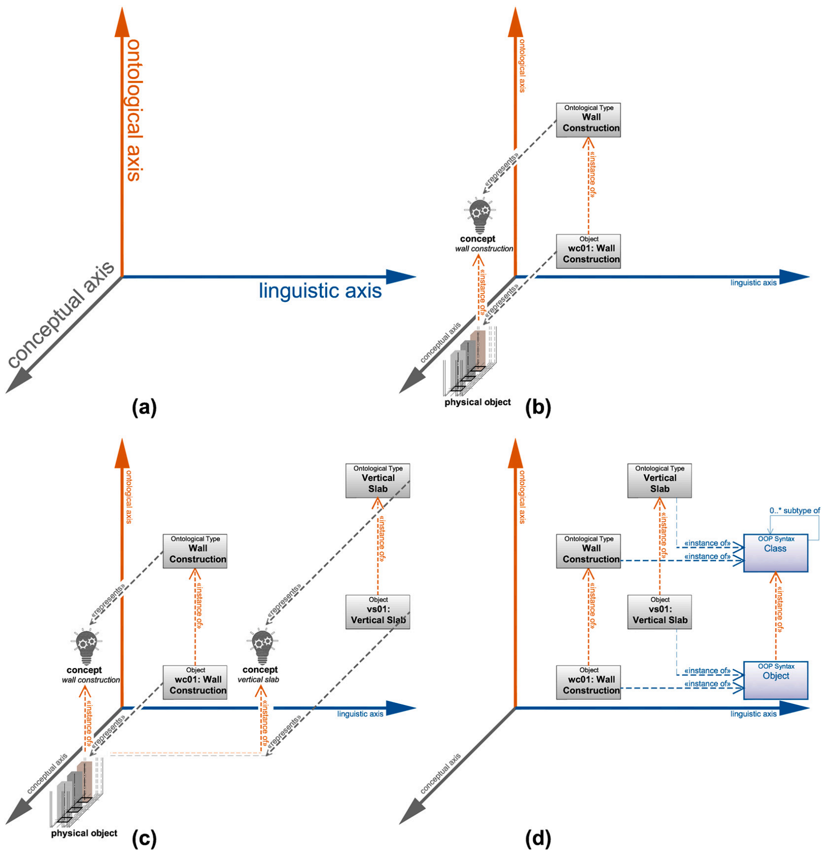


Fig. 3. The three axes of integration. (a) ontology, linguistics and conceptual thinking (b) a typical relationship between a concept and its model (c) the same object classified as different concepts (d) a typical relationship between a model and its formal language.

multiple levels of instantiating. Fig. 4(b) shows that we cannot actually instantiate all ontological types we need by employing the object-oriented approach alone. Fig. 4(c) demonstrates the model-driven engineering approach. The *Typed Component* element allows an arbitrary number of instantiating steps via the relationship *Of Type*. Thus, *Layered Design*, *Structural Design* and *Design* can all be syntactically constructed as syntactic instances of *Typed Component* and semantically connected along the ontological axis.

The declaration of additional types we described above is an example of handling the pragmatic dimension of a data model by making one possible implicit assumption explicit, e.g., that *Wall Construction* and *Vertical Slab* are aspects of the same concept (see the motivating example in Section 1.2), and integrating it in the model of the semantics, even if it is not expressed in the type system of the target application.

The pragmatics can be expressed along the conceptual axis as well, since our approach allows representational relationships along the conceptual axis. Fig. 4(d) shows a different type of relationship between the ontological types *Wall Construction* and *Vertical Slab*, via the *Representing* relationship declared between different syntactic instances of element *Typed Component*. Here, the pragmatics of another user can be made explicit by stating that the vertical slab is regarded as a representation of the wall construction (in a particular context). The situations depicted in Fig. 4(c) and (d) can be regarded as conflicting pragmatics made explicit. This, in turn, makes it possible to resolve the conflict as part of the model and to produce a coherent integration facade of the target application.

Finally, our approach allows us to confine the data model syntax purely to its role as a linguistic tool and to model ontology and representation separately from it. This removes the syntax of the data model from the list of potential semantic error sources during translation between data models.

In the next section, we present an embedded case study consisting of three cases, or units of analysis, on the basis of the guidelines of [26]. The case study enables us to evaluate the approach we presented in this section in a real-world environment, and to examine the separated workflow steps in detail.

4. Case study

The prototype of the modeling framework, we introduced in Section 3 is implemented as a Windows Presentation Foundation (WPF) application in C#. Therefore, we focus our attention on software written in C#, or on applications that offer a C# API for direct access (e.g., Microsoft Excel™).

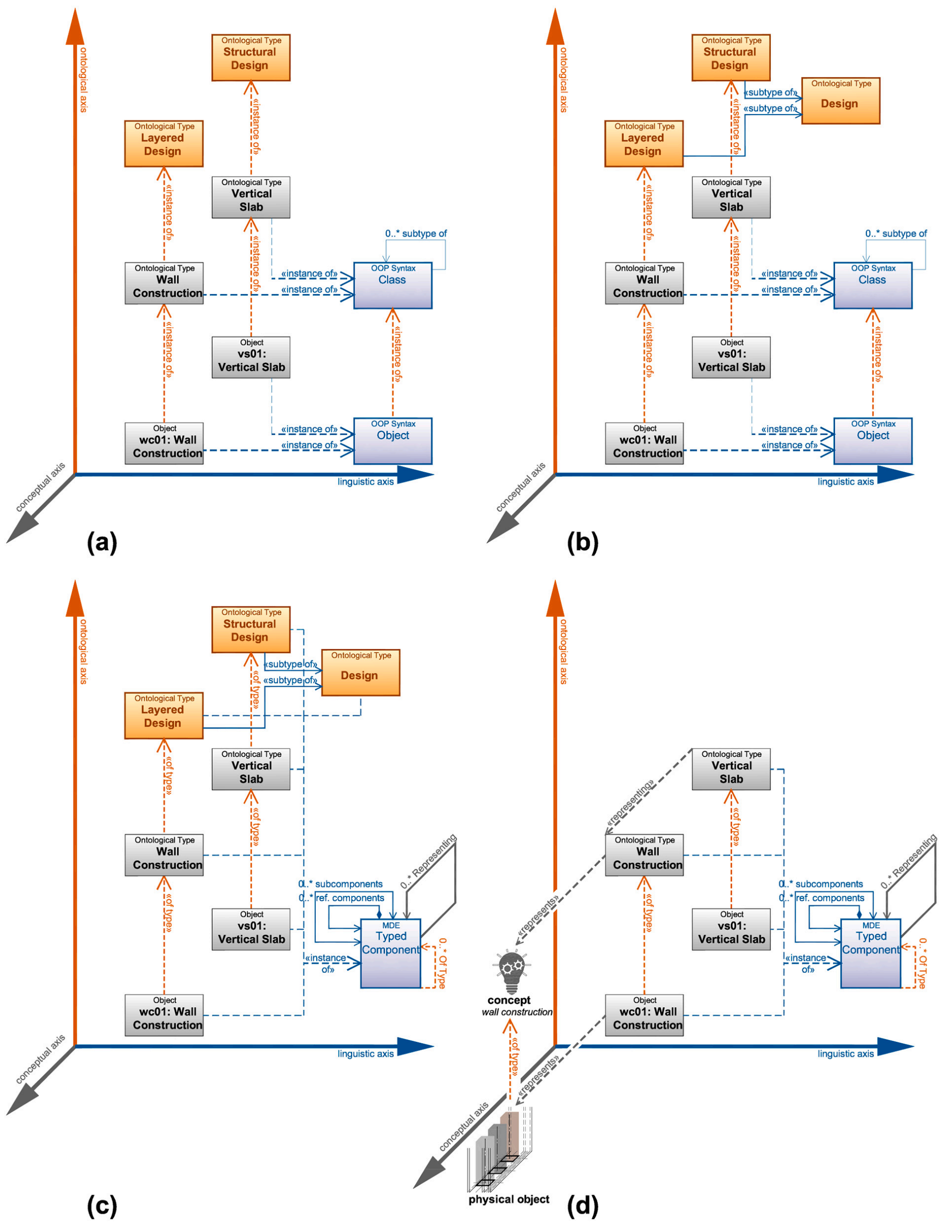


Fig. 4. The three axes of integration continued. (a) adding ontological types (b) connecting the ontological types via a common super-type is not possible with the typical class-object syntax (c) realising the additional types via the syntax of our modeling language (d) realising a representational relationship between ontological types along the conceptual axis.

4.1. Design

We chose small non-web based simulation software, which is typical for simulation tools in the AEC domains. Usually, such tools are prototypically implemented as part of a research project, but seldom advance from this stage into a product one. Furthermore, as we outlined in Section 1.1, maintaining them in the context of very complex and constantly evolving data exchange standards is not cost-effective. According to our own online search, such tools are effectively not available beyond the duration of the project they accompany.

For this reason, we have chosen two simulation applications available on the website Code Project[39] that closely mimic the typical features of simulation software in the field of building physics. The first one is a gravity simulator, and the second a sound wave propagation simulator (from the acoustics sub-domain of buildng physics). Both handle input and output via a Graphical User Interface (GUI). The third application is a Microsoft Excel™ tool for calculating the temperature, humidity and CO2 concentration in a single space developed by domain experts at TU Wien [22].

4.1.1. Case 1: a windows forms application simulating particle motion under gravity

This software simulates the motion of a swarm of particles, each with an initial mass and velocity, under the influence of gravity. The result is represented by an animation on a two-dimensional canvas and as a table containing mass, positions and velocities. Both in its functionality and in the presentation of the simulation results, it resembles existing tools in the AEC industries, e.g., a tool for the calculation of light distribution. The main difference lies in the input handling. This tool allows the user to set the initial position of each particle per mouse-click, which makes the result dependent on the user's hand movements. However, simulation tools in the AEC industries aim to deliver reproducible results. For this reason, they require predictable input, most commonly in the form of a human-readable text file. Therefore, a first step in adapting a tool with an interactive user interface would be to replace the imprecise interaction by precise textual instructions in an input file. This requires the tool to implement custom serialization, which is a non-trivial task depending on multiple factors, e.g., the tool's data model complexity. In any case, a direct communication with the application's data model is not possible.

4.1.2. Case 2: a windows presentation foundation application simulating the propagation of sound waves

This software simulates the propagation of sound waves generated by user-defined sound emitting point or line sources. The resulting interference pattern is calculated numerically over a discrete grid and is displayed as an animation on a canvas in one, two, or three dimensions. Its functionality is very similar to existing tools in the AEC industries. There is an entire sub-domain in building physics dedicated to sound protection, which employs similar methods for the calculation of the sound-proofing properties of materials and constructions. As in the previous case, the main difference, from the viewpoint of the user, is the input handling. Sound emitters and sound blockers are defined by mouse-click. An exact numerical definition in a text file is not possible, as the tool is not equipped with a custom serializer.

Cases 1 and 2 will demonstrate each step of **Workflow 1** in detail. In particular, we will focus on the part of the implementation that addresses **Challenge 1**: *Exposing the semantics* and **Challenge 2**: *Exposing the functionality*.

4.1.3. Case 3: a Microsoft Excel™ application simulating the thermal behavior of a single space

This case takes a deeper look at a Microsoft Excel™ appli-cation

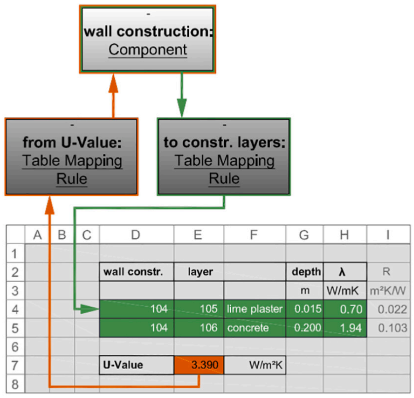


Fig. 5. Case 3. A sample calculation of the U-Value of a wall construction consisting of two layers in an Excel sheet.

developed as part of a research project in the domain of building physics [22]. It determines the temperature, humidity and CO2 concentration in a single enclosed space over the course of a week during a heat wave. In addition, it takes climate, wall construction, orientation, user behavior, and building services into account.

Since this configuration is far too complex to depict here, we have used a small example of one possible configuration of the input and output as a representation of the results we obtained in this study. The bottom part of Fig. 5 shows an Excel sheet prepared for the calculation of the U-Value[40] of a wall construction consisting of two layers. The input cells are green, with each row corresponding to a material layer in the wall construction. The output cell is orange. The green and orange arrows indicate the input and output information flow, respectively. Both the source and the target of the information flow is a wall construction object. In the full case configuration, the input is distributed over multiple sheets and multiple cell ranges within each sheet. The simulation routine is written in a global module in VBA and is called by clicking a button. The output is a time series saved in a dedicated output sheet. The application supplying the input BIM model is object-oriented and defines buildings, spaces, walls etc. as objects.

This case demonstrates the steps involved in addressing **Challenge 4**: *Making the pragmatics explicit*. In particular, it focuses on the expression of the pragmatics as a representational relationship. This means it shows the mechanisms involved in declaring one semantic concept as representative of another.

4.2. Evaluation focus

In all three cases, we will make adaptations in the source code and evaluate both the adaptation itself and the implementation of the workflow described in Section 3. During the evaluation, we will underline the aspects that directly address the challenges we listed in Section 2.2.1. Since **Challenge 2**: *Exposing the functionality* is entirely dependent on the target application, we will answer the following additional questions:

CQ 1. What amount of effort, measured in change in program length as defined in [15], is necessary to expose an alternative entry point in an existing software that accepts an object of arbitrary complexity as input?

[39] https://www.codeproject.com/ (last accessed 2020-11-13).

[40] EN ISO 6946:2017 https://www.iso.org/standard/65708.html (last accessed 2020-11-13).

CQ 2. What amount of effort, measured in change in program length (see above), is necessary to adapt an existing software, not conforming to object-oriented programming conventions, e.g., using arrays of primitive types instead of objects, to an object-oriented entry point requirement?

4.3. Case study procedures

4.3.1. Data collection procedures

The subjects of this study are pieces of software. Our selection criteria included (*i*) full access to the source code, (*ii*) the software not being familiar to us or, if it is familiar to us, also having a real-world application, (*iii*) the software either being of the AEC domains or very similar in function and (*iv*) the software simulating a physical process based on a one-time input and returning a one-time output. We searched for prototypes of simulation tools from the building physics domain online. For the first two cases we chose applications published on the Code Project Website. For the third, TU Wien provided us with access to a simulation tool for the purpose of this study (the Microsoft Excel™ tool, [22]). In all cases, we were able to examine the code fully and test various adaptations. The purpose of the examination was not to review the quality of the code, but to work with unfamiliar and/or real-world examples and be confronted with real-world challenges.

4.3.2. Analysis procedures

For the analysis of the cases we gathered quantitative data: the change in program length, the change in lines of code, and the number of new types created in the target software (see questions **CQ 1** and **CQ 2**). We further evaluated qualitative data. We tested various approaches to creating an alternative entry point in the target application. We also documented the programming changes required for exposing enough of the underlying semantics through the entry point to enable passing all necessary input data and receiving output.

4.3.3. Validity procedures

The validity of the obtained results can be compromised by several factors. Firstly, the programming expertise of the researchers has an influence on the data collection. For this reason we do not measure the time spent in unsuccessful adaptation attempts, but analyze only the most efficient adaptations achieved. Secondly, the applied technologies can impact both the qualitative and the quantitative evaluation. Quantitative evaluation measured in lines of code can vary greatly based on the programming language and the developer's programming style. Therefore we have added a more robust measure, the program length, which is defined by [15] as the sum of the total number of operators and the total number of operands in the program.

The creation of the entry point depends strongly on technology as well–therefore we only answer the question, if it is possible, not how expensive it is. The difficulty in exposing the underlying semantics via an entry point, on the other hand, depends on the programming style. We have selected cases with very different programming styles to demonstrate the wide range of challenges an adaptation could face.

4.4. Evaluation

In this section we apply the workflow steps we presented in Section 3 and evaluate the results. We start with **Workflow 1** (see Fig. 2) and Case 1 (see Section 4.1.1).

4.4.1. Evaluation of case 1: a windows forms application simulating particle motion under gravity

Workflow 1 starts with the discovery of the semantics of the target application. Step 1 is shown in Fig. 6. The involved parts of the modeling framework and of the target application are highlighted in blue and green, respectively, in the overview image in the top right corner of the figure. Each of the highlighted modules is expanded to show the relevant portion of its content. For example, the semantics of the target application (the green block in the overview image labeled *Types*) can be seen

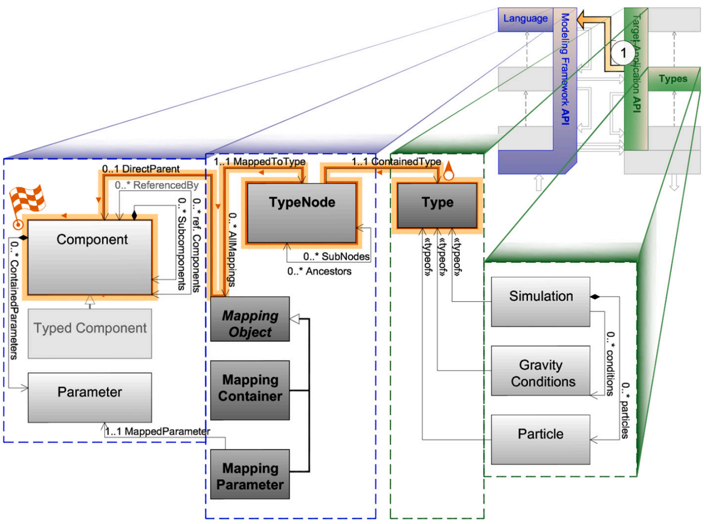


Fig. 6. Case 1. Realization of step 1 of Workflow 1.

in detail on the right in the main portion of the figure, in the box with a dashed greed border. This semantic data model consists of three types. There is a *Simulation*, which refers to *Gravity Conditions* and contains some *Particles*. The language of the modeling framework we presented in Section 3 is the in the top left corner of the overview image. The relevant details are depicted on the left in the main portion of the figure.

In addition to the data models, Fig. 6 shows the interfaces involved in step 1 of **Workflow 1**. On the side of the target application, we use the. net Reflection API to gather information about the target application's type system via type *Type*. The modeling framework realizes the transfer to its own language by employing the types *TypeNode*, *MappingObject* and its subtypes *MappingContainer* and *MappingParameter*. The application of step 1 of **Workflow 1** is depicted as the path highlighted in orange that originates at *Type* and ends in *Component*, thereby completing the information transfer from the target application to the modeling framework.

The result from this transfer is shown in Fig. 7. In step 2 of **Workflow 1**, the instances of *TypeNode* extract the information necessary both for the construction of the corresponding metamodel of each target type and for its instantiation. Once the type structure has been discovered, the modeling framework uses a subtype of *Component*, *Typed Component*, to set a direct relationship between a type and its metamodel element (see the highlighted association *OfType* between *Typed Component* and *Type* in Fig. 7). This is a generic procedure, completely independent of the specific types of the target application. The only requirement is that the application has types exposable by some mechanism, e.g., the meta-information extracted by the Reflection API.

Fig. 8 depicts step 3 of **Workflow 1**. We put particular emphasis on the completed metamodel corresponding to the type system of the target application and represented by the blue block labeled *Metamodel* in the overview image. It consists of elements **M1**, **M2** and **M3**. **M1** models type *Simulation*, with attribute *Name* set to 'M1' and attribute *OfTypeName* set to'Simulation'. Effectively, **M1** has two types, *Typed Component* and *Simulation*. Those types, however, play different roles. The linguistic type is indicated by the dashed orange line connecting **M1** and the language element *Typed Component*. It provides structure, or syntax. The ontological type is indicated by the association *OfType* between **M1** and *Simulation* highlighted in orange. It determines the meaning, or semantics. Therefore, **M1** corresponds to, or is a model of, type *Simulation*.

Type *Simulation* includes the attributes *Name* and *Size*. Those are modeled by the Parameter instances **pM1** and **pM2**, respectively, contained in **M1** via the relationship *ContainedParameters*. Also included in the structure of **M1** is the reference to type *Gravity Conditions*. It is modeled by **M2** of linguistic type *Typed Component* and ontological type *Gravity Conditions*. It is associated with **M1** via the *ref. Components* relationship. Fig. 8 shows the model **M3** of type *Particle* as well.

In effect, the metamodel is the model of the target application's type system. It is shown both in Fig. 8 and in Fig. 9. It enables the creation of run-time instances via reflection. The process begins with the instantiation of this metamodel, or type model, into instance models (see the expanded module *Model* in blue in Fig. 9).

Let us take a closer look at the instance models in Fig. 9 and compare them to the automatically generated target application run-time instances on the right, in module *Instances*. Model instance **s1 Model** is of syntactic type **M1**, which is itself of semantic type *Simulation*. The syntactic *instance-of* relationship between **s1 Model** and **M1** relies on the association *OfType* between *Typed Component* and *Component* in the language utility of the modeling framework (see Fig. 2). The *instance-of* relationship between an object and its type in the the target application, on the other hand, is enforced by the type system of its programming language. All such relationships are depicted as dashed orange lines annotated *« instance-of»* in Fig. 9.

The structure of the run-time instance of *Simulation*, **s1**, includes the slots *Name* and *Size*. Those are modeled by the *Parameter* instances **pM Name** and **pM Size**, respectively, contained in **s1 Model** via the relationship *ContainedParameters*. Also included in the structure of **s1** is the reference to instance **gc** of type *Gravity Conditions*. It is modeled by **gc Model** of syntactic type **M2**, associated with **M1** via the *ref. Components* relationship. In addition, Fig. 9 shows the models **pa Model** and **pb Model** of the instances **pa** and **pb** of type *Particle*, respectively. Each slot *Mass* in a run-time instance is modeled by a *Parameter* instance with the Name 'Mass' and the corresponding value.

Just as the target application permits editing of its run-time data

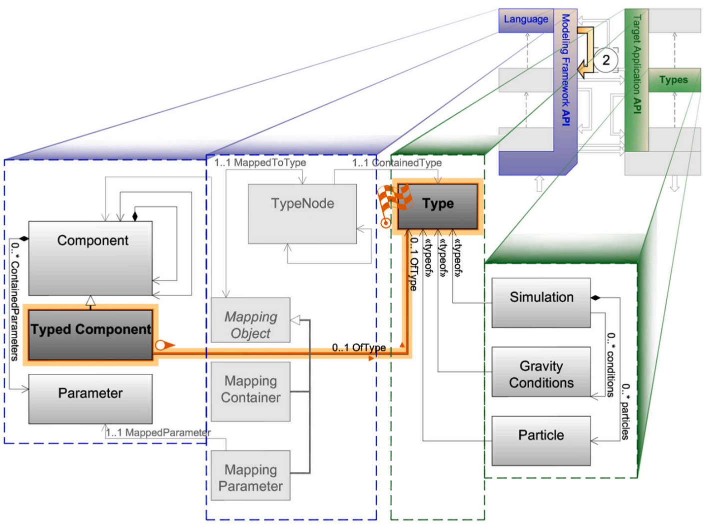


Fig. 7. Case 1. Realization of step 2 of Workflow 1.

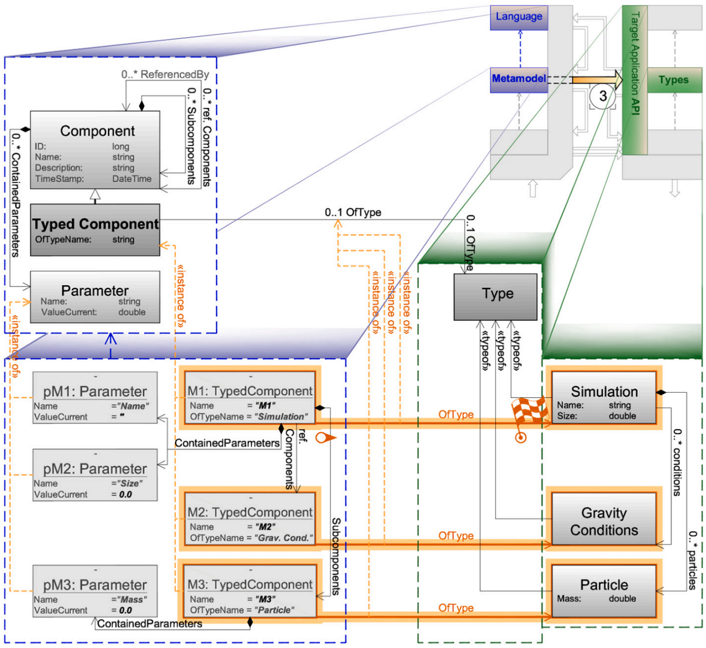


Fig. 8. Case 1. Realization of step 3 of Workflow 1.

structure, so does the modeling framework. The size of the simulation, the number and mass of the particles as well as the gravitational conditions can all be adjusted by the user. In steps 4 and 5 of **Workflow 1**, the model instances receive the user's input (see the orange highlight annotated *4* in the bottom left corner of Fig. 9) and convert it to model instances. In step 6, the corresponding run-time instances of the target application are updated via reflection. In effect, the user input is injected into the target application.

An alternative, simplified solution, which dispenses with the separate modeling of the types and run-time instances of the target application, is shown in Fig. 12 in Appendix A.

The correspondence in structure between the target application data structure and its model in Fig. 9 is syntactic, or linguistic. The semantic correspondence is guaranteed by the association to the same types - *Simulation*, *Gravity Conditions* and *Particle*. Both the target application's data structure and the modeling framework model carry the same meaning - a particle simulation named 'Earth' with size 10.5 containing two particles with mass 0.25 and 0.5 and referencing some gravitational conditions.

This concludes the demonstration of steps 1 to 6 of **Workflow 1**. Those steps fulfill the requirements of **Challenge 1**: *Exposing the semantics*. The modeling framework makes the semantics of the target application available in real-time. This is achieved by producing artifacts conforming to it, e.g., valid run-time instances of its semantic types, containing user input, on demand.

Challenge 2: *Exposing the functionality* requires actual adaptations of the target application and involves steps 7 to 9 of **Workflow 1**. So far, we have gained full access to the run-time instances without implementing any semantics by hand. However, calling an application's functionality depends on the public methods it exposes. In this case, the application uses Windows Forms for the GUI design. This means that there is a standard entry point without input parameters, which calls the standard constructor of the form:

```
[ STAThread ]
static void Main () {
Application .Run ( new Form1 ());
}
```

An alternative entry point, which takes an object, possibly representing the entire simulation, as an input parameter, and calls an adapted form constructor could look like this:

```
public static void Go( Type_1 input ) {
Application . Run( new Form1 ( input ));
}
```

The programming style of this tool is object-oriented and there is, in fact, a type representing the entire particle simulation. Therefore, the introduction of the alternative entry point is straightforward. The type representing a particle simulation, *Simulation*, simply replaces *Type_1* in the code above. An additional method for triggering the simulation programmatically completes the list of necessary changes in the source code. After those, in steps 6 to 9 of **Workflow 1**, the tool can be passed input and executed from the modeling framework. The quantitative evaluation of the code adaptation is shown in Table 3. The additional type serves to organize input and output and implements the above mentioned method.

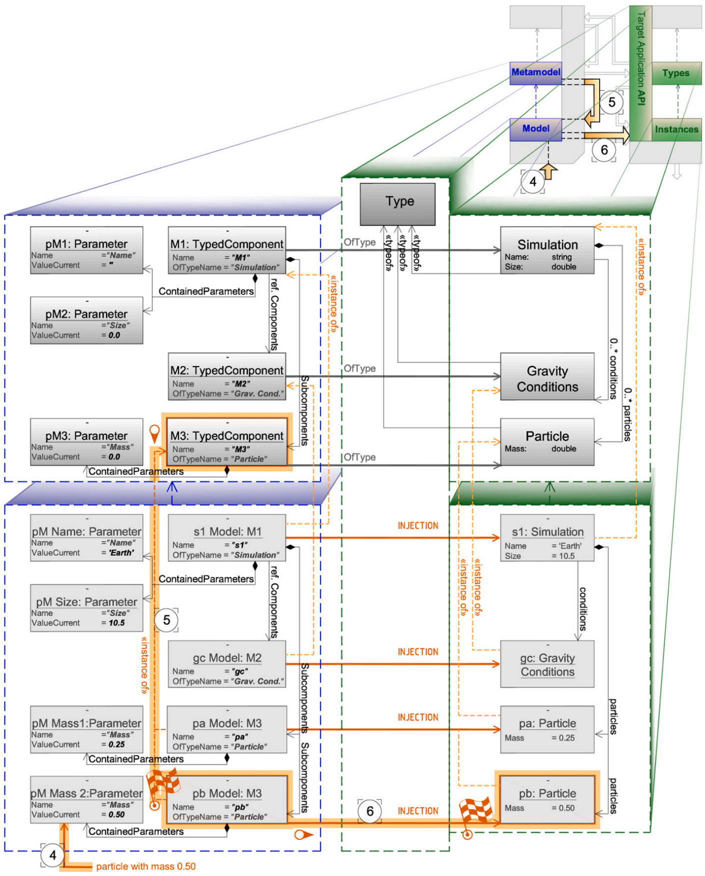


Fig. 9. Case 1. Realization of steps 4, 5 and 6 of Workflow 1.

Table 3
Case 1. Quantitative evaluation of the code adaptation.

Metric	Original	Adapted	Diff.	Increase in %
Program length	7791	8125	334	4.29
Lines of code	714	755	41	5.74
Number of types	8	9	1	–

In conclusion, we present the adapted workflow.

(1) Initially, the modeling framework establishes the connection to the particle simulation tool.
(2) It creates a default model of a particle simulation containing only one particle with a default mass zero. This can be viewed as a model of the types of the particle simulation.
(3) The user edits the instance model to change the properties of the single particle as well as to add more particles and modify the gravitational conditions. Since the modeling framework enables copying, it means that the user can produce an arbitrary number of simulation instances from that one initial model.
(4) The user triggers the simulation with any one of those instances via the alternative entry point **Go**, which accepts a particle simulation instance as input.
(5) The particle simulation tool creates both an animation and a table containing the results.

4.4.2. Evaluation of case 2: a windows presentation foundation application simulating the propagation of sound waves

The programming style of this application is data-oriented. It utilizes numerical methods for the calculation of the interference pattern of sound waves and, therefore, uses a grid, implemented as a two-dimensional array, as its main data structure and input source. There are different approaches to handling two-dimensional arrays, or matrices. If the matrix is sparse, the non-zero entries can be represented as objects. In this case, those would be points in two or more dimensions. In this manner, the entire matrix can be represented by a collection of objects. This scenario is very similar to the one in Case 1. However, instead of using already existing types in the target application, such as *Simulation*, *Gravity Conditions* and *Particle*, we can adapt the source code and add a new type *Point2D* to represent a two-dimensional emitting point source, and a new type *Simulation* that wraps a list of *Point2D* instances.

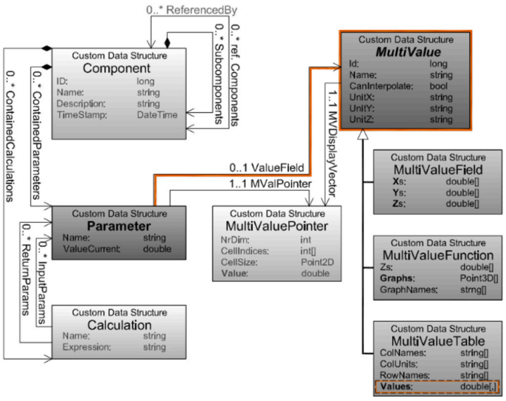


Fig. 10. Handling large numeric data in the modeling framework.

However, we want to address **Challenge 1**: *Exposing the semantics* without modifying the target application. In addition, the scenario described above is less suitable for large dense matrices. Such matrices are produced, for example, by simulation or monitoring software in the building physics domain. A matrix of monitoring measurements taken every 15 s over the course of a year has more than 2 million rows and may have many hundreds or even thousands of columns.

The language utility of the modeling framework we presented in Section 3 has classes for handling data-oriented approaches. Those are the sub-types of *MultiValue*, an abstract container for multiple numeric values, shown in Fig. 10. The values can be organized in scalar fields (*MultiValueField*), graph fields (*MultiValueFunction*) or tables (*MultiValueTable*). An instance of *MultiValue* can be referenced by an instance of *Parameter* by means of the *ValueField* relationship. In this way, it allows the parameter access to all the data via a pointer (see *MultiValuePointer*). Since a parameter can exist only when contained in a component, the data is automatically associated with one or more components, which can take on multiple roles (see Section 2.1.5). On the one hand, the component can act as a simple container, or wrapper, of the data. On the other hand, if the component has a type, i.e., if it is an instance of *Typed Component*, it can act as a typing mechanism.

Using a component as a wrapper of the two-dimensional array produces a model comparable to Case 1. Since the data-oriented programming style still uses the C# typing utility, the Reflection API allows the discovery of all relevant types. As we will see below, we need to add at least one new type to the application to act as input parameter of the alternative entry point. We can utilize this additional type, *Simulation*, as a wrapper for the two-dimensional binary array that carries the input information.

Just as in the previous case, meeting **Challenge 2**: *Exposing the functionality* requires code adaptations. The application uses WPF for the implementation of the GUI. This means that the standard entry point is located in an automatically generated file:

```
[ System.STAThreadAttribute ()]
[ System.Diagnostics.DebuggerNonUserCodeAttribute
()]
[ System.CodeDom.Compiler.
GeneratedCodeAttribute ("PresentationBuildTasks ",
" 4.0.0.0 ")]
public static void Main () {
WpfTransmissionLineMatrixCS.App app =
new WpfTransmissionLineMatrixCS.App ();
app.InitializeComponent();
app.Run ();
}
```

An alternative entry point definition may involve generating a new application domain with its own domain manager for the WPF application to run in. The type of the object we use for passing the input to the application has to be declared as serializable; however, no custom serialization needs to be implemented.

```
public static void Go( Simulation input )
{
var dm_type = typeof ( DomainManager );
string codeBase = Assembly.GetExecutingAssembly().
CodeBase ;
UriBuilder uri = new UriBuilder( codeBase );
string path = Uri.UnescapeDataString( uri. Path );
string ab_path = Path.GetDirectoryName( path );

var setup = new AppDomainSetup
{
ApplicationBase = ab_path,
```

Table 4
Case 2. Quantitative evaluation of the code adaptation.

Metric	Original	Adapted	Diff.	Increase in %
Program length	8422	8783	361	4.29
Lines of code	1085	1133	48	4.44
Number of types	6	10	4	–

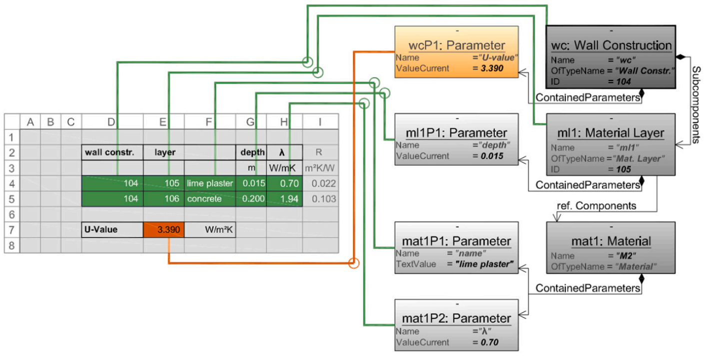


Fig. 11. Case 3. Translating between type *Wall Construction* and an Excel sheet.

```
AppDomainManagerAssembly = dm_type.Assembly.
FullName,
AppDomainManagerType = dm_type.FullName
};
AppDomain domain =
AppDomain.CreateDomain("TempDomain", null, setup);
CallBackContext ctx = new CallBackContext();
ctx.SourceContainer = input;
CrossAppDomainDelegate action = ctx.AppEntry;
domain.DoCallBack(action);
}
```

As mentioned above, the class *Simulation* (see the input parameter type in the alternative entry point **Go** above) plays the role of a wrapper for the information we need to pass. These adaptations result in the addition of 4 classes to the tool (see Table 4): two for the entry point, one as a wrapper for the input and one for translation from the data to the object-oriented paradigm. Both in terms of lines of code and program length increase, this case is very similar to the previous one.

This completes the adaptation of **Workflow 1** to this case. As in the previous case, **Challenges 1** and **2** were overcome. **Challenge 3**: *Staying up-to-date* is met as a direct consequence of the dynamic connection between the modeling framework and the target application. Any changes in the type structure of the target application results in an automatic update of the dynamically created models in the modeling framework.

In conclusion, we present the adapted workflow.

(1) Initially, the modeling framework establishes the connection to the sound wave propagation simulation tool.
(2) The modeling framework creates an empty table referenced by a parameter in a component instance of type *Simulation*. This configuration can be viewed as a model of the default instance of the sound simulation.
(3) The user populates the table by marking the sound emitting sources as ones and leaving all other entries zero.
(4) As in case 1, the user triggers the simulation by injecting a model of a *Simulation* instance into the alternative entry point, **Go**.
(5) The tool creates the resulting sound interference pattern.

4.4.3. Evaluation of case 3: a Microsoft Excel™ application simulating the thermal behavior of a single space

In this case, we evaluate the ability of the modeling framework to overcome **Challenge 4**: *Making the pragmatics explicit*. As we laid out in Section 4.1.3, the setup in Case 3 is complex. The source application has an object-oriented type system for representing an entire building. The target application, which simulates the thermal behavior of the building, consists of multiple Excel sheets and VBA macros. As an illustration of the task, we chose one small excerpt, which translates a wall construction to an Excel sheet, as shown in Fig. 5 and, in more detail, in Fig. 11.

On the right in Fig. 11 is the model of the wall construction, consisting of two material layers, of which only one (**ml1**) is shown. On the left is the Excel sheet that calculates the U-value of wall constructions. The translation pairs are connected by colored lines. Green stands for translation to the spreadsheet, orange for translation from the spreadsheet. For example, the wall construction **wc** itself maps to the cell under the label *wall constr.*, by supplying only its ID. Parameter **mlP1**, contained in the material layer **ml1**, maps to the cell under the label *depth*, by supplying only its current numeric value, 0.015. Parameter **wcP1**, contained in the wall construction **wc**, on the other hand, is mapped to from the cell where the U-value is located in the spreadsheet.

This case example presents us with the task of dealing with a collection of cells in a table, whose meaning is known only to the designer of the simulation the table realizes. In essence, we need to extract the pragmatics and make them explicit in the corresponding model of the modeling framework. As mentioned in Section 2.1.1, Francis et al. were confronted with a similar challenge in the case study presented in [13]. Their approach involved the creation of a metamodel–a *Spreadsheet* containing multiple *Worksheets*, which contain *Columns*, which can reference each other by means of *Reference*. This metamodel allows the automatic detection of dependencies between cells and tables. However, the semantics of those connections still needs to be supplied by the user.

Addressing **Challenge 4**: *Making the pragmatics explicit* involves making the an explicit connection between cells or ranges of cells within a spreadsheet and a semantic type (see Fig. 4(c)) or concept (see Fig. 4 (d)). For example, the user can create a model element in the modeling framework that corresponds to the concept of an U-Value. Subsequently, the user can define a semantic instance-of relationship between this element and the models of cells in a particular range that store U-Values. This would make the implicit assumptions of the simulation designer explicit and greatly aid any user by removing a significant potential error source, the misinterpretation of the simulation's data model.

This concludes the evaluation of our approach. In the following subsection we will discuss some threats to validity with respect to the four challenges (see their definitions in Section 2.2.1) we aim to overcome.

4.5. Evaluation of validity

It is of note that in all three cases we work with applications with clearly defined functionality and a data model consisting of less than 15 types.

4.5.1. Construct validity

As we stated in Section 2.2.1, the goal of our approach is to provide full integration or *integration facade* for any data model, i.e., a semantic and pragmatic consensus, as this is the backbone of interoperability in any data exchange environment, including *Big Open BIM*. In order to account for the arbitrary component in our evaluation, in Case 1 and Case 2 we chose applications that were not developed by us or known to us prior to the study. In Case 3 we chose an Excel simulation (see [22]) well known to us. The reason for this is threefold. First, the algorithm and its input and output are far more complex than those in Case 1 or Case 2–this is the complexity of a real-world applications. Second, the Excel environment is typically used as a prototyping tool in multiple AEC domains. And third, the programming paradigm of this tool is data-oriented instead of object-oriented and gives us the opportunity to account for information flow between semantic-carrying and semantic-agnostic applications.

4.5.2. Hidden factors

In spite of being confronted with a data-oriented programming paradigm in Case 2, we still had the ability, due to the chosen programming language, C#, to utilize object-oriented methods. Our results do not apply to software written in languages that do not allow object-oriented access of any kind.

We have not considered any security aspects in this study.

4.5.3. Generalization

A data exchange standard cannot rely on our approach unless it is generalizable to include much more complex applications.

Challenge 1: *Exposing the semantics*. Our approach employs automated methods for extracting the data model of any application. Even a data model as extensive as IFC4.3 can be automatically loaded.

Challenge 2: *Exposing the functionality*. The three cases we examined allow us to formulate the following requirements that need to be fulfilled by a software with no API, no implementations of memory sharing or of data exchange standards, in order to be fully accessible from other applications:

R2.1. The type structure should contain one type with a pre-defined name, e.g., *MainObject*, providing controlled access to all objects managing input and output.

R2.2. The application should have an entry point that accepts an instance of type *MainObject* as input.

R2.3. The application should trigger its main functionality automatically on receiving input over the entry point described above.

Challenge 3: *Staying up-to-date*. Since both Challenge 1 and Challenge 2 are generalizable, staying up-to-date, in terms of data model and functionality, is generalizable as well.

Challenge 4: *Making the pragmatics explicit*. The modeling framework allows the definition of additional semantic types as well as relationships along the ontological and conceptual axes. These tools allow the extraction of the pragmatics and making it explicit for all users, thus contributing to the consensus of the *integration facade*.

4.5.4. Reliability

The researchers involved in this case study have expertise in the following fields: architecture, building physics, model driven engineering, multi-level modeling and ontology engineering.

5. Conclusion and future work

In Section 2.2.1 we formulated four challenges that need to be overcome in any data exchange process in order to achieve uninterrupted multi-directional information flow. This also applies to Big Open BIM in the AEC industries. The modeling framework we presented in Section 3 addresses those challenges to varying degrees, as summarized in Section 4.5. It enables the integration of semantics at run-time and of pragmatics at design-time. The result is that implicit assumptions are made explicit. Consequently, they can be discussed and a consensus both in semantic and pragmatic terms can be reached, since, as stated by Fetzer [12], *"meaning is at the heart of both semantics and pragmatics"*. In our approach we regard syntax, semantics and representation as the three independent axes of the space in which data exchange operates. This independence, or separation of concerns, allows the semantics to be modeled independently of the notation or representation of information in any available tool, which makes our approach universally applicable to data exchange processes. In addition, we can model pragmatics both along the semantic and the representational dimensions, and produce a full integration of both semantics and pragmatics into an *integration facade*, which is a prerequisite for producing a single source of truth.

Our future work will focus on utilizing the integration facades in the translation between the data models of various tools and standards, even in cases where no one-to-one correspondence between concepts exists. In addition, we will develop algorithms for the detection of semantic patterns as basis for automated translation procedures and of translation rules between semantic models involving calculations. Furthermore, we will examine alternative approaches to defining a semantic model. In this work we demonstrated the power of Model-Driven Engineering (MDE). We intend to perform a comparison between MDE and ontology design in our future work.

CRediT authorship contribution statement

Galina Paskaleva: Conceptualization, Methodology, Software. **Alexandra Mazak-Huemer**: Investigation, Writing - review & editing, Supervision. **Manuel Wimmer**: Supervision. **Thomas Bednar**: Supervision, Software.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Workflow 1: non-level-respecting modeling style

An alternative, simplified solution to the one in Fig. 9 in Section 4.4.1, which dispenses with the separate modeling of the types and runtime instances of the target application, is shown in Fig. 12. The model presented in the bottom left corner contains both the type and instance information and is ready for user input injection. This is a more efficient, however, not level-respecting, model according to [19].

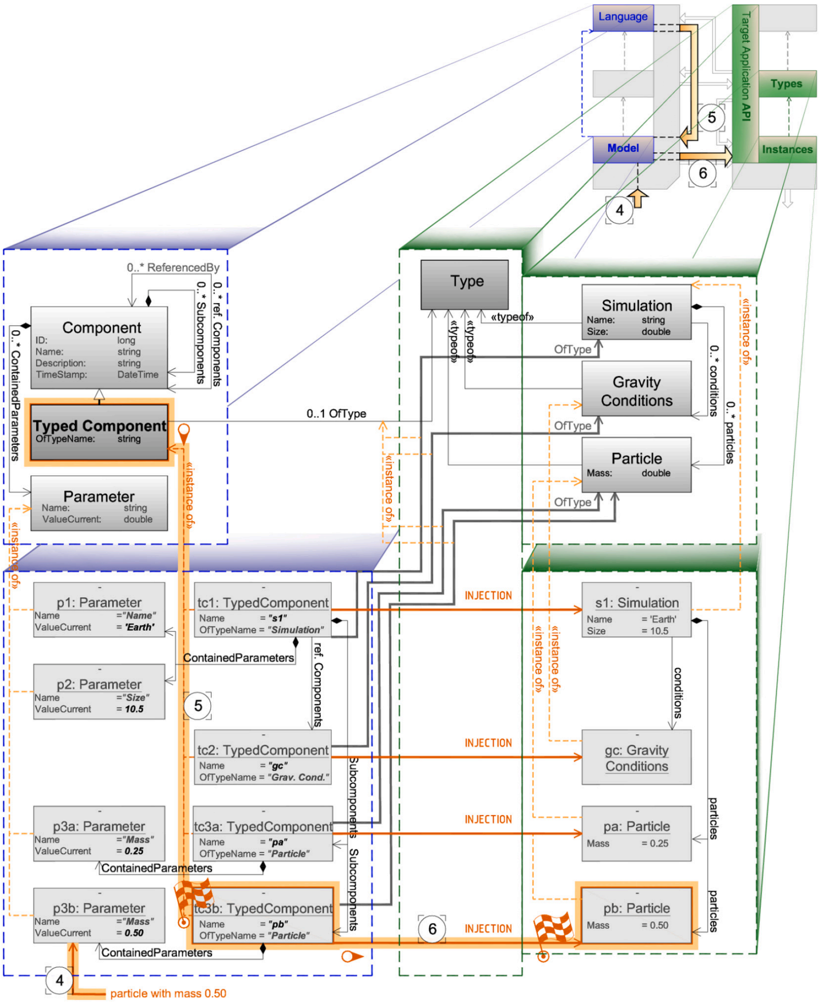


Fig. 12. Case 1. An alternative solution to the one shown in Fig. 8 and Fig. 9 in Section 4.4.1.

References

[1] T. Akanbi, J. Zhang, Y.C. Lee, Data-driven reverse engineering algorithm development method for developing interoperable quantity takeoff algorithms using IFC-based BIM, J. Comput. Civ. Eng. 34 (2020), https://doi.org/10.1061/(ASCE)CP.1943-5487.0000909.

[2] A. Borrmann, T. Kolbe, A. Donaubauer, H. Steuer, J. Jubierre, M. Flurl, Multi-scale geometric-semantic modeling of shield tunnels for GIS and BIM applications, Comput.-Aided Civil Infrastruct. Eng. 30 (2015) 263–281, https://doi.org/10.1111/mice.12090.

[3] M. Bracht, A. Melo, R. Lamberts, A metamodel for building information modeling-building energy modeling integration in early design stage, Autom. Constr. 121 (2021), https://doi.org/10.1016/j.autcon.2020.103422.

[4] M. Brambilla, J. Cabot, M. Wimmer, Model-Driven Software Engineering in Practice, 2 ed., Morgan & Claypool, USA, 2017. ISBN: 978-1627057080.

[5] buildingSMART, IFC4 Release, URL, https://standards.buildingsmart.org/IFC/RELEASE/IFC4/FINAL/HTML/, 2013. last accessed on November 13, 2020.

[6] buildingSMART, IFC4.3 Release Candidate 2, URL: https://standards.buildingsmart.org/IFC/DEV/IFC4_3/RC2/HTML/, 2020. last accessed on February 26, 2021.

[7] T. Cerovsek, A review and outlook for a 'building information model' (BIM): a multi-standpoint framework for technological development, Adv. Eng. Inform. 25 (2011) 224–244, https://doi.org/10.1016/j.aei.2010.06.003.

[8] W. Chen, M. Das, V.J.L. Gan, J.C.P. Cheng, Integrated data model and mapping for interoperable information exchange between BIM and energy simulation tools, in: E. Toledo Santos, S. Scheer (Eds.), Proceedings of the 18th International Conference on Computing in Civil and Building Engineering, Springer International Publishing, 2021, pp. 496–506, https://doi.org/10.1007/978-3-030-51295-8_35.

[9] A. Costin, N. Nawari, A. Adibfar, C. Eastman, Preliminary evaluation of the industry foundation classes (IFC) to enable smart city applications, in: CIB World Building Congress, 2019, pp. 1–10.

[10] F. Dong, R. Zhang, R. Xiao, B. Lei, Mesh simplification with global contour feature preservation, in: WRI World Congress on Computer Science and Information Engineering, 2009, pp. 679–685, https://doi.org/10.1109/CSIE.2009.636.

[11] W. East, Construction Operations Building Information Exchange (COBIE), Defence Technical Information Center URL, https://apps.dtic.mil/sti/citations/ADA491899, 2007. last accessed on February 26, 2021.

[12] A. Fetzer, Recontextualizing Context, John Benjamins Publishing Company, Amsterdam, The Netherlands, 2004. ISBN 9789027295712.

[13] M. Francis, D.S. Kolovos, N. Matragkas, R.F. Paige, Adding spreadsheets to the MDE toolkit, in: A. Moreira, B. Schätz, J. Gray, A. Vallecillo, P. Clarke (Eds.),

Model-Driven Engineering Languages and Systems, Springer, Berlin Heidelberg, 2013, pp. 35–51. ISBN 978-3-642-41533-3.
[14] T.R. Gruber, Toward principles for the design of ontologies used for knowledge sharing? Int. J. Hum.-Comput. Stud. 43 (1995) 907–928, https://doi.org/10.1006/ijhc.1995.1081.
[15] M.H. Halstead, Elements of software science, in: Elsevier Computer Science Library: Operational Programming Systems Series, North-Holland, New York, NY, 1977. ISBN 978-0444002150.
[16] J. Haymaker, P. Keel, E. Ackermann, W. Porter, Filter mediated design: generating coherence in collaborative design, Des. Stud. 21 (2000) 205–220, https://doi.org/10.1016/S0142-694X(99)00042-3.
[17] S. Kaewunruen, P. Rungskunroch, J. Welsh, A digital-twin evaluation of net zero energy building for existing buildings, Sustainability 11 (2018), https://doi.org/10.3390/su11010159.
[18] H.R. Kranz, BACnet Gebäudeautomation 1.12. cci Dialog, 2013. ISBN 978-3922420255.
[19] T. Kühne, Matters of (meta-) modeling, Softw. Syst. Model. 5 (2006) 369–385, https://doi.org/10.1007/s10270-006-0017-9.
[20] H. Lai, X. Deng, Interoperability analysis of IFC-based data exchange between heterogeneous BIM software, J. Civ. Eng. Manag. 24 (2018) 537–555, https://doi.org/10.3846/jcem.2018.6132.
[21] J. Lee, Y. Jeong, User-centric knowledge representations based on ontology for AEC design collaboration, Comput. Aided Des. 44 (2012) 735–748, https://doi.org/10.1016/j.cad.2012.03.011.
[22] J.N. Nackler, Sommerlicher Wärmeschutz: Vergleich von Berechnungsansätzen eines Planungsinstrumentes für Entwurfsfindung, Ph.D. thesis, TU Wien, Vienna, Austria, 2017.
[23] R.F. Paige, A. Zolotas, D.S. Kolovos, The changing face of model-driven engineering, in: Present and Ulterior Software Engineering, Springer, 2017, pp. 103–118, https://doi.org/10.1007/978-3-319-67425-4_7.
[24] E. Petrova, P. Pauwels, K. Svidt, R. Jensen, Towards data-driven sustainable design: decision support based on knowledge discovery in disparate building data, Architect. Eng. Design Manag. 15 (2019) 334–356, https://doi.org/10.1080/17452007.2018.1530092.
[25] M. Rasmussen, M. Lefrançois, P. Pauwels, C. Hviid, J. Karlshøj, Managing interrelated project information in AEC knowledge graphs, Autom. Constr. 108 (2019), https://doi.org/10.1016/j.autcon.2019.102956.
[26] P. Runeson, M. Höst, Guidelines for conducting and reporting case study research in software engineering, Empir. Softw. Eng. 14 (2008) 131–164, https://doi.org/10.1007/s10664-008-9102-8.
[27] J. Steel, R. Drogemuller, B. Toth, Model interoperability in building information modelling, Softw. Syst. Model. 11 (2012) 99–109, https://doi.org/10.1007/s10270-010-0178-4.
[28] J. Wu, H. Sadraddin, R. Ren, J. Zhang, X. Shao, Invariant signatures of architecture, engineering, and construction objects to support BIM interoperability between architectural design and structural analysis, J. Constr. Eng. Manag. 147 (2021), https://doi.org/10.1061/(ASCE)CO.1943-7862.0001943.
[29] N. Yabuki, T. Aruga, H. Furuya, Development and application of a product model for shield tunnels, in: 30th ISARC, 2013, pp. 435–447, https://doi.org/10.22260/ISARC2013/0047.
[30] Q. Yang, Y. Zhang, Semantic interoperability in building design: methods and tools, Comput. Aided Des. 38 (2006) 1099–1112, https://doi.org/10.1016/j.cad.2006.06.003.
[31] A. Zolotas, H.H. Rodriguez, D.S. Kolovos, R.F. Paige, S. Hutchesson, Bridging proprietary modelling and open-source model management tools: the case of PTC integrity modeller and epsilon, in: 20th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems (MODELS), IEEE, 2017, pp. 237–247, https://doi.org/10.1109/MODELS.2017.18.

Galina Paskaleva, M.Sc. from TU Wien, is a researcher at Montanuniversität Leoben at the chair of Subsurface Engineering. She is an architect and a student of informatics. Her main research focus is the application of model-driven software development methods to the data exchange in the domains of Architecture, Engineering and Construction. She is a member of the development team for a multi-stakeholder IT-environment SIMULTAN at TU Wien.

Alexandra Mazak-Huemer, is a professor at Montanuniversität Leoben at the chair of Subsurface Engineering. Before she headed Module 3 in the Christian Doppler Laboratory for Model-Integrated Smart Production (CDL-MINT) at JKU Linz. Her research field focuses on digital transformation in tunnel information modeling as well as in modeling of production systems. Her research interests are on information integration, model-driven engineering and model-based data analytics. She headed numerous interdisciplinary national funded projects in the field of digital transformation.

Manuel Wimmer is a full professor leading the Institute of Business Informatics - Software Engineering at the Johannes Kepler University Linz and he is the head of the Christian Doppler Laboratory CDL-MINT. His research interests comprise foundations of model engineering techniques as well as their application in domains such as tool interoperability, legacy modeling tool modernization, model versioning and evolution, and industrial engineering.

Thomas Bednar, M.Sc. and PhD from TU Wien, is a full professor for Building Physics at TU Wien. His main research focus are prediction models for hygrothermal and acoustic performance of buildings. He has written many papers related to energy use, moisture and building acoustics both on calculations and measurements and is now finishing projects on plus-plus energy high rise office buildings or safeguarding wooden beam ends after renovations of brick-buildings from 1900. He is head of the development of multi-stakeholder IT-environment SIMULTAN at TU Wien for developing and understanding the built environment.

5 A View on Model-Driven Vertical Integration: Alignment of Production Facility Models and Business Models

B. Wally, C. Huemer and A. Mazak;
Proceedings of the 13th IEEE International Conference on Automation Science and Engineering (CASE), IEEE, (2017), pp. 1012–1018.
DOI: 10.1109/COASE.2017.8256235

2017 13th IEEE Conference on Automation Science and Engineering (CASE)
Xi'an, China, August 20-23, 2017

A View on Model-Driven Vertical Integration: Alignment of Production Facility Models and Business Models

Bernhard Wally[1], Christian Huemer[1] and Alexandra Mazak[2]

***Abstract*— Smart manufacturing requires deeply integrated IT systems in order to foster flexibility in the setup, rearrangement and use of attached manufacturing systems. In a vertical integration scenario, IT systems of different vendors might be in use and proprietary interfaces need to defined in order to allow the exchange of relevant information from one system to another. In this paper we present a model-driven approach for vertical integration of IT systems. It is based on the application of industry standards for the representation of hierarchy level specific system properties and an alignment of their key concepts in order to provide bridging functions for the transformation between the different systems.**

I. INTRODUCTION

Quite a few names have been coined for the concept of deeply integrated, networked manufacturing systems: industrial internet of things, cyber-physical production systems, digital production, smart manufacturing, or Industrie 4.0—just to name a few. They all share the common vision of the automation and individualization of the complete manufacturing process—from product description, over order processing and production to product delivery. To make this vision a reality, different business partners are required that execute specific processes and provide these capabilities as services. Abstractly speaking, two kinds of system integration are required: *horizontal integration* for the linking of systems on the same hierarchy level and for seamless communication between different parties and *vertical integration* for the integration within one partner, from the business floor to the shop floor.

Model-driven engineering (MDE) has developed a rich palette of tools and techniques for the description and manipulation of software models. Formalized cross-disciplinary engineering is supported by translating between the different engineering fields through common meta-metamodels and clearly specified sets of operations for model-to-model transformations, model validations, model querying, etc.

In this work, we will showcase the application of MDE techniques in the field of automated production systems (aPS) and their implications up to the business layer. With regards to "digital production", the externalizing of internal processes through services that can be queried becomes more and more important. Flexible automation systems that can be adapted much faster than it has been the case in the past require rapid adaptation of corresponding business models in order to provide up-to-date service descriptions.

[1]Bernhard Wally and Christian Huemer are with the Business Informatics Group, Institute of Software Technology and Interactive Systems, TU Wien, 1040 Vienna, Austria `{wally, huemer}@big.tuwien.ac.at`.
[2]Alexandra Mazak is affiliated with the Christian Doppler Laboratory MINT at TU Wien funded by the Austrian Federal Ministry of Science, Research and Economy (BMWFW) `mazak@big.tuwien.ac.at`.

II. RELATED WORK

Since this paper is cross-disciplinary, we present the related work in several subsections, starting with information about the chosen metamodels and their features, followed by aligning the contributions of this paper with related work on *vertical integration*, *metamodel alignment*, and *model co-evolution*.

A. CAEX and AutomationML

Computer Aided Engineering Exchange (CAEX) is a data format that has been defined in the scope of IEC 62424:2008 and provides structures (i) for information exchange between Piping and Instrumentation Diagram (P&ID) tools and Process Control Engineering (PCE) related Computer Aided Engineering (CAE) tools, as well as (ii) for the representation of PCE requests in P&I diagrams [1]. CAEX is based on XML[1] and enables the metamodeling and modeling of e.g., the hierarchical architecture of a plant, including involved machines and controllers and their physical and logical connections.

IEC 62714 is based on CAEX and defines sets of role classes and interface classes with certain restrictions regarding their application [2], [3]. It is more commonly known as Automation Markup Language (AutomationML, AML), which is the term we will use in the remainder of this paper. AML defines an abstract interface class `ExternalDataConnector` which is used to reference external documents and elements therein. Two use cases of this external data connector have been defined so far in separate whitepapers: (i) `COLLADAInterface` specifies how external COLLADA[2] documents are referenced [4] and (ii) `PLCopenXMLInterface` defines how PLCopen[3] XML documents can be referenced from AML documents [6]. These whitepapers provide a rough guideline on the referencing and integration of external data into AML documents and serve as a starting point for the work presented here.

[1]cf. https://www.w3.org/TR/2008/REC-xml-20081126/
[2]COLLADA—Collaborative Design Activity: an XML based exchange format for 3D assets (cf. https://www.khronos.org/collada/).
[3]PLCopen is a vendor- and product-independent association active in industrial control (cf. http://www.plcopen.org/). PLCopen XML is a data exchange format for the storage of PLC program information according to IEC 61131-3 [5].

978-1-5090-6781-7/17/$31.00 ©2017 IEEE

B. ISA-95 and B2MML

ISA-95 is a series of standards that addresses the integration of the enterprise domain with the manufacturing and control domains. It defines a set of object models for the exchanging of information between these domains—it provides a standard terminology and set of concepts for system integration [7]. The relevant part of ISA-95 for this work is part 2, as it is specified in IEC 62264-2:2013 [8].

Part 2 of ISA-95 specifies common objects and attributes, mainly by a set of commented UML[4] class diagrams, that can be roughly differentiated between (i) basic resources that depict the static definitions of an enterprise with regards to its production facilities (e.g., personnel, equipment, and material) and (ii) operations management information that resembles operational data (e.g., operations capabilities, schedules, and performance).

An XML serialization of ISA-95 has been defined in [9], the business to manufacturing markup language (B2MML). The current version of B2MML is compliant with the current version of ISA-95 and has been used in [10] to link ISA-95 information into AML models.

C. Resource-Event-Agent

Resource-Event-Agent (REA) was coined in the early 80's, by consolidating the then current ideas of accounting research within a unified framework [11]. In its initial and very condensed form, REA describes three concepts: economic resources, economic events, and economic agents. Following accounting theory, an economic event resembles something that has actually happened and that causes a record in the general ledger, such as paying for raw material, receiving money for selling finished products, or renting offices.

The initial REA model was extended and refined to a more complete business ontology comprising new types of events for production and a planning layer that allows the specification of contracts, schedules, policies, etc. [12], [13], [14], [15]. REA thus resembles a link between vertical integration (supports the modeling of production processes) and horizontal integration (supports the modeling of resource exchange and internal/external business modeling).

The runtime configurability of well-designed REA-based information systems has been successfully demonstrated in [16], by running a configurable retail information system (RIS) [17]. The RIS domain model can be evolved by adding/removing/altering types, objects, and their properties.

D. Type-Object Modeling Pattern

As a modeling pattern, the type-object (or power type) pattern [18], [19], [20] is often used in runtime configurable systems in order to allow the dynamic creation and manipulation of classifiers (the types) and instances thereof (the objects). The type-object pattern is one solution to the three-level case of multi-level software engineering [21], [22]. The type-object pattern is very convenient for a number of use-cases, which is why it has been adopted in all the previously mentioned technologies: in REA, in ISA-95 (there it is called class layer), and in AML (system unit classes and role classes can be used for creating instances in the type layer).

For REA, the type-object implementation has best been described in [23], [15], [17]. In short, the type-object pattern proposes to provide a type class and an object class for the specification of a specific entity type and its instances. E.g., in order to support multiple types of agents and instances of these agents, a class `Agent Type` and a class `Agent` would be defined. Each `Agent` instance (e.g., a person called "John Smith" who is employed as a salesman at a company) would be associated with the specific `Agent Type` instance "Salesman". That way, a new type of agent (e.g., "Cashier") could be added at runtime by creating a new instance of `Agent Type`; then, `Agent` instances could impersonate this type of agent.

E. Model-Driven Vertical Integration

The importance of vertical integration in production processes is emphasized in [24] and [25], where a domain specific modeling language, derived from *business process model and notation*[5], is introduced.

In [26] some key challenges for software evolution in aPS are collected, including the co-evolving of interdisciplinary engineering models, which is the challenge addressed in this paper. One of the research goals stated in [26] is the development of automatic consistency mechanisms for domain specific systems. With our approach we can contribute to this research goal.

System evolution in the context of aPS and information systems (IS) is investigated in [27]: (i) hardware changes in a pick and place unit require an evolution of the the state chart model as well as changes at the code level and (ii) the migration of IS components to the cloud demands changes in deployment, configuration, etc. Their approach stresses the importance of architecture models (from IS to control systems) and their use in the estimation of change effort estimation and impact analysis. The examples given comprise manual co-evolution of different systems based on changes in the architectural model. In our approach (i) changes in one system should automatically propagate to other systems, where this is possible and (ii) the overall architectural model is not modeled explicitly, but to be inferred from multiple domain models.

Integration of the various models in aPS is studied in [28] by employing a linking metamodel that allows the explicit linking of model elements from different modeling domains in order to track consistency, constraint satisfaction, etc. Their approach could be used in conjunction with the approach presented in this paper by providing explicit mappings between model elements and not relying solely on the metamodel level. It would be worthwhile to investigate an integration of their linking metamodel into our approach.

Co-evolution of production system models and their libraries is examined in [29], where AML models and their

[4] cf. http://www.omg.org/spec/UML/

[5] cf. http://www.omg.org/spec/BPMN/

AML model library prototypes are checked for different kinds of inconsistencies. Repair operations are executed or warning messages are displayed using the Epsilon Validation Language[6] (EVL). The tools and techniques used in their work overlap with what we have used in our approach, however in contrast to their work, our approach targets the comparison and balancing of diverging metamodels with different main focuses.

In [10], an approach for the alignment of ISA-95 and AML has been proposed, where the different metamodel elements are matched against each other, and a technique is presented to reference ISA-95 information from AML documents. Their work can be used as a basis for the formulation of transformation rules between AML and ISA-95.

A high level alignment of a reduced set of REA and ISA-95 has been presented in [30] and [31], showcasing an application scenario where vertical and horizontal integration are brought together to provide an integrated engineering view on internal processes and external dependencies. Parts of their work are used as a basis for the transformation rules presented in this work.

In [32], the usage of ISA-95 as task layer execution script is explored, and a high level alignment of REA and ISA-95 is presented. The idea is to rely on ISA-95 for representing detailed production information and provide only relevant information to the business layer. In their work, high level elements of ISA-95 are aligned with REA, while lower level elements of ISA-95 are used to describe aspects that are beyond the usual application of REA. For our approach this means that these lower level concepts are usually not transformed between ISA-95 and REA, but might be relevant for the information exchange between ISA-95 and AML.

III. Alignment of AML, ISA-95 and REA

Given the alignments already defined in [10] (AML and ISA-95) and [30], [31], [32] (REA and ISA-95), we want to showcase how specific entities and their properties propagate between the different systems of interest in order to keep then in sync. Fig. 1 depicts a high level view on the various alignments that are involved in the translation between the systems in vertical integration scenarios.

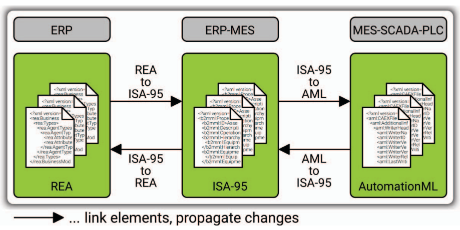


Fig. 1. Overview of model links defined for the purpose of vertical integration.

[6]cf. https://www.eclipse.org/epsilon/doc/evl/

In the remainder, we are using the Epsilon Object Language[7] (EOL) for querying model states, Epsilon Transformation Language[8] (ETL) for model-to-model transformations, and EVL for the validation of models.

Given an entity PT_{ISA95} that resembles an ISA-95 Material Definition with the ID attribute set to "Pine Timber":

- in AML this would be correspondingly instantiated as PT_{AML} of type SystemUnitClass with a reference to RoleClass "ProductStructure",
- in REA, this would be resembled as entity PT_{REA} of type Resource Type.

In order to determine whether a given entity exists in one of the other system levels, we can query its state space and look for a corresponding item. In the case of PT_{ISA95}, we can look into AML documents and query the registered SystemUnitClasses for an attribute with the name "ID" and a matching value. It could also be that the link between ISA-95 and AML has been explicitly defined by an ExternalDataReference of a SystemUnitClass to a B2MML document or element representing PT_{ISA95}.

In the case of REA, the approach would be similar: the REA model would be searched for a Resource Type with a name or "ID" attribute matching "Pine Timber". The link could also be explicitly defined in either the REA element using an attribute "ISA-95 ID" that would hold the value "Pine Timber" (cf. Lst. 1). Vice versa, the link could also be defined in ISA-95, by adding a Material Definition Property with the ID "REA ID" and the corresponding value.

Lst. 1. Querying the REA model for Material Definitions with a matching name or "ISA-95 ID" attribute, expressed in EOL.

```
var needle = "Pine Timber";
var supn = "Material Definition";
for( m in Model.all )
{
   for( rt in m.resourceTypes
      .select( n | not n.superType.isUndefined()
         and n.superType.name = supn
         and n.name = needle ) )
   {
      rt.println( "Found by name: " );
   }
   for( rt in m.resourceTypes
      .select( n | not n.superType.isUndefined()
         and n.superType.name = supn
         and not n.attributes.isUndefined()
         and n.attributes.selectOne( a | a.key =
            "ISA-95 ID" ).value.stringValue =
            needle ) )
   {
      rt.println( "Found by attribute: " );
   }
}
```

Using MDE techniques, it is possible to generate stub models in various domains, generated from a single base model. Using this approach, the different domain models can be created from a common, synchronized understanding

[7]cf. https://www.eclipse.org/epsilon/doc/eol/
[8]cf. https://www.eclipse.org/epsilon/doc/etl/

of core concepts. Lst. 2 shows an excerpt of the transformation from an ISA-95 model (exemplified with Material Definitions) into an REA model. The showcased rule creates a new Resource Type and sets its name to the ID of an input Material Definition (line 7). Additionally, the parent Resource Type “superType” is set to the Resource type with name “Material Definition” (line 8–9). Finally, the Resource Type is added to the REA model (line 10). The whole rule is executed only if there exists no other Resource Type with that name already in the REA model (realized through a *guard* in line 5).

Lst. 2. Transformation of Material Definitions to Resource Types, expressed int ETL.

```
 1 rule MaterialDefinition2ResourceType
 2    transform md : isa!MaterialDefinitionType
 3    to rt : rea!ResourceType
 4 {
 5    guard: rea.resourceTypes.select( rt |
 6       rt.name = md.iD.value ).size() == 0
 7    rt.name = md.iD.value;
 8    rt.superType =
 9       rt.getMaterialDefinition( rea );
10    rea.resourceTypes.add( rt );
11 }
```

In order to determine if two models of different domains are in sync, they can be validated against each other. Lst. 3 checks whether an REA model under test contains all the Material Definitions of a given ISA-95 model, and using Epsilon[9] tooling, a human domain expert would get the chance to fix occurring issues with the click of his/her mouse. Lines 5–6 check whether a Resource Type with a name corresponding to the ID of the current Material Definition exists—and if not, it displays the error message defined in lines 7–8: the user interface integration of Epsilon enables executing the fix defined in lines 9–22 (cf. Fig. 2). The fix creates a new Resource Type with a name corresponding to the ID of the current Material Definition (lines 14–15), sets its parent to the generic “Material Definition” Resource Type (lines 16–18) and adds it to the underlying REA model (lines 19–20).

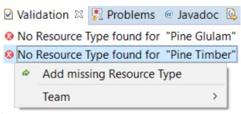


Fig. 2. The error messages of the epsilon validation engine provide a clickable “quick fix” facility.

IV. Application Scenario

An explicit example shall clarify the effect of model-driven synchronized production and business systems under the influence of changes in any of the given subsystems. The use case describes the evolution of involved systems and the

[9]cf. https://www.eclipse.org/epsilon/

Lst. 3. Excerpt of a cross-model validation between ISA-95 and REA, expressed in EVL.

```
 1 context isa!MaterialDefinitionType
 2 {
 3    constraint MaterialDefinitionExists
 4    {
 5       check : rea!ResourceType.all.select( rt |
 6          rt.name = self.iD.value ).size() = 1
 7       message : "No Resource Type found for "+
 8          " \"" + self.iD.value + "\""
 9       fix
10       {
11          title : "Add missing Resource Type"
12          do
13          {
14             var rt = new rea!ResourceType;
15             rt.name = self.iD.value;
16             rt.superType = rea!ResourceType.all
17                .selectOne( md | md.name =
18                "Material Definition" );
19             rea!Model.all.first()
20                .resourceTypes.add( rt );
21          }
22       }
23    }
24 }
```

services they provide: the addition of a new business service offering requires changes in the production facilities. These changes occur at different hierarchy levels, and we show how these changes can be propagated to systems of other hierarchy levels using model-driven engineering techniques.

A. Initial System State

The example is based on a fictitious company “Glulam Ltd.” that has specialized in the production of glued laminated timer (*glulam*). The core production process consists of pieces of timber as raw material that are fed into a continuous finger jointer that produces an endless so called “lamella” that is cut into pieces of required length. Several of these pieces of lamella are then laminated (glued together) to form a thicker piece of wood, a “glulam”, that is often used for building construction work. This production process is depicted in Fig. 3, that roughly correlates with how this process would be modeled in ISA-95 using the Process Segment model. For the sake of simplicity, Personnel Segment Specifications have been omitted.

The corresponding model in REA of the “Lamination” Process Segment is depicted in Fig. 4. It maps to a Transformation Duality Type that consists of a Produce Event Type, a Consume Event Type, and a Use Event Type. The rationale behind this mapping is: a Process Segment is not a specific instance of a production run, but resembles the blueprint for specific production runs (Operations Performances). Similarly, a Transformation Duality Type serves as the blueprint for specific Transformation Dualities that relate Transformation Events to each other. Here, the “Lamination” Transformation Duality Type comprises:

- incremental Produce Event Type “Produce Glulam” that indicates the production of a certain amount of “Glulam” Resource Types,

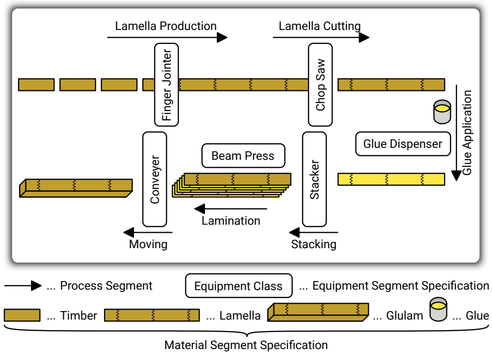


Fig. 3. Overview of the underlying production process of the application scenario in its initial state.

- decremental Consume Event Type "Consume Lamination Material" that explicates the consumption of input material (lamella and glue), and
- decremental Use Event Type "Use Lamination Equipment" that resembles the use of required machinery.

The associated participants for all these Event Types are the "Lamination Operator" and the "Shift Supervisor" Agent Types. In the incremental Event Type the "Lamination Operator" resembles the *providing* Agent Type and the "Shift Supervisor" is the *receiving* Agent Type. In the decremental Event Types the participation roles are inverted.

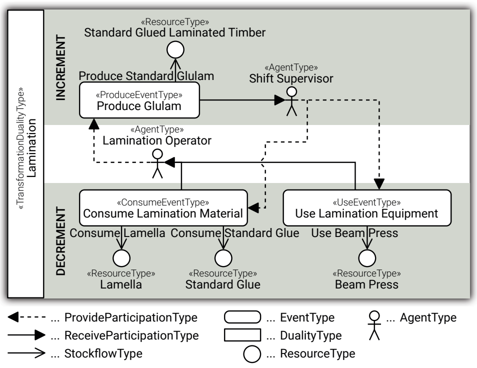


Fig. 4. A view on the REA model of the given application scenario in its initial state.

B. Model-Driven Co-Evolution

So far, the glulam lamination process consisted of pressing the lamellas for a given amount of time. However, recently a new type of glue has been suggested to meet the requirements of some important customers with very specific needs. This type of glue requires a minimum temperature of 25°C while curing; however, it yields a much stronger glulam. Therefore, a heating device is installed next to the beam press in order to ensure the required temperature.

1) Evolution of the business system: For the business system, this means adding three new Resource Types "Heating Device", "Warm Curing Glue", and "Strong Glulam" and deciding on how the new elements should be integrated into the business model. Here, it is decided to make a copy of the "Lamination" Transformation Duality Type called "Minimum Temperature Lamination" and adapt it by adding and changing Stockflow Types (cf. Fig. 5): (i) the glue to be used is now a warm curing glue, (ii) the final product is now a strong glulam, and (iii) a heating device is added to the Use Event Type.

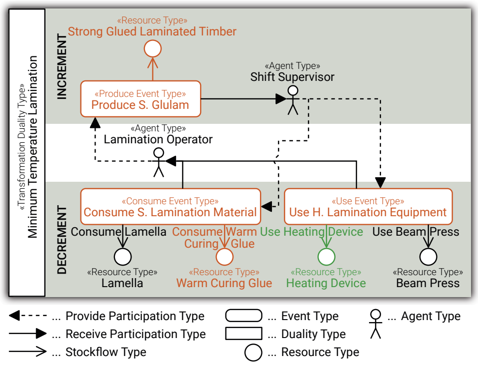


Fig. 5. The REA model of the given application scenario in its evolved state. Green items depict elements that have been added compared to the initial state, orange items resemble elements that have been present in the initial state in a structurally similar way, but that now resemble other entities.

2) Evolution of the ERP-MES transfer model: For the ERP-MES transfer model (expressed in ISA-95), the heating device needs to be integrated into the production process, and this change must also be reflected in the corresponding ISA-95 models, specifically the material model, the equipment model and the process segment model need to be changed. Based on an alignment of the meta-models of REA and ISA-95, an initial skeleton of the ISA-95 model can be created by transforming the REA model. The new Transformation Duality Type is transformed into a Process Segment, the Resource Types into elements of the material and equipment model, and the Stockflow Types into Material Segment Specifications and Equipment Segment Specifications. Some transformations require manual intervention or a specific naming or structuring convention, because it is not intrinsically clear, whether a Resource Type is mapped into a Material Class/Definition, Equipment Class, or Physical Asset Class. Lst. 4 shows an excerpt of a cross-model validation between REA and ISA-95, with the convention that the IDs of ISA-95 elements correspond to names of REA elements. The validation code checks whether all REA Transformation Duality Types have a Process Segment counterpart in ISA-95 and if not, offers a quick fix for generating a skeleton Process

Segment. It does that by traversing all Event Types and adding all of their Stockflow Types as Equipment, Physical Asset, or Material Segment Specification. In order to keep the example concise, only the traversal of Use Event Types is depicted.

Lst. 4. Excerpt of a cross-model validation between REA and ISA-95, expressed in EVL. A backslash (\) denotes a soft line break required to adapt to the line width.

```
context rea!TransformationDualityType
{
   constraint ProcessSegmentExists
   {
      check : isa!ProcessSegmentType.all.exists(
         ps | ps.iD.value = self.name )
      message : "No Process Segment found for "+
         " \"" + self.name + "\""
      fix
      {
         title : "Add missing Process Segment"
         do
         {
            var ps = new isa!ProcessSegmentType;
            ps.iD = new isa!IDType;
            ps.iD.value = self.name;
            for( uet in self.useEventTypes )
            {
               for( sft in uet.stockflowTypes )
               {
                  var ess = new isa!Equipment\
                     SegmentSpecificationType;
                  ess.equipmentClassID = new isa
                     !EquipmentClassIDType;
                  ess.equipmentClassID.value =
                     sft.resourceType.name;
                  ess.equipmentUse = new isa!
                     EquipmentUseType;
                  ess.equipmentUse.value = uet
                     .name + ": " +
                     sft.resourceType.name;
                  ps.materialSegment\
                  Specification.add( ess );
               }
            }
            /* Code intentionally left out */
            isa!ProcessSegmentInformationType
               .all.first().processSegment
               .add( ps );
         }
      }
   }
}
```

3) Evolution of the Plant Topology: The plant topology that is expressed in AML can benefit from the adapted ERP-MES transfer model by following the mapping rules presented in [10]. As a result, SystemUnitClasses for the heating device, strong glulam, warm curing glue, and minimum temperature lamination need to be created. This can again be achieved by a set of transformation rules that transform an ISA-95 model into a corresponding AML model.

V. Critical Discussion

The presented approach provides an initial setup for the transformation between systems of various layers of automated production systems. Some restrictions apply that prohibit a fully automated transformation between the different systems. E.g., the mapping between REA and ISA-95 sports syntactic inter-model heterogeneities including *1:n*, *I2I*, and *BreadthDifference*, following the classification presented in [33], regarding the mapping between REA Resource Types and ISA-95 Material Class/Definition, Equipment Class, and Physical Asset Class. The 1:n heterogeneity causes a decision that needs to be taken in order to determine what kind of output instance should be created for a given input instance. One way to solve this issue is by defining company specific conventions or by providing generic Resource Types (e.g., named “Equipment Class”) that serve as parent types for respective instances.

Another problem that cannot be handled generically are different levels of indirection, such as REA Events, that have no equivalent in ISA-95. While event entities can be skipped when transforming from REA to ISA-95, they cannot be generically created when transforming from ISA-95 to REA: it is not clear, which ISA-95 Segment Specifications should be bundled together in single Event Types as Stockflow Types and Participation Types. Modeling conventions could help in resolving parts of these issues, or additional reasoning steps could be introduced that would provide a meaningful modeling structure for the entities in question.

We have chosen to present the application of our approach on a very specific fictitious company, however, the approach itself and most of its implementation are company and domain agnostic. This can be verified by examining the code listings and asserting that only metamodel classes and features are referenced, except for e.g., the statement in line 18 of Lst. 3, where a company specific modeling convention is exemplified. This is inevitable in some cases in order to e.g., accommodate to a basic set of rules on how company assets are modeled, or to explicitly denote that specific elements of different domains represent in fact the same entity.

We have refrained from presenting details about the transformation between AML and ISA-95 models, as the alignment between these two metamodels has been presented in more detail in [10] already. Corresponding transformation rules can be inferred from the mappings defined there.

The benefit of our approach is that a common understanding of concepts from different domains is accomplished by relating *metamodel* elements with each other. This approach is agnostic to the kind of business a company is involved in. Specific implementations could provide industry related information in order to better acknowledge peculiarities and conventions.

VI. Conclusion and Outlook

We have presented a model-driven approach for the co-evolution of models residing on different levels with respect to the automation hierarchy, based on a generic alignment of corresponding metamodels. While the given technique does not provide a “single point of intervention” when it comes to changes in the models, it facilitates the creation of stub models and provides means for cross-model validation. The main contribution is thus the model-driven propagation of basic model elements and changes of model elements between models of different hierarchy levels. By defining clear rules it might even be possible to overcome some of the main limitations with respect to the automatic propagation of changes. Some of these conventions might be defined in a generic way, others might only be possible in a company's environment.

For the future, we would like to specify the cross-model validations and transformations more exhaustively in order to provide a rather complete setup for the co-evolution of models situated in different hierarchy levels in production systems, with the goal of vertical integration. Further, we plan the incorporation of communication stacks such as OPC Unified Architecture[10]. Here, we would like to investigate cross-domain model mining based on runtime information captured in structured communication streams. E.g., the business layer could be adapted via ISA-95 based on operations performance information sourced in the production process.

References

[1] International Electrotechnical Commission (IEC), *Representation of process control engineering—Requests in P&I diagrams and data exchange between P&ID tools and PCE-CAE tools*, International Standard, Rev. 1.0, August 2008, IEC 62424:2008.

[2] International Electrotechnical Commission (IEC), *Engineering data exchange format for use in industrial automation systems engineering—Automation markup language—Part 1: Architecture and general requirements*, International Standard, Rev. 1.0, June 2014, IEC 62714-1:2014.

[3] International Electrotechnical Commission (IEC), *Engineering data exchange format for use in industrial automation systems engineering—Automation markup language—Part 2: Role class libraries*, International Standard, Rev. 1.0, March 2015, IEC 62714-2:2015.

[4] AutomationML consortium, *AutomationML Whitepaper Part 3—Geometry and Kinematics*, Whitepaper, Rev. 2.0, August 2015, AML Part 3:2015.

[5] PLCopen Technical Committee 6, *XML Formats for IEC 61131-3*, Technical Paper, Rev. 2.01, May 2009, PLCopen XML:2009.

[6] AutomationML consortium, *AutomationML Whitepaper Part 4—AutomationML Logic Description*, Whitepaper, Rev. 2.0, May 2010, AML Part 4:2010.

[7] International Electrotechnical Commission (IEC), *Enterprise-control system integration—Part 1: Models and terminology*, International Standard, Rev. 2.0, May 2013, IEC 62264-1:2013.

[8] International Electrotechnical Commission (IEC), *Enterprise-control system integration—Part 2: Objects and attributes for enterprise-control system integration*, International Standard, Rev. 2.0, June 2013, IEC 62264-2:2013.

[9] Manufacturing Enterprise Solutions Association (MESA) International, *Business To Manufacturing Markup Language*, Standards and Tools, Rev. V0600, May 2013.

[10] B. Wally, C. Huemer, and A. Mazak, "Entwining plant engineering data and ERP information: Vertical integration with AutomationML and ISA-95," in *Proceedings of the 3rd IEEE International Conference on Control, Automation and Robotics (ICCAR 2017)*, 2017.

[11] W. E. McCarthy, "The REA accounting model: A generalized framework for accounting systems in a shared data environment," *The Accounting Review*, vol. 57, no. 3, pp. 554–578, 1982.

[12] G. L. Geerts and W. E. McCarthy, "Modeling business enterprises as value-added process hierarchies with resource-event-agent object templates," in *Business Object Design and Implementation*, 1997.

[13] G. L. Geerts and W. E. McCarthy, "An accounting object infrastructure for knowledge-based enterprise models," *IEEE Intelligent Systems*, vol. 14, no. 4, pp. 89–94, 1999.

[10]cf. https://opcfoundation.org/about/opc-technologies/opc-ua/

[14] G. L. Geerts and W. E. McCarthy, "The ontological foundation of REA enterprise information systems," in *Annual Meeting of the American Accounting Association, Philadelphia, PA*, vol. 362, 2000.

[15] G. L. Geerts and W. E. McCarthy, "Policy-level specifications in REA enterprise information systems," *Journal of Information Systems*, vol. 20, no. 2, pp. 37–63, 2006.

[16] B. Wally, A. Mazak, B. Kratzwald, C. Huemer, P. Regatschnig, and D. Mayrhofer, "REAlist—a tool demo," in *Proc. of the 9th Int. Workshop on Value Modeling and Business Ontology*, 2015.

[17] B. Wally, A. Mazak, B. Kratzwald, and C. Huemer, "Model-driven retail information system based on REA business ontology and Retail-H," in *Proceedings of the 17th IEEE Conference on Business Informatics (CBI 2015)*, vol. 1, 2015, pp. 116–124.

[18] P. Coad, "Object-oriented patterns," *Communications of the ACM*, vol. 35, no. 9, pp. 152–159, 1992.

[19] R. Johnson and B. Woolf, "Type object," in *Pattern Languages of Program Design 3*, R. C. Martin, D. Riehle, and F. Buschmann, Eds. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1997, ch. Type Object, pp. 47–65.

[20] Object Management Group, Inc., *OMG Unified Modeling Language (OMG UML)*, Specification, Rev. 2.5, March 2015.

[21] C. Atkinson and T. Kühne, "The essence of multilevel metamodeling," in *UML 2001—The Unified Modeling Language. Modeling Languages, Concepts, and Tools*, ser. Lecture Notes in Computer Science, 2001.

[22] T. Kühne and F. Steimann, "Tiefe Charakterisierung," in *Modellierung 2004*, ser. Lecture Notes in Informatics, B. Rumpe and W. Hesse, Eds., vol. P-45. Gesellschaft für Informatik, 2004, pp. 109–120.

[23] G. L. Geerts and W. E. McCarthy, "An ontological analysis of the economic primitives of the extended-REA enterprise information architecture," *International Journal of Accounting Information Systems*, vol. 3, no. 1, pp. 1–16, 2002.

[24] M. Witsch and B. Vogel-Heuser, "Towards a formal specification framework for manufacturing execution systems," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 2, pp. 311–320, May 2012.

[25] M. Witsch, "Funktionale Spezifikation von Manufacturing Execution Systems im Spannungsfeld zwischen IT, Geschäftsprozess und Produktion," Ph.D. dissertation, Fakultät für Wirtschaftswissenschaften, Technische Universität München, 2013.

[26] B. Vogel-Heuser, S. Feldmann, J. Folmer, J. Ladiges, A. Fay, S. Lity, M. Tichy, M. Kowal, I. Schaefer, C. Haubeck, W. Lamersdorf, T. Kehrer, S. Getir, M. Ulbrich, V. Klebanov, and B. Beckert, "Selected challenges of software evolution for automated production systems," in *Proc. of the 13th IEEE Int. Conf. on Industrial Informatics*, 2015.

[27] B. Vogel-Heuser, S. Feldmann, J. Folmer, S. Rösch, R. Heinrich, K. Rostami, and R. Reussner, "Architecture-based assessment and planning of software changes in information and automated production systems state of the art and open issues," in *Proc. of the 2015 IEEE Int. Conference on Systems, Man, and Cybernetics (SMC 2015)*, 2015.

[28] S. Feldmann, M. Wimmer, K. Kernschmidt, and B. Vogel-Heuser, "A comprehensive approach for managing inter-model inconsistencies in automated production systems engineering," in *Proceedings of the 12th IEEE International Conference on Automation Science and Engineering (CASE 2016)*, 2016, pp. 1120–1127.

[29] L. Berardinelli, S. Biffl, E. Maetzler, T. Mayerhofer, and M. Wimmer, "Model-based co-evolution of production systems and their libraries with AutomationML," in *Proceedings of the 20th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2015)*, 2015.

[30] A. Mazak and C. Huemer, "From business functions to control functions: Transforming REA to ISA-95," in *Proceedings of the 17th IEEE Conference on Business Informatics (CBI 2015)*, D. Aveiro, U. Frank, K. J. Lin, and J. Tribolet, Eds., vol. 1. IEEE, 2015.

[31] A. Mazak and C. Huemer, "HoVer: A modeling framework for horizontal and vertical integration," in *Proceedings of the 13th IEEE International Conference on Industrial Informatics (INDIN 2015)*. IEEE, 2015, pp. 1642–1647.

[32] B. Wally, C. Huemer, and A. Mazak, "ISA-95 based task specification layer for REA in production environments," in *Proceedings of the 11th International Workshop on Value Modeling and Business Ontologies (VMBO 2017)*, 2017.

[33] M. Wimmer, G. Kappel, A. Kusel, W. Retschitzegger, J. Schoenboeck, and W. Schwinger, "Towards an expressivity benchmark for mappings based on a systematic classification of heterogeneities," in *Proceedings of the First International Workshop on Model-Driven Interoperability*, ser. MDI '10. New York, NY, USA: ACM, 2010, pp. 32–41.

6 Flexible Production Systems: Automated Generation of Operations Plans Based on ISA-95 and PDDL

B. Wally, J. Vyskocil, P. Novák, C. Huemer, R. Sindelár, P. Kadera, A. Mazak-Huemer and M. Wimmer;
IEEE Robotics and Automation Letters (RA-L), IEEE, 4, (2019), pp. 4062–4069.
DOI: 10.1109/LRA.2019.2929991

4062 IEEE ROBOTICS AND AUTOMATION LETTERS, VOL. 4, NO. 4, OCTOBER 2019

Flexible Production Systems: Automated Generation of Operations Plans Based on ISA-95 and PDDL

Bernhard Wally, Jiří Vyskočil, Petr Novák, Christian Huemer, Radek Šindelář, Petr Kadera, Alexandra Mazak, and Manuel Wimmer

***Abstract*—Model-driven engineering (MDE) provides tools and methods for the manipulation of formal models. In this letter, we leverage MDE for the transformation of production system models into flat files that are understood by general purpose planning tools and that enable the computation of "plans", i.e., sequences of production steps that are required to reach certain production goals. These plans are then merged back into the production system model, thus enriching the formalized production system knowledge.**

***Index Terms*—AI-based methods, factory automation, intelligent and flexible manufacturing.**

I. Introduction

MANUFACTURING systems of the future are required to be more and more flexible, regarding both the products they produce and the production systems themselves [1], [2]. According to the principles of smart manufacturing, products and their recipes are not required to be known at design time, product variants may be edited at runtime, and production planning and scheduling are to be invoked on-the-fly, when a new production order appears. As such, the use of automated planning systems seems very natural, however, current commercial industrial planning systems are not sufficient [3].

Processing production orders on-the-fly means that a flexible manufacturing line does not need to be in a predefined initial state before starting a new production. Moreover, the manufacturing line can even be already producing other orders, and thus the state of all resources such as shuttles on a transportation system, or locations of material can vary. Moving these shuttles back to an artificial initial state, as it is done in industrial practice currently, would mean time and energy loss that could and should be avoided. Such high degrees of freedom disqualify traditional ways of programming manufacturing lines and strengthen the need for using automated planning systems being able to react on changing initial conditions and targets. Further, a declarative way of programming related to planners and industrial specification languages is essential for fulfilling the challenging demands of smart manufacturing systems.

In this letter, we are presenting a model-driven approach to automatically transform a manufacturing system specification to a production plan via automated planning. To formulate the manufacturing line planning task, a specification of all industrial components and their actions and interactions is needed. In this environment a number of methods, tools and standards are well established:

1) *Production systems engineering*: specification of industrial components and processes using industry standards or domain specific modeling languages [4], [5];
2) *Model-driven engineering* (MDE): generic methods for the specification of discrete models, their validation, manipulation and transformation, etc. [6]–[8];
3) *Automated reasoning*: methods for realizing reasoning tasks, covering various classes of problems with different computational complexities, from NP-complete propositional logic, EXPSPACE-complete classical planning, semi-decidable first-order logic, up to undecidable halting problems [9]–[11].

Our proposed approach covers both the engineering phase of systems as well as their runtime. With respect to the engineering phase, it can be considered as a verification tool for the fulfillment of functional requirements, with respect to the runtime it can make use of automated reasoning in order to find sequences of production steps to reach initially unknown production system states or to produce products that have not been known at design time. The overview of the proposed approach is depicted in Fig. 1.

This contribution is structured as follows: after a discussion of related work in Section II, Section III describes the rules for transforming production system models into planning problems, while Section IV discusses their application on a specific use case. Section V presents concrete problem statements as well as

Manuscript received February 15, 2019; accepted June 27, 2019. Date of publication July 22, 2019; date of current version August 8, 2019. This letter was recommended for publication by Associate Editor A. Agostini and Editor T. Asfour upon evaluation of the reviewers' comments. This letter was supported in part by the Austrian Federal Ministry for Digital and Economic Affairs and the Austrian National Foundation for Research, Technology and Development, in part by the DAMiAS project funded by the Technology Agency of the Czech Republic, in part by the H2020 project DIGICOR, and in part by the OP VVV DMS project Cluster 4.0. *(Corresponding author: Bernhard Wally.)*

B. Wally, R. Šindelář, A. Mazak, and M. Wimmer are with the CDL for Model-Integrated Smart Production, Department of Business Informatics – Software Engineering, Johannes Kepler University Linz, Linz 4040, Austria (e-mail: bernhard.wally@jku.at; radek.sindelar@jku.at; alexandra.mazak@jku.at; manuel.wimmer@jku.at).

J. Vyskočil, P. Novák, and P. Kadera are with the Department of Intelligent Systems for Industry and Smart Distribution Networks, Czech Institute of Informatics, Robotics, and Cybernetics, Czech Technical University in Prague, Prague 160 00, Czech Republic (e-mail: jiri.vyskocil@cvut.cz; petr.novak@cvut.cz; petr.kadera@cvut.cz).

C. Huemer is with the Business Informatics Group, Institute of Information Systems Engineering, Technische Universitat Wien, Vienna 1040, Austria (e-mail: huemer@big.tuwien.ac.at).

Digital Object Identifier 10.1109/LRA.2019.2929991

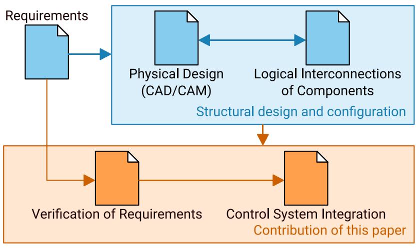


Fig. 1. High-level view on the proposed approach.

performance data of our approach. Finally, Section VI concludes and provides hints for future research directions.

II. Related Work

A. Model-Driven Engineering

Model-driven engineering (MDE) has long been investigated and practiced. In the early 2000s, standardization efforts finally culminated in widely adopted standards such as the Meta Object Facility and the Unified Modeling Language (UML) [6]. Similar to the object-oriented paradigm of the 1980s ("everything is an object") the new paradigm was "everything is a model" [7]. Based on this foundation, MDE tools may impose domain-specific constraints and perform model checking that can detect and prevent many errors early in the life cycle [8]. This is exactly the main reason why we employ MDE techniques in this letter. We aim to formally provide knowledge of a production system and use this knowledge to verify certain properties thereof.

MDE in the context of smart production is, of course, not new. For instance, IEC 62264 (also known as ISA-95) is a series of international standards describing data structures, activities and a communication protocol in the field of manufacturing execution systems (MES) and their interfaces with enterprise resource planning (ERP) systems [4]. Specifically, parts 2 and 4 of of ISA-95 define a set of UML-based metamodels that enable the modeling of MES related information [12], [13]. With AutomationML[1] a standardized data format was introduced for representing engineering information in the area of process automation and control [5]. An integration layer for ISA-95 and AutomationML [14] has been presented in [15] and [16], enabling AutomationML to act as a container format for encoding ISA-95 information.

Model-driven transformation of transportation system knowledge from the proprietary tool PX5 Configurator has been discussed in [17], where it was converted into AutomationML before being further processed within another proprietary simulation tool. While we are also using the PX5 Configurator in this letter in our case study, we are not implementing a toolchain to achieve integration between two proprietary tools, but we define generic transformation rules between standardized (modeling) languages.

Model-driven alignment of structural production system information was further presented in [18]–[20], where business models were aligned to MES models. This letter could be an interesting extension to our work, when it comes to the integration of business information. A compatible ERP-like system has been presented in [21].

B. Automated Planning

Automated planning is a branch of artificial intelligence that deals with the issue of finding *plans*, which are strategies or sequences of actions. Typical application scenarios are, e.g., plans that are executed by autonomous robots [22]. "Classical" planning [23] describes automated planning where a set of assumptions and restrictions have to hold.

The worst case complexity of classical planning is EXPSPACE-complete, for plan existence problems [24]. On one hand, many planning systems allow to relax some of the "classical" properties that can even lead to semi-decidability [24] of plan existence problems. On the other hand, many well-known planning problems are typically much easier (NP-complete or even better) [25].

The Planning Domain Definition Language (PDDL) is a standardized classical planning language that has been used for approximately 20 years at the international planning competitions.[2] In PDDL, the planning problem is divided into two parts: (i) the *domain* part holds all available predicates as well as the allowed actions on the state-space with preconditions and effects and (ii) one or more related *problem* part, which define the initial state and the goal state conditions. PDDL solvers then try to find sequences of actions that lead from the initial state to the goal state.

Only some of the automated reasoning methods can be utilized for (semi-)automated solutions at industrial scale. Therefore, we focused on PDDL-based classical planning problems in this letter. The latest version of the language is PDDL 3.1 [26] but there exist many variants/extensions that support various features like goal rewards, probabilistic effects, multi-agent planning, temporal planning, etc. An overview of several extensions of PDDL including explanation of techniques in successful solvers is provided in [27].

A study on usage of PDDL for a collection of typical basic industrial problems is presented in [3]. Compared to [3], the approach proposed in this letter (i) utilizes PDDL as an intermediate format rather than a tool for direct modeling by experts, (ii) we use a real system of industrial scale, and (iii) we are focused on classical (i.e., non-temporary) planners due to the efficiency. While a "bridge between automation and AI planning" is described in [3], we are enhancing this concept by incorporating a standardized specification and its translation into automation and AI planning. Various systems for executing production plans have been proposed. Some of them are discussed in [28] that mainly addresses execution of production plans based on PDDL.

[1]cf. https://www.automationml.org/

[2]http://www.icaps-conference.org/index.php/Main/Competitions

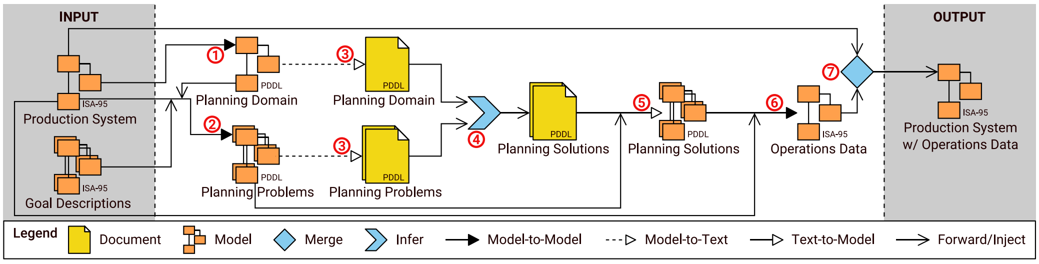


Fig. 2. Implemented core workflow of our approach. Input files need to be provided, all other artifacts are generated automatically.

C. Synopsis

While both, model-driven engineering and automated planning, have been applied to industrial engineering, we are not aware of any particular approach which allows for automated planning solely on the model-based domain representations such as provided by ISA-95. Our proposed approach uses PDDL in a fully transparent way, i.e., the input needed by PDDL solvers is fully derived from the model and also the output provided by PDDL solvers is automatically translated back to the model. Thus, design-time and runtime decisions can be performed by domain experts without requiring knowledge about the underlying solver technology.

III. Model-Driven Engineering of Flexible Production Systems

Based on the structural description of a production plant sequences of actions shall be derived that enable reaching certain production goals. We have tackled this task by leveraging (i) ISA-95 models of production systems as input and output models and (ii) PDDL as the technology for inferring sequences of actions. For this letter, the most important concepts of ISA-95 are the *equipment* and *process segment* models. Among other entities, they provides concepts for describing the machinery available in production environments such as robots, transportation systems, etc., as well as structures depicting the production steps that can be performed using the equipment. In Section III-A we describe the general approach; a detailed description follows in Section III-B, while a concrete example is presented in Section V.

A. Approach

Our model-driven approach requires the formulation of metamodels for the involved domain models. Therefore, we have created metamodels (i) for ISA-95, following the specification given in the standards' documents and (ii) for PDDL 3.1, based on the Backus–Naur form given in [26].

We are taking two input files into account: (i) an ISA-95 model describing the production system (including equipment, material, process segments, and resource connections) and (ii) one or more ISA-95 models that describe the envisioned goal states. We will show in Section IV-A how the production system model can be automatically derived from a proprietary source model (this is an optional pre-processing step). The output is an ISA-95 model that is derived from the initial ISA-95 model, but now includes information about operations definitions. The applied "core" workflow is depicted in Fig. 2, the individual processing steps (circled numbers) are described below, accordingly.

1) **Production system → Planning domain:** the production system is parsed and relevant information extracted and transformed into PDDL domain concepts.
2) **(Production system + Goal descriptions) → Planning problem:** the production system is parsed and relevant information extracted and transformed into the initial state of a PDDL problem. For each goal description that is provided, a separate planning problem is created, with the corresponding goal specifications. The initial state of these planning problems is reused from the initially created PDDL problem.
3) **PDDL code generation:** so far, the planning domain and problems have been described by means of models. In this step, the models are serialized as plain-text PDDL documents that can be read by standard-conforming PDDL solvers.
4) **PDDL solving:** for each planning problem a planning solution is calculated by a PDDL solver. If no solution could be found for certain problems, this is also recorded. The solutions are created as plain-text files.
5) **Planning solutions → Planning solution models:** the plain-text files are "reverse-engineered" into formal PDDL models in order to be useable in the subsequent processing steps.
6) **PDDL solution models → Operations data:** the sequence of actions found by the solver is transformed into operations that are collected in an ISA-95 model.
7) **(Production system + Operations data) → Integrated model:** the original production system model and the operations data model are merged into a single ISA-95 model containing both the static production system information and the behavioral information of goal-oriented production steps.

Since step 6 generates operations data, it might be desired to generate this data at runtime instead of at design time, in order to enable flexible production systems that are able to compute production plans online. Fortunately, our approach can be applied at design time and at runtime.

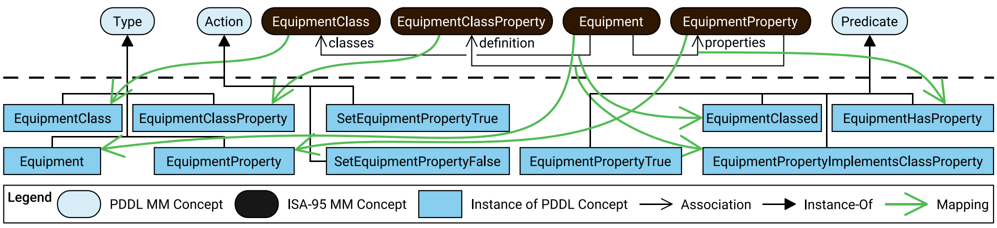


Fig. 3. Mapping of ISA-95 metamodel elements to a PDDL domain description.

B. Implementation

We have implemented the workflow previously described based on metamodels of ISA-95 and PDDL that have been formalized using Ecore/EMF.[3] However, our approach could have been realized using any capable technology, including, e.g., ontologies. The transformation of the initial production system information into a planning domain model has been realized threefold, as described in the following two sub-sections for generic information and in Section IV-B for domain-specific concepts. Our approach assumes that ISA-95 ProcessSegments are defined in a way that they refer to EquipmentClasses rather than to Equipment, and that the runtime information uses pieces of Equipment rather than EquipmentClasses. This is typically the case.

1) Metamodel Concepts: relevant metamodel concepts of ISA-95 are converted to certain PDDL statements (cf. Fig. 3). (i) relevant metamodel classes (that are used by the ISA-95 model under observation) are implemented as PDDL Types. (ii) ISA-95 associations are converted to PDDL Predicates. (iii) boolean properties are supported by a dedicated Predicate, e.g., `EquipmentPropertyTrue` for equipment properties. (iv) for the manipulation of these properties, two Actions are defined: `SetEquipmentPropertyTrue` and `SetEquipmentPropertyFalse`. These two actions enable explicit setting of boolean equipment properties that are not tagged with the term `pddl:implicit` in their description attribute. A PDDL encoding of these transformed concepts is given in Lst. 1. Lines 15–16 and 21–22 encode information that takes into account instance data: properties that are tagged as being set implicitly must not be supported by the generic `SetEquipmentProperty*` actions.

It is important to note, that this is only one way of encoding an ISA-95 model in PDDL. For instance, boolean properties could instead be translated as specific Predicates and not as objects that are related to equipment instances via generic Predicates.

2) Instance Data: apart from preparing the PDDL environment with generic concept directly inferred from the ISA-95 metamodel, also instance data of the ISA-95 model has an impact on the planning domain description and requires proper mapping (cf. Fig. 4). Examples for the PDDL representation of this mapping are given in Lst. 2, the single mapping statements refer to specific lines in this listing (the instance data used refers to the example given in Section V):

[3]cf. https://www.eclipse.org/modeling/emf/

```
1 (:types EquipmentClass Equipment
2   EquipmentClassProperty EquipmentProperty)
3 (:predicates
4   (EquipmentClassed ?E - Equipment ?C - EquipmentClass)
5   (EquipmentPropertyImplementsClassProperty
6     ?EP - EquipmentProperty
7     ?ECP - EquipmentClassProperty)
8   (EquipmentHasProperty
9     ?E - Equipment ?P - EquipmentProperty)
10  (EquipmentPropertyTrue ?P - EquipmentProperty))
11 (:action SetEquipmentPropertyTrue
12  :parameters (?EP - EquipmentProperty)
13  :precondition (and
14    (not (EquipmentPropertyTrue ?EP))
15    (not (EquipmentPropertyImplementsClassProperty
16        ?EP ECP_PositioningUnitOccupied)))
17  :effect (EquipmentPropertyTrue ?EP))
18 (:action SetEquipmentPropertyFalse
19  :parameters (?EP - EquipmentProperty)
20  :precondition (and (EquipmentPropertyTrue ?EP)
21    (not (EquipmentPropertyImplementsClassProperty
22        ?EP ECP_PositioningUnitOccupied)))
23  :effect (not (EquipmentPropertyTrue ?EP)))
```

Listing 1. Excerpt of the PDDL domain description, showing types, predicates and actions that have been generated from the ISA-95 metamodel.

(i) for each class instance (e.g., instances of EquipmentClass, EquipmentClassProperty), a Constant is created, using the instance's id with a suitable prefix as identifier (lines 2–4). (ii) ProcessSegments are implemented as PDDL Actions, using the id as name (line 7). The process's segment specifications are converted to parameters, as they represent the required resources for the process (line 8). Relevant ISA-95 relations are checked via specific Preconditions, using the Predicates defined in the metamodel mapping (line 11). Segment specification properties are checked for specific tags that need to be implemented in the ISA-95 model in order for the transformation process to behave as expected: if the `pddl:pre` or the `pddl:post` tag is detected, a corresponding Precondition or Effect is created, respectively (lines 12–16 and lines 18–22). Finally, the duration-related attributes of the ProcessSegment are interpreted as cost of the Action, uniformly converted to seconds (line 23).

Again, the presented conversion is just one example of encoding. For instance, if the properties were implemented as specialized predicates (as mentioned earlier) instead of as dedicated objects, the exists statement could be avoided and replaced by a simple predicate condition, as well as the forall statement could be replaced by a simple predicate effect statement issued on the corresponding Parameter, derived from the segment specification.

4066 IEEE ROBOTICS AND AUTOMATION LETTERS, VOL. 4, NO. 4, OCTOBER 2019

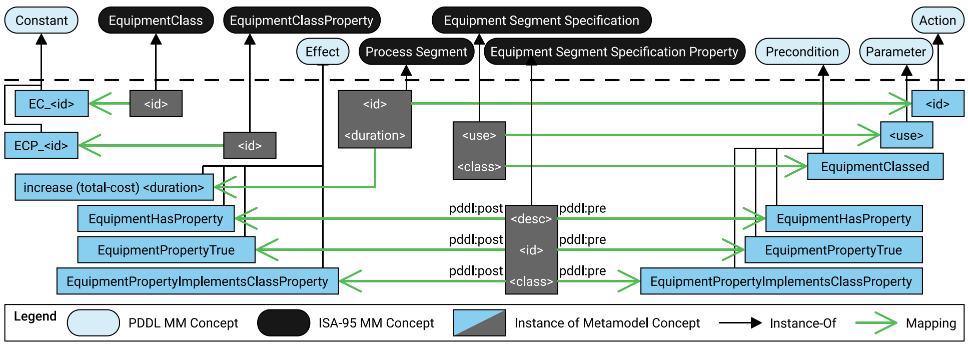


Fig. 4. Mapping of ISA-95 model elements to a PDDL domain description ("MM" means metamodel).

```
;... skipping requirements and types
(:constants
  EC_PositioningUnit EC_Shuttle - EquipmentClass
  ECP_PositioningUnitOccupied - EquipmentClassProperty)
;... skipping predicates
(:functions (total-cost))
(:action MoveShuttle
  :parameters (?SHUTTLE ?FROM ?TO - Equipment)
  :precondition (and
    ;... skipping similar preconditions
    (EquipmentClassed ?FROM EC_PositioningUnit)
    (exists (?P - EquipmentProperty) (and
      (EquipmentPropertyTrue ?P)
      (EquipmentPropertyImplementsClassProperty
        ?P ECP_PositioningUnitOccupied)
      (EquipmentHasProperty ?FROM ?P))))
  :effect (and ;... skipping similar effects
    (forall (?P - EquipmentProperty) (when (and
      (EquipmentPropertyImplementsClassProperty
        ?P ECP_PositioningUnitOccupied)
      (EquipmentHasProperty ?FROM ?P))
      (not (EquipmentPropertyTrue ?P))))
    (increase (total-cost) 10))
```

Listing 2. Excerpt of the PDDL domain description generated from ISA-95 instance data.

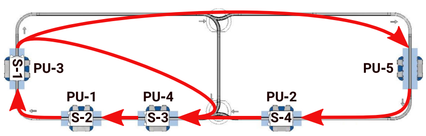


Fig. 5. Layout of the use case, rendered from within the PX5 Configurator by Montratec: positioning units PU-1 to PU-5, shuttles S-1 to S-4. Red arrows depict directed edges of the routing topology.

IV. Use Case – Industry 4.0 Testbed

We are applying the mapping defined above in a use case that is derived from a real production system deployed at the Technical University in Prague, the *Industry 4.0 Testbed*. It is reduced to only the transportation system and purposely leaves out any robots or material. The physical layout of the chosen use case is depicted in Fig. 5.

Section IV-A explains how a proprietary transportation system model is attached to the workflow as an optional preprocessing step, while Section IV-B explains domain specific knowledge that is to be introduced to the core workflow.

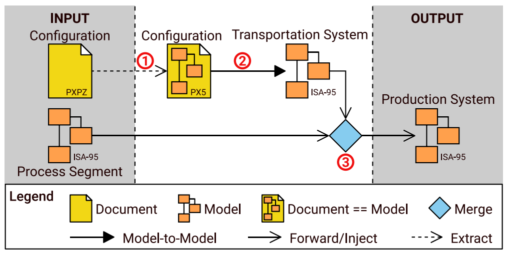


Fig. 6. Conversion workflow used to pre-process proprietary information into an ISA-95 model. The input models have been hand-crafted, the resulting output model can be used as an input model for the core workflow.

A. PX5 Configurator

So far, the process of mapping ISA-95 elements to PDDL has been domain-agnostic. For the chosen use case of the evaluation which is situated in the field of automated intra-logistics, we need to add a few extra conversion rules in order to get meaningful results. For this, it is important to understand how the system under observation works. It is an automated transportation system centered around a monorail track that can carry one or more shuttles. These shuttles can move on the rail between so called "positioning units" (PU), which are mechatronic systems with a well-defined location on the rail that can physically lock shuttles once they are located at one of these PUs.

In order to simplify the development of a corresponding "Production System" ISA-95 model, we have implemented a converter for the proprietary tool "PX5 Configurator for montratec"; the conversion workflow is depicted in Fig. 6. In short, ① we are reading the contents of the PX5 project file, and extracting relevant information in terms of a PX5 model (a corresponding metamodel has been reverse-engineered from the underlying proprietary XML document). Then ② this PX5 model is transformed into an ISA-95 model and ③ enriched with separately modeled process information. The result of this workflow is an ISA-95 model of a production system that can be used as an input for the core workflow described in Section III and depicted in Fig. 2.

```
1 (:predicates
2   ;... skipping already defined predicates
3   (PositioningUnitConnection ?F ?T - Equipment)
4   (ShuttleLocation ?S ?PU - Equipment))
5 (:action MoveShuttle
6   ;... skipping already defined parameters
7   :precondition (and
8     ;... skipping already defined preconditions
9     (PositioningUnitConnection ?FROM ?TO)
10    (ShuttleLocation ?SHUTTLE ?FROM)
11    (not (ShuttleLocation ?SHUTTLE ?TO)))
12  :effect (and
13    ;... skipping already defined effects
14    (not (ShuttleLocation ?SHUTTLE ?FROM))
15    (ShuttleLocation ?SHUTTLE ?TO)))
```

Listing 3. Domain-specific PDDL snippets.

B. Domain-Specific Concepts

The ISA-95 representation of this transportation system is strongly supported by the concept of ResourceRelationship-Networks. Track elements (straight line, curve and switch) are connected to each other by ResourceNetworkConnection instances of type `Track-Connection`. Positioning units and shuttles are described as "being attached" to a track element. This is realized by ResourceNetworkConnection instances of type `Positioning-Unit-Connection` and `Shuttle-Connection`, respectively, that connect these entities to corresponding track elements. Since multiple positioning units or shuttles can be located at one element, the (x, y, z) coordinates of the entities are stored as FromResourceReferenceProperties. This is required for creating correct routing graphs between the PUs, as well as assigning the shuttles to the correct PUs. In the process of converting an ISA-95 model to PDDL, the track- and PU-connections are simplified to a directed graph containing only PU nodes that are connected with each other. Also, the locations of the shuttles are reduced to those of the PUs, i.e., a shuttle is only in a well-known location if it is physically located at a PU. Locations in-between are not important in the context of our planning problem. The additional mapping rules are described below; they are implemented as part of steps 1 and 2 of the *core* workflow. Line numbers below refer to Lst. 3:

i) Two Predicates are defined that correspond to the previously described simplifications: `PositioningUnit-Connection` and `ShuttleLocation`. The former allows the definition of a directed graph representing the routing scheme of the transport system (line 3). The latter describes where a certain shuttle is currently located (line 4).

ii) The ProcessSegment `MoveShuttle` defines a boolean ProcessSegmentParameter with the id `movement` and value `true`. This parameter is recognized in the first transformation step of the core workflow, from the ISA-95 model to the PDDL domain model. This process segment also specifies three EquipmentSegmentSpecifications: the shuttle `S` to move and two PUs: the source `FROM` and the destination `TO`. Based on this information, three additional Preconditions and two additional Effects are created. For the preconditions, the following statements are added: first, it is checked, whether the two positioning units are directly connected with each other (line 9). Second, it is checked whether the shuttle is currently located in the source PU (line 10). Third, it is checked if the shuttle is not already in the destination location (line 11). The last two statements are somewhat redundant in the case that all actions that manipulate the `ShuttleLocation` predicate are correctly implemented (a shuttle should never be in two places at the same time). However, this redundancy can be considered a safety net and might support comprehensiveness for human readers. The effects are clearly related: first, the shuttle is set to be no longer in the source PU (line 14) and second, the shuttle is now located in the destination PU (line 15).

V. Evaluation

We are evaluating our approach threefold: (i) we are evaluating the use case previously described, (ii) we are evaluating similar use cases of various sizes in order to discuss scalability aspects and (iii) we are extending these use cases to include manufacturing operations in order to exemplify transferability of the approach.

All performance numbers mentioned in the remainder have been collected on a standard portable computer, equipped with a 2.4 GHz CPU. *Fast Downward*[4] was chosen as the PDDL solver, configured with "`astar(ipdb)`".

A. Use Case "Industry 4.0 Testbed"

Given an initial state as depicted in Fig. 5, with the four shuttles 1, 2, 3, 4 (encoded as 1234), located in PUs 3, 1, 4 and 2. The question that arose during the design time of this layout was, whether it is possible to bring the shuttles into an arbitrary order, with only one spare PU (5). While the answer to this problem might be obvious to experts, frequently, engineers are not able to answer such questions with confidence, especially if the layout is more complex. Thus, industrial systems are often equipped with additional features in an "ad-hoc" way in the hope that this would solve specific production-related problems. However, there are two problems with this approach: (i) it remains unclear whether the problem is really solved and (ii) these additional features are often quite expensive and might represent over-engineering. Therefore, an approach where the envisioned solution can be *formally verified* is clearly an advantage.

In our use case we want to find an answer to the question, and we would like to know how expensive (in terms of time) each of the reorderings is, given that each shuttle movement takes 10 s. For that, we have created 23 "goal description" ISA-95 models, each representing one of the desired goal states (excluding the goal state that is equivalent to the initial state): 1243, 1324, 1342, 1423, etc.

Lst. 4 depicts an excerpt of the generated problem definition that formulates the reordering of the shuttles from 1234 to 2341. The initial state is represented in lines 3–6, while the goal description is given in lines 8–11. Other initialization statements are left out—they are generated in step 2 of the core workflow. The 23 generated problem definitions are all exact copies of one

[4]cf. http://www.fast-downward.org/

```
(:init
   ; other initialization left out
  (ShuttleLocation E_Shuttle-01 E_PositioningUnit-03)
  (ShuttleLocation E_Shuttle-02 E_PositioningUnit-01)
  (ShuttleLocation E_Shuttle-03 E_PositioningUnit-04)
  (ShuttleLocation E_Shuttle-04 E_PositioningUnit-02))
(:goal (and
  (ShuttleLocation E_Shuttle-02 E_PositioningUnit-03)
  (ShuttleLocation E_Shuttle-03 E_PositioningUnit-01)
  (ShuttleLocation E_Shuttle-04 E_PositioningUnit-04)
  (ShuttleLocation E_Shuttle-01 E_PositioningUnit-02)))
```

Listing 4. Excerpt of the PDDL init state and the complete goal description that has been generated from one of the ISA-95 goal description models.

```
(moveshuttle e_shuttle-01
  e_positioningunit-03 e_positioningunit-05)
(moveshuttle e_shuttle-02
  e_positioningunit-01 e_positioningunit-03)
(moveshuttle e_shuttle-03
  e_positioningunit-04 e_positioningunit-01)
(moveshuttle e_shuttle-04
  e_positioningunit-02 e_positioningunit-04)
(moveshuttle e_shuttle-01
  e_positioningunit-05 e_positioningunit-02)
; cost = 50 (general cost)
```

Listing 5. The generated plan for re-ordering the shuttles from 1-2-3-4 to 2-3-4-1. Note that the chosen solver converts all entities to lower case.

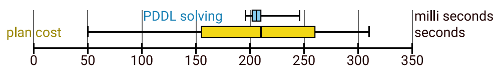


Fig. 7. Box plots of collected data from the 23 problem statements: (top) performance data of the PDDL solver, (bottom) cost of the solutions. Solving time was collected from within the Java-based workflow, i.e., it includes calling the solver as an external program, parsing its output, etc.

another, except for the goal statements that depict the desired order of the shuttles.

Lst. 5 depicts the resulting plan for the given problem. It shows that it is possible to reorder the shuttles by using five sequential steps, each costing 10s, thus resulting in a total time of 50 s, if all movements are executed sequentially.

Running the complete workflow on our concrete use case, from reading in the `.pxpz` file and the 23 goal statement models to the final production system model including the knowledge gained from the problem solver takes about 10 s, of which approximately 5 s are dedicated to the PDDL solver. More detailed PDDL-related data is given in Fig. 7: there, the distribution of the 23 solving times and the distribution of the 23 plan cost are depicted using box plots.

B. *Use Case "Scalability"*

Scalability has been tested by generating the above mentioned use case in various sizes; not fixed to 5 PUs. We have created six instances of the transportation system, with 5 to 15 positioning units, respectively. The layout template is depicted in the lower part of Fig. 8. A load factor of 65% (ratio of the number of shuttles and the number of PUs) is implemented. The task was to find a sequence of actions that would reverse the order of the shuttles, just as in the previous use case. The results of these experiments are depicted in the upper part of Fig. 8, represented by solid lines.

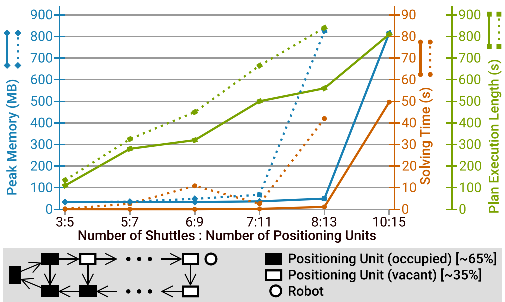


Fig. 8. Performance data gained from setups of different sizes. In these experiments, the solver has been called directly, not from within the Java-based workflow. Plan execution length is given in seconds (divide by 10 to get the number of steps required).

It can be seen that it is feasible to compute transportation plans for the given topology and load factor for settings as large as 10 shuttles on 15 PUs. This would account for small to medium sized systems. In the largest case that has been tested, computation took $\approx$ 50 s, which can be considered very responsive, given that the plan execution length of the corresponding solution amounts to 810 s. What can also be seen is that the length of the computed plans increases linearly, while the computational effort (memory and runtime) grows exponentially. In order to compute solutions for larger systems, it will be necessary to find either a better streamlined encoding of the ISA-95 model in PDDL, or to divide larger problems into smaller sub-problems and solving them independently from each other.

C. *Use Case "Transferability"*

Production systems usually handle and alter material, which is why we have created an extended version of the use case previously described. This extended version adds material to the setting, namely wooden boards that are mounted to the shuttles. An additional ProcessSegment `DrillBoard` is defined that can be executed by the production system in order to drill a hole into the board. This process segment requires a drilling robot and the shuttle carrying the board needs to be located in a positioning unit that is within the reach of this robot. In our experiments, we have located the drilling robot next to the top right positioning unit, as is depicted in the lower part of Fig. 8. The results of the experiments are depicted using dashed lines. The task for the solver was to find a sequence of actions that would drill a hole in each of the boards.

Most importantly, the results show that it is feasible to convert ISA-95 models that include both inventory movements and manufacturing operations to PDDL and have it successfully solved. The new concepts (Predicates) and entities (Objects) required to formulate the new kind of operation have a significant impact on the solving performance. In fact, we could not compute a solution for the largest experiment (10:15) within our timeout frame of 300 s, which is why the diagram does not show values for this setting.

VI. Conclusion

We have presented a conceptual mapping and a workflow for the transformation of ISA-95 models into a PDDL formalism in order to find sequences of production steps to fulfill certain manufacturing goals. We have successfully tested our approach in a use case where a chosen transportation system layout was tested whether it would fulfill certain logistics requirements.

To verify the proposed approach, we have developed a simple software tool that is able to execute the computed plan with real machinery. While the entire workflow has been tested and verified up to the point where real shuttles move in the aforementioned Industry 4.0 Testbed, we have focused on the conceptual model transformation part in this letter.

It is also worth noting that, while this letter has been focused on transportation systems, the generic approach and mapping strategy between ISA-95 and PDDL can be leveraged for other production-related problems as well. Briefly, we have already considered material manipulation (drilling) in one of our evaluation scenarios. Consequently, in a next step we would like to consider product assembly tasks in the production process. This should enable a flexible manufacturing system to create production plans for assembly-based lot-size 1 products automatically. What would be needed for such a scenario, would be the construction plan of the final product, as well as the consideration of machinery capabilities with respect to assembly operations.

The performance data presented in Section V is based on an non-optimized implementation: (i) the PDDL solver could be tweaked by experimenting with the parameters of its search algorithm. (ii) improvements could be achieved by parallelizing the tasks assigned to the PDDL solver. Currently, the 23 problems are solved sequentially in a simple for-loop—of course, the invocation of the solver could be done for several problems in parallel; most easily based on the number of cores the underlying platform provides. (iii) the encoding of the ISA-95 model in PDDL could be streamlined in a way that is more convenient to the solver (i.e., steps 1 and 2 of the core workflow could be improved). This argument has already been teased in Section III-B, where alternative PDDL encodings for specific ISA-95 constructs are mentioned.

Future work could also take into consideration more advanced versions of PDDL that would, e.g., enable the specification of *durative actions* [29], ultimately supporting parallelism of production tasks at the planning level. While such an approach could lead to finding highly efficient production plans, it might be too computationally expensive. Nevertheless, experiments in this direction seem worthwhile.

References

[1] M. Schleipen, "Adaptivität und Semantische Interoperabilität von Manufacturing Execution Systemen (MES)," Ph.D. dissertation, Dept. Inform. Karlsruhe Inst. Technol., Karlsruhe, Germany, 2012.

[2] F. Shrouf, J. Ordieres, and G. Miragliotta, "Smart factories in Industry 4.0: A review of the concept and of energy management approached in production based on the Internet of Things paradigm," in *Proc. IEEE Int. Conf. Ind. Eng. Eng. Manage.*, 2014, pp. 697–701.

[3] A. Rogalla, A. Fay, and O. Niggemann, "Improved domain modeling for realistic automated planning and scheduling in discrete manufacturing," in *Proc. 23rd IEEE Int. Conf. Emerg. Technol. Factory Autom.*, 2018, pp. 464–471.

[4] *Enterprise-control System Integration—Part 1: Models and Terminology*, IEC Std. IEC 62264-1:2013, International Electrotechnical Commission, 2013.

[5] *Engineering Data Exchange Format for Use in Industrial Automation Systems Engineering—Automation Markup Language—Part 1: Architecture and General Requirements*, IEC Std. IEC 62714-1:2018, 2018.

[6] S. Kent, "Model driven engineering," in *Proc. 3rd International Conference on Integrated Formal Methods (IFM)*, (Lecture Notes in Computer Science 2335), M. Butler, L. Petre, and K. Sere, Eds., Berlin, Germany: Springer, 2002, pp. 286–298.

[7] J. Bézivin, "In search of a basic principle for model driven engineering," *Novatica J.*, vol. 5, no. 2, pp. 21–24, 2004.

[8] D. C. Schmidt, "Model-driven engineering," *Computer*, vol. 39, no. 2, pp. 25–31, Feb. 2006.

[9] M. Kaufmann, P. Manolios, and J. S. Moore, Eds., *Computer-Aided Reasoning: ACL2 Case Studies*. Berlin, Germany: Springer, 2000.

[10] K. Claessen, N. Eén, M. Sheeran, N. Sörensson, A. Voronov, and K. Åkesson, "SAT-solving in practice, with a tutorial example from supervisory control," *Discr. Event Dyn. Syst.*, vol. 19, no. 4, pp. 495–524, 2009.

[11] M. Helmert, "The fast downward planning system," *J. Artif. Intell. Res.*, vol. 26, no. 1, pp. 191–246, Jul. 2006.

[12] *Enterprise-Control System Integration—Part 2: Objects and Attributes for Enterprise-Control System Integration*, IEC Std. IEC 62264-2:2013, International Electrotechnical Commission, 2013.

[13] *Enterprise-Control System Integration—Part 4: Object Model Attributes for Manufacturing Operations Management Integration*, IEC Std. IEC 62264-4:2015, 2015.

[14] B. Wally, C. Huemer, and A. Mazak, "Entwining plant engineering data and ERP information: Vertical integration with AutomationML and ISA-95," in *Proc. 3rd IEEE Int. Conf. Control, Autom. Robot.*, 2017, pp. 356–364.

[15] B. Wally, "Provisioning for MES and ERP," TU Wien and AutomationML e.V., Vienna, Austria, Application Recommendation, 2018.

[16] B. Wally, C. Huemer, A. Mazak, and M. Wimmer, "IEC 62264-2 for AutomationML," in *Proc. 5th Autom. ML User Conf.*, 2018, pp. 1–7.

[17] P. Novák, P. Kadera, and M. Wimmer, "Model-based engineering and virtual commissioning of cyber-physical manufacturing systems—Transportation system case study," in *Proc. 22nd IEEE Int. Conf. Emerg. Technol. Factory Autom.*, 2017, pp. 1–4.

[18] B. Wally, C. Huemer, and A. Mazak, "Aligning business services with production services: The case of REA and ISA-95," in *Proc. 10th IEEE Int. Conf. Service Oriented Comput. Appl.*, 2017, pp. 9–17.

[19] B. Wally, C. Huemer, and A. Mazak, "A view on model-driven vertical integration: Alignment of production facility models and business models," in *Proc. 13th IEEE Int. Conf. Autom. Sci. Eng.*, 2017, pp. 1012–1018.

[20] B. Wally, C. Huemer, and A. Mazak, "ISA-95 based task specification layer for REA in production environments," in *Proc. 11th Int. Workshop Value Model. Bus. Ontol.*, 2017, pp. 1–5.

[21] B. Wally, A. Mazak, B. Kratzwald, and C. Huemer, "Model-driven retail information system based on REA business ontology and Retail-H," in *Proc. 17th IEEE Conf. Bus. Inform.*, 2015, pp. 116–124.

[22] M. Ghallab, D. S. Nau, and P. Traverso, *Automated Planning and Acting*. New York, NY, USA: Cambridge Univ. Press, 2016.

[23] M. Ghallab, D. S. Nau, and P. Traverso, *Automated Planning—Theory and Practice*. Amsterdam, The Netherlands: Elsevier, 2004.

[24] K. Erol, D. S. Nau, and V. S. Subrahmanian, "Complexity, decidability and undecidability results for domain-independent planning," *Artif. Intell.*, vol. 76, no. 1/2, pp. 75–88, 1995.

[25] M. Heusner, T. Keller, and M. Helmert, "Best-case and worst-case behavior of greedy best-first search," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 1463–1470.

[26] D. L. Kovacs, "Complete BNF description of PDDL 3.1," Dept. Meas. Inf. Syst., Budapest Univ. Technol. Econ., Budapest, Hungary, 2011.

[27] A. R. Sousa and J. J. P. Z. S. Tavares, "Toward automated planning algorithms applied to production and logistics," in *IFAC Proc. Vol.*, vol. 46, no. 24, pp. 165–170, 2013.

[28] T. Niemueller, T. Hofmann, and G. Lakemeyer, "CLIPS-based execution for PDDL planners," in *Proc. Workshop Integr. Planning, Acting, Execution*, 2018, pp. 41–49.

[29] M. Fox and D. Long, "PDDL 2.1: An extension to PDDL for expressing temporal planning domains," *J. Artif. Intell. Res.*, vol. 20, pp. 61–124, 2003.

7 Thirteen years of SysML: A systematic mapping study

S. Wolny, A. Mazak, C. Carpella, V. Geist and M. Wimmer;
Journal of Software and Systems Modeling, Springer, 19(1), (2020), pp. 111–169.
DOI: 10.1007/s10270-019-00735-y

Software & Systems Modeling (2020) 19:111–169
https://doi.org/10.1007/s10270-019-00735-y

REGULAR PAPER

Thirteen years of SysML: a systematic mapping study

Sabine Wolny[1] · Alexandra Mazak[1] · Christine Carpella[2] · Verena Geist[3] · Manuel Wimmer[1]

Received: 7 March 2018 / Revised: 11 February 2019 / Accepted: 9 April 2019 / Published online: 13 May 2019

Abstract
The OMG standard *Systems Modeling Language (SysML)* has been on the market for about thirteen years. This standard is an extended subset of UML providing a graphical modeling language for designing complex systems by considering software as well as hardware parts. Over the period of thirteen years, many publications have covered various aspects of SysML in different research fields. The aim of this paper is to conduct a systematic mapping study about SysML to identify the different categories of papers, (i) to get an overview of existing research topics and groups, (ii) to identify whether there are any publication trends, and (iii) to uncover possible missing links. We followed the guidelines for conducting a systematic mapping study by Petersen et al. (Inf Softw Technol 64:1–18, 2015) to analyze SysML publications from 2005 to 2017. Our analysis revealed the following main findings: (i) there is a growing scientific interest in SysML in the last years particularly in the research field of Software Engineering, (ii) SysML is mostly used in the design or validation phase, rather than in the implementation phase, (iii) the most commonly used diagram types are the SysML-specific requirement diagram, parametric diagram, and block diagram, together with the activity diagram and state machine diagram known from UML, (iv) SysML is a specific UML profile mostly used in systems engineering; however, the language has to be customized to accommodate domain-specific aspects, (v) related to collaborations for SysML research over the world, there are more individual research groups than large international networks. This study provides a solid basis for classifying existing approaches for SysML. Researchers can use our results (i) for identifying open research issues, (ii) for a better understanding of the state of the art, and (iii) as a reference for finding specific approaches about SysML.

Keywords SysML · Systematic mapping study · Systems engineering

1 Introduction

The *Systems Modeling Language (SysML)* is a standard from the Object Management Group (OMG) to support the design, the analysis, and verification of complex systems which may include software and hardware components. SysML reuses parts of UML and additionally offers new language elements like value types, quantity kind, as well as the opportunity to describe the functionality of continuous systems [29]. One of the first intention for SysML was to give systems engineers a modeling language in hand which is not too software oriented [51]. SysML enables to model a wide variety of systems from different perspectives such as behavior, structure, or requirement. The temporarily last version 1.5 was released in May 2017. SysML has been in place for about thirteen years, and various papers capturing different aspects of this standard have been published at different venues by different research communities. Since SysML is used in multi-disciplinary engineering, there are large application fields where the language is used.

To get a better overview of this huge number of contributions as well as to identify the relevance of SysML in scientific communities, we carried out a systematic mapping study by analyzing the abstracts of the different contributions. The study helps to generate knowledge by determining the application fields in which SysML is commonly used, which research groups are involved, etc. These insights help to iden-

Communicated by Jean-Michel Bruel.

Sabine Wolny
sabine.wolny@jku.at

1 Christian Doppler Laboratory for Model-Integrated Smart Production (CDL-MINT) Institute of Business Informatics - Software Engineering, Johannes Kepler University Linz, Altenberger Straße 69, 4040 Linz, Austria

2 ENGEL AUSTRIA GmbH, Schwertberg, Austria

3 Software Competence Center Hagenberg GmbH (SCCH), Softwarepark 21, 4232 Hagenberg, Austria

tify trends to which direction SysML should be developed in future, also with respect to the ongoing discussion about SysML 2.0.

To put the aim of this article in a nutshell, we present inputs as well as outputs of the SysML mapping study and show a comprehensive overview of the evolution of SysML over a period of more than 10 years. Additionally, we identify open issues and discuss these issues in the conclusion of this article with regard to SysML 2.0. According to Kitchenham et al. [28], the findings and outlook may support the work of the following stakeholders:

- Research: Scientists just started with research in the field of SysML may use this study as an overview and starting point for their work. Experienced researchers may also use it as reference to save time for in-depth studies and to accelerate the search for open issues.
- Industry: For industry, the findings give a good outline of the state of the art in SysML research. This may enable to transfer knowledge between academia and industry. Such knowledge transfer may push forward the realization of open issues in the vision of Industry 4.0 and cyber-physical systems [10]. At least, industry stakeholders may identify relevant and suitable research outputs for practical settings.

The remainder of this article is structured as follows: Section 2 discusses the related work. In Sect. 3, we present the research method, define the research questions, and describe the process of conducting the mapping study. In Sect. 4, we describe and analyze the extracted data and visualize the results. Section 5 covers possible threats to validity. In Sect. 6, we present the conclusions and an outlook to future work. In Appendices A and B we present references of all covered SysML papers, a list of books, and theses, which were not part of this survey.

2 Related work

In this section, we give an outline on the method of systematic literature review compared to the method of a systematic mapping study. Furthermore, we take a closer look on these methods applied to UML and to its profiles (e.g., SysML, MARTE).

2.1 Systematic literature review versus systematic mapping study

Evidence-based practices, originating from the medicine discipline, have been widely adopted in software engineering (SE) since 2004. In order to address evidence-based SE in the form of *systematic literature reviews* (SLRs), the corresponding techniques were re-formulated by Kitchenham [26]. SLR is a well-defined methodology to identify, analyze, and interpret evidences in an unbiased and repeatable way [28]. A large majority of published SLRs in the domain of SE has been performed by following the approaches introduced by Kitchenham et al. [25,27]. In addition, there are some authors who have adopted surveys from medicine [35] as well as from social sciences [46], or they have applied refined guidelines like introduced in [11,14,60].

In this article, we apply a broader form of SLR which is known as *systematic mapping study* (SMS) according to [8], since our intention is to focus on evidences for a specific research topic instead of answering detailed research questions. Based on a set of primary studies, a SMS identifies gaps in the research area under consideration and discovers potential research trends. By doing so, we follow the guidelines for conducting SMS in SE introduced by Petersen et al. [44,45]. Additionally, we apply the survey of Kuhrmann et al. [31] for performing our SMS for SysML (see Sect. 3).

It seems that there are similarities between SMS and SLR; however, the approaches of these two methodologies and also their goals are quite different. For instance, in contrast to SLR, a SMS uses general research questions to classify and aggregate relevant studies to high-level categories [40].

2.2 SMSs and SLRs applied to UML

In empirical studies concerning the maintenance of UML diagrams and their use in the maintenance of code, Fernández-Sáez et al. [17] conducted a SMS. For this purpose, the authors studied 38 already published studies for discovering an empirical evidence by applying the guidelines of [25]. As a result, the authors identified the need for more experiments and case studies in industrial contexts.

In the particular research field of UML-driven software performance engineering, Garousi et al. [18] conducted a SMS to systematically categorize the current state of the art. Thereby, the authors applied the guidelines provided by Kitchenham and Charters [25] and Petersen et al. [44]. Among others, the authors identified emerging trends in this specialized research field based on a set of 90 (from 114 identified) papers published between 1998 and 2011 [18].

Torre et al. [56] deliver a comprehensive summary of UML consistency rules (regarding the different diagram types) by performing a SMS including 94 primary studies published until December 2012. For their SMS, they used in total seven search engines and followed the guidelines of Kitchenham [25]. There are related research works that address, e.g., a SLR on UML consistency management [33] by covering an earlier publication period (2001–2007), as well as, a SLR about the quality of UML diagrams [37]. Finally mentioned, there exist prior works on empirical evidence related to UML in general, e.g., a SLR [9] and a SMS [47], which consider papers on UML properties and features published until 2008.

In the area of Software Product Lines (SPL), a SMS on business process variability is conducted by Valença et al. [57]. This SMS includes 80 primary studies and considers one empirical study on a hierarchical representation method for UML 2.0 activity diagrams. They based their work mainly on the surveys presented in [8,44] as well as on SMS best practice as introduced in [28].

All of these related works have in common that they do not consider SysML as main topic of the survey and that they apply other techniques than we follow in our mapping study. However, they represent interesting related work, not least because UML provides the basis for SysML.

2.3 SMSs and SLRs applied to UML profiles

Ameller et al. [1] classify UML and UML profiles used to specify functional and non-functional requirements based on SMS to assess the state of the art in the development of services-oriented architectures using model-driven development. The authors selected and analyzed 129 papers by adopting the guidelines presented in [25] and those described in [28,44]. There are related SMS investigating the alignment of requirements specification and testing such as presented in [5]. In [52], the authors conducted a survey to examine the use of UML profiles for testing Web services composition.

In the research field of domain-specific languages (DSLs), Nascimento et al. [13] perform a SMS to identify the most popular application domains of DSLs. The authors categorize 1440 (from 4450 identified) primary studies by applying the guidelines described in [25,44]. The technique of UML profiles is mentioned in 21 publications of their catalog. An extensive SLR in the specialized research area of model-driven security was conducted by Nguyen et al. [39], where the authors also consider UML profiles (e.g., UMLSec, SecureUML, etc.) for the definition of security-oriented DSLs. In addition, Souag et al. [55] surveyed UML-based extensions for modeling security in the field of security requirements engineering.

The UML profile SysML is addressed as topic in a mapping study, which investigated the usability requirements elicitation [41]. The study was conducted based on the guideline presented in [25]. The authors formulated a sub-question on notations to elicit usability requirements, and they identified model-based notations and natural language as the most widely used notations in SE. There are similar SLRs related to this topic such as presented in [2], which covers model-driven requirements engineering.

Regarding model-based requirements specifications, Rashid et al. [48] investigated how UML, SysML, and MARTE profiles have been used to specify aspects of embedded systems in the context of early design verification by considering papers published between 2008 and 2015. In an additional SLR on tool selection in model-based systems engineering, Rashid et al. [49] classified selected research work in different categories like "modeling category," where modeling aspects of embedded systems using UML and its profiles SysML and MARTE were discussed. Additionally to model-based or model-driven requirements engineering and specification, SysML as topic was also investigated in the field of model-based testing like in the work presented in [54]. Wortmann et al. [62] explore in their SMS the state of the art of using modeling languages for model-based systems engineering of smart factories. The authors found out that SysML and its variants play a key role as modeling technique for realizing Industry 4.0 approaches.

In the research field of systems engineering, several SLRs include specific research questions concerning UML as well as its profiles SysML and MARTE. For instance, Guessi et al. [20] conducted a SLR on the topic of describing software architectures for systems of systems (SoS). The authors' second research question targets the techniques that have been used for describing SoS. They identified that most primary studies use UML or SysML as semi-formal architecture description languages.

In the previous past, SMSs were applied on safety and security topics in the research field of systems engineering. For instance, Nguyen et al. [40] conducted a SMS by covering primary studies that focus on several SysML profiles like SysML-Sec. Other SMSs such as presented in [16,22] only touch SysML in their explanations and findings.

There are a lot of SMSs addressing UML-based approaches and UML profiles (e.g., SysML), e.g., (i) an SMS on functional safety conducted by [7], (ii) a survey on SPL evolution presented by [32], or two SMS on SPL testing conducted by [15,38].

2.4 Synopsis

In this section, we relate existing research to our mapping study. The presented research includes guidelines for conducting SLRs and SMSs such as the work of Kitchenham et al. [25,27] and Petersen et al. [44,45], or Kuhrmann et al [31]. We discussed works including empirical studies, case studies, and surveys on UML and UML profiles, in particular, applied in the domain of software engineering as well as systems engineering. The conducted studies and mentioned surveys investigate in the research fields of requirements engineering, embedded systems in the context of early design verification, model-based systems engineering, security engineering, performance engineering, and software testing, e.g., the quality and usability of UML and UML profiles.

All of these studies and surveys have in common that they do not consider SysML exclusively and that they apply other guidelines than we follow in our mapping study. For instance, the presented SLRs answer detailed research questions but they give no evidence on various aspects for SysML for systems and software engineering. However, they represent interesting related work and provide relevant entry points to our own mapping study.

3 Research method

As research method, we used the previously introduced SMS that enables to cover and classify publications in a specific area. In our study, we are focusing on the abstracts of publications, published in the time period from 2005 to 2017, that have SysML as their main topic.

The process for conducting this SMS is shown as SysML activity diagram in Fig. 1, which mainly bases on the guidelines introduced in Petersen et al. [44]. We modified this mapping process by adapting the last two activities.

Our systematic mapping process consists of five steps (see Fig. 1). It starts by the activity of defining research questions. The output of this activity are appropriate research questions that define the review scope for the next step. In that activity, we conduct a literature search. The output are all publications related to the previously defined research questions. The next step is the screening of those publications in order to select the relevant ones. These relevant publications are the input for the activity called "classification using abstracts," where we categorize the relevant publications by their abstracts based on the research type facets introduced by Petersen et al. [44] (see Table 1). We enhance this activity by further investigating to classify the abstracts based on systems engineering phases related to the VDI guideline 2206 [58] and contribution types as introduced by Shaw [53]. As output, we get classified abstracts of selected publications, which we use as input for the last activity "mapping of papers," After this final step, we get a systematic map, which enables us to extract main findings related to our research questions.

In the following, we describe four (Sects. 3.1–3.3) of the five SMS activities based on the research topic of our survey. Afterward, in Sect. 4, we present the final activity "mapping of papers."

All data (i.e., founded results, search strings, screened paper, classifications) can be also found on figshare[1] at https://figshare.com/s/871aa0c03aa18eb3edf6.

3.1 Activity 1: defining research questions

In this subsection, we define our research questions to specify the review scope of the mapping study and we provide an insight into the intentions behind these questions.

- **RQ 1**: *What are the bibliometric key facts of SysML publications?*
  The intentions of this research question is to find out (i) the number of SysML publications that were contributed in the period from 2005 to 2017, (ii) the type of those publications (e.g., article, book chapter), (iii) the main venues where the publications have been submitted, and (iv) the main research background (i.e., communities) of these venues.
- **RQ 2**: *Where are the scientific communities of SysML located and are there main contributors, who scientifically promote SysML topics?*
  The intention is to identify and analyze scientific communities working on topics of SysML, e.g., we are interested in the location of these communities. Moreover, we address the question if there are more single authors working on SysML topics, or rather (small) research groups. For instance, we are interested in the number of publications and their authors to identify those publications published by one and the same author. Last but not least, we consider the number of citations of each of the publications to identify the relevance for the respective community. By doing this analysis, we want to find out if there exists a huge network spanning over the world which is working on SysML approaches, or not.
- **RQ 3**: *Which research type facets do the identified publications address?*
  The main intention is to categorize the different publications by a solid and already well-established schema ([31,59]). Therefore, we use the research type facets introduced by Petersen et al. [44] as described in detail in Sect. 3.3 (see Table 1). Based on this type facets, we want to find out in which research contexts SysML topics are used, e.g., validation, evaluation, etc.
- **RQ 4**: *What are the key aspects of applying SysML in the classified publications?*
  In addition to assigning the publications to type facets, we are interested to get a deeper insight in the research contribution of those publications. This research question aims to identify (i) in which phase of the engineering process [58] SysML is used, and (ii) the contribution type [53] of the publications.

3.2 Activities 2 and 3: conducting search and screening of publications

After identifying our research questions, the next activity is the definition of appropriate keywords to find all published papers regarding topics about SysML.

[1] https://figshare.com/.

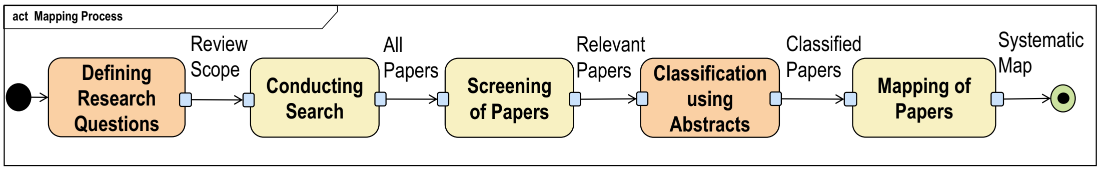


Fig. 1 Activity diagram of the systematic mapping process [44]

Table 1 *Research Type Facet* [44, p.4]

Category	Description
Validation Research	Techniques investigated are novel and have not yet been implemented in practice. Techniques used are, for example, experiments, i.e., work done in the laboratory
Evaluation Research	Techniques are implemented in practice, and an evaluation of the technique is conducted. That means, it is shown how the technique is implemented in practice (solution implementation) and what are the consequences of the implementation in terms of benefits and drawbacks (implementation evaluation). This also includes to identify problems in industry
Solution Proposal	A solution for a problem is proposed; the solution can be either novel or a significant extension of an existing technique. The potential benefits and the applicability of the solution are shown by a small example or a good line of argumentation
Philosophical Papers	These papers sketch a new way of looking at existing things by structuring the field in form of a taxonomy or conceptual framework
Opinion Papers	These papers express the personal opinion of somebody whether a certain technique is good or bad, or how things should been done. They do not rely on related work and research methodologies
Experience Papers	Experience papers explain on what and how something has been done in practice. It has to be the personal experience of the author

Conducting search In contrast to existing work (see Sect. 2), we do not want to cover just a single aspect of SysML. Our aim is to provide an overview of all published papers. Thus, we decide to search for the following keywords:

- SysML
- "Systems Modeling Language" (case insensitive)
- "System Modeling Language" (case insensitive)

There are many different digital literature libraries available on the Web for conducting a literature search. We have opted for the following four established ones:

- Scopus (www.scopus.com)
  One of the largest abstract and citation databases of peer-reviewed literature.
- ACM Digital Library (http://dl.acm.org/)
  ACM is a research, discovery, and networking platform where a collection of full-text articles and bibliographic records can be found.
- IEEE Xplore Digital Library (http://ieeexplore.ieee.org)
  IEEE Xplore provides a full-text access to technical literature in engineering as well as technology.
- DBLP (http://dblp.uni-trier.de/)
  The computer science bibliography database dblp offers open bibliographic information on computer science journals and proceedings.

In our piloting phase, we got more than 2000 papers resulting from the conducted search process in these libraries. In order to obtain more precise results regarding our intention to find out the state of the art of research on SysML in academia, we decided to restrict the search string by the following criteria:

- *Publication in the period from 2005 to 2017*: The first SysML Specification[2] v.0.9. is online since January 2005. Thus, we use this year as starting point in our systematic mapping study. Since the survey was conducted in late 2017/early 2018, we decided to define 2017 as end date.
- *Title*: In order to get as output more specific SysML publications, and not only papers mentioning SysML as related

[2] http://sysml.org/sysml-specifications/.

work, we restrict the search query to publications where the previously defined keywords are in the title. This decision should ensure that only publications that focus on SysML are included in our result set.

We updated our result set several times to receive a complete set of all relevant publications to answer our research questions. In addition during the revision process, we also made several updates for finding any further publication published in 2017. The final state of our result set, aligned with all libraries, was checked the last time on the 21 of January 2019.

Screening of publications For screening the publications, we defined the following exclusion criteria:

- *Duplicates*
- *Papers:*
  - without available abstract
  - without an English, German, or French abstract
  - without any context to the language SysML
    For instance, SysML as abbreviation for "System Machine Learning."
  - with similar abstracts
    Some papers are covering different development stages of a project, and therefore, their abstracts are identical or have been just slightly extended. We deleted the older or shorter version and always kept the newer or longer version in the result set.
  - with identical abstracts
    There are papers with identical content and abstracts; however, they have been published at different venues (e.g., conferences and journals). We decided to leave one of them in the results set and deleted the other publication.
- *Books:* Books are deleted because they are not peer-reviewed (e.g., A Practical Guide to SysML [B4]). The whole list of retrieved books can be found in Appendix B.
- *Theses:* Theses often cover several different aspects and therefore can be assigned to different type facets. In addition, most theses are also (partly) published as conference or journal papers and would be duplicates. Thus, we removed them from the result set. The list of excluded theses can be found in Appendix B.

Based on these exclusion criteria, we double-checked (extractor/checker) all extracted papers in order to ensure that there is consensus on all findings. After performing this screening process, our result set comprises 579 papers. For these papers, we additionally considered the number of citations provided by Google Scholar[3] (see Sect. 4.2). The overall list of the 579 publications is provided in Appendix A.

3.3 Activity 4: classification using abstracts

According to the guidelines of Petersen et al. [44], it is sufficient to search only the abstract of a publication for categorization. In order to get a deeper insight of the research context of those publications and for a better mapping, we decide to apply the research type facets of Petersen et al., as presented in Table 1, already in this phase of the SMS. This means that we deviate from the original mapping process by using the research type facets as classification schema to categorize the abstracts of the selected publications. Thus, we modified the activity of "keywording of abstracts" in that we use an already established classification schema.

Besides the assignment of abstracts to these research type facets, it is important to find out in which engineering phase SysML is mainly used and to which contribution type the publications belong in order to get a deeper insight in the research field of the selected publications. For this purpose, we examine the different topics of the papers by analyzing the keywords of the abstracts and cluster the publications based on systems engineering phases and contribution types. In this respect, we adapted the mapping process introduced by Petersen et al. [44] by making a more fine grained categorization, as we describe in the following:

Systems engineering phase Based on the V-model related to the VDI guideline 2206 [58], we distinguish the following phases:

- *Requirements*: Defining the requirements and system properties such as the scope of functions and interfaces.
- *Design*: Designing the architecture of the system.
- *Implementation*: Phase of realization and integration to which simulations and code generators belong.
- *Validation and Verification*: The final phase of the V-model to analyze and check the system.

Contribution type On the basis of the types of research results introduced by Shaw [53], we define our categories for the contribution types. Shaw defines seven different types, in which she is also distinguishing between different data models (empirical, analytic, qualitative model). In our study, we do not focus on different data models. Thus, we adapt these types for our classification process. To give an overview, we briefly describe our contribution types in the following:

[3] https://scholar.google.at/.

- *Technique*: Definition of a method or procedure.
- *Process*: Sequence of both interdependent as well as linked procedures.
- *Notation*: A formal language or graphical notation to support a method (e.g., SysML Profile) or to map SysML to other languages (e.g., translation).
- *Tool*: A specific implemented tool based on a certain technique.
- *Specific Solution*: Solution for an application problem, e.g., result of a specific analysis, evaluation, or comparison.
- *Other*: For all publications of our result set, which cannot be assigned to one of the contribution types specified above. For instance, this includes publications that use SysML in an educational context, or that compare SysML with other modeling languages.

The outcome of the first four activities of the mapping process is a result set of publications classified based on research type facets, systems engineering phases, as well as on contribution types. This outcome serves as an input to the final activity called "mapping of papers," which is described in detail in the next section.

4 Mapping of papers

In this section, we describe analysis and results to answer the research questions (RQ1–RQ4). This output bases on the last activity of our adapted SMS. Additionally, we briefly summarize the main findings related to these questions at the end of each subsection.

4.1 RQ 1: bibliometrics of SysML publications

To answer the first research question, we start with analyzing the distribution of published papers in the period from 2005 to 2017. In a second step, we relate the result set to the type of publications. Furthermore, we make a list of venues, where the publications have been submitted. Figure 2 depicts the absolute number of publications per year. The plot shows that this number subjects to fluctuations. We found out that in the years, in which a new version of the SysML standard was published, the number of publications is mostly higher than in the years before. The peak in Fig. 2 indicates that most of the publications were published in 2013.

For further analysis of these results, Fig. 3 illustrates the relationship among the number of publications, the years, and the type of publication. The orange line depicts that in the period from 2005 to 2017 there have been submitted much more inproceedings to scientific conferences than articles to scientific journals (see the blue line) or book chapters (see the green line). Regarding the publication type, 80% of the

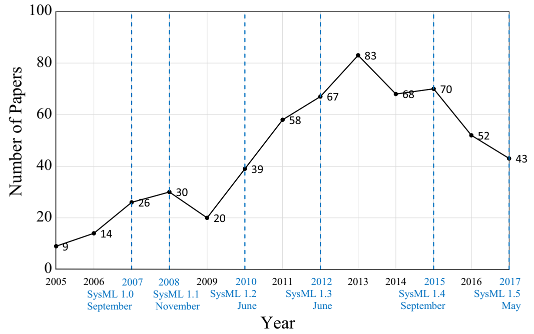


Fig. 2 Number of publications per year in the period from 2005 to 2017 (included number of studies: 579)

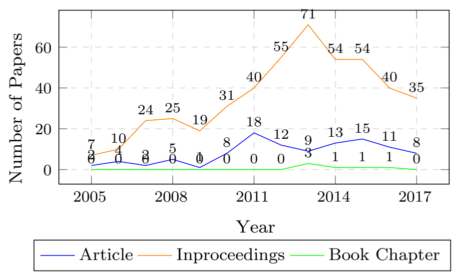


Fig. 3 Number of publications per year regarding publication type (included number of studies: 579)

screened SysML publications were published as inproceeding, 19% as article, and only 1% as book chapter.

Furthermore, we want to find out if there are few selected venues promoted by a handful research communities, or if the publications spread over various conferences, workshops, and journals which are promoted by very different research communities. The result set shows that there are 316 different venues, where the papers have been submitted. For the sake of relevance and clarity, we list in Table 2 those conferences with at least 8 SysML publications and in Table 3 the journals with at least 4 publications. In total, we present 12 different venues. The category column shows the main research community of these venues.

The main venue is the *Annual International Symposium of the International Council on Systems Engineering (INCOSE)*, where more than 30 publications have been submitted in the last 10 years. A possible reason for INCOSE being so prominent may be that the development of the SysML language specification was a collaborative effort between members of OMG and INCOSE. Thus, the INCOSE community is interested in applications and innovations of SysML. Additionally, one of the main research topics of this conference is systems engineering, where SysML plays a key role. From a statistical point of view, we underpin INCOSE's

Table 2 Most Prominent Conferences regarding the Number of Publications (at least 8 Papers) Sys. Eng. = Systems Engineering, Sof. Eng. = Software Engineering, Sim. = Simulation, Aut. = Automation

Venue	Category	Number of Publications
INCOSE (Annual International Symposium of the International Council on Systems Engineering)	Sys. Eng.	32
IDETC/CIE (International Design Engineering Technical Conferences)	Sys. Eng.	14
ETFA (International Conference on Emerging Technologies and Factory Automation)	Aut.	13
SysCon (International Systems Conference)	Sof. Eng.	12
MODELSWARD (Conference on Model-Driven Engineering and Software Development)	Sof. Eng.	11
CSER (Conference on Systems Engineering Research)	Sys. Eng.	10
ISSE (International Symposium on Systems Engineering)	Sys. Eng.	9
WSC (Winter Simulation Conference)	Sim.	9
ICEIS (International Conference on Enterprise Information Systems)	Sof. Eng.	8

Table 3 Most Prominent Journals regarding the Number of Publications (at least 4 Papers) Sys. Eng. = Systems Engineering, Sof. Eng. = Software Engineering, Sim. = Simulation, Aut. = Automation

Venue	Category	Number of Publications
Systems Engineering	Sys. Eng.	11
Innovations in Systems and Software Engineering	Sof. Eng.	5
Software and Systems Modeling	Sof. Eng.	4

main position by considering the statistical distribution based on the number of papers per venues, listed in Table 2. We get a mean value of 13 (13,1) and a standard deviation of 7 (6,9). Since we identified a spread of 6 to 20 publications per venue in this descriptive statistical analysis, we can classify INCOSE as an outlier compared to the averages of the other conferences.

The second prominent venue is the *International Design Engineering Technical Conferences (IDETC/CIE)*, where 14 papers were submitted and presented. This conference is, among others, one of the main conferences for design engineering mostly related to the manufacturing domain, where SysML fits thematically well, since it is often used in the design phase of automation systems (see Sect. 4.3.1).

The third venue is the *International Conference on Emerging Technologies and Factory Automation (ETFA)*, where 13 papers were submitted. Approaches based on SysML are in line with this conference, since the main topic of this conference is complex systems, and among others, one goal of SysML is to support the modeling of systems considering software as well as hardware components.

All other venues listed in Table 2 have at most 12 publications. Even though these venues are focusing on different subjects, all of them capture the main topics of SysML such as design, simulation, and complex systems. With regard to journals (see Table 3), the listed ones all deal with systems engineering or software engineering topics, whereby the journal Systems Engineering (with a number of 11 publications) has the most published articles with a focus on SysML.

The distribution curve across all these publications (most prominent venues: conferences, journals) in the time frame from 2005 to 2017 regarding the main research communities is shown in Fig. 4. Regarding the number of publications per year, it is obvious that most contributions were published in the field of systems engineering.

Based on the provided information in the relevant abstract of each publication, we classified the publication in various application fields in a double-checking process (extractor/checker). Those publications that do not clearly belong to a specific application field are discussed in the group. If no unambiguous assignment is possible even after this dis-

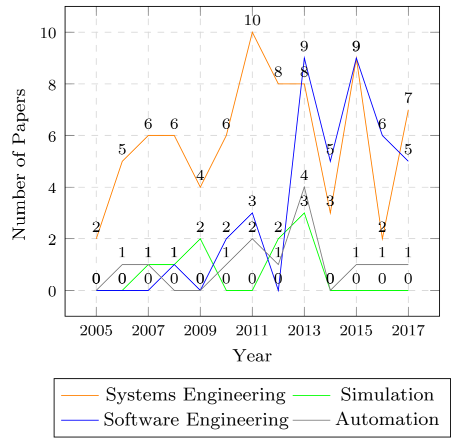


Fig. 4 Number of publications per year regarding the research fields of the 12 most prominent venues (included number of studies: 579)

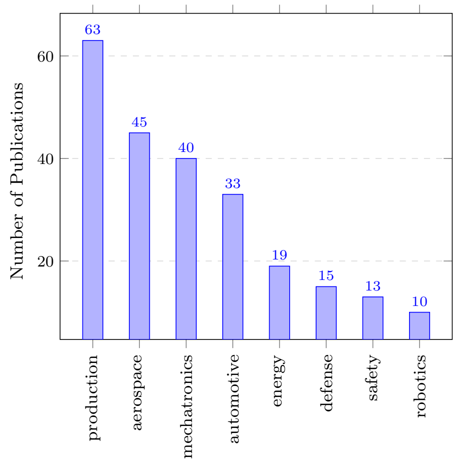


Fig. 5 Application domains with at least 10 publications (included number of studies: 579)

cussion, the paper is identified as not classified. Based on these two processes, unfortunately 229 paper are not classified in our result set. Figure 5 shows the different application domains that have at least 10 publications. Summarized, SysML is experiencing a strong application in the production area, followed by the aerospace sector (both aircraft and space applications), which is closely followed by the application field of mechatronics. In addition, SysML is frequently used for system engineering modeling in the areas of automotive and energy.

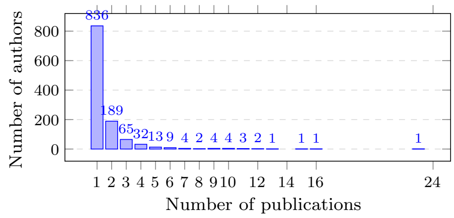


Fig. 6 Number of publications per author (included number of studies: 579)

RQ 1—Main Findings: Generally, SysML is a modeling language to support systems engineers in their work. The findings related to RQ1 show that most of the publications were published in venues with a systems engineering background. However, the analysis of contributions submitted to the main venues of Tables 2 and 3 shows that there is an emerging interest on SysML in the research field of software engineering since 2013 (see Fig. 4). This could be the fact, since systems are becoming more and more software intensive. Another indication is associated with the fact that from 2013 German initiatives increasingly have started to implement the concepts of Industry 4.0 in the production area such as Internet of Things (IoT), Cyber Physical Systems (CPS), Cyber Physical Production Systems (CPPS), which are focusing on both software and hardware. Besides the work of the German initiatives, changes to SysML 1.3 were made in June 2012 concerning the redefinition of physical flows and architectural alignments with UML, which could also be another effect for increasing the attractiveness of SysML in software engineering and systems engineering.

4.2 RQ 2: research communities and main contributors of SysML topics

In a first step, we analyzed the number of authors and their publications. We identified in total 1167 authors, of whom 30 are single authors without relationship to any other author of the result set. Twenty-seven of these single or "non-related" authors have published only one publication with a SysML topic. The "related" authors have at least one relationship to an author, who also has published a paper about SysML. Figure 6 illustrates the number of publications per author.

It should be noted that most of the related as well as non-related authors (in total 836) have published only a single publication about SysML. However, there are 13 authors, who worked more closely on the topic and wrote at least 10 papers. The specific affiliations of these authors are shown in Fig. 7. It is worth to mention that these 13 authors belong to eight different institutions. Some of them like Hammad,

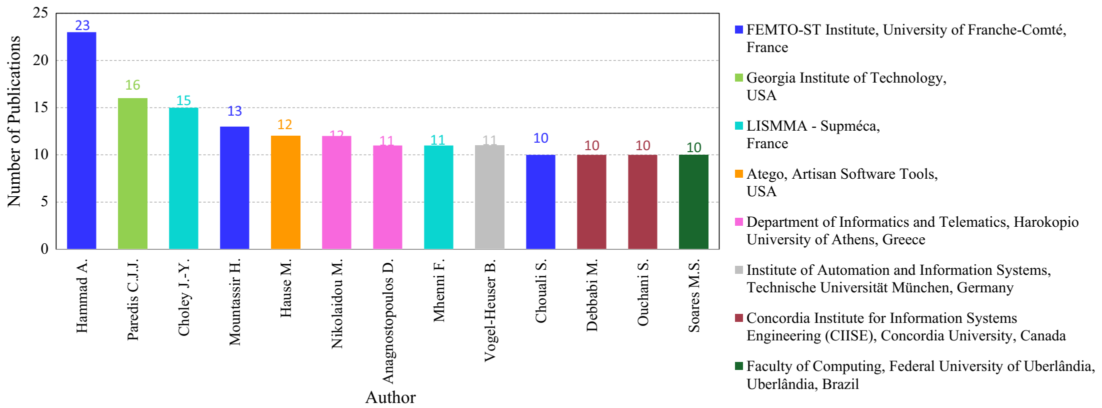


Fig. 7 Affiliation of authors with at least 10 papers (included number of studies: 579)

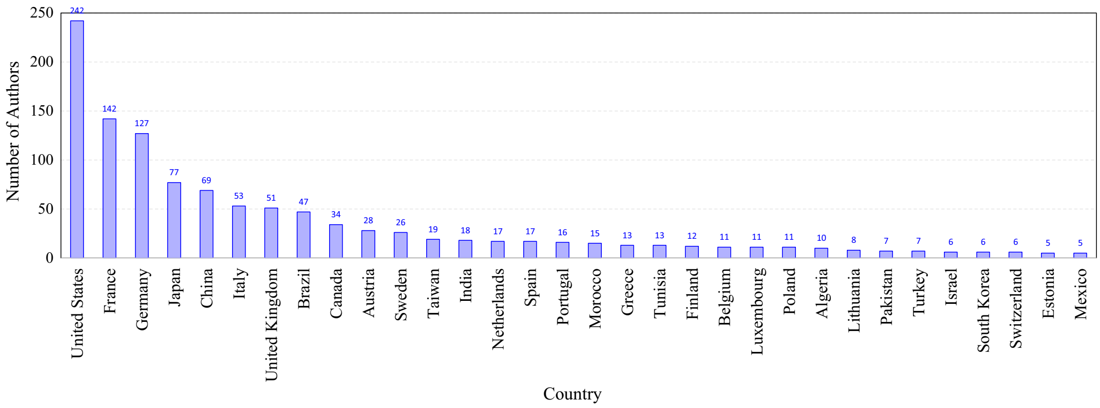


Fig. 8 Number of authors per country with at least 5 authors (included number of studies: 579)

Mountassir, and Chouali are working together in the same research group, whereas other prominent ones like Paredis, Hause, Vogel-Heuser, and Soares publish on behalf of their own research groups.

For deriving the distribution of related and non-related authors over the world, we took a look at their affiliation to a country. Thereby, we found out that most of the authors are from the USA, followed by France, and Germany (see Fig. 8). Figure 9 illustrates the distribution of authors from a continental perspective. Most of the authors are from Europe (48.8%) followed by North America (23.7%).

In a next step, we analyzed the relationship among these authors to get an overview of networks between them. For this network analysis, we used the free tool Gephi.[4] The results of this analysis are shown in Fig. 10. It illustrates all links among the 1167 authors. A link exists as soon as

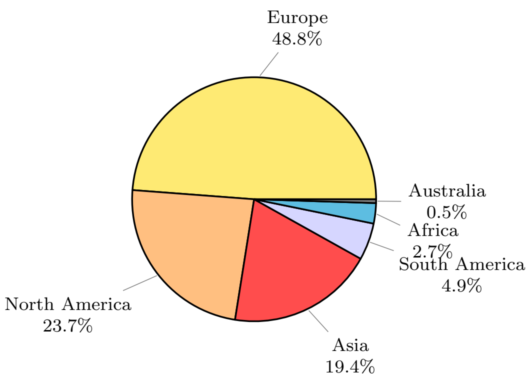


Fig. 9 Percentage of numbers of authors per continent (included number of studies: 579)

[4] https://gephi.org.

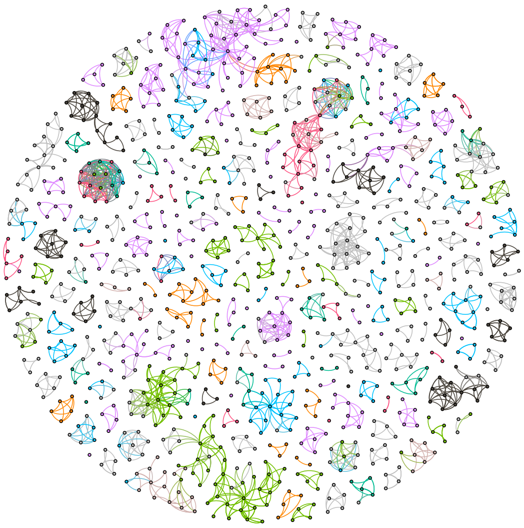


Fig. 10 Connections between authors (created by Gephi, included number of studies: 579)

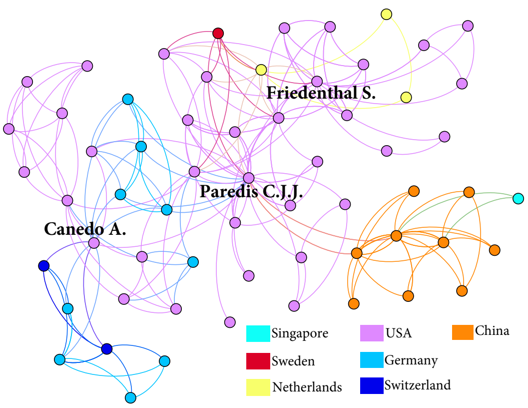


Fig. 11 Biggest network between authors with bridge builder (created by Gephi, included number of studies: 579)

one author worked with another author on the same publication. Based on Fig. 10, we identified that there are several research networks for SysML, but not a single big one. For a deeper analysis, we took a closer look at the largest network in the entire graph. This research network consists of 61 authors and is shown in Fig. 11. In this figure, we only name the so-called "bridge builders," who are the authors Paredis from the Georgia Institute of Technology in Atlanta (USA), Friedenthal from the Lockheed Martin Corporation in Fairfax (USA), and Canedo, who is working at the Siemens Corporation Research in Princeton (USA). These three authors are the anchor points linking the research networks across the world.

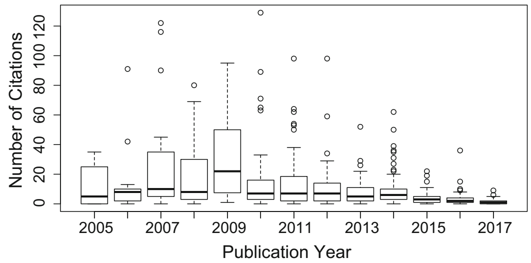


Fig. 12 Distribution of citations of publications by year (included number of studies: 579)

Finally, we analyzed the influence of the selected publications on the scientific community. We used Google Scholar for counting the citations, since we found no information about citation count in the other used research libraries, except Scopus. In order to make the distribution of citations more comprehensible and to show which publications are cited most frequently in an annual comparison (see Fig. 12, outliers), we have chosen a boxplot for visualization as shown in Fig. 12. The top three papers [12,21,43] are each cited more than 100 times. These three papers are focusing on the SysML topics: simulation, physical systems, and design. These topics, as well as other SysML topics, are one of the most important issues, as the tag cloud shows (see Fig. 14), which we will discuss later on when presenting the results of RQ 4.

RQ 2—Main Findings: SysML research takes place worldwide with major contributors especially located in the USA, France, and Germany. However, we discovered that the research interest on SysML topics seems to be stronger in Europe rather than in other continents (see Fig. 9). The social network analysis shows an active community that fostered the discussion on SysML over the years. This network created impact in several engineering domains, ranging from frequently cited basic knowledge (e.g., system and simulation modeling using SysML) to contributions on very specific approaches (e.g., SysML4Modelica, SysML4Mechatronic). In addition, the network analysis shows that there are many smaller research groups working on SysML topics. These groups are partly interconnected by so-called "bridge-builders." It can be concluded that the interest on SysML has been expanded, since researchers moved to other research groups, bringing their knowledge and interest on SysML topics in these groups in order to work on further SysML approaches.

4.3 RQ 3: classification of SysML publications

For the classification process, we used the definition of categories as described in Sect. 3.3. In a first step, each of us

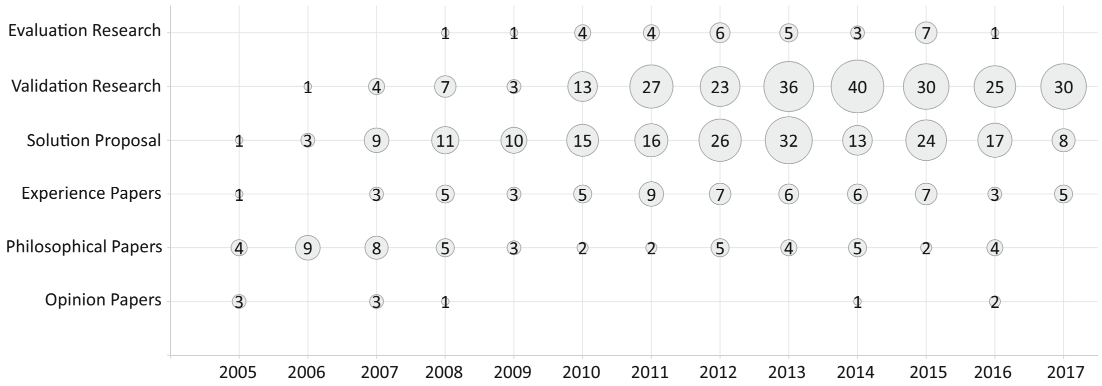


Fig. 13 Number of publications per year according to the type facet classification (included number of studies: 579)

Fig. 14 Tag cloud of most important terms (created with http://tagcrowd.com/, included number of studies: 579)

individually categorized the abstracts of the selected publications based on one of the six different type facets introduced by Petersen et al. [44] (see Table 1). This classification offers the possibility to find out whether SysML is used in own approaches, in experiments, or in theoretical considerations. In a second step, we discussed the categorizations and potential conflicts in the group. Based on these discussions, the conflicting papers were finally assigned to one category. The results of this classification process are shown in Fig. 13.

The result set comprises 10 opinion papers, 32 evaluation papers, 53 philosophical papers, and 60 experience papers. We found out that philosophical papers are so to speak the "pioneers" in the introductory phase of the standard until about 2007. There are negligibly few opinion papers in the result set. The majority of the publications are assigned to the categories solution proposals (185 papers) and validation research (239 papers). This means that the SysML topic of the majority of papers is an own approach and its sample implementation. Figure 13 shows the result set and its chronological sequence which indicates that from 2010 onward validation research and solution proposal become more and more prominent. There are only few evaluation papers, since this category requires a preceding solution implementation and based on this groundwork an evaluation within a practical setting with an industry partner.

> *RQ 3—Main Findings:* The higher number of philosophical papers in the time period from 2005 to 2007 could be explained by the fact that at the beginning of a new modeling language standard the comparison with other modeling standards is usually the focus. Over time when the standard is established, researchers are more interested to take advantage of the standard to realize own approaches and their implementations. This development has been in the foreground since 2010.

4.3.1 RQ 4: key aspects of applying SysML

Additionally to the categorization of publications based on research type facets, we want to detect the key aspects for applying SysML. For this purpose, we firstly created a tag cloud based on all abstracts of the publications of our result set. (German and French abstracts were translated.) Figure 14 shows this tag cloud, which gives us an overview of the 50 most important keywords (with a frequency of at least 50 times) of the abstracts. It should be mentioned that we have deleted conjunctions and keywords like "SysML," "Systems," "Modeling," "Language" as well as "UML," since we already used them for our general keyword searching when conducting the second activity of the mapping process (see Sect. 3.2).

The most frequently used keywords are *design*, *requirements*, *process*, and *simulation*. Based on the frequency of these keywords, it can be derived that SysML is most frequently used in connection with design and requirement problems. It should be noticed that in this tag cloud every term is counted as often as it occurs in the selected abstracts. In addition, keywords like *implementation*, *verification*, and *validation* frequently appear.

For an even more detailed analysis of the application of SysML topics in the selected publications, we clustered the result set according to systems engineering phases and contribution types, as defined and described in Sect. 3. In the following, we give a summary of the result set analyzed based on engineering phases:

- *Requirements:* In this initial phase of the engineering process, SysML is used to describe system requirements. The requirements representation is enhanced by a graphical view and by an explicit mapping of the relationships between them. Additionally, the traceability is significantly improved by the so-called "requirements tables." This in SysML newly introduced diagram type helps to bridge the gap between documents written in natural language and modeled use cases. SysML is also used for modeling non-functional requirements. Besides the requirement diagram, the parametric diagram is used to formally describe design requirements for verification and validation purposes.
- *Design:* We found out that in the design phase, SysML is often used to get a better system understanding and to improve interoperability. In many publications, SysML is used to get a detailed picture of the designed system. Increasingly, SysML is used (i) as modeling language for hardware systems, (ii) for concurrent design processes, (iii) for mechanical concept designs, and (iv) for solving aerospace development problems. Additionally to fulfill special design requirements, SysML is extended by profiles, used in combination with, e.g., MARTE, or mapped to other models.
- *Implementation:* In the implementation phase, SysML is often used in combination with other languages like SystemC, Modelica, or DEVS to support the implementation of an executable architecture that provides a feasible systems engineering solution. Generally, SysML models are used as basis for the structural and behavioral description of systems. Based on SysML models, executable code is generated by code generators or model transformations are performed by model transformation languages like QVT.
- *Validation and Verification:* Regarding the V&V phase, we identified that different approaches deal with (i) model checking for the assessment and evaluation of performance characteristics, (ii) generating automated test cases out of models, and (iii) reliability analysis. The formalization of SysML models allows to building frameworks for the verification and validation of systems design.

Regarding the research results of the contribution type, there are different research fields addressed in the result set, briefly described as follows:

- *Technique:* There are a lot of different techniques presented in the publications. Most of them deal with (i) efficient modeling of requirements (functional and non-functional), (ii) performing parametric analysis of complex systems, and (iii) verification of designed models.
- *Process:* It could be identified that the support of the development process of systems stands in the foreground. Most of the presented approaches deal with the development of requirements up to the entire design phase, whereas only few publications address the process beyond the design phase.
- *Notation:* We found out that in relation to language engineering, most of the publications of the result set deal with SysML profiles. There are extensions and profiles for (i) facilitating the verification of non-functional quantitative requirements, (ii) improving the application of SysML to complex systems, and (iii) using SysML in the automation, mechatronic system, or embedded system domain. In addition to profiles, there are approaches focusing on translation like transformation to Petri Net or Matlab/Simulink. SysML is also used in combination with OCL, OPM, or MARTE.
- *Tool:* Approaches in this category mostly engage in the development of tools, e.g., to create and versioning SysML models. There are, for example, requirements modeling tools based on SysML and also tools integrating SysML in a process and design optimization framework. Additionally, there exist approaches that use SysML in combination with simulation frameworks or engines like fUML[5] and James II.[6]
- *Specific Solution:* There are some specific solutions based on SysML, for example, for space systems, automotive systems, or embedded systems. It can be said that the focus of these solutions is on describing the special requirements of the respective projects.
- *Other:* The main aspects for assigning publications to the contribution type called "other" are: (i) the comparisons

[5] http://www.omg.org/spec/FUML/.

[6] http://jamesii.informatik.uni-rostock.de/jamesii.org/.

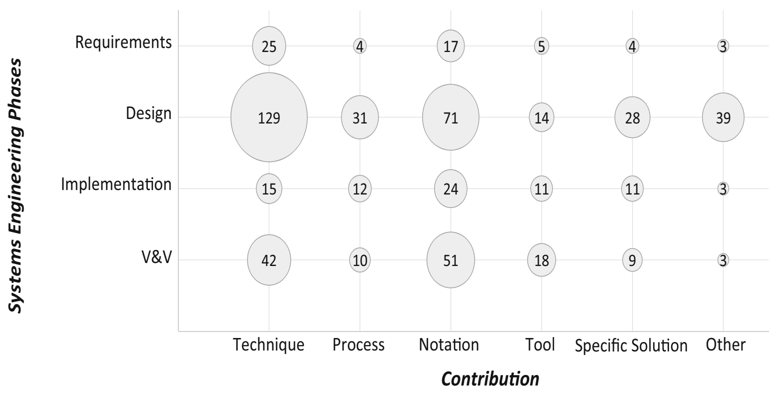


Fig. 15 Distribution of publications by systems engineering and contribution (included number of studies: 579)

of SysML to other modeling languages, (ii) the analysis of the usability of SysML diagrams like requirement view and parametric diagram, and (iii) teaching systems modeling in SysML.

We connected the results of the systems engineering phases and the contribution types together and visualized it in Fig. 15. The distribution shows that most of the publications deal with problems in the design phase followed by the V&V phase. To realize their approaches, the authors mostly develop their own techniques and notations.

There are many papers in our result set dealing with SysML extensions or transformations to other languages, techniques, tools, and concepts. Therefore, we have analyzed the information provided in the abstracts for creating a formalism transformation graph (FTG) [34], and additionally based on the same principle, we created a formalism extension graph (FEG).

The FTG graph in Fig. 16 shows the various transformations of SysML to other languages, techniques, tools, and concepts for different application scenarios such as simulation, verification, analysis, and extracting code.[7] In addition, the FEG in Fig. 17 shows the different extensions of SysML used in the approaches, techniques, and methods introduced and presented in the papers of our result set.[8] Besides the shown transformations and extensions, there are two publications describing linking techniques for SysML to other languages, one to Relax and one to Simulink.

It can be summarized that most of the SysML publications are directed toward individual approaches for the design or validation of systems. In most cases, established languages, mechanisms for extension, and transformations are used. To illustrate these main findings, we give an overall view in Fig. 18 where we show the systematic map of SysML publications regarding type facets, systems engineering phases, and contribution types. This figure presents the interplay of all the probed categories and their classification as output of the last activity of the systematic mapping process (see Sect. 3, Fig. 1).

RQ 4—Main Findings: It turns out that in the area of systems engineering the phases design and validation are predominant topics in all type facet categories. Thus, design and validation are clearly the dominant engineering phases in the usage of SysML. Regarding contribution types, we concluded that the types focusing on technology and notation are more likely to be found in the research type facets "Solution Proposals" and "Validation Research." In the other type facets categories, the main contribution type can differ. For instance, presentations of specific solutions is an important input for "Experience Papers."

5 Threads to validity

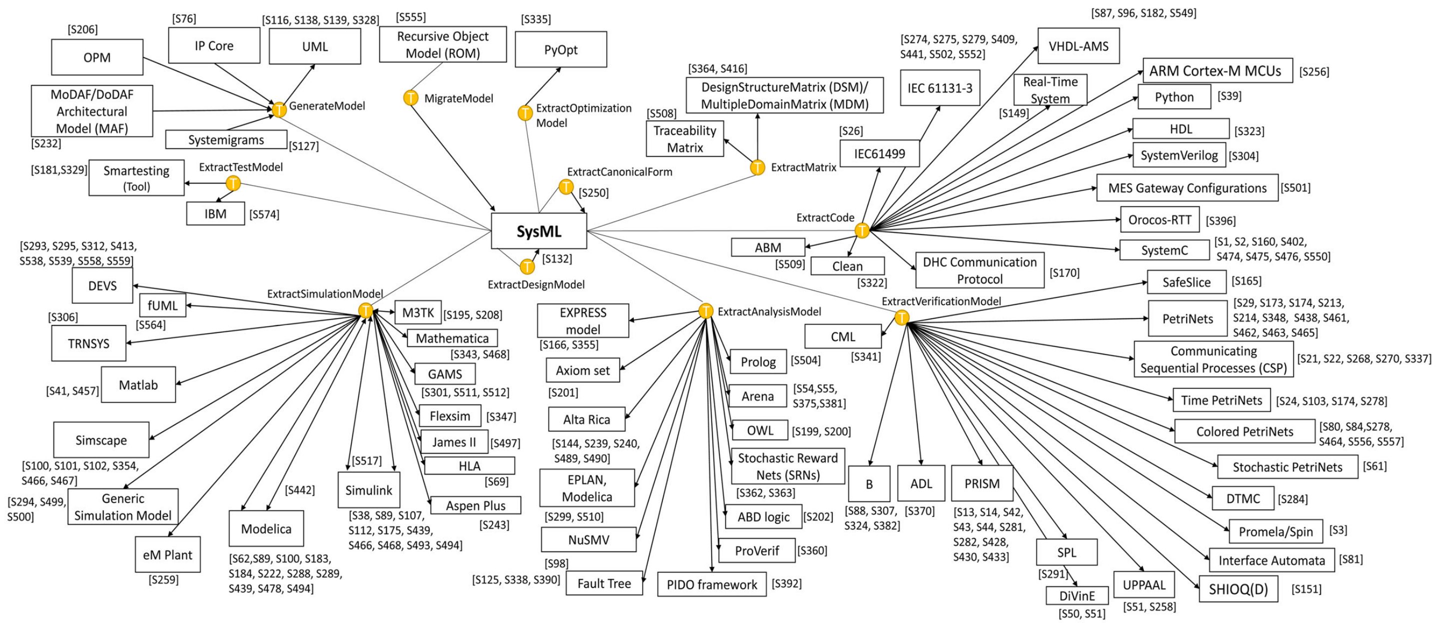
For identifying the threats of validity of our SMS, we follow the four basic types of validity threats according to Wohlin et al. [61]. We address each of these threats in the following subsections.

5.1 Conclusion validity

Conclusion validity takes care of issues that might arise when drawing conclusions and whether the SMS can be repeated. According to Wohlin et al. [61], the main focus is to draw a correct conclusion regarding relations between the design and outcome of the study. In the given SMS, threats to conclusion validity include:

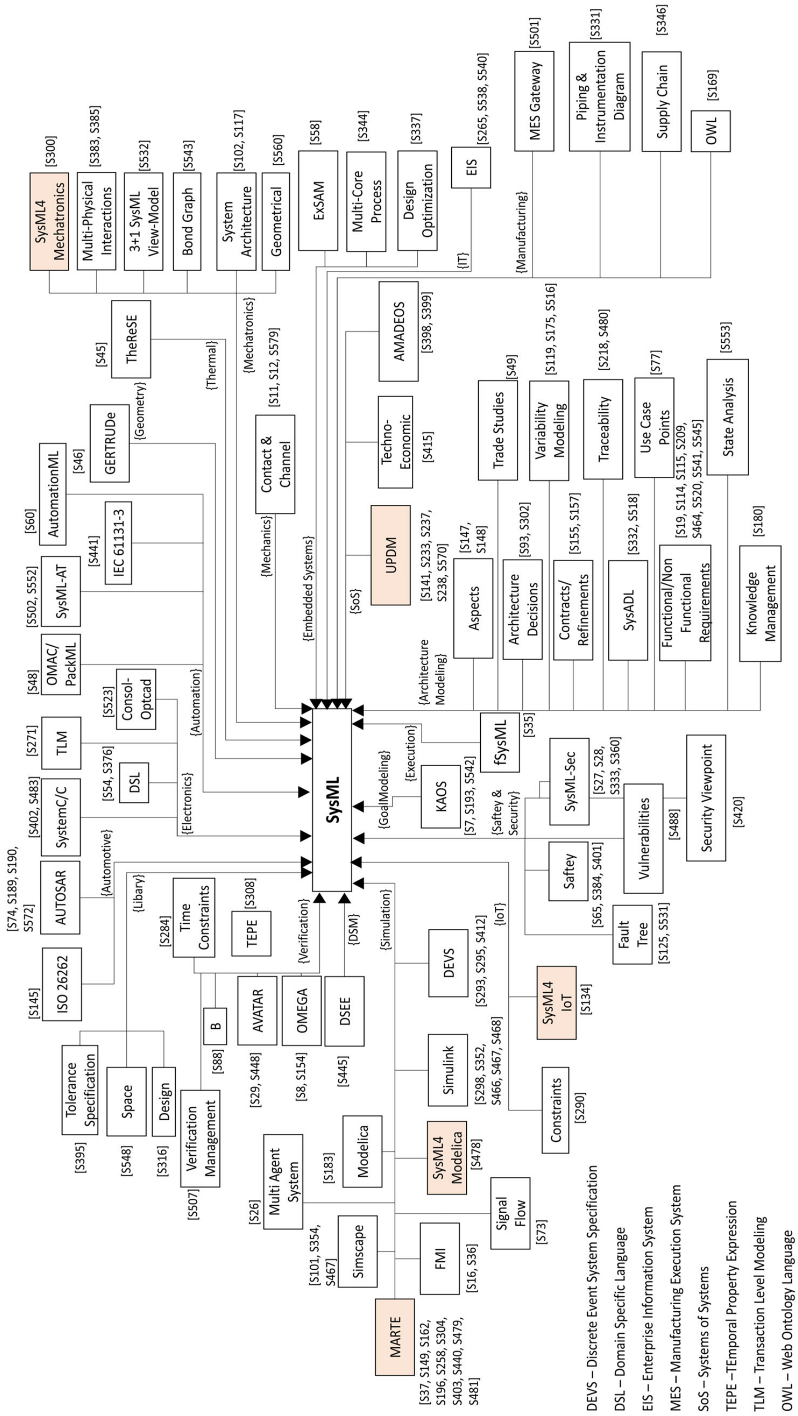
[7] Graph also available at https://figshare.com/s/5de5b35ed2ef8cdf8317.

[8] Graph also available at https://figshare.com/s/0f0f13ea189b891e312f.



- GAMS – General Algebraic Modeling System
- M3TK – Mobility Mechanics Modeling Toolkit
- DEVS –Discrete Event System Specification
- TRNSYS – Transient System Simulation Tool
- HLA – High Level Architecture
- OWL – Web Ontology Language
- ABD –Abstract Block Diagram
- PIDO – Process Integration and Design Optimization
- IBM –Intermediate Black box Model
- OPM – Object-Process Methodology
- MoDAF/DoDAF – Department of Defense Architecture Framework
- VHDL-AMS – Very High Speed Integrated Circuit Hardware Description Language, mixed-signal extensions
- HDL – Hardware Description Language
- MES – Manufacturing Execution System
- DHC – Digital Home Compliant
- Orocos-RTT – Orocos Real-Time Toolkit
- ABM – Agent-Based Modeling
- DTMC – Discrete-Time Markov Chains
- SPL – Sequence Planner Language
- PRISM – Probabilistic Symbolic Model-checker
- ADL – Architectural Description Language
- CML – COMPASS Modelling Language

Fig. 16 Formalism transformation graph (FTG) of SysML publications (included number of studies: 579)



- DEVS – Discrete Event System Specification
- DSL – Domain Specific Language
- EIS – Enterprise Information System
- MES – Manufacturing Execution System
- SoS – Systems of Systems
- TEPE –TEmporal Property Expression
- TLM – Transaction Level Modeling
- OWL – Web Ontology Language
- UPDM - Unified Profile for DoDAF and MODAF

Fig. 17 Formalism extension graph (FEG) of SysML publications (included number of studies: 579)

- *Subjective measures*, such as the manual categorization of abstracts to the research type facets of Petersen et al. [44].
- *Low statistical power*, due to the restricted amount of identified publications (e.g., a few publications may influence the ranking of prominent contributors).
- *Fishing (searching) for specific results*, since the results are influenced by the chosen selection of publications (see internal validity).

An additional threat to the validity of the conclusion of a SMS is the publication bias. The term "publication bias" occurs when studies with non-significant findings are either be not submitted by their authors, or may be rejected by reviewers and/or editors and then this could be a risk considering our research type facets. For instance, the risk could be based on the reason that opinion papers are less frequent, since they are more often rejected and become either unofficial technical reports or unpublished studies. To counteract to this risk, we use different databases with various scopes.

As mitigation strategy against subjective measures, the papers of the result set were classified by each of us based on the strategy introduced in Petersen et al. [44], presented in Table 1. Subsequently, these classifications were discussed among themselves. Thereby, occurred discrepancies were considered in more detail and discussed in the group before we re-classified them. Once again, our mitigation strategy against the low statistical power is the use of four different digital libraries to obtain the most complete possible set of papers focusing on SysML as research topic. A comparison with a sampling method such as introduced in [30] would be interesting in order to see whether the same publications would be found. Unfortunately, this investigation goes beyond the scope of this article.

5.2 Internal validity

Threats to internal validity address issues that indicate a causal relationship, such as hidden factors. This phenomenon is also known as spurious correlation. Therefore, the main goal is to guarantee that the methods used in the SMS cause the outcome of the survey. It should be mentioned that factors which impact the internal validity are also significantly influencing the process of the research subjects' (i.e., publications') selection. For a better understanding of the internal validity regarding our SMS, we describe in more detail the two influencing factors, selection and instrumentation, based on Wohlin et al. [61], in the following:

- *Publication selection* based on:
  - *Keywords*: only the title of publications were searched for the following keywords: 'SysML' or 'System[s] Modeling Language'.
  - *Time frame*: restricted from 2005 to 2017, since the first draft of SysML specification was published in 2005. The idea of UML for Systems Engineering was already issued in 2003 but with a different naming.
  - *Literature repositories*: we took into account four different literature repositories, which are Scopus, ACM Digital Library, IEEE Xplore Digital Library, as well as DBLP.
  - *Publication language*: only publications with English, German, or French abstracts were considered, even though the repositories have provided additional abstracts satisfying the keywords as well as time frame, like Chinese or Spanish publications.
  - *Manual filtering*: we deleted duplicates, books, and theses, as well as publications without abstract or research context to SysML.
- *Instrumentation* caused by design of artifacts:
  This includes, for instance, *timeliness* and *completeness* of literature repositories to answer the question which venues are considered by those libraries. It may be possible to delay previously published articles like in the case of post-proceedings, and therefore, they are not available online.

Our mitigation strategy to address risks of publication selection and instrumentation was to avoid too tight restrictions by considering alternatives. For instance, (i) three different keywords based on our mapping scope were used for the search process, (ii) a time frame was applied that started with the first draft of SysML, and (iii) four broad-based literature repositories were taken into account for conducting the search. In contrast to [63], we use the four libraries ACM, IEEE, DBLP, and Scopus and not, for example, SpringerLink. However, SpringerLink references are included in DBLP and Scopus and therefore implicit in our mapping study. In addition, Scopus contains many publications in the field of systems engineering that do not appear in the other libraries. Thus, by this mitigation strategy, the result set may cover a representative set of relevant publications.

Regarding manual filtering, a certain bias remains according to publications with heterogeneous titles and abstracts, but identical content. We discussed this issue in the group. However, this uncertainty remains open due to method we have chosen for this SMS based on Petersen et al. [44].

5.3 Construct validity

Construct validity concerns the relationship between theory and observation. According to Wohlin et al. [61], construct threats to validity cope with issues that might arise during research design. Thus, it should be checked if the used concept is sufficient. There are two kinds of threats to construct

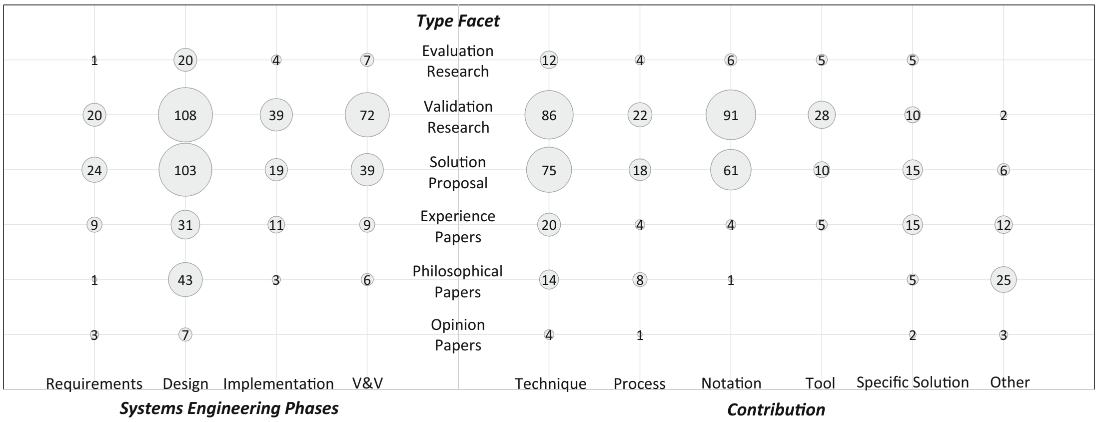


Fig. 18 Systematic map of SysML papers (included number of studies: 579)

validity, which are (i) *design threats* such as mono-operation bias, mono-method bias, or confounding constructs as well as levels of constructs, and (ii) *social threats* such as hypothesis guessing, evaluation apprehension, and experimenter expectancies [61]. It should be mentioned that social threats do not apply to non-personal subjects (such as publications); however, they may be relevant regarding the authors of this mapping study [61].

In the given SMS, threats to construct validity include:

- *Mono-method bias*: the study is mainly based on the systematic mapping process introduced by Petersen et al. [44]. In this context, the mitigation strategy was that two independent research groups have worked cooperatively in this study. In doing so, the first literature review has been independently carried out by each group.
- *Confounding constructs and levels of constructs*: for instance, in the case of categorization, there is more than only one type facet applicable (e.g., validation research vs. solution proposal). In cases where the levels of applicable type facets are relevant, we selected the most fitting category based on objective aspects and discussion within the group.
- *Hypothesis guessing*: since the authors of this article are familiar with systems engineering and SysML, some outcomes might be expected such as increasing publications over the years, or close relationships among research groups. We minimized this risk by using an open research design where we have generated knowledge instead of only checking it.

5.4 External validity

The external validity is concerned with "generalization," and whether the result of a study can be generalized outside the scope of the study or not. According to this validity, there are three main risk types [61]: (i) interaction of participants and treatment, (ii) interaction of environment/setting and treatment, and (iii) interaction of history/timing and treatment. However, in the presented SMS, we do not aim for generalization. Given our scope (keywords, time frame, etc.), the SMS aimed for completeness; however, no extensive literature survey can ever claim to be complete. Our SMS is concerned with scientific research on SysML and can not be generalized to closely related research field. Although some conclusions could be generalized to a broader topic (e.g., lack of evaluation research studies), we did not draw such general conclusions.

6 Conclusion and outlook

In this article, we report on our findings regarding the investigated research topics on SysML over the last thirteen years by performing a systematic mapping study. We found out that initially most of the publications were published in systems engineering venues, but since 2013, the research interest on SysML topics moves more toward software engineering. It may be concluded that this moving interest results from the fact that in 2013 Industry 4.0 initiatives started to implement their visions such as CPPS, IoT, IIoT, and others. Therefore, SysML has been very strongly represented in the production application area since that time. Also it seems that the research interest on SysML topics seems to be stronger in Europe than on other continents, since the Industry 4.0 vision started in Germany. However, in Asia and the USA, there started also similar initiatives known under the umbrella "advanced manufacturing" [10] which also stimulate research on software engineering and SysML.

It can be summarized that out of the nine SysML diagram types, the following ones are mainly used: requirement diagram, parametric diagram, activity diagram, state machine diagram, block definition diagram, and internal block diagram. It turned out that the two newly introduced diagram types—requirement diagram and parametric diagram—are accepted and frequently used by the academia research community. SysML is well established as modeling language for designing, analyzing, and verifying complex systems. However, many researchers customize SysML for their purposes, and therefore, define their own profiles since SysML seems still too generic for some domain-specific tasks (e.g., SysML4Modelica [42,50], SysML4Mechatronics [4,24] to mention just a few approaches). An additional finding is that SysML is lacking of operational semantics. Some approaches aim to overcome this gap such as fSysML [3] which is similar to fUML (a foundational subset of UML for executable UML models).

Toward SysML v2

The OMG is currently working on a new version of SysML in version 2 (abbreviated SysML v2). Based on the first insights from the draft *SysML v2 Requirements*[9], it becomes apparent that the main challenges regarding the usage of SysML, which we have identified and discussed in the presented mapping study, were also admitted in the current work of the standardization group. For instance, SysML v2 is intended to expand the requirement diagram by formal definitions of non-textual requirements in order to make these requirements more general and subject to automated validation. Additionally, the draft addresses the issue of ambiguous operational semantics of SysML, trying to solve this ambiguity similarly to the fUML initiative. There are also planned enhancements to have a timing component in models, which is an important issue, e.g., when modeling continuous systems in combination with discrete systems.
Based on the presented SMS and its results and main findings, we identified the following research directions for future work.

Research direction 1: life cycle support

The results show that there is only limited support when using SysML in the implementation phase, and very limited support for describing the whole life cycle of a system from design until operation and back again, for implementing so-called "liquid models" [36]. Therefore, a future research direction is to exploit and adapt SysML for supporting the execution and analysis of systems during runtime and to align operational data with design models.

[9] http://www.omg.org/cgi-bin/doc?ad/2017-12-02.

Research direction 2: modeling hybrid systems

Most of the selected publications consider either discrete or continuous challenges when designing systems [6,23]. This means that very rarely hybrid solutions in systems design are taken into account [19]. Therefore, further investigations should be undertaken for defining formal semantics for SysML to close the gap when combining discrete and continuous modeling and simulation.

Research direction 3: operational semantics for SysML

Currently, there is no support, e.g., to shift property specification and verification tasks up to the model level. There is still a rule-based operational semantics missing to ensure a step-wise, state-based semantics, e.g., to describe a finite execution trace through a sequence of changes. In this context, a future research direction is to define a rule-based operational semantics for SysML, e.g., based on foundations done in the context of fUML.

Research direction 4: deeper analysis of the publication corpus for further research questions Our result set provides a good foundation for deeper analysis for specific topics related to SysML regarding particular research fields. For example, the contribution category notation can be further differentiated into various language engineering aspects (e.g., profiling, translation to another languages, etc.). Based on such analysis, it is possible to characterize similarities and differences among various approaches.

Acknowledgements Open access funding provided by Johannes Kepler University Linz. This work has been supported by the Austrian Federal Ministry for Digital and Economic Affairs, the National Foundation for Research, Technology and Development. The research reported in this paper has also partly been supported by the FWF in the Project TETRABox under the Grant Number P28519-N31, the Austrian Ministry for Transport, Innovation and Technology, and the Province of Upper Austria in the frame of the COMET center SCCH.

A SysML-papers

[S1] Abdulhameed, A., Hammad, A., Mountassir, H., Tatibouët, B.: An Approach based on SysML and SystemC to Simulate Complex Systems. In: Proceedings of the 2nd International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2014), Lisbon, Portu-

gal, January 7–9, pp. 555–560. SciTePress (2014). https://doi.org/10.5220/0004809205550560
[S2] Abdulhameed, A., Hammad, A., Mountassir, H., Tatibouët, B.: An Approach Combining Simulation and Verification for SysML using SystemC and Uppaal. In: Proceedings of the 8th Conference Francophone sur l'Architecture Logicielle (CAL), Paris, France, June 10–11. Revue des Nouvelles Technologies de l'Information (RNTI) (2014)
[S3] Abdulhameed, A., Hammad, A., Mountassir, H., Tatibouet, B.: An approach to verify SysML functional requirements using Promela/SPIN. In: Proceedings of the 12th International Symposium on Programming and Systems (ISPS), Apr., pp. 323–331 (2015). https://doi.org/10.1109/ISPS.2015.7245003
[S4] Abid, A., Barkallah, M., Hammadi, M., Choley, J.Y., Louati, J., Rivière, A., Haddar, M.: Conceptual design of an intelligent welding cell using SysML and holonic paradigm. In: Proceedings of the 6th International Congress on Design and Modeling of Mechanical Systems, Mar 2015, Hammamet, Tunisia, *Lecture Notes in Mechanical Engineering*, vol. 789, pp. 3–10 (2015). https://doi.org/10.1007/978-3-319-17527-0_1
[S5] Abid, A., Hammadi, M., Choley, J.Y., Rivière, A., Barkallah, M., Louati, J., Haddar, M.: A SysML based-methodology for modelling disturbances in manufacturing systems using ADACOR holonic control architecture. In: Proceedings of the 11th France-Japan 9th Europe-Asia Congress on Mechatronics (MECATRONICS) /17th International Conference on Research and Education in Mechatronics (REM), pp. 97–102 (2016). https://doi.org/10.1109/MECATRONICS.2016.7547123
[S6] Abid, H., Pernelle, P., Noterman, D., Campagne, J.P., Ben Amar, C.: SysML approach for the integration of mechatronics system within PLM systems. International Journal of Computer Integrated Manufacturing **28**(9), 972–987 (2015). https://doi.org/10.1080/0951192X.2014.941938
[S7] Ahmad, M., Bruel, J., Laleau, R., Gnaho, C.: Using RELAX, SysML and KAOS for Ambient Systems Requirements Modeling. In: Proceedings of the 3rd International Conference on Ambient Systems, Networks and Technologies (ANT 2012), the 9th International Conference on Mobile Web Information Systems (MobiWIS-2012), Niagara Falls, Ontario, Canada, August 27–29, *Procedia Computer Science*, vol. 10, pp. 474–481. Elsevier (2012). https://doi.org/10.1016/j.procs.2012.06.061
[S8] Ahmad, M., Dragomir, I., Bruel, J., Ober, I., Belloir, N.: Early Analysis of Ambient Systems SYSML Properties using OMEGA2-IFx. In: SIMULTECH 2013 - Proceedings of the 3rd International Conference on Simulation and Modeling Methodologies, Technologies and Applications, Reykjavík, Iceland, July 29–31, pp. 147–154. SciTePress (2013). https://doi.org/10.5220/0004483101470154
[S9] Ahram, T., Karwowski, W.: Human Systems Integration Modeling Using Systems Modeling Language. In: Proceedings of the 53rd Human Factors and Ergonomics Society Annual Meeting, vol. 3, pp. 1849–1853 (2009)
[S10] Al-Fedaghi, S.: Systems Design: SysML vs. Flowthing Modeling. International Journal of Software Engineering and its Applications **8**(1), 355–370 (2014). https://doi.org/10.14257/ijseia.2014.8.1.31
[S11] Albers, A., Zingel, C.: Interdisciplinary Systems Modeling Using the Contact & Channel-Model for SysML. In: Proceedings of the 18th International Conference on Engineering Design - Impacting Society Through Engineering Design (ICED 11), vol. 9, pp. 196–207 (2011)
[S12] Albers, A., Zingel, C.: Extending SysML for Engineering Designers by Integration of the Contact & Channel - Approach (C&C^2-A) for Function-Based Modeling of Technical Systems. In: Proceedings of the Conference on Systems Engineering Research, CSER 2013, Atlanta, Georgia, USA, March 19–22, *Procedia Computer Science*, vol. 16, pp. 353–362. Elsevier (2013). https://doi.org/10.1016/j.procs.2013.01.037
[S13] Ali, S., Basit-Ur-Rahim, M., Arif, F.: Formal verification of internal block diagram of SysML for modeling real-time system. In: Proceedings of the IEEE/ACIS 16th International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, SNPD 2015, Takamatsu, Japan, June 1–3, pp. 617–622. IEEE Computer Society (2015). https://doi.org/10.1109/SNPD.2015.7176271
[S14] Ali, S., Basit-Ur-Rahim, M., Arif, F.: Formal Verification of Time Constrains SysML Internal Block Diagram Using PRISM. In: Proceedings of the 15th International Conference on Computational Science and Its Applications, ICCSA 2015, Banff, AB, Canada, June 22–25, pp. 62–66. IEEE Computer Society (2015). https://doi.org/10.1109/ICCSA.2015.11
[S15] Alt, O.: Integration textueller Anforderungen und Modell-basiertem Testen mit SysML. Softwaretechnik-Trends **28**(3) (2008)
[S16] Amálio, N., Payne, R., Cavalcanti, A., Woodcock, J.: Checking SysML Models for Co-simulation. In: Proceedings of the Formal Methods and Software Engineering: 18th International Conference on For-

mal Engineering Methods, ICFEM 2016, Tokyo, Japan, November 14–18, *LNCS*, vol. 10009, pp. 450–465. Springer International Publishing (2016). https://doi.org/10.1007/978-3-319-47846-3_28

[S17] Ambert, F., Bouquet, F., Lasalle, J., Legeard, B., Peureux, F.: Applying a Def-Use Approach on Signal Exchange to Implement SysML Model-Based Testing. In: Proceedings of the 9th European Conference on Modelling Foundations and Applications (ECMFA 2013), Montpellier, France, July 1–5, *Lecture Notes in Computer Science*, vol. 7949, pp. 134–151. Springer (2013). https://doi.org/10.1007/978-3-642-39013-5_10

[S18] Ammar, N., Chaieb, H., Bouallegue, R.: From Modeling with SysML to Simulation with Contiki Cooja Simulator of Wireless Sensor Networks. In: Proceedings of the 30th International Conference on Advanced Information Networking and Applications Workshops (WAINA), pp. 760–765. IEEE Computer Society (2016). https://doi.org/10.1109/WAINA.2016.64

[S19] Amyot, D., Anda, A.A., Baslyman, M., Lessard, L., Bruel, J.: Towards Improved Requirements Engineering with SysML and the User Requirements Notation. In: Proceedings of the 24th IEEE International Requirements Engineering Conference (RE 2016), Beijing, China, September 12–16, pp. 329–334. IEEE (2016). https://doi.org/10.1109/RE.2016.58

[S20] Andersson, H., Herzog, E., Johansson, G., Johansson, O.: Experience from introducing unified modeling language/systems modeling language at saab aerosystems. Systems Engineering **13**(4), 369–380 (2010). https://doi.org/10.1002/sys.20156

[S21] Ando, T., Yatsu, H., Kong, W., Hisazumi, K., Fukuda, A.: Formalization and Model Checking of SysML State Machine Diagrams by CSP#. In: Proceedings of the 13th International Conference on Computational Science and Its Applications (ICCSA 2013), Ho Chi Minh City, Vietnam, June 24–27, Part III, *Lecture Notes in Computer Science*, vol. 7973, pp. 114–127. Springer (2013). https://doi.org/10.1007/978-3-642-39646-5_9

[S22] Ando, T., Yatsu, H., Kong, W., Hisazumi, K., Fukuda, A.: Translation rules of SysML state machine diagrams into CSP# toward formal model checking. International Journal of Web Information Systems **10**(2), 151–169 (2014). https://doi.org/10.1108/IJWIS-02-2014-0004

[S23] Andrade, E.C., Machida, F., Kim, D.S., Trivedi, K.S.: Modeling and Analyzing Server System with Rejuvenation through SysML and Stochastic Reward Nets. In: Proceedings of the 6th International Conference on Availability, Reliability and Security (ARES 2011), Vienna, Austria, August 22–26, pp. 161–168. IEEE Computer Society (2011). https://doi.org/10.1109/ARES.2011.28

[S24] Andrade, E.C., Maciel, P.R.M., de Almeida Callou, G.R., e Silva Nogueira, B.C.: A Methodology for Mapping SysML Activity Diagram to Time Petri Net for Requirement Validation of Embedded Real-Time Systems with Energy Constraints. In: Proceedings of the 3rd International Conference on the Digital Society (ICDS 2009), Cancun, Mexico, February 1–7, pp. 266–271. IEEE Computer Society (2009). https://doi.org/10.1109/ICDS.2009.19

[S25] Antonio, E., Rovina, R., Fabbri, S.: Verification and Validation Activities for Embedded Systems - A Feasibility Study on a Reading Technique for SysML Models. In: Proceedings of the 16th International Conference on Enterprise Information Systems (ICEIS 2014), Volume 2, Lisbon, Portugal, April 27–30, pp. 233–240. SciTePress (2014). https://doi.org/10.5220/0004887302330240

[S26] Antonova, I., Batchkova, I.: Development of multi-agent control systems using UML/SysML. In: Proceedings of the 4th International IEEE Conference Intelligent Systems (IS 2008), vol. 1, pp. 6-26–6-31 (2008). https://doi.org/10.1109/IS.2008.4670435

[S27] Apvrille, L., Roudier, Y.: Designing Safe and Secure Embedded and Cyber-Physical Systems with SysML-Sec. In: Proceedings of the 3rd International Conference on Model-Driven Engineering and Software Development, MODELSWARD 2015, Angers, France, February 9–11, *Communications in Computer and Information Science*, vol. 580, pp. 293–308. Springer (2015). https://doi.org/10.1007/978-3-319-27869-8_17

[S28] Apvrille, L., Roudier, Y.: SysML-Sec Attack Graphs: Compact Representations for Complex Attacks. In: Proceedings of the 2nd International Workshop on Graphical Models for Security, GraMSec 2015, Verona, Italy, July 13, *Lecture Notes in Computer Science*, vol. 9390, pp. 35–49. Springer (2015). https://doi.org/10.1007/978-3-319-29968-6_3

[S29] Apvrille, L., de Saqui-Sannes, P.: Static Analysis Techniques to Verify Mutual Exclusion Situations within SysML Models. In: SDL 2013: Model-Driven Dependability Engineering - Proceeding of the 16th International SDL Forum, Montreal, Canada, June 26–28, *Lecture Notes in Computer Science*, vol. 7916, pp. 91–106. Springer (2013). https://doi.org/10.1007/978-3-642-38911-5_6

[S30] Arita, M., Kikuchi, M., Kato, T., Hui, P., Matsumoto, A., Sato, K., Matsuoka, Y.: Timeaxis Design of Health Monitoring Seat System Using M Method and SysML. In: Proceedings of the 6th International Conference on Applied Human Factors and Ergonomics and the Affiliated Conferences, AHFE 2015, *Procedia Manufacturing*, vol. 3, pp. 5435–5442. Elsevier B.V. (2015). https://doi.org/10.1016/j.promfg.2015.07.670

[S31] Artery, G., De Spain, M.: Integrating the Life-cycle Process Utilizing SysML. In: Proceedings of the 21st Annual International Symposium of the International Council on Systems Engineering (INCOSE 2011), *INCOSE International Symposium*, vol. 21, pp. 597–609 (2011)

[S32] Artery, G., De Spain, M., Griego, R.: Product life-cycle modeling utilizing SysML modeling. In: Proceedings of the 18th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2008), *INCOSE International Symposium*, vol. 18, pp. 1406–1420 (2008)

[S33] Auriol, G., Baron, C.: New SysML based approach for integrated system design. In: Proceedings of the 7th International Industrial Simulation Conference (ISC 2009), Loughborough, UK, June 1–3, pp. 111–115 (2009)

[S34] Azevedo, K., Bras, B., Doshi, S., Guldberg, T.: Modeling Sustainability of Complex Systems: A Multi-Scale Framework Using SysML. In: Proceedings of the ASME Design Engineering Technical Conference and Computers and Information in Engineering Conference (DETC 2009), vol. 2, pp. 1437–1448 (2009). https://doi.org/10.1115/DETC2009-87496

[S35] Badreddin, O., Abdelzad, V., Lethbridge, T., Elaasar, M.: fSysML: Foundational Executable SysML for Cyber-Physical System Modeling. In: Proceedings of the 4th International Workshop on the Globalization Of Modeling Languages co-located with ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems (MODELS 2016), Saint Malo, France, October 4th, *CEUR Workshop Proceedings*, vol. 1731, pp. 38–51. CEUR-WS.org (2016)

[S36] Bagnato, A., Brosse, E., Quadri, I., Sadovykh, A.: SysML for modeling co-simulation orchestration over FMI: The INTO-CPS approach. In: Proceedings of the 21st International Conference on Reliable Software Technologies, Ada-Europe 2016, *Ada User Journal*, vol. 37, pp. 215–218 (2016)

[S37] Bagnato, A., Quadri, I., Brosse, E., Sadovykh, A., Indrusiak, L., Paige, R., Audsley, N., Gray, I., Kolovos, D., Matragkas, N., Rossi, M., Baresi, L., Crippa, M., Genolini, S., Hansen, S., Meisel-Blohm, G.: MADES FP7 EU project: Effective high level SysML/MARTE methodology for real-time and embedded avionics systems. In: Handbook of Research on Embedded Systems Design, pp. 181–208. IGI Global (2014). https://doi.org/10.4018/978-1-4666-6194-3.ch008

[S38] Bailey, W.C., Che, J., Tsou, P., Jennings, M.: A Framework for Automated Model Interface Coordination Using SysML. In: Proceedings of the ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE), vol. 1 (2017)

[S39] Balestrini-Robinson, S., Freeman, D., Browne, D.: An object-oriented and executable SysML framework for rapid model development. In: Proceedings of the 2015 Conference on Systems Engineering Research, *Procedia Computer Science*, vol. 44, pp. 423–432 (2015). https://doi.org/10.1016/j.procs.2015.03.062

[S40] Balmelli, L.: An overview of the systems modeling language for products and systems development. Journal of Object Technology **6**(6), 149–177 (2007)

[S41] Bank, D., Blumrich, F., Kress, P., Stöferle, C.: A systems engineering approach for a dynamic co-simulation of a sysml tool and matlab. In: Proceedings of the 10th Annual International Systems Conference, SysCon 2016, Orlando, FL, USA, April 18–21, pp. 1–6. IEEE (2016). https://doi.org/10.1109/SYSCON.2016.7490534

[S42] Baouya, A., Bennouar, D., Mohamed, O., Ouchani, S.: A probabilistic and timed verification approach of SysML state machine diagram. In: Proceedings of the 12th International Symposium on Programming and Systems (ISPS), Apr., pp. 304–312 (2015). https://doi.org/10.1109/ISPS.2015.7245001

[S43] Baouya, A., Bennouar, D., Mohamed, O., Ouchani, S.: A quantitative verification framework of SysML activity diagrams under time constraints. Expert Systems with Applications **42**(21), 7493–7510 (2015). https://doi.org/10.1016/j.eswa.2015.05.049

[S44] Baouya, A., Bennouar, D., Mohamed, O., Ouchani, S.: On the Probabilistic Verification of Time Constrained SysML State Machines. In: Proceedings of the 14th International Conference on Intelligent Software Methodologies, Tools and Techniques (SoMeT), Naples, Italy, September 15–17, *Communications in Computer and Information Science*, vol. 532, pp. 425–441. Springer (2015). https://doi.org/10.1007/978-3-319-22689-7_33

[S45] Barbedienne, R., Penas, O., Choley, J.Y., Gasser, L.: TheReSE: SysML extension for thermal modeling. In: Proceedings of the 9th Annual IEEE Systems Conference, SysCon 2015, Vancouver, BC, Canada, April 13–16, pp. 301–308. IEEE (2015). https://doi.org/10.1109/SYSCON.2015.7116768

[S46] Barbedienne, R., Penas, O., Choley, J.Y., Riviere, A., Warniez, A., Della Monica, F.: Introduction of geometrical contraints modeling in SysML for mechatronic design. In: Proceedings of the 10th France-Japan / 8th Europe-Asia Congress on Mecatronics (MECATRONICS), pp. 145–150 (2014). https://doi.org/10.1109/MECATRONICS.2014.7018580

[S47] Barbieri, G., Kernschmidt, K., Fantuzzi, C., Vogel-Heuser, B.: A SysML based design pattern for the high-level development of mechatronic systems to enhance re-usability. In: Proceedings of the 19th IFAC World Congress, *IFAC Proceedings Volumes*, vol. 47, pp. 3431–3437 (2014)

[S48] Bareiß, P., Schütz, D., Priego, R., Marcos, M., Vogel-Heuser, B.: A model-based failure recovery approach for automated production systems combining SysML and industrial standards. In: Proceedings of the 21st IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2016, Berlin, Germany, September 6–9, pp. 1–7. IEEE (2016). https://doi.org/10.1109/ETFA.2016.7733720

[S49] Barnes, P.: A NASA Space Communications and Navigation SysML Profile Adaptation. In: Proceedings of the 22nd Annual International Symposium of the International Council on Systems Engineering (INCOSE 2012) and the 8th Biennial European Systems Engineering Conference (EuSEC 2012), *INCOSE International Symposium*, vol. 22, pp. 1711–1725 (2012)

[S50] Basit Ur Rahim, M., Arif, F., Ahmad, J.: Modeling of Embedded System Using SysML and Its Parallel Verification Using DiVinE Tool. In: Proceedings of the 14th International Conference on Computational Science and Its Applications (ICCSA 2014), Guimarães, Portugal, June 30–July 3, Part V, *Lecture Notes in Computer Science*, vol. 8583, pp. 541–555. Springer (2014). https://doi.org/10.1007/978-3-319-09156-3_38

[S51] Basit-Ur-Rahim, M., Arif, F., Ahmad, J.: Modeling of real-time embedded systems using SysML and its verification using UPPAAL and DiVinE. In: Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS, pp. 132–136. IEEE (2014). https://doi.org/10.1109/ICSESS.2014.6933529

[S52] Bassam, S., Herrmann, J., Schmidt, L.: Using SysML for Model-based Vulnerability Assessment. In: Proceedings of the Conference on Systems Engineering Research, *Procedia Computer Science*, vol. 44, pp. 413–422 (2015). https://doi.org/10.1016/j.procs.2015.03.025

[S53] Bassi, L., Secchi, C., Bonfé, M., Fantuzzi, C.: A SysML-Based Methodology for Manufacturing Machinery Modeling and Design. IEEE/ASME Transactions on Mechatronics **16**(6), 1049–1062 (2011). https://doi.org/10.1109/TMECH.2010.2073480

[S54] Batarseh, O., McGinnis, L.: SysML to discrete-event simulation to analyze electronic assembly systems. In: Proceedings of the 2012 Symposium on Theory of Modeling and Simulation - DEVS Integrative M&S Symposium, Orlando, FL, USA, March 26–29, pp. 48:1–48:8. SCS/ACM (2012)

[S55] Batarseh, O., McGinnis, L.F.: System modeling in SysML and system analysis in Arena. In: Winter Simulation Conference, WSC '12, Berlin, Germany, December 9–12, pp. 258:1–258:12. WSC (2012). https://doi.org/10.1109/WSC.2012.6465139

[S56] Batarseh, O.G., Goldlust, E.J., Day, T.E.: SysML for conceptual modeling and simulation for analysis: A case example of a highly granular model of an emergency department. In: Proceedings of the Winter Simulations Conference: Simulation Making Decisions in a Complex World, WSC 2013, Washington, DC, USA, December 8–11, pp. 2398–2409. IEEE (2013). https://doi.org/10.1109/WSC.2013.6721614

[S57] Batchkova, I., Antonova, I.: Improving the software development life cycle in process control using UML/SysML. In: Proceedings of the 18th IFAC World Congress, *IFAC Proceedings Volumes*, vol. 44, pp. 14,133–14,138 (2011). https://doi.org/10.3182/20110828-6-IT-1002.03190

[S58] Behjati, R., Yue, T., Nejati, S., Briand, L.C., Selic, B.: Extending SysML with AADL Concepts for Comprehensive System Architecture Modeling. In: Proceedings of the 7th European Conference on Modelling Foundations and Applications (ECMFA 2011), Birmingham, UK, June 6–9, *Lecture Notes in Computer Science*, vol. 6698, pp. 236–252. Springer (2011). https://doi.org/10.1007/978-3-642-21470-7_17

[S59] Belloir, N., Bruel, J., Hoang, N., Pham, C.: Utilisation de SysML pour la modélisation des réseaux de capteurs. In: Langages et Modèles à Objets, LMO 2008, Montréal, Québec, Canada, mars 5–7, *RNTI*, vol. L-1, pp. 169–184. Cépaduès-Éditions (2008)

[S60] Berardinelli, L., Biffl, S., Lüder, A., Mätzler, E., Mayerhofer, T., Wimmer, M., Wolny, S.: Cross-disciplinary engineering with AutomationML and SysML. At-Automatisierungstechnik **64**(4), 253–269 (2016). https://doi.org/10.1515/auto-2015-0076

[S61] Berrachedi, A., Rahim, M., Ioualalen, M., Ahmed, H.: Validation of a SysML based design for wireless sensor networks. In: AIP Conference Proceedings of the International Conference of numerical analysis and applied mathematics (ICNAAM 2016), vol. 1863 (2017)

[S62] Berrani, S., Hammad, A., Mountassir, H.: Mapping SysML to modelica to validate wireless sensor networks non-functional requirements. In: Proceedings of the 11th International Symposium on Programming and Systems, ISPS 2013, pp. 177–186 (2013). https://doi.org/10.1109/ISPS.2013.6581484

[S63] Bhatia, G., Mesmer, B.: Integrating SysML and value-based design with an NEA scout small satellite example. In: Proceedings of the AIAA SPACE and Astronautics Forum and Exposition, Orlando, FL. American Institute of Aeronautics and Astronautics (2017). https://doi.org/10.2514/6.2017-5234

[S64] Bhuvaneshwari, S., Vaideki, K., Kumar, K., Margret Anouncia, A.: Problem frames analysis over SysML model for critical goods transportation monitoring system. Journal of Theoretical and Applied Information Technology **43**(2), 222–228 (2012)

[S65] Biggs, G., Sakamoto, T., Kotoku, T.: A profile and tool for modelling safety information with design information in SysML. Software & Systems Modeling **15**(1), 147–178 (2016). https://doi.org/10.1007/s10270-014-0400-x

[S66] Bijan, Y., Yu, J., Graves, H., Stracener, J., Woods, T.: Using MBSE with SysML parametrics to perform requirements analysis. In: Proceedings of the 21st Annual International Symposium of the International Council on Systems Engineering (INCOSE 2011), *INCOSE International Symposium*, vol. 21, pp. 769–782 (2011)

[S67] Biswas, N., Chattopadhyay, S., Mahapatra, G., Chatterjee, S., Mondal, K.C.: SysML Based Conceptual ETL Process Modeling. In: J.K. Mandal, P. Dutta, S. Mukhopadhyay (eds.) Computational Intelligence, Communications, and Business Analytics, pp. 242–255. Springer Singapore (2017)

[S68] Bleakley, G., Lapping, A., Whitfield, A.: Determining the right solution using SysML and model based systems engineering (MBSE) for trade studies. In: Proceedings of the 21st Annual International Symposium of the International Council on Systems Engineering (INCOSE 2011), *INCOSE International Symposium*, vol. 21, pp. 783–795 (2011)

[S69] Bocciarelli, P., D'Ambrogio, A., Fabiani, G.: A Model-driven Approach to Build HLA-based Distributed Simulations from SysML Models. In: Proceedings of the 2nd International Conference on Simulation and Modeling Methodologies, Technologies and Applications, Rome, Italy, July 28–31, pp. 49–60. SciTePress (2012)

[S70] Bocciarelli, P., D'Ambrogio, A., Giglio, A., Gianni, D.: A SaaS-based automated framework to build and execute distributed simulations from SysML models. In: Proceedings of the Winter Simulations Conference: Simulation Making Decisions in a Complex World, WSC 2013, Washington, DC, USA, December 8–11, pp. 1371–1382. IEEE (2013). https://doi.org/10.1109/WSC.2013.6721523

[S71] Bock, C.: SysML and UML 2 support for activity modeling. Systems Engineering **9**(2), 160–186 (2006). https://doi.org/10.1002/sys.20046

[S72] Bock, C.: Componentization in the systems modeling language. Systems Engineering **17**(4), 392–406 (2014). https://doi.org/10.1002/sys.21276

[S73] Bock, C., Barbau, R., Matei, I., Dadfarnia, M.: An Extension of the Systems Modeling Language for Physical Interaction and Signal Flow Simulation. Systems Engineering **20**(5), 395–431 (2017). https://doi.org/10.1002/sys.21380

[S74] Boldt, R.: Creating AUTOSAR systems models using the combined power of UML and SysML. Automotive Industries AI **187**(11) (2007)

[S75] Bombieri, N., Ebeid, E., Fummi, F., Lora, M.: On the Reuse of RTL IPs for SysML Model Generation. In: Proceedings of the 13th International Workshop on Microprocessor Test and Verification (MTV 2012), Austin, TX, USA, December 10–13, pp. 54–59. IEEE Computer Society (2012). https://doi.org/10.1109/MTV.2012.10

[S76] Bombieri, N., Ebeid, E., Fummi, F., Lora, M.: On the reuse of heterogeneous IPs into SysML models for integration validation. Journal of Electronic Testing: Theory and Applications (JETTA) **29**(5), 647–667 (2013). https://doi.org/10.1007/s10836-013-5409-5

[S77] Bone, M., Cloutier, R.: Applying Systems Engineering Modeling Language (SysML) to System Effort Estimation Utilizing Use Case Points. In: Proceedings of the 21st Annual International Symposium of the International Council on Systems Engineering (INCOSE 2011), *INCOSE International Symposium*, vol. 21, pp. 114–127 (2011)

[S78] Bonnet, S., Voirin, J.L., Exertier, D., Normand, V.: Not (strictly) relying on SysML for MBSE: Language, tooling and development perspectives: The Arcadia/Capella rationale. In: Proceedings of the 10th Annual International Systems Conference, SysCon 2016, Orlando, FL, USA, April 18–21, pp. 1–6. IEEE (2016). https://doi.org/10.1109/SYSCON.2016.7490559

[S79] Bostelman, R., Foufou, S., Hong, T., Shah, M.: Model of Mobile Manipulator Performance Measurement using SysML. Journal of Intelligent &

Robotic Systems (2017). https://doi.org/10.1007/s10846-017-0705-4
[S80] Bouabana-Tebibel, T., Rubin, S.H., Bennama, M.: Formal modeling with SysML. In: Proceedings of the 13th IEEE International Conference on Information Reuse & Integration (IRI 2012), Las Vegas, NV, USA, August 8–10, pp. 340–347. IEEE (2012). https://doi.org/10.1109/IRI.2012.6303029
[S81] Bouaziz, H., Chouali, S., Hammad, A., Mountassir, H.: Compatibility verification of SysML blocks using hierarchical interface automata. In: Proceedings of the 12th International Symposium on Programming and Systems (ISPS), Apr., pp. 313–322 (2015). https://doi.org/10.1109/ISPS.2015.7245002
[S82] Bouaziz, H., Chouali, S., Hammad, A., Mountassir, H.: Exploitation de la Hiérarchie pour la Vérification de la Compatibilité des Blocs SysML. In: Avancées récentes dans le domaine des Architectures Logicielles : articles sélectionnés et étendus de CAL 2015 et MODA 2015, Hammamet, Tunisie, 13-15 Mai, *RNTI*, vol. L-8, pp. 99–118. Hermann-Éditions (2015)
[S83] Bouaziz, H., Chouali, S., Hammad, A., Mountassir, H.: SysML Blocks Adaptation. In: Proceedings of the 17th International Conference on Formal Engineering Methods (ICFEM), Paris, France, November 3–5, *Lecture Notes in Computer Science*, vol. 9407, pp. 417–433. Springer (2015). https://doi.org/10.1007/978-3-319-25423-4_27
[S84] Bouaziz, H., Chouali, S., Hammad, A., Mountassir, H.: A model-driven approach to adapt SysML blocks. In: Proceedings of the 22nd International Conference on Information and Software Technologies, ICIST 2016, Druskininkai, Lithuania, October 13–15, *Communications in Computer and Information Science*, vol. 639, pp. 255–268 (2016). https://doi.org/10.1007/978-3-319-46254-7_21
[S85] Bougain, S., Gerhard, D.: A CBR approach for supporting ecodesign with SysML. In: Proceedings of the 21st International Conference on Engineering Design (ICED 17): Product, Services and Systems Design, Vancouver, Canada, 21–25.08, vol. 3, pp. 111–120 (2017)
[S86] Bougain, S., Gerhard, D.: Integrating Environmental Impacts with SysML in MBSE Methods. In: Proceedings of the 24th CIRP Conference on Life Cycle Engineering, *Procedia CIRP*, vol. 61, pp. 715 – 720 (2017). https://doi.org/10.1016/j.procir.2016.11.196
[S87] Bouquet, F., Gauthier, J.M., Hammad, A., Peureux, F.: Transformation of SysML structure diagrams to VHDL-AMS. In: Proceedings of the 2nd Workshop on Design, Control and Software Implementation for Distributed (MEMS, dMEMS 2012), pp. 74–81. IEEE (2012). https://doi.org/10.1109/dMEMS.2012.12
[S88] Bousse, E., Mentré, D., Combemale, B., Baudry, B., Katsuragi, T.: Aligning SysML with the B method to provide V&V for systems engineering. In: Proceedings of the Workshop on Model-Driven Engineering, Verification and Validation, MoDeVVa 2012, Innsbruck, Austria, September, pp. 11–16 (2012). https://doi.org/10.1145/2427376.2427379
[S89] Branscomb, J.M., Paredis, C., Che, J., Jennings, M.J.: Supporting Multidisciplinary Vehicle Analysis Using a Vehicle Reference Architecture Model in SysML. In: Proceedings of the Conference on Systems Engineering Research, CSER 2013, Atlanta, Georgia, USA, March 19–22, *Procedia Computer Science*, vol. 16, pp. 79–88. Elsevier (2013). https://doi.org/10.1016/j.procs.2013.01.009
[S90] Brecher, C., Nittinger, J.A., Karlberger, A.: Model-based Control of a Handling System with SysML. In: Proceedings of the Conference on Systems Engineering Research, CSER 2013, Atlanta, Georgia, USA, March 19–22, *Procedia Computer Science*, vol. 16, pp. 197–205. Elsevier (2013). https://doi.org/10.1016/j.procs.2013.01.021
[S91] Breckenridge, J., Johnson, S.: Implementation of a goal-based systems engineering process using the Systems Modeling Language (SysML). In: Proceedings of the AIAA Infotech at Aerospace Conference, Boston, MA, USA, August 19–22 (2013)
[S92] Briand, L., Falessi, D., Nejati, S., Sabetzadeh, M., Yue, T.: Traceability and SysML Design Slices to Support Safety Inspections: A Controlled Experiment. ACM Transactions on Software Engineering and Methodology (TOSEM) **23**(1), 9:1–9:43 (2014). https://doi.org/10.1145/2559978
[S93] Broadwell, D., Paredis, C.: Using SysML to Elicit a Value Model in Multi-Stakeholder Value-Driven System Design. In: Proceedings of the ASME 2014 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Buffalo, New York, USA, August 17–20, vol. 1B (2014). https://doi.org/10.1115/DETC201434370
[S94] Browne, D., Kempf, R., Hansen, A., O'Neal, M., Yates, W.: Enabling Systems Modeling Language Authoring in a Collaborative Web-based Decision Support Tool. In: Proceedings of the Conference on Systems Engineering Research, CSER 2013, Atlanta, Georgia, USA, March 19–22, *Procedia Computer Science*, vol. 16, pp. 373–382. Elsevier (2013). https://doi.org/10.1016/j.procs.2013.01.039

[S95] Bryans, J., Fitzgerald, J., Payne, R., Miyazawa, A., Kristensen, K.: SysML contracts for systems of systems. In: Proceedings of the 9th International Conference on System of Systems Engineering: The Socio-Technical Perspective, SoSE 2014, Glenelg, Australia, June 9–13, pp. 73–78. IEEE (2014). https://doi.org/10.1109/SYSOSE.2014.6892466

[S96] Café, D.C., Hardebolle, C., Jacquet, C., dos Santos, F.V., Boulanger, F.: Discrete-Continuous Semantic Adaptations for Simulating SysML Models in VHDL-AMS. In: Proceedings of the 8th Workshop on Multi-Paradigm Modeling co-located with the 17th International Conference on Model Driven Engineering Languages and Systems, MPM@ MODELS 2014, Valencia, Spain, September 30, *CEUR Workshop Proceedings*, vol. 1237, pp. 11–20. CEUR-WS.org (2014)

[S97] Calà, A., Lüder, A., Vollmar, J., Foehr, M.: Evaluation of migration scenarios towards cyber-physical production systems using SysML. In: Proceedings of the IEEE International Systems Engineering Symposium (ISSE), pp. 1–5 (2017). https://doi.org/10.1109/SysEng.2017.8088287

[S98] Caltais, G., Leitner-Fischer, F., Leue, S., Weiser, J.: SysML to NuSMV Model Transformation via Object-Orientation. In: Proceedings of the 6th International Workshop on Cyber Physical Systems. Design, Modeling, and Evaluation (CyPhy 2016), Pittsburgh, PA, USA, October 6, *Lecture Notes in Computer Science*, vol. 10107, pp. 31–45. Springer (2016). https://doi.org/10.1007/978-3-319-51738-4_3

[S99] Cano, L.: Using SysML to model complex systems for security. In: Proceedings of the 44th Annual International Carnahan Conference on Security Technology, pp. 66–70. IEEE (2010). https://doi.org/10.1109/CCST.2010.5678679

[S100] Cao, Y., Liu, Y., Fan, H., Fan, B.: SysML-based uniform behavior modeling and automated mapping of design and simulation model for complex mechatronics. CAD Computer Aided Design **45**(3), 764–776 (2013). https://doi.org/10.1016/j.cad.2012.05.001

[S101] Cao, Y., Liu, Y., Paredis, C.: Integration of System-Level Design and Analysis Models of Mechatronic System Behavior Based on SysML and Simscape. In: Proceedings of the ASME Design Engineering Technical Conference and Computers and Information in Engineering Conference, vol. 3, pp. 1099–1108 (2010). https://doi.org/10.1115/DETC2010-28213

[S102] Cao, Y., Liu, Y., Paredis, C.: System-level model integration of design and simulation for mechatronic systems based on SysML. Mechatronics **21**(6), 1063–1075 (2011). https://doi.org/10.1016/j.mechatronics.2011.05.003

[S103] Carneiro, E., Maciel, P., Callou, G., Tavares, E., Nogueira, B.: Mapping SysML State Machine Diagram to Time Petri Net for Analysis and Verification of Embedded Real-Time Systems with Energy Constraints. In: Proceedings of the International Conference on Advances in Electronics and Microelectronics (ENICS 2008), pp. 1–6 (2008). https://doi.org/10.1109/ENICS.2008.19

[S104] Carrillo, O., Chouali, S., Mountassir, H.: Formalizing and verifying compatibility and consistency of SysML blocks. ACM SIGSOFT Software Engineering Notes **37**(4), 1–8 (2012). https://doi.org/10.1145/2237796.2237813

[S105] Carrillo, O., Chouali, S., Mountassir, H.: Incremental Modeling of System Architecture Satisfying SysML Functional Requirements. In: Proceedings of the 10th International Symposium on Formal Aspects of Component Software FACS 2013, Nanchang, China, October 27–29, 2013, *Lecture Notes in Computer Science*, vol. 8348, pp. 79–99. Springer (2013). https://doi.org/10.1007/978-3-319-07602-7_7

[S106] Chabibi, B., Anwar, A., Nassar, M.: Towards an alignment of SysML and simulation tools. In: Proceedings of the 12th IEEE/ACS International Conference of Computer Systems and Applications, AICCSA 2015, Marrakech, Morocco, November 17–20, pp. 1–6. IEEE (2015). https://doi.org/10.1109/AICCSA.2015.7507216

[S107] Chabibi, B., Douche, A., Anwar, A., Nassar, M.: Integrating SysML with Simulation Environments (Simulink) by Model Transformation Approach. In: Proceedings of the 25th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), pp. 148–150. IEEE Computer Society (2016). https://doi.org/10.1109/WETICE.2016.39

[S108] Chami, M., Ammar, H., Voos, H., Tuyls, K., Weiss, G.: A Nonparametric Evaluation of SysML-based Mechatronic Conceptual Design. In: Proceedings of the 24th Benelux Conference on Artificial Intelligence (BNAIC) (2012)

[S109] Chami, M., Ammar, H., Voos, H., Tuyls, K., Weiss, G.: Swarm-based evaluation of nonparametric SysML mechatronics system design. In: Proceedings of the IEEE International Conference on Mechatronics (ICM), pp. 436–441. IEEE (2013). https://doi.org/10.1109/ICMECH.2013.6518576

[S110] Chami, M., Seemüller, H., Voos, H.: A SysML-based integration framework for the engineering of mechatronic systems. In: Proceedings of IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA), pp. 245–250. IEEE (2010). https://doi.org/10.1109/MESA.2010.5552066

[S111] Chamis, C.: Application of SysML Standards to Space Mission Operations. In: Proceedings of the SpaceOps Conference, Huntsville, Alabama (2010). https://doi.org/10.2514/6.2010-2024

[S112] Chandler, S., Matthews, P.: Through-life systems engineering design & support with SysML. In: Proceedings of the 2nd International Through-life Engineering Services Conference, *Procedia CIRP*, vol. 11, pp. 425–430 (2013). https://doi.org/10.1016/j.procir.2013.07.002

[S113] Chang, C., Lu, C., Hsueh, N., Chu, W.C., Shih, C., Yang, C., Hsiung, P., Koong, C.: SysML-based requirement modeling environment for multicore embedded system. In: Proceedings of the ACM Symposium on Applied Computing (SAC), Sierre, Switzerland, March 22–26, pp. 2224–2228. ACM (2010). https://doi.org/10.1145/1774088.1774555

[S114] Chang, C.H., Lu, C.W., Chu, W.C., Hsiung, P.A., Chang, D.M.: SysML-Based Requirement Management to Improve Software Development. International Journal of Software Engineering and Knowledge Engineering **26**(3), 491–511 (2016). https://doi.org/10.1142/S0218194016500200

[S115] Chang, C.H., Lu, C.W., Kao, K.F., Chu, W.C., Yang, C.T., Hsueh, N.L., Hsiung, P.A., Koong, C.S.: A SysML-Based Requirement Supporting Tool for Embedded Software. In: Proceedings of the 5th International Conference on Secure Software Integration and Reliability Improvement (SSIRI), Jeju Island, Korea, June 27–29, pp. 202–206. IEEE Computer Society (2011). https://doi.org/10.1109/SSIRI-C.2011.34

[S116] Chang, C.H., Lu, C.W., Yang, W., Chu, W.C., Yang, C.T., Tsai, C.T., Hsiung, P.A.: A SysML Based Requirement Modeling Automatic Transformation Approach. In: Proceedings of the IEEE 38th Annual Computer Software and Applications Conference, COMPSAC Workshops 2014, Vasteras, Sweden, July 21–25, pp. 474–479. IEEE Computer Society (2014). https://doi.org/10.1109/COMPSACW.2014.80

[S117] Chen, R., Liu, Y., Cao, Y., Xu, J.: A SysML-based modeling language for mechatronic system architecture. In: Proceedings of the ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE), Boston, Massachusetts, USA, August 2–5, vol. 1B (2015). https://doi.org/10.1115/DETC2015-46738

[S118] Chhaniyara, S., Saaj, C., Maediger, B., Althoff-Kotzias, M., Langpap, B., Ahrns, I.: SysML based system engineering: A case study for Space Robotics Systems. In: Proceedings of the 62nd International Astronautical Congress (IAC 2011), Cape Town, South Africa, vol. 9, pp. 7271–7278 (2011)

[S119] Chiquitto, A., Gimenes, I., Oliveira, E.: SyMPLES-CVL: A SysML and CVL Based Approach for Product-Line Development of Embedded Systems. In: Proceedings of the 9th Brazilian Symposium on Components, Architectures and Reuse Software (SBCARS), Sept., pp. 21–30 (2015). https://doi.org/10.1109/SBCARS.2015.13

[S120] Chiron, F., Kouiss, K.: Design of IEC 61131-3 function blocks using SysML. In: Proceedings of the Mediterranean Conference on Control and Automation (MED 2007), pp. 1–5. IEEE (2007). https://doi.org/10.1109/MED.2007.4433695

[S121] Chouali, S., Carrillo, O., Mountassir, H.: Specifying System Architecture from SysML Requirements and Component Interfaces. In: Proceedings of the 7th European Conference on Software Architecture (ECSA 2013), Montpellier, France, July 1–5, *Lecture Notes in Computer Science*, vol. 7957, pp. 348–352. Springer (2013). https://doi.org/10.1007/978-3-642-39031-9_36

[S122] Chouali, S., Hammad, A.: Formal verification of components assembly based on SysML and interface automata. Innovations in Systems and Software Engineering **7**(4), 265–274 (2011). https://doi.org/10.1007/s11334-011-0170-3

[S123] Chouali, S., Hammad, A., Mountassir, H.: Assembling components using SysML with non-functional requirements. Electronic Notes in Theoretical Computer Science **295**, 31–47 (2013). https://doi.org/10.1016/j.entcs.2013.04.003

[S124] Christophe, F., Sell, R., Coatanéa, E.: Conceptual design framework supported by dimensional analysis and system modelling language. Estonian Journal of Engineering **14**(4), 303–316 (2008). https://doi.org/10.3176/eng.2008.4.02

[S125] Clark, T., Rabelo, L., Yazici, H.: Extending SysML Models to Enable Automatic Generation of Fault Trees. In: Proceedings of the IIE Annual Conference, pp. 1085–1090 (2017)

[S126] Claver, C., Dubois-Felsmann, G., Delgado, F., Hascall, P., Marshall, S., Nordby, M., Schalk, T., Schumacher, G., Sebag, J.: Using SysML for MBSE analysis of the LSST system. In: Proceedings of SPIE - The International Society for Optical Engineering, vol. 7738, pp. 77,381D–77,381D–10 (2010). https://doi.org/10.1117/12.857227

[S127] Cloutier, R., Sauser, B., Bone, M., Taylor, A.: Transitioning systems thinking to model-based systems engineering: Systemigrams to SysML models. IEEE Transactions on Systems, Man, and Cybernetics: Systems **45**(4), 662–674 (2015). https://doi.org/10.1109/TSMC.2014.2379657
[S128] Colombo, P., Del Bianco, V., Lavazza, L.: Towards the integration of SysML and Problem Frames. In: Proceedings of the 3rd International Workshop on Applications and Advances of Problem (IWAAPF 2008), pp. 1–7 (2008). https://doi.org/10.1145/1370811.1370813
[S129] Colombo, P., Del Bianco, V., Lavazza, L., Coen-Porisini, A.: A Methodological Framework for SysML: a Problem Frames-based Approach. In: Proceedings of the 14th Asia-Pacific Software Engineering Conference (APSEC 2007), Nagoya, Japan, December 5–7, pp. 25–32. IEEE Computer Society (2007). https://doi.org/10.1109/APSEC.2007.9
[S130] Colombo, P., Khendek, F., Lavazza, L.: Requirements Analysis and Modeling with Problem Frames and SysML: A Case Study. In: Proceedings of the 6th European Conference on Modelling Foundations and Applications (ECMFA 2010), Paris, France, June 15–18, *Lecture Notes in Computer Science*, vol. 6138, pp. 74–89. Springer (2010). https://doi.org/10.1007/978-3-642-13595-8_8
[S131] Colombo, P., Khendek, F., Lavazza, L.: Generating Early Design Models from Requirements Analysis Artifacts Using Problem Frames and SysML. In: Proceedings of the 7th European Conference on Modelling Foundations and Applications (ECMFA 2011), Birmingham, UK, June 6–9, *Lecture Notes in Computer Science*, vol. 6698, pp. 97–114. Springer (2011). https://doi.org/10.1007/978-3-642-21470-7_8
[S132] Colombo, P., Khendek, F., Lavazza, L.: Bridging the gap between requirements and design: An approach based on Problem Frames and SysML. Journal of Systems and Software **85**(3), 717–745 (2012). https://doi.org/10.1016/j.jss.2011.09.046
[S133] Constantine, J., Solak, S.: SysML modeling of Off-the-Shelf-Option acquisition for risk mitigation in military programs. Systems Engineering **13**(1), 80–94 (2010). https://doi.org/10.1002/sys.20134
[S134] Costa, B., Pires, P.F., Delicato, F.C.: Modeling IoT Applications with SysML4IoT. In: Proceedings of the 42th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2016, Limassol, Cyprus, August 31–Sept. 2, pp. 157–164. IEEE Computer Society (2016). https://doi.org/10.1109/SEAA.2016.19
[S135] Costa, T., Sampaio, A., Alves, G.: Using SysML in Systems Design. In: Proceedings of the International Conference on Information Management, Innovation Management and Industrial Engineering (ICIII 2009), vol. 4, pp. 615–618. IEEE (2009). https://doi.org/10.1109/ICIII.2009.607
[S136] Ćwikła, G., Gwiazda, A., Banaś, W., Monica, Z., Foit, K.: Analysis of the possibility of SysML and BPMN application in formal data acquisition system description. In: IOP Conference Series: Materials Science and Engineering, vol. 227 (2017)
[S137] Da Silva, A., Linhares, M., Padilha, R., Roqueiro, N., De Oliveira, R.: An Empirical Study of SysML in the Modeling of Embedded Systems. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Taipei, Taiwan, October 8–11, pp. 4569–4574. IEEE (2006). https://doi.org/10.1109/ICSMC.2006.384866
[S138] Da Silva Melo, M., França, J., Oliveira E., J., Soares, M.: A Model-driven Approach to Transform SysML Internal Block Diagrams to UML Activity Diagrams. In: Proceedings of the 17th International Conference on Enterprise Information Systems (ICEIS), Barcelona, Spain, April 27–30, vol. 2, pp. 92–101. SciTePress (2015). https://doi.org/10.5220/0005372700920101
[S139] Da Silva Melo, M., Soares, M.: Model-driven Structural Design of Software-intensive Systems Using SysML Blocks and UML Classes. In: Proceedings of the 16th International Conference on Enterprise Information Systems (ICEIS), Lisbon, Portugal, April 27–30, vol. 2, pp. 193–200. SciTePress (2014). https://doi.org/10.5220/0004871301930200
[S140] Dahlweid, M., Brauer, J., Peleska, J.: Model-Based Testing: Automatic Generation of Test Cases, Test Data and Test Procedures from SysML Models. In: Proceedings of the SAE 2015 AeroTech Congress & Exhibition (AEROTECH), no. 2015-01-2553 in SAE Technical Papers, Sept. SAE International (2015). https://doi.org/10.4271/2015-01-2553
[S141] Dahmann, J., Markina-Khusid, A., Doren, A., Wheeler, T., Cotter, M., Kelley, M.: SysML executable systems of system architecture definition: A working example. In: Proceedings of the Annual IEEE International Systems Conference, SysCon 2017, Montreal, QC, Canada, April 24–27, pp. 1–6. IEEE (2017). https://doi.org/10.1109/SYSCON.2017.7934816
[S142] David, P., Idasiak, V., Kratz, F.: Towards a better interaction between design and dependability analysis: Fmea derived from uml/sysml models. In: Proceedings of the ESREL 2008 and 17th SRA-

Europe Conference, Valencia, Spain, September, pp. 2259–2266 (2008)

[S143] David, P., Idasiak, V., Kratz, F.: Improving reliability studies with SysML. In: Proceedings of the Annual Reliability and Maintainability Symposium, pp. 527–532. IEEE (2009). https://doi.org/10.1109/RAMS.2009.4914731

[S144] David, P., Idasiak, V., Kratz, F.: Reliability study of complex physical systems using SysML. Reliability Engineering and System Safety **95**(4), 431–450 (2010). https://doi.org/10.1016/j.ress.2009.11.015

[S145] David, P., Shawky, M.: Supporting ISO 26262 with SysML, benefits and limits. In: Proceedings of the ESREL 2010, pp. 2000–2008 (2010)

[S146] Day, J., Donahue, K., Ingham, M., Kadesch, A., Kennedy, A., Post, E.: Modeling off-nominal behavior in SysML. In: Proceedings of the AIAA Infotech at Aerospace Conference and Exhibit 2012, Garden Grove, CA, USA (2012)

[S147] De Oliveira, K., França, J., Soares, M.: Extensions of SysML for Modeling an Aspect Oriented Software Architecture with Multiple Views. In: Proceedings of the 10th International Conference on Information Technology: New Generations, ITNG 2013, Las Vegas, Nevada, USA, April 15–17, pp. 680–685. IEEE Computer Society (2013). https://doi.org/10.1109/ITNG.2013.105

[S148] De Oliveira, K., Soares, M.: Modeling Aspects in Requirements using SysML Extensions. In: Proceedings of the 15th International Conference on Enterprise Information Systems (ICEIS 2013), Volume 2, Angers, France, July 4–7, pp. 126–133. SciTePress (2013). https://doi.org/10.5220/0004419601260133

[S149] Detommasi, G., Vitelli, R., Boncagni, L., Neto, A.: Modeling of marte-based real-time applications with sysml. IEEE Transactions on Industrial Informatics **9**(4), 2407–2415 (2013). https://doi.org/10.1109/TII.2012.2235073

[S150] Deveci, O., Onkol, M., Unver, H., Ozturk, Z.: Design and development of a low-cost solar powered drip irrigation system using Systems Modeling Language. Journal of Cleaner Production **102**, 529–544 (2015). https://doi.org/10.1016/j.jclepro.2015.04.124

[S151] Ding, S., Tang, S.Q.: An approach for formal representation of SysML block diagram with description logic SHIOQ(D). In: Proceedings of the 2nd International Conference on Industrial and Information Systems (IIS 2010), vol. 2, pp. 259–261 (2010). https://doi.org/10.1109/INDUSIS.2010.5565700

[S152] Dominguez-Bonilla, C., Gutierrez, A., Jimenez, F., Chamorro, H.R.: SysML methodology for FPGA-based Controller design for quadcopters. In: Proceedings of the 7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), Oct., pp. 1–6 (2016). https://doi.org/10.1109/IEMCON.2016.7746362

[S153] Dou, Z., Li, H.: Optimization of the border port logistics and the key-factors recognition based-on HLA/SysML. Journal of Coastal Research pp. 104–107 (2015). https://doi.org/10.2112/SI73-019.1

[S154] Dragomir, I., Ober, I., Lesens, D.: A Case Study in Formal System Engineering with SysML. In: Proceedings of the 17th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS 2012), Paris, France, July 18–20, pp. 189–198. IEEE Computer Society (2012). https://doi.org/10.1109/ICECCS.2012.1

[S155] Dragomir, I., Ober, I., Percebois, C.: Integrating verifiable Assume/Guarantee contracts in UML/SysML. In: Proceedings of the 6th International Workshop on Model Based Architecting and Construction of Embedded Systems co-located with ACM/IEEE 16th International Conference on Model Driven Engineering Languages and Systems (MoDELS 2013), Miami, Florida, USA, September 29th, *CEUR Workshop Proceedings*, vol. 1084. CEUR-WS.org (2013)

[S156] Dragomir, I., Ober, I., Percebois, C.: Safety Contracts for Timed Reactive Components in SysML. In: Proceedings of SOFSEM 2014: Theory and Practice of Computer Science - 40th International Conference on Current Trends in Theory and Practice of Computer Science, Nový Smokovec, Slovakia, January 26–29, *Lecture Notes in Computer Science*, vol. 8327, pp. 211–222. Springer (2014). https://doi.org/10.1007/978-3-319-04298-5_19

[S157] Dragomir, I., Ober, I., Percebois, C.: Contract-based modeling and verification of timed safety requirements within SysML. Software & Systems Modeling pp. 1–38 (2015). https://doi.org/10.1007/s10270-015-0481-1

[S158] Durugbo, C.: Integrated product-service analysis using SysML requirement diagrams. Systems Engineering **16**(1), 111–123 (2013). https://doi.org/10.1002/sys.21229

[S159] Dutenhoffer, C., Tirona, J.: The value of SysML modeling during system operations: A case study. In: Proceedings of the IEEE Aerospace Conference, pp. 1–10. IEEE (2013). https://doi.org/10.1109/AERO.2013.6496850

[S160] é, D.C., dos Santos, F.V., Hardebolle, C., Jacquet, C., Boulanger, F.: Multi-paradigm semantics for simulating SysML models using SystemC-AMS. In: Proceedings of the Forum on specification and Design

Languages, FDL 2013, Paris, France, September 24–26, pp. 1–8. IEEE (2013)
[S161] Eisenbart, B., Mandel, C., Gericke, K., Blessing, L.: Integrated function modelling: Comparing the IFM framework with SYSML. In: Proceedings of the International Conference on Engineering Design, ICED, 27–30 July, Politecnico di Milano, Italy, vol. 5, pp. 145–156. Design Society (2015)
[S162] Espinoza, H., Cancila, D., Selic, B., Gérard, S.: Challenges in Combining SysML and MARTE for Model-Based Design of Embedded Systems. In: Proceedings of the 5th European Conference on Model Driven Architecture - Foundations and Applications (ECMDA-FA 2009), Enschede, The Netherlands, June 23–26, *Lecture Notes in Computer Science*, vol. 5562, pp. 98–113. Springer (2009). https://doi.org/10.1007/978-3-642-02674-4_8
[S163] Evensen, K., Weiss, K.: A comparison and Evaluation of Real-Time Software Systems Modeling Languages. In: Proceedings of the AIAA Infotech at Aerospace 2010, Atlanta, Georgia, April 20–22 (2010)
[S164] Evrot, D., Petin, J.F., Morel, G., Lamy, P.: Using SysML for identification and refinement of machinery safety properties. In: Proceedings of the 1st IFAC Workshop on Dependable Control of Discrete Systems, *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol. 40, pp. 127–132 (2007)
[S165] Falessi, D., Nejati, S., Sabetzadeh, M., Briand, L.C., Messina, A.: SafeSlice: a model slicing and design safety inspection tool for SysML. In: Proceedings of the 19th ACM SIGSOFT Symposium on the Foundations of Software Engineering (FSE-19) and 13th European Software Engineering Conference (ESEC-13), Szeged, Hungary, September 5–9, pp. 460–463. ACM (2011). https://doi.org/10.1145/2025113.2025191
[S166] Fan, H., Liu, Y., Liu, D., Ye, X.: Automated generation of the computer-aided design model from the system structure for mechanical systems based on systems modeling language. Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture **230**(5), 883–908 (2016). https://doi.org/10.1177/0954405414560619
[S167] Fan, H., Liu, Y., Liu, Y.: SysML-based model integration for online collaborative design of mechatronic systems. In: Proceedings of the 19th International Conference on Engineering Design (ICED), Seoul, Korea, August 19–22, vol. 9 DS75-09, pp. 237–246 (2013)
[S168] Faria, J., Mahomad, S., Silva, N.: Practical Results from the Application of Model Checking and Test Generation from UML/SysML Models of On-Board Space Applications. In: Proceedings of the Conference on DAta Systems In Aerospace (DASIA 2009), Noordwijk, Netherlands, *ESA SP*, vol. 669 (2009)
[S169] Feldmann, S., Kernschmidt, K., Vogel-Heuser, B.: Combining a SysML-based Modeling Approach and Semantic Technologies for Analyzing Change Influences in Manufacturing Plant Models. In: Proceedings of the 47th CIRP Conference on Manufacturing Systems - Variety Management in Manufacturing, *Procedia CIRP*, vol. 17, pp. 451–456 (2014). https://doi.org/10.1016/j.procir.2014.01.140
[S170] Fernández, M., Alonso, I., Casanova, E.: Improving the Interoperability in the Digital Home Through the Automatic Generation of Software Adapters from a SysML Model. Journal of Intelligent & Robotic Systems pp. 1–11 (2016). https://doi.org/10.1007/s10846-016-0419-z
[S171] Florian, M., Albert, A., Daniel, W., Matthias, B.: Multi-View Modeling in SysML: Thematic Structuring for Multiple Thematic Views. In: Proceedings of the Conference on Systems Engineering Research, CSER 2014, Redondo Beach, CA, USA, March 20–22, *Procedia Computer Science*, vol. 28, pp. 531–538. Elsevier (2014). https://doi.org/10.1016/j.procs.2014.03.065
[S172] Follmer, M., Hehenberger, P., Punz, S., Zeman, K.: Using SysML in the product development process of mechatronic systems. In: Proceedings of the 11th International Design Conference (DESIGN 2010), pp. 1513–1522 (2010)
[S173] Foures, D., Albert, V., Pascal, J., Nketsa, A.: Automation of SysML activity diagram simulation with model-driven engineering approach. In: Proceedings of the Symposium on Theory of Modeling and Simulation - DEVS Integrative M&S Symposium, Orlando, FL, USA, March 26–29, pp. 11:1–11:6. SCS/ACM (2012)
[S174] Foures, D., Albert, V., Pascal, J.C.: Activitydiagram2petrinet : Transformation-based model in accordance with the OMG SysML specifications. In: Proceedings of the European Simulation and Modelling Conference (ESM 2011), Guimaraes, Portugal, October, pp. 429–433 (2011)
[S175] Fragal, V.H., Silva, R.F., de Souza Gimenes, I.M., de Oliveira Junior, E.A.: Application Engineering for Embedded Systems - Transforming SysML Specification to Simulink within a Product-Line based Approach. In: Proceedings of the 15th International Conference on Enterprise Information Systems (ICEIS 2013), Volume 2, Angers, France, July 4–7, pp. 94–101. SciTePress (2013). https://doi.org/10.5220/0004402600940101

[S176] Friedl, M., Kellner, A., Weingartner, L.: Integration of domain-specific simulation models into descriptive system models by using SysML. In: Proceedings of the IEEE International Systems Engineering Symposium (ISSE), pp. 1–5 (2017). https://doi.org/10.1109/SysEng.2017.8088256

[S177] Friedland, B., Eschbacher, G.: Implementing SysML into a feasible systems engineering solution. In: Proceedings of the IIE Annual Conference and Expo 2013, pp. 4066–4075 (2013)

[S178] Gaeta, J., Czarnecki, K.: Modeling aerospace systems product lines in SysML. In: Proceedings of the 19th International Conference on Software Product Line, SPLC 2015, Nashville, TN, USA, July 20–24, pp. 293–302. ACM (2015). https://doi.org/10.1145/2791060.2791104

[S179] Gans, H.: Development of a Space Vehicle CONOPS Using SysML and the Unified Profile for DoDAF and MoDAF (UPDM). In: Proceedings of the AIAA SPACE and Astronautics Forum and Exposition, Orlando, FL. American Institute of Aeronautics and Astronautics (2017). https://doi.org/10.2514/6.2017-5298

[S180] Gardan, J., Matta, N.: Enhancing Knowledge Management into Systems Engineering through New Models in SysML. In: Proceedings of the 27th CIRP Design Conference - Complex Systems Engineering and Development, Cranfield University, UK 10th – 12th May, *Procedia CIRP*, vol. 60, pp. 169–174 (2017). https://doi.org/10.1016/j.procir.2017.01.052

[S181] Gauthier, J.M.: Test Generation for RTES from SysML Models: Context, Motivations and Research Proposal. In: Proceedings of the 6th IEEE International Conference on Software Testing, Verification and Validation, ICST 2013, Luxembourg, Luxembourg, March 18–22, pp. 503–504. IEEE Computer Society (2013). https://doi.org/10.1109/ICST.2013.83

[S182] Gauthier, J.M., Bouquet, F., Hammad, A., Peureux, F.: Verification and Validation of Meta-model based Transformation from SysML to VHDL-AMS. In: Proceedings of the 1st International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2013), Barcelona, Spain, February 19–21, pp. 123–128. SciTePress (2013). https://doi.org/10.5220/0004317601230128

[S183] Gauthier, J.M., Bouquet, F., Hammad, A., Peureux, F.: A SysML Formal Framework to Combine Discrete and Continuous Simulation for Testing. In: Proceedings of the 17th International Conference on Formal Engineering Methods, ICFEM 2015, Paris, France, November 3–5, *Lecture Notes in Computer Science*, vol. 9407, pp. 134–152. Springer (2015). https://doi.org/10.1007/978-3-319-25423-4_9

[S184] Gauthier, J.M., Bouquet, F., Hammad, A., Peureux, F.: Tooled Process for Early Validation of SysML Models Using Modelica Simulation. In: Proceedings of the 6th International Conference on Fundamentals of Software Engineering,FSEN 2015 Tehran, Iran, April 22–24, *Lecture Notes in Computer Science*, vol. 9392, pp. 230–237. Springer (2015). https://doi.org/10.1007/978-3-319-24644-4_16

[S185] Gaye, K., Coulibaly, A., Gardoni, M.: Product meta-model based on the coupling of the extended design matrix X -DSM and SysML formalism. In: Proceedings of the 5th International Conference on Industrial Engineering and Operations Management (IEOM), pp. 1–7 (2015). https://doi.org/10.1109/IEOM.2015.7093881

[S186] Geyer, P.: Systems modeling for building design: A method based on the systems modeling language. In: 18th International Workshop of the European Group for Intelligent Computing in Engineering, EG-ICE 2011, Twente UniversityEnschede, Netherlands, July 6–8 2011 (2014)

[S187] Geyer, P.: Using the Systems Modelling Language (SysML) for decision modelling for sustainable building design. In: eWork and eBusiness in Architecture, Engineering and Construction - Proceedings of the 10th European Conference on Product and Process Modelling, ECPPM 2014, pp. 361–366 (2015)

[S188] Gezer, D., Unver, H., Tascioglu, Y., Celebioglu, K., Aradag, S.: Design and simulation of a SCADA system using SysML and Simulink. In: Proceedings of International Conference on Renewable Energy Research and Applications (ICRERA 2013), pp. 1058–1062 (2013). https://doi.org/10.1109/ICRERA.2013.6749909

[S189] Giese, H., Hildebrandt, S., Neumann, S.: Towards Integrating SysML and AUTOSAR Modeling via Bidirectional Model Synchronization. In: Proceedings of the Dagstuhl-Workshop MBEES: Modellbasierte Entwicklung eingebetteter Systeme V, Schloss Dagstuhl, Germany, 2009, Tagungsband Modellbasierte Entwicklung eingebetteter Systeme, *Informatik-Bericht*, vol. 2009-01, pp. 155–164. TU Braunschweig, Institut für Software Systems Engineering (2009)

[S190] Giese, H., Hildebrandt, S., Neumann, S.: Model Synchronization at Work: Keeping SysML and AUTOSAR Models Consistent. In: Graph Transformations and Model-Driven Engineering - Essays Dedicated to Manfred Nagl on the Occasion of his 65th Birthday, *Lecture Notes in Computer Science*, vol. 5765, pp. 555–579. Springer (2010). https://doi.org/10.1007/978-3-642-17322-6_24

[S191] Giorgetti, A., Hammad, A., Tatibouët, B.: Using SysML for Smart Surface Modeling. In: Proceedings of the 1st Workshop on Hardware and Software Implementation and Control of Distributed MEMS (dMEMS 2010), pp. 100–107. IEEE (2010). https://doi.org/10.1109/dMEMS.2010.22

[S192] Globe, J.: Using SysML to create a simulation conceptual model of a basic ISR survivability test thread. In: Proceedings of the Spring Simulation Interoperability Workshop (Spring SIW), pp. 356–363 (2007)

[S193] Gnaho, C., Semmak, F.: Une extension sysml pour l'ingénierie des exigences non fonctionnelles orientée but. Ingénierie des Systèmes d'Information **16**(1), 9–32 (2011). https://doi.org/10.3166/isi.16.1.9-32

[S194] Gnaho, C., Semmak, F., Laleau, R.: An overview of a SysML extension for goal-oriented NFR modelling: Poster paper. In: Proceedings of the IEEE 7th International Conference on Research Challenges in Information Science, RCIS 2013, Paris, France, May 29–31, pp. 1–2. IEEE (2013). https://doi.org/10.1109/RCIS.2013.6577734

[S195] Godart, P., Gross, J., Mukherjee, R., Ubellacker, W.: Generating real-time robotics control software from SysML. In: Proceedings of the IEEE Aerospace Conference, pp. 1–11 (2017). https://doi.org/10.1109/AERO.2017.7943610

[S196] Gomez, C., DeAntoni, J., Mallet, F.: Multi-view Power Modeling Based on UML, MARTE and SysML. In: Proceedings of the 38th Euromicro Conference on Software Engineering and Advanced Applications (SEAA 2012), Cesme, Izmir, Turkey, September 5–8, pp. 17–20. IEEE Computer Society (2012). https://doi.org/10.1109/SEAA.2012.66

[S197] González Alonso, I., García Fuente, M., Brugos, J.: Using SysML to Describe a New Methodology for Semiautomatic Software Generation from Inferred Behavioral and Data Models. In: Proeceedings of the 4th International Conference on Systems (ICONS 2009), Gosier, Guadeloupe, France, March 1–6, pp. 210–215. IEEE Computer Society (2009). https://doi.org/10.1109/ICONS.2009.50

[S198] Gotoh, T., Eguchi, T., Koga, T., Aoyama, K.: Modeling for product requirements based on logical structure of product (Model-driven development method for mechanical/electrical/soft integrated products using SysML). Nihon Kikai Gakkai Ronbunshu, C Hen/Transactions of the Japan Society of Mechanical Engineers, Part C **76**(771), 2754–2763 (2010)

[S199] Graves, H.: Integrating SysML and OWL. In: Proceedings of the 6th International Conference on OWL: Experiences and Directions, Chantilly, VA, *OWLED'09*, vol. 529, pp. 117–124. CEUR-WS.org (2009)

[S200] Graves, H.: Structural Models in Axiomatic SysML. In: Proceedings of the 24th International Workshop on Description Logics (DL 2011), Barcelona, Spain, July 13–16, *CEUR Workshop Proceedings*, vol. 745. CEUR-WS.org (2011)

[S201] Graves, H.: Integrating Reasoning with SysML. In: Proceedings of the 22nd Annual International Symposium of the International Council on Systems Engineering (INCOSE 2012) and the 8th Biennial European Systems Engineering Conference (EuSEC 2012), *INCOSE International Symposium*, vol. 22, pp. 2228–2242 (2012)

[S202] Graves, H., Bijan, Y.: Using formal methods with SysML in aerospace design and engineering. Annals of Mathematics and Artificial Intelligence **63**(1), 53–102 (2011). https://doi.org/10.1007/s10472-011-9267-5

[S203] Grecki, M., Geng, Z., Ayvazyan, G., Simrock, S., Aminov, B.: Application of SysML to design of ATCA based LLRF control system. In: 2008 IEEE Nuclear Science Symposium Conference Record, pp. 44–52 (2008). https://doi.org/10.1109/NSSMIC.2008.4775248

[S204] Griego, R., Mayes, L., McGrath, D.: Enterprise domain modelling process using sysml for the tooling enterprise at the u.s. nnsa's pantex plant. In: Proceedings of the 17th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2007), *INCOSE International Symposium*, vol. 17, pp. 1358–1372 (2007)

[S205] Grobshtein, Y., Dori, D.: Evaluating aspects of systems modeling languages by example: SysML and OPM. In: Proceedings of the 18th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2008), *INCOSE International Symposium*, vol. 18, pp. 1023–1037 (2008)

[S206] Grobshtein, Y., Dori, D.: Generating SysML views from an OPM model: Design and evaluation. Systems Engineering **14**(3), 327–340 (2011). https://doi.org/10.1002/sys.20181

[S207] Grobshtein, Y., Perelman, V., Safra, E., Dori, D.: Systems Modeling Languages: OPM Versus SysML. In: Proceedings of the 1st International ICST Conference on Systems Engineering and Modeling (ICSEM 2007), Herzliyya-Haifa, Israel, March 20–23, pp. 102–109. IEEE (2007). https://doi.org/10.1109/ICSEM.2007.373339

[S208] Gross, J., Mukherjee, R.: Integrating Multibody Simulations With SysML. In: Proceedings of the

11th International Conference on Multibody Systems, Nonlinear Dynamics, and Control, Boston, Massachusetts, USA, August 2–5, *ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 6 (2015). https://doi.org/10.1115/DETC2015-48095

[S209] Gruber, K., Huemer, J., Zimmermann, A., Maschotta, R.: Integrated Description of Functional and Non-Functional Requirements for Automotive Systems Design Using SysML. In: Proceedings of the 7th International Conference on System Engineering and Technology (ICSET 2017), Shah Alam, Malaysia, October 2–3, pp. 1–5. IEEE (2017)

[S210] Güdemann, M., Kegel, S., Ortmeier, F., Poenicke, O., Richter, K.: SysML in digital engineering. In: Proceedings of the 1st International Workshop on Digital Engineering (IWDE 2010), Magdeburg, Germany, June 14, pp. 1–8. ACM (2010). https://doi.org/10.1145/1837154.1837155

[S211] Guillerm, R., Demmou, H., Sadou, N.: Sysml Knowledge base for Designing Dependable Complex System. CoRR **abs/1212.4246** (2012)

[S212] Gulias, E., Torreblanca, L., Aguilar, J., Fernandez, C.: Using SysML Modeling to Accurately Represent Automotive Safety Requirements. In: Proceedings of the 4th International Conference in Software Engineering Research and Innovation (CONISOFT), pp. 21–26 (2016). https://doi.org/10.1109/CONISOFT.2016.12

[S213] Gutierrez, A., Chamorro, H., Jimenez, J.: Hardware-in-the-Loop based SysML for model and control design of interleaved boost converters. In: Proceedings of the IEEE 15th Workshop on Control and Modeling for Power Electronics (COMPEL). IEEE (2014). https://doi.org/10.1109/COMPEL.2014.6877155

[S214] Gutierrez, A., Chamorro, H., Jimenez, J., Villa, L., Alonso, C.: Hardware-in-the-loop simulation of PV systems in micro-grids using SysML models. In: Proceedings of the IEEE 16th Workshop on Control and Modeling for Power Electronics (COMPEL) (2015). https://doi.org/10.1109/COMPEL.2015.7236466

[S215] Gutierrez, A., Chamorro, H.R., Villa, L.F.L., Jimenez, J.F., Alonso, C.: SysML methodology for HIL implementation of PV models. In: Proceedings of the 17th European Conference on Power Electronics and Applications (EPE'15 ECCE-Europe), Sept., pp. 1–7 (2015). https://doi.org/10.1109/EPE.2015.7309196

[S216] Haan, B.: Examination of the interplay of reliability and security using system modeling language. In: Proceedings of the Annual Reliability and Maintainability Symposium, pp. 475–480. IEEE (2008). https://doi.org/10.1109/RAMS.2008.4925842

[S217] Haidrar, S., Anwar, A., Roudiès, O.: Towards a generic framework for requirements traceability management for SysML language. In: Proceedings of the 4th IEEE International Colloquium on Information Science and Technology (CiSt 2016), Tangier, Morocco, October 24–26, pp. 210–215. IEEE (2016). https://doi.org/10.1109/CIST.2016.7805044

[S218] Haidrar, S., Anwar, A., Roudiès, O.: A SysML-based Approach to Manage Stakeholder Requirements Traceability. In: 14th IEEE/ACS International Conference on Computer Systems and Applications, AICCSA 2017, Hammamet, Tunisia, October 30–Nov. 3, 2017, pp. 202–207. IEEE (2017). https://doi.org/10.1109/AICCSA.2017.183

[S219] Haley, T., Cerenzia, J., Diederich, D., Friedenthal, S.: Using SysML and UML to develop and implement interoperable system components for engagement simulations. In: Proceedings of the 18th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2008), *INCOSE International Symposium*, vol. 18, pp. 1011–1022 (2008)

[S220] Haley, T., Friedenthal, S.: Assessing the application of SysML to systems of systems simulations. In: Proceedings of the Simulation Interoperability Workshop Spring 2008, pp. 651–662 (2008)

[S221] Hamilton, M., Hackler, W.: A formal universal systems semantics for SysML. In: Proceedings of the 17th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2007), *INCOSE International Symposium*, vol. 17, pp. 1333–1357 (2007)

[S222] Hammad, A., Mountassir, H., Chouali, S.: Combining SysML and Modelica to Verify the Wireless Sensor Networks Energy Consumption. In: Proceedings of the 1st International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2013), Barcelona, Spain, February 19–21, pp. 198–201. SciTePress (2013). https://doi.org/10.5220/0004319601980201

[S223] Hampson, K.: Technical evaluation of the Systems Modeling Language (SysML). In: Proceedings of the Conference on Systems Engineering Research, *Procedia Computer Science*, vol. 44, pp. 403–412 (2015). https://doi.org/10.1016/j.procs.2015.03.054

[S224] Hanai, R., Saito, H., Nakabo, Y., Fujiwara, K., Ogure, T., Mizuguchi, D., Homma, K., Ohba, K.: RT-component based integration for IEC61508 ready system using SysML and IEC61499 function blocks. In: Proceedings of the IEEE/SICE International Symposium on System Integration (SII 2012), pp.

105–110 (2012). https://doi.org/10.1109/SII.2012.6426952
[S225] Handley, H., Amissah, M., Kandemir, C.: Levels of SysML Compatibility for Collaborative Human System Development. In: Proceedings of the Human Factors and Ergonomics Society Annual Meeting, vol. 60, pp. 1711–1715 (2016)
[S226] Hause, M.: Cross-Cutting Concerns and Ergonomic Profiling Using UML/SysML. In: Proceedings of the 16th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2006), *INCOSE International Symposium*, vol. 16, pp. 179–192 (2006)
[S227] Hause, M.: The OMG Systems Modeling Language (SysML). In: Proceedings of the Conference on DAta Systems In Aerospace (DASIA 2007), no. 638 in ESA SP (2007)
[S228] Hause, M.: Designing mission-critical systems using OMG SysML. Electronics World **114**(1865), 20–22 (2008)
[S229] Hause, M., Hummell, J.: Simulation of an electrical network and control system in SysML. In: Proceedings of the 2012 Symposium on Theory of Modeling and Simulation - DEVS Integrative M&S Symposium, Orlando, FL, USA, March 26–29, pp. 47:1–47:7. SCS/ACM (2012)
[S230] Hause, M., Thom, F.: Modeling High Level Requirements in UML/SysML. In: Proceedings of the 15th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2005), *INCOSE International Symposium*, vol. 15, pp. 316–327 (2005)
[S231] Hause, M., Thom, F.: An integrated safety strategy to model driven development with SysML. In: Proceedings of the 2nd Institution of Engineering and Technology International Conference on System Safety, pp. 124–129. IET (2007). https://doi.org/10.1049/cp:20070452
[S232] Hause, M., Thom, F.: Bridging the chasm - tracing from architectural frameworks to SysML. In: Proceedings of the 17th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2007), *INCOSE International Symposium*, vol. 17, pp. 1317–1332 (2007)
[S233] Hause, M., Thom, F.: HCI aspects of SysML and architectural frameworks. In: Proceedings of the 17th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2007), *INCOSE International Symposium*, vol. 17, pp. 351–366 (2007)
[S234] Hause, M., Thom, F.: Building bridges between systems and software with SysML and UML. In: Proceedings of the 18th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2008), *INCOSE International Symposium*, vol. 18, pp. 797–811 (2008)
[S235] Hause, M., Thom, F., Moore, A.: Inside SysML. IEE Electronics Systems and Software **3**(3), 20–25 (2005). https://doi.org/10.1049/ess:20050302
[S236] Hause, M.C., Thom, F.: An Integrated MDA Approach with SysML and UML. In: Proceedings of the 13th International Conference on Engineering of Complex Computer Systems (ICECCS 2008), Belfast, Northern Ireland, March 31–April 3, pp. 249–254. IEEE Computer Society (2008). https://doi.org/10.1109/ICECCS.2008.21
[S237] Hayden, J., Jeffries, A.: On using SysML, DoDAF 2.0 and UPDM to model the architecture for the NOAA's Joint Polar Satellite System (JPSS) Ground System (GS). In: Proceedings of the 12th International Conference on Space Operations (SpaceOps), Stockholm, Sweden, June 11–15 (2012). https://doi.org/10.2514/6.2012-1289592
[S238] He, Z.H., Wang, M.Z.: Dynamic performance and effectiveness evaluation on SysML design. Xi Tong Gong Cheng Yu Dian Zi Ji Shu/Systems Engineering and Electronics **28**(11), 1712–1716 (2006)
[S239] Hecht, M., Dimpfl, E., Pinchak, J.: Automated Generation of Failure Modes and Effects Analysis from SysML Models. In: Proceedings of the 25th IEEE International Symposium on Software Reliability Engineering Workshops, ISSRE Workshops, Naples, Italy, November 3–6, pp. 62–65. IEEE Computer Society (2014). https://doi.org/10.1109/ISSREW.2014.117
[S240] Hecht, M., Nguyen, E., Chuidian, A., Pinchak, J., Dimpfl, E.: Creation of Failure Modes and Effects Analyses from SysML. In: Proceedings of the SAE 2015 AeroTech Congress & Exhibition (AEROTECH), no. 2015-01-2444 in SAE Technical Papers, Sept. SAE International (2015). https://doi.org/10.4271/2015-01-2444
[S241] Hecht, M., Tamaki, J., Lo, D.: Modeling of Failure detection and recovery in SysML. In: Proceedings of the IEEE 24th International Symposium on Software Reliability Engineering, ISSRE 2013, Pasadena, CA, USA, November 4–7, pp. 85–95. IEEE Computer Society (2013). https://doi.org/10.1109/ISSREW.2013.6688879
[S242] Helle, P.: Automatic SysML-based safety analysis. In: Proceedings of the 5th International Workshop on Model Based Architecting and Construction of Embedded Systems, ACES-MB@MoDELS 2012, Innsbruck, Austria, September 30, pp. 19–24. ACM (2012). https://doi.org/10.1145/2432631.2432635

[S243] Hernandez, C., M., R., Diaz, I., Sanz, R.: Model Based Engineering of Process Plants using SysML. In: Proceedings of the 26th European Symposium on Computer Aided Process Engineering, *Computer Aided Chemical Engineering*, vol. 38, pp. 1281–1286. Elsevier (2016). https://doi.org/10.1016/B978-0-444-63428-3.50218-6

[S244] Herzig, S., Karban, R., Trancho, G., Dekens, F., Jankevificius, N., Troy, M.: Analyzing the operational behavior of the alignment and phasing system of the thirty meter telescope using SysML. In: 5th Adaptive Optics for Extremely Large Telescopes, 2017 AO4ELT5, vol. 2017-June, pp. 1–12 (2017). https://doi.org/10.26698/AO4ELT5.0023

[S245] Herzog, E., Andersson, H., Hallonquist, J.: Experience from introducing SysML into a large project organisation. In: Proceedings of the 20th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2010), *INCOSE International Symposium*, vol. 20, pp. 595–604 (2010)

[S246] Herzog, E., Hallonquist, J., Naeser, J.: Systems Modeling with SysML - an experience report. In: Proceedings of the 22nd Annual International Symposium of the International Council on Systems Engineering (INCOSE 2012) and the 8th Biennial European Systems Engineering Conference (EuSEC 2012), *INCOSE International Symposium*, vol. 22, pp. 600–611 (2012)

[S247] Herzog, E., Pandikow, A.: SysML - an assessment. In: Proceedings of the 15th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2005), *INCOSE International Symposium*, vol. 15, pp. 293–305 (2005)

[S248] Hetherinton, D.: SysML requirements for training game design. In: Proceedings of the 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), Qingdao, China, October 8–11, pp. 162–167. IEEE (2014). https://doi.org/10.1109/ITSC.2014.6957684

[S249] Hilken, C., Peleska, J.: Model-Based Testing Against Complex SysML Models. In: Proceedings of the Formal Modeling and Verification of Cyber-Physical Systems, 1st International Summer School on Methods and Tools for the Design of Digital Systems, Bremen, Germany, Sept., pp. 284–286. Springer (2015). https://doi.org/10.1007/978-3-658-09994-7_14

[S250] Hilken, C., Peleska, J., Wille, R.: A Unified Formulation of Behavioral Semantics for SysML Models. In: Proceedings of the 3rd International Conference on Model-Driven Engineering and Software Development, MODELSWARD 2015, ESEO, Angers, Loire Valley, France, 9–11 February, pp. 263–271. SciTePress (2015). https://doi.org/10.5220/0005241602630271

[S251] Hinckel, E., Borsato, M., Schmidt, J., MacCari, F., Storrer, P., Onofre, E.: Driving Product Design and Requirements Management with SysML. In: Proceedings of the 23rd ISPE Inc. International Conference on Transdisciplinary Engineering, At Curitiba, Brazil, *Advances in Transdisciplinary Engineering*, vol. 4, pp. 1071–1080 (2016). https://doi.org/10.3233/978-1-61499-703-0-1071

[S252] Hiron, E., Miramont, P.: Process Based on SysML for New Launchers System and Software Developments. In: Proceedings of the Conference on DAta Systems In Aerospace (DASIA 2010), Budapest, Hungary, June 1–4, *ESA SP*, vol. 682 (2010)

[S253] Hochwallner, M., Hörl, M., Dierneder, S., Scheidl, R.: Some Aspects of SysML Application in the Reverse Engineering of Mechatronic Systems. In: Proceedings of the 13th International Conference on Computer Aided Systems Theory (EUROCAST 2011), Las Palmas de Gran Canaria, Spain, February 6–11, Revised Selected Papers, Part II, *Lecture Notes in Computer Science*, vol. 6928, pp. 81–88. Springer (2012). https://doi.org/10.1007/978-3-642-27579-1_11

[S254] Hoffmann, H.P.: SysML-based Systems Engineering Using a Model-Driven Development Approach. In: Proceedings of the 16th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2006), *INCOSE International Symposium*, vol. 16, pp. 804–814 (2006)

[S255] Hörl, M., Hochwallner, M., Dierneder, S., Scheidl, R.: Integration of SysML and Simulation Models for Mechatronic Systems. In: Proceedings of the 13th International Conference on Computer Aided Systems Theory (EUROCAST 2011), Las Palmas de Gran Canaria, Spain, February 6–11, Revised Selected Papers, Part II, *Lecture Notes in Computer Science*, vol. 6928, pp. 89–96. Springer (2012). https://doi.org/10.1007/978-3-642-27579-1_12

[S256] Hossein, M., Hemmat, A., Mohamed, O., Boukadoum, M.: Towards code generation for ARM Cortex-M MCUs from SysML activity diagrams. In: Proceedings of the IEEE International Symposium on Circuits and Systems, ISCAS 2016, Montréal, QC, Canada, May 22–25, pp. 970–973. IEEE (2016). https://doi.org/10.1109/ISCAS.2016.7527404

[S257] Hsu, J.: Applying systems modeling language to a simple hardware system. In: Proceedings of the 16th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2006), *INCOSE International Symposium*, vol. 16, pp. 595–605 (2006)

[S258] Huang, C., Huang, Z., Hu, J., Wu, Z., Wang, S.: A MDE-Based Approach to the Safety Verification of Extended SysML Activity Diagram. JSW **10**(1), 56–70 (2015)
[S259] Huang, E., Ramamurthy, R., McGinnis, L.F.: System and simulation modeling using SysML. In: Proceedings of the Winter Simulation Conference (WSC 2007), Washington, DC, USA, December 9–12, pp. 796–803. WSC (2007). https://doi.org/10.1109/WSC.2007.4419675
[S260] Huang, X., Sun, Q., Li, J., Zhang, T.: MDE-Based Verification of SysML State Machine Diagram by UPPAAL. In: Proceedings of the International Conference on Trustworthy Computing and Services (ISCTCS 2012), *Communications in Computer and Information Science*, vol. 320, pp. 490–497 (2013). https://doi.org/10.1007/978-3-642-35795-4_62
[S261] Huckaby, J., Christensen, H.: Modeling robot assembly tasks in manufacturing using SysML. In: Proceedings of the ISR/ROBOTIK 2014, Munich, Germany, June 2–3, pp. 743–749 (2014)
[S262] Huckaby, J., Christensen, H.I.: A case for SysML in robotics. In: Proceedings of the 2014 IEEE International Conference on Automation Science and Engineering (CASE), New Taipei, Taiwan, August 18–22, pp. 333–338. IEEE (2014). https://doi.org/10.1109/CoASE.2014.6899347
[S263] Ingram, C., Andrews, Z., Payne, R.J., Plat, N.: SysML fault modelling in a traffic management system of systems. In: Proceedings of the 9th International Conference on System of Systems Engineering, SoSE 2014, Glenelg, Australia, June 9–13, pp. 124–129. IEEE (2014)
[S264] Iqbal, M., Khan, M.U., Sher, M.: System Analysis and Modeling Using SysML. In: Proceedings of the International Conference on IT Convergence and Security (ICITCS 2012), Pyeong Chang, Korea, December 5–7, *Lecture Notes in Electrical Engineering*, vol. 215, pp. 1211–1220. Springer (2012). https://doi.org/10.1007/978-94-007-5860-5_145
[S265] Izukura, S., Yanoo, K., Osaki, T., Sakaki, H., Kimura, D., Xiang, J.: Applying a Model-Based Approach to IT Systems Development Using SysML Extension. In: Proceedings of the 14th International Conference on Model Driven Engineering Languages and Systems (MODELS 2011), Wellington, New Zealand, October 16–21, *Lecture Notes in Computer Science*, vol. 6981, pp. 563–577. Springer (2011). https://doi.org/10.1007/978-3-642-24485-8_41
[S266] Jackson, M., Fernández, M., McVittie, T., Sindiy, O.: Architecting the Human Space Flight program with Systems Modeling Language (SysML). In: Proceedings of the AIAA Infotech at Aerospace Conference and Exhibit 2012 (2012)
[S267] Jacobs, J., Simpson, A.: Towards a Process Algebra Framework for Supporting Behavioural Consistency and Requirements Traceability in SysML. In: Proceedings of the 15th International Conference on Formal Engineering Methods, ICFEM 2013, Queenstown, New Zealand, October 29–November 1, *Lecture Notes in Computer Science*, vol. 8144, pp. 265–280. Springer (2013). https://doi.org/10.1007/978-3-642-41202-8_18
[S268] Jacobs, J., Simpson, A.: A formal model of sysml blocks using CSP for assured systems engineering. In: Proceedings of the 3rd International Workshop on Formal Techniques for Safety-Critical Systems, FTSCS 2014, Luxembourg, November 6–7, *Communications in Computer and Information Science*, vol. 476, pp. 127–141. Springer (2015). https://doi.org/10.1007/978-3-319-17581-2_9
[S269] Jacobs, J., Simpson, A.: On the formal interpretation and behavioural consistency checking of SysML blocks. Software and System Modeling **16**(4), 1145–1178 (2017). https://doi.org/10.1007/s10270-015-0511-z
[S270] Jacobs, J., Simpson, A.C.: On the Formal Interpretation of SysML Blocks Using a Safety Critical Case Study. In: Proceedings of the 8th Brazilian Symposium on Software Components, Architectures and Reuse, SBCARS 2014, Maceió, Alagoas, Brazil, September 29–30, pp. 95–104. IEEE Computer Society (2014). https://doi.org/10.1109/SBCARS.2014.14
[S271] Jain, V., Kumar, A., Panda, P.R.: A SysML Profile for Development and Early Validation of TLM 2.0 Models. In: Proceedings of the 7th European Conference on Modelling Foundations and Applications (ECMFA 2011), Birmingham, UK, June 6–9, *Lecture Notes in Computer Science*, vol. 6698, pp. 299–311. Springer (2011). https://doi.org/10.1007/978-3-642-21470-7_21
[S272] Jakjoud, A., Zrikem, M., Baron, C., Ayadi, A.: SysPEM: A SysML and SPEM based process modelling language for systems engineering. International Journal of Services Operations and Informatics **7**(4), 330–348 (2012). https://doi.org/10.1504/IJSOI.2012.052183
[S273] Jakob, F., Mazzini, S., Jung, A.: A sysml-based methodology in a concurrent satellite design process. In: Proceedings of the Aerospace Technology Conference and Exposition, SAE Technical Papers (2011). https://doi.org/10.4271/2011-01-2713
[S274] Jamro, M.: Automatic generation of implementation in SysML-based model-driven development for

IEC 61131-3 control software. In: Proceedings of the 19th International Conference On Methods and Models in Automation and Robotics, MMAR 2014, Międzyzdroje, Poland, September 2–5, pp. 468–473. IEEE (2014). https://doi.org/10.1109/MMAR.2014.6957399

[S275] Jamro, M.: SysML modeling of POU-oriented unit tests for IEC 61131-3 control software. In: Proceedings of the 19th International Conference On Methods and Models in Automation and Robotics, MMAR 2014, Międzyzdroje, Poland, September 2–5, pp. 82–87. IEEE (2014). https://doi.org/10.1109/MMAR.2014.6957329

[S276] Jamro, M.: SysML Modeling of Functional and Non-functional Requirements for IEC 61131-3 Control Systems. In: Progress in Automation, Robotics and Measuring Techniques - Control and Automation, *Advances in Intelligent Systems and Computing*, vol. 350, pp. 91–100. Springer (2015). https://doi.org/10.1007/978-3-319-15796-2_10

[S277] Jamro, M., Rzonca, D.: SysML-Based Modeling of Token Passing Paradigm in Distributed Control Systems. In: P. Gaj, A. Kwiecien, P. Stera (eds.) Proceedings of the 22nd International Conference on Computer Networks, CN 2015, Brunów, Poland, June 16–19, *Communications in Computer and Information Science*, vol. 522, pp. 139–149. Springer (2015). https://doi.org/10.1007/978-3-319-19419-6_13

[S278] Jamro, M., Rzonca, D., Rząsa, W.: Testing communication tasks in distributed control systems with SysML and Timed Colored Petri Nets model. Computers in Industry **71**, 77–87 (2015). https://doi.org/10.1016/j.compind.2015.03.007

[S279] Jamro, M., Trybus, B.: An approach to SysML modeling of IEC 61131-3 control software. In: Proceedings of 18th International Conference on Methods & Models in Automation & Robotics, MMAR 2013, Międzyzdroje, Poland, August 26–29, pp. 217–222. IEEE (2013)

[S280] Jankevicius, N.: Resource Analysis and Automated Verification for the Thirty Meter Telescope using Executable SysML Models. In: Proceedings of the 2nd International Workshop on Executable Modeling co-located with ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems (MODELS 2016), Saint-Malo, France, October 3, *CEUR Workshop Proceedings*, vol. 1760, pp. 2–4. CEUR-WS.org (2016)

[S281] Jarraya, Y., Debbabi, M.: Formal Specification and Probabilistic Verification of SysML Activity Diagrams. In: Proceedings of the 6th International Symposium on Theoretical Aspects of Software Engineering (TASE 2012), Beijing, China, July 4–6, pp. 17–24. IEEE Computer Society (2012). https://doi.org/10.1109/TASE.2012.34

[S282] Jarraya, Y., Debbabi, M.: Quantitative and qualitative analysis of SysML activity diagrams. International Journal on Software Tools for Technology Transfer **16**(4), 399–419 (2014). https://doi.org/10.1007/s10009-014-0305-6

[S283] Jarraya, Y., Debbabi, M., Bentahar, J.: On the Meaning of SysML Activity Diagrams. In: Proceedings of the 16th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ECBS 2009), San Francisco, California, USA, April 14–16, pp. 95–105. IEEE Computer Society (2009). https://doi.org/10.1109/ECBS.2009.25

[S284] Jarraya, Y., Soeanu, A., Debbabi, M., Hassaïne, F.: Automatic Verification and Performance Analysis of Time-Constrained SysML Activity Diagrams. In: Proceedings of the 14th Annual IEEE International Conference and Workshop on Engineering of Computer Based Systems (ECBS 2007), Tucson, Arizona, USA, March 26–29, pp. 515–522. IEEE Computer Society (2007). https://doi.org/10.1109/ECBS.2007.22

[S285] de Jesus, T.O., Soares, M.S.: An Event-Based Technique to Trace Requirements Modeled with SysML. In: O. Gervasi, B. Murgante, S. Misra, G. Borruso, C.M. Torre, A.M.A.C. Rocha, D. Taniar, B.O. Apduhan, E.N. Stankova, A. Cuzzocrea (eds.) Proceedings of the 17th International Conference on Computational Science and Its Applications (ICCSA), Trieste, Italy, July 3–6, *Lecture Notes in Computer Science*, vol. 10409, pp. 145–159. Springer (2017). https://doi.org/10.1007/978-3-319-62407-5_10

[S286] Jiang, C.Y., Wang, W.P., Li, Q.: SysML: a new systems modeling language. Xitong Fangzhen Xuebao / Journal of System Simulation **18**(6), 1483–1487+1492 (2006)

[S287] Jobe, J., Johnson, T., Paredis, C.: Multi-Aspect Component Models: A framework for model reuse in SysML. In: Proceedings of the ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference (DETC 2008), Brooklyn, New York, USA, August 3–6, vol. 3, pp. 943–955 (2008). https://doi.org/10.1115/DETC2008-49339

[S288] Johnson, T., Jobe, J., Paredis, C., Burkhart, R.: Modeling continuous system dynamics in SysML. In: Proceedings of the ASME International Mechanical Engineering Congress and Exposition (IMECE 2007), vol. 3, pp. 197–205 (2008). https://doi.org/10.1115/IMECE2007-42754

[S289] Johnson, T., Kerzhner, A., Paredis, C., Burkhart, R.: Integrating models and simulations of continuous dynamics into SysML. Journal of Computing and Information Science in Engineering **12**(1) (2012). https://doi.org/10.1115/1.4005452

[S290] Kakiuchi, Y.: Constraint Analysis of System Requirement in SysML with Formal Methods. IEEJ Transactions on Electronics, Information and Systems **137**(6), 809–814 (2017). https://doi.org/10.1541/ieejeiss.137.809

[S291] Kanthabhabhajeya, S., Berglund, J., Falkman, P., Lennartson, B.: Interface between SysML and Sequence Planner Language for Formal Verification. In: Proceedings of the 23rd Annual International Symposium of the International Council on Systems Engineering, INCOSE 2013, *INCOSE International Symposium*, vol. 23, pp. 918–932 (2013)

[S292] Kanthabhabhajeya, S., Falkman, P., Lennartson, B.: System Modeling Specification in SysML and Sequence Planner Language - Comparison Study. In: Proceedings of the 14th IFAC Symposium on Information Control Problems in Manufacturing, *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol. 14, pp. 1543–1550 (2012). https://doi.org/10.3182/20120523-3-RO-2023.00189

[S293] Kapos, G.D., Dalakas, V., Nikolaidou, M., Anagnostopoulos, D.: An integrated framework for automated simulation of SysML models using DEVS. Simulation **90**(6), 717–744 (2014). https://doi.org/10.1177/0037549714533842

[S294] Kapos, G.D., Dalakas, V., Tsadimas, A., Nikolaidou, M., Anagnostopoulos, D.: Model-based system engineering using SysML: Deriving executable simulation models with QVT. In: Proceedings of the IEEE International Systems Conference, SysCon 2014, Ottawa, ON, Canada, March 31–April 3, pp. 531–538. IEEE (2014). https://doi.org/10.1109/SysCon.2014.6819307

[S295] Kapos, G.D., Nikolaidou, M., Dalakas, V., Anagnostopoulos, D.: An integrated framework to simulate SysML models using DEVS simulators. In: Formal Languages for Computer Simulation: Transdisciplinary Models and Applications, pp. 305–332. IGI Global (2013). https://doi.org/10.4018/978-1-4666-4369-7.ch010

[S296] Karwowski, W., Ahram, T.: Interactive Management of Human Factors Knowledge for Human Systems Integration Using Systems Modeling Language. Information Systems Management **26**(3), 262–274 (2009). https://doi.org/10.1080/10580530903018128

[S297] Karwowski, W., Amaba, B., Ahram, T., Bedny, G.: Human reliability assessment using systems modeling language & tasks based systemic-structural activity theory. In: Proceedings of the 8th International Topical Meeting on Nuclear Plant Instrumentation, Control, and Human-Machine Interface Technologies (NPIC and HMIT 2012): Enabling the Future of Nuclear Energy, vol. 3, pp. 1464–1475 (2012)

[S298] Kawahara, R., Nakamura, H., Dotan, D., Kirshin, A., Sakairi, T., Hirose, S., Ono, K., Ishikawa, H.: Verification of embedded system's specification using collaborative simulation of SysML and simulink models. In: Proceedings of the International Conference on Model-Based Systems Engineering (MBSE 2009), pp. 21–28. IEEE (2009). https://doi.org/10.1109/MBSE.2009.5031716

[S299] Kernschmidt, K., Barbieri, G., Fantuzzi, C., Vogel-Heuser, B.: Possibilities and Challenges of an Integrated Development Using a Combined SysML-Model and Corresponding Domain Specific Models. In: Proceedings of the 7th IFAC Conference on Manufacturing Modelling, Management, and Control, MIM 2013, Saint Petersburg, Russia, June 19–21, pp. 1465–1470. International Federation of Automatic Control (2013). https://doi.org/10.3182/20130619-3-RU-3018.00391

[S300] Kernschmidt, K., Vogel-Heuser, B.: An interdisciplinary SysML based modeling approach for analyzing change influences in production plants to support the engineering. In: Proceedings of the 2013 IEEE International Conference on Automation Science and Engineering, CASE 2013, Madison, WI, USA, August 17–20, pp. 1113–1118. IEEE (2013). https://doi.org/10.1109/CoASE.2013.6654030

[S301] Kerzhner, A., Paredis, C.: Combining SysML and Model Transformations to Support Systems Engineering Analysis. ECEASST **42** (2011)

[S302] Kerzhner, A., Paredis, C.: A SysML-Based Language for Modeling System-Level Architecture Selection Decisions. In: Proceedings of the ASME Design Engineering Technical Conference, vol. 2, pp. 1263–1276 (2012). https://doi.org/10.1115/DETC2012-71005

[S303] Khan, A., Mallet, F., Rashid, M.: Combining SysML and Marte/CCSL to Model Complex Electronic Systems. In: Proceedings of the International Conference on Information Systems Engineering (ICISE), pp. 12–17 (2016). https://doi.org/10.1109/ICISE.2016.13

[S304] Khan, A., Rashid, M.: Generation of SystemVerilog Observers from SysML and MARTE/CCSL. In: Proceedings of the 19th International Symposium on Real-Time Distributed Computing (ISORC), pp. 61–68. IEEE Computer Society (2016). https://doi.org/10.1109/ISORC.2016.18

[S305] Kim, H., Fried, D., Menegay, P.: Connecting SysML models with engineering analyses to support multidisciplinary system development. In: Proceedings of the 12th AIAA Aviation Technology, Integration and Operations (ATIO) Conference and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Indianapolis, IN, September 17–19 (2012)

[S306] Kim, S.: Automating building energy system modeling and analysis: An approach based on SysML and model transformations. Automation in Construction **41**, 119–138 (2014). https://doi.org/10.1016/j.autcon.2013.10.018

[S307] Kinoshita, S., Nishimura, H., Takamura, H., Mizuguchi, D.: Describing Software Specification by Combining SysML with the B Method. In: Proceedings of the 25th IEEE International Symposium on Software Reliability Engineering Workshops, ISSRE Workshops, Naples, Italy, November 3–6, pp. 146–151. IEEE Computer Society (2014). https://doi.org/10.1109/ISSREW.2014.66

[S308] Knorreck, D., Apvrille, L., de Saqui-Sannes, P.: TEPE: a SysML language for time-constrained property modeling and formal verification. ACM SIGSOFT Software Engineering Notes **36**(1), 1–8 (2011). https://doi.org/10.1145/1921532.1921556

[S309] Korff, A.: UML for Systems Engineering (SysML) Eine Notation zur Beschreibung von Systemen. In: Echtzeitaspekte bei der Koordinierung Autonomer Systeme - PEARL 2005, Fachtagung der GI-Fachgruppe Echtzeitsysteme und PEARL (EP), Boppard, 1./2. Dezember 2005, Informatik Aktuell, pp. 115–123. Springer (2005). https://doi.org/10.1007/3-540-29595-X_11

[S310] Korff, A.: Re-using SysML System Architectures. In: Proceedings of the 4th International Conference on Complex Systems Design & Management (CSD&M 2013), Paris, France, December 4–6, pp. 257–266. Springer (2013). https://doi.org/10.1007/978-3-319-02812-5_19

[S311] Kößler, J., Paetzold, K.: Supporting SysML model generation in early phases of the development process. In: Proceedings of NordDesign, Trondheim, Norway, 10–12 August, *NordDesign*, vol. 1, pp. 300–309 (2016)

[S312] Kotronis, C., Tsadimas, A., Kapos, G., Dalakas, V., Nikolaidou, M., Anagnostopoulos, D.: Simulating SysML transportation models. In: Proceedings of the International Conference on Systems, Man, and Cybernetics, SMC 2016, Budapest, Hungary, October 9–12, pp. 1674–1679. IEEE (2016). https://doi.org/10.1109/SMC.2016.7844478

[S313] Kretzenbacher, M., Findlay, R., Lange, C., Yan, W.: Model Based Systems Engineering (MBSE) applied through a SysML model to the mascot asteroid lander. In: Proceedings of the 64th International Astronautical Congress (IAC), vol. 10, pp. 8015–8022 (2013)

[S314] Kruse, B., Gilz, T., Shea, K., Eigner, M.: Systematic comparison of functional models in SysML for design library evaluation. In: Proceedings of the 24th CIRP Design Conference, *Procedia CIRP*, vol. 21, pp. 34–39 (2014). https://doi.org/10.1016/j.procir.2014.03.175

[S315] Kruse, B., Münzer, C., Wölkl, S., Canedo, A., Shea, K.: A Model-Based Functional Modeling and Library Approach for Mechatronic Systems in SysML. In: Proceedings of the ASME Design Engineering Technical Conference and Computers and Information in Engineering Conference, Chicago, Illinois, USA, August 12–15, vol. 2, pp. 1217–1227 (2012). https://doi.org/10.1115/DETC2012-70378

[S316] Kruse, B., Shea, K.: Design Library Solution Patterns in SysML for Concept Design and Simulation. In: Proceedings of the 26th CIRP Design Conference, *Procedia CIRP*, vol. 50, pp. 695–700 (2016). https://doi.org/10.1016/j.procir.2016.04.132

[S317] Kruus, H., Jervan, G.: Evaluation of SysML software for teaching systems engineering basics. In: Proceedings of the 25th International Conference on European Association for Education in Electrical and Information Engineering, EAEEIE 2014, pp. 29–32. IEEE (2014). https://doi.org/10.1109/EAEEIE.2014.6879379

[S318] Kruus, H., Robal, T., Jervan, G.: SysML in systems engineering course. In: Proceedings of the 10th European Workshop on Microelectronics Education (EWME), pp. 177–181 (2014). https://doi.org/10.1109/EWME.2014.6877421

[S319] Kruus, H., Robal, T., Jervan, G.: Teaching modeling in SysML/UML and problems encountered. In: Proceedings of the 25th International Conference on European Association for Education in Electrical and Information Engineering, EAEEIE 2014, pp. 33–36 (2014). https://doi.org/10.1109/EAEEIE.2014.6879380

[S320] Kwon, K.S., McGinnis, L.F.: SysML-based Simulation Framework for Semiconductor Manufacturing. In: Proceedings of the IEEE Conference on Automation Science and Engineering (CASE 2007), Scottsdale, Arizona, USA, September 22–25, pp. 1075–1080. IEEE (2007). https://doi.org/10.1109/COASE.2007.4341777

[S321] Lafi, S., Champagne, R., Kouki, A., Belzile, J.: Modeling radio-frequency front-ends using SysML: A case study of a UMTS transceiver. In: Proceedings of the 1st International Workshop on Model Based Architecting and Construction of Embedded Systems, *CEUR Workshop Proceedings*, vol. 503, pp. 115–128 (2008)

[S322] Lakhdara, Z., Merniz, S.: A SysML and CLEAN based methodology for RISC processor micro- Architecture design. International Journal of Embedded and Real-Time Communication Systems **6**(1), 101–131 (2015). https://doi.org/10.4018/IJERTCS.2015010105

[S323] Lakhdara, Z., Merniz, S.: A SysML and CLEAN-based methodology for digital circuits design. IJHPSA **6**(4), 222–237 (2016). https://doi.org/10.1504/IJHPSA.2016.10002660

[S324] Laleau, R., Semmak, F., Matoussi, A., Petit, D., Hammad, A., Tatibouet, B.: A first attempt to combine SysML requirements diagrams and B. Innovations in Systems and Software Engineering **6**(1), 47–54 (2010). https://doi.org/10.1007/s11334-009-0119-y

[S325] Lane, J., Bohn, T.: Using SysML modeling to understand and evolve systems of systems. Systems Engineering **16**(1), 87–98 (2013). https://doi.org/10.1002/sys.21221

[S326] de Lange, D., Guo, J., de Koning, H.: Applicability of SysML to the Early Definition Phase of Space Missions in a Concurrent Environment. In: Proceedings of the 2nd International Conference on Complex Systems Design & Management (CSDM 2011), Paris, France, December 7–9, pp. 173–185. Springer (2011). https://doi.org/10.1007/978-3-642-25203-7_12

[S327] Larisch, M., Hänle, A., Siebold, U., Häring, I.: SysML aided functional safety assessment. In: Proceedings of the ESREL 2008 and 17th SRA-Europe Conference, Valencia, Spain, September, pp. 1547–1554 (2008)

[S328] Lasalle, J., Bouquet, F., Legeard, B., Peureux, F.: SysML to UML model transformation for test generation purpose. ACM SIGSOFT Software Engineering Notes **36**(1), 1–8 (2011). https://doi.org/10.1145/1921532.1921560

[S329] Lasalle, J., Peureux, F., Fondement, F.: Development of an automated MBT toolchain from UML/SysML models. Innovations in Systems and Software Engineering **7**(4), 247–256 (2011). https://doi.org/10.1007/s11334-011-0164-1

[S330] Le Mair, A., Fraanje, R.: Using SysML to teach Systems Engineering skills. In: Proceedings of the 11th France-Japan 9th Europe-Asia Congress on Mechatronics (MECATRONICS) /17th International Conference on Research and Education in Mechatronics (REM), pp. 1–5 (2016). https://doi.org/10.1109/MECATRONICS.2016.7547106

[S331] Lee, T., Cha, J.M., Kim, J.Y., Shin, J., Kim, J., Yeom, C.: Plant modeling based on SysML domain specific language. In: Proceedings of the IEEE International Systems Engineering Symposium (ISSE), pp. 1–5 (2017). https://doi.org/10.1109/SysEng.2017.8088289

[S332] Leite, J.C., Oquendo, F., Batista, T.V.: SysADL: A SysML Profile for Software Architecture Description. In: Proceedings of the 7th European Conference on Software Architecture (ECSA 2013), Montpellier, France, July 1–5, *Lecture Notes in Computer Science*, vol. 7957, pp. 106–113. Springer (2013). https://doi.org/10.1007/978-3-642-39031-9_9

[S333] Lemaire, L., Lapon, J., Decker, B.D., Naessens, V.: A SysML Extension for Security Analysis of Industrial Control Systems. In: Proceedings of the 2nd International Symposium for ICS & SCADA Cyber Security Research 2014, ICS-CSR 2014, St. Pölten, Austria, September 11–12, Workshops in Computing. BCS (2014)

[S334] Lempia, D., Jorgensen, R.: Practical SysML Applications: A Method to Describe the Problem Space. In: Proceedings of the 21st Annual International Symposium of the International Council on Systems Engineering (INCOSE 2011), *INCOSE International Symposium*, vol. 21, pp. 85–98 (2011)

[S335] Leserf, P., De Saqui-Sannes, P., Hugues, J.: Multi domain optimization with SysML modeling. In: Proceedings of the IEEE 20th Conference on Emerging Technologies Factory Automation (ETFA), Sept., pp. 1–8 (2015). https://doi.org/10.1109/ETFA.2015.7301406

[S336] Leserf, P., De Saqui-Sannes, P., Hugues, J., Chaaban, K.: Architecture Optimization with SysML Modeling: A Case Study Using Variability. In: Proceedings of the 3rd International Conference on Model-Driven Engineering and Software Development (MODELSWARD), Angers, France, February 9–11, *Communications in Computer and Information Science*, vol. 580, pp. 311–327. Springer (2015). https://doi.org/10.1007/978-3-319-27869-8_18

[S337] Leserf, P., De Saqui-Sannes, P., Hugues, J., Chaaban, K.: SysML Modeling for Embedded Systems Design Optimization: A Case Study. In: Proceedings of the 3rd International Conference on Model-Driven Engineering and Software Development (MODELSWARD), ESEO, Angers, Loire Valley, France, 9–11 February, pp. 449–457. SciTePress (2015). https://doi.org/10.5220/0005229204490457

[S338] Li, G., Wang, B.: SysML Aided Safety Analysis for Safety-Critical Systems. In: Proceedings of the

3rd International Conference on Artificial Intelligence and Computational Intelligence (AICI 2011), Taiyuan, China, September 24–25, Part I, *Lecture Notes in Computer Science*, vol. 7002, pp. 270–275. Springer (2011). https://doi.org/10.1007/978-3-642-23881-9_35

[S339] Li, L., Wang, N., Ma, L., Yang, Q.: Modeling method of military aircraft support process based SysML. In: Proceedings of 9th International Conference on Reliability, Maintainability and Safety (ICRMS 2011), pp. 1247–1251. IEEE (2011). https://doi.org/10.1109/ICRMS.2011.5979460

[S340] Li, X., Liu, J.: A method of SysML-based visual transformation of system design-simulation models. Jisuanji Fuzhu Sheji Yu Tuxingxue Xuebao/Journal of Computer-Aided Design and Computer Graphics **28**(11), 1973–1981 (2016)

[S341] Lima, L., Didier, A., Cornélio, M.: A Formal Semantics for SysML Activity Diagrams. In: Formal Methods: Foundations and Applications - Proceedings of the 16th Brazilian Symposium, SBMF 2013, Brasilia, Brazil, September 29–October 4, *Lecture Notes in Computer Science*, vol. 8195, pp. 179–194. Springer (2013). https://doi.org/10.1007/978-3-642-41071-0_13

[S342] Lima, L., Miyazawa, A., Cavalcanti, A., Cornélio, M., Iyoda, J., Sampaio, A., Hains, R., Larkham, A., Lewis, V.: An integrated semantics for reasoning about SysML design models using refinement. Software and Systems Modeling **26**(2), 367–405 (2015). https://doi.org/10.1007/s10270-015-0492-y

[S343] Limère, V., Balachandran, S., McGinnis, L., Van Landeghem, H.: In-plant logistics systems modeling with SysML. In: ESM 2010 - European Simulation and Modelling Conference, pp. 383–387 (2010)

[S344] Lin, C.S., Lu, C.H., Lin, S.W., Chen, Y.R., Hsiung, P.A.: VERTAF/multi-core: A SysML-based application framework for multi-core embedded software development. Journal of Computer Science and Technology **26**(3), 448–462 (2011). https://doi.org/10.1007/s11390-011-1146-3

[S345] Lin, H., Sorouri, M., Vyatkin, V., Salcic, Z.: Model-based customisation of intelligent mechatronic systems using SysML. In: Proceedings of 2013 IEEE 18th Conference on Emerging Technologies & Factory Automation, ETFA 2013, Cagliari, Italy, September 10–13, pp. 1–4. IEEE (2013). https://doi.org/10.1109/ETFA.2013.6648138

[S346] Lin, H.Y., Sierla, S., Papakonstantinou, N., Vyatkin, V.: A SysML profile supporting change orders in model driven engineering. In: Proceedings of the IEEE International Conference on Automation Science and Engineering (CASE), Aug., pp. 1054–1059 (2015). https://doi.org/10.1109/CoASE.2015.7294238

[S347] Lin, J., Shih, P.H., Huang, E., Chiu, C.C.: Airport baggage handling system simulation modeling using SysML. In: Proceedings of the 5th International Conference on Industrial Engineering and Operations Management (IEOM), March, pp. 1–10 (2015). https://doi.org/10.1109/IEOM.2015.7093764

[S348] Linhares, M.V., de Oliveira, R.S., Farines, J., Vernadat, F.: Introducing the modeling and verification process in SysML. In: Proceedings of 12th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2007), Patras, Greece, September 25–28, pp. 344–351. IEEE (2007). https://doi.org/10.1109/EFTA.2007.4416788

[S349] Linhares, M.V., da Silva, A.J., de Oliveira, R.S.: Empirical Evaluation of SysML through the Modeling of an Industrial Automation Unit. In: Proceedings of 11th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2006), Diplomat Hotel Prague, Czech Republic, September 20–22, pp. 145–152. IEEE (2006). https://doi.org/10.1109/ETFA.2006.355190

[S350] Liu, J., Wang, S., Fu, C.: Design Analysis Method for Multidisciplinary Complex Product using SysML. In: Proceedings of the 3rd International Conference on Mechanical, Electronic and Information Technology Engineering (ICMITE 2017), *MATEC Web Conference*, vol. 139 (2017)

[S351] Liu, X., Ren, Y., Wang, Z., Liu, L.: Modeling method of SysML-based reliability block diagram. In: Proceedings of the International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC 2013), pp. 206–209. IEEE (2013). https://doi.org/10.1109/MEC.2013.6885073

[S352] Liu, X.H., Cao, Y.F., Wang, B., Zhuang, L.K., Zhou, Z.H.: Flight control system conceptual prototype design based on SysML and Simulink. Dianzi Keji Daxue Xuebao/Journal of the University of Electronic Science and Technology of China **40**(6), 887–891+910 (2011). https://doi.org/10.3969/j.issn.1001-0548.2011.06.016

[S353] Liu, Y., Irudayaraj, P., Zhou, F., Jiao, R., Goodman, J.: SysML-based Model Driven Discrete-Event Simulation. In: Proceedings of the 21st ISPE Inc. International Conference on Concurrent Engineering, Beijing Jiaotong University, China, September 8–11, *Advances in Transdisciplinary Engineering*, vol. 1, pp. 617–626. IOS Press (2014). https://doi.org/10.3233/978-1-61499-440-4-617

[S354] Liu, Y., Yuan, W., Fan, H., Cao, Y.: Research on information integration framework of SysML based model driven design of complex products. Zhongguo Jixie Gongcheng/China Mechanical Engineering **23**(12), 1438–1445 (2012). https://doi.org/10.3969/j.issn.1004-132X.2012.12.011

[S355] Liu, Y.S., Fan, H.R.: Integration of System-Level Design and Detailed Design Models of Mechatronic Systems Based on SysML and Step AP 203 Standard. In: Applied Mechanics and Mechanical Engineering III, *Applied Mechanics and Materials*, vol. 249, pp. 1160–1165. Trans Tech Publications (2013). https://doi.org/10.4028/www.scientific.net/AMM.249-250.1160

[S356] Liu, Y.S., Yuan, W.Q.: Automatic Integration of System-Level Design and System Optimization Based on SysML. In: Applied Mechanics and Mechanical Engineering III, *Applied Mechanics and Materials*, vol. 249, pp. 1154–1159. Trans Tech Publications (2013). https://doi.org/10.4028/www.scientific.net/AMM.249-250.1154

[S357] Lopata, A., Ambraziunas, M., Veitaite, I., Masteika, S., Butleris, R.: SysML and UML models usage in knowledge based MDA process. Elektronika ir Elektrotechnika **21**(2), 50–57 (2015). https://doi.org/10.5755/j01.eee.21.2.5629

[S358] Lovric, T., Schneider-Scheyer, M., Sarkic, S.: SysML as backbone for engineering and safety - Practical experience with TRW braking ECU. In: Proceedings of the SAE 2014 World Congress & Exhibition, *SAE Technical Papers*, vol. 1 (2014). https://doi.org/10.4271/2014-01-0212

[S359] Lu, X., Piétrac, L., Niel, E.: A new approach of modeling supervisory control for manufacturing systems based on SysML. In: Proceedings of the 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), pp. 1–9 (2017). https://doi.org/10.1109/ETFA.2017.8247626

[S360] Lugou, F., Li, L., Apvrille, L., Ameur-Boulifa, R.: SysML Models and Model Transformation for Security. In: Proceedings of the 4th International Conference on Model-Driven Engineering and Software Development - MODELSWARD 2016, Rome, Italy, 19–21 February, pp. 331–338. SciTePress (2016). https://doi.org/10.5220/0005748703310338

[S361] Lukas, S., Lukas, H.: Proposal of System Testing Integration into Safety Critical System Design Process Supported by SysML. In: Proceedings of the UKSim-AMSS 8th European Modelling Symposium on Computer Modelling and Simulation, EMS 2014, Pisa, Italy, October 21–23, pp. 251–256. IEEE (2014). https://doi.org/10.1109/EMS.2014.74

[S362] Machida, F., Andrade, E.C., Kim, D.S., Trivedi, K.S.: Candy: Component-based Availability Modeling Framework for Cloud Service Management Using SysML. In: Proceedings of the 30th IEEE Symposium on Reliable Distributed Systems (SRDS 2011), Madrid, Spain, October 4–7, pp. 209–218. IEEE Computer Society (2011). https://doi.org/10.1109/SRDS.2011.33

[S363] Machida, F., Xiang, J., Tadano, K., Maeno, Y.: Composing hierarchical stochastic model from SysML for system availability analysis. In: Proceedings of the IEEE 24th International Symposium on Software Reliability Engineering, ISSRE 2013, Pasadena, CA, USA, November 4–7, pp. 51–60. IEEE Computer Society (2013). https://doi.org/10.1109/ISSRE.2013.6698904

[S364] Maisenbacher, S., Kernschmidt, K., Kasperek, D., Vogel-Heuser, B., Maurer, M.: Using DSM and MDM methodologies to analyze structural SysML models. In: Proceedings of the IEEE International Conference on Industrial Engineering and Engineering Management, Bangkok, Thailand, pp. 351–355 (2014). https://doi.org/10.1109/IEEM.2013.6962432

[S365] Maissa, Y., Mouline, S.: A SysML profile for wireless sensor networks modeling. In: Proceedings of the 5th International Symposium on I/V Communications and Mobile Networks (ISIVC 2010), pp. 1–4. IEEE (2010). https://doi.org/10.1109/ISVC.2010.5656281

[S366] Makartetskiy, D., Sisto, R.: An approach to refinement checking of SysML requirements. In: Proceedings of the 16th Conference on Emerging Technologies & Factory Automation (ETFA 2011), Toulouse, France, September 5–9, pp. 1–4. IEEE (2011). https://doi.org/10.1109/ETFA.2011.6059147

[S367] Mandutianu, S., Morillo, R., Simpson, K., Liepack, O., Bonanne, K.: Modeling complex cross-systems software interfaces using SysML. In: Proceedings of the AIAA Infotech at Aerospace Conference, Boston, MA, USA, August 19–22 (2013)

[S368] Marco, J., Vaughan, N.: Integration of architectural modelling using the SysML within the traditional automotive CACSD process. In: Proceedings of the UKACC International Conference on Control 2010, pp. 668–673. IET (2010). https://doi.org/10.1049/ic.2010.0361

[S369] Marco, J., Vaughan, N.: Architectural modelling of an energy management control system using the SysML. International Journal of Vehicle Design **55**(1), 1–22 (2011). https://doi.org/10.1504/IJVD.2011.038044

[S370] Maroui, R., Ayeb, B.: Integrating the SysML and ACME in a Model Driven Engineering Approach to Verify the Web Service Composition. In: Proceedings of the 24th IEEE International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE 2015, Larnaca, Cyprus, June 15–17, pp. 183–189. IEEE Computer Society (2015). https://doi.org/10.1109/WETICE.2015.41

[S371] Matthiesen, S., Schmidt, S., Moeser, G.: SysKIT 2.0 - Implementation of a SysML teaching approach and observations on systems modelling by mechatronic teams. In: Proceedings of the 17th International Conference on Engineering and Product Design Education: Great Expectations: Design Teaching, Research and Enterprise, Loughborough, UK, E&PDE, pp. 518–523 (2015)

[S372] Matthiesen, S., Schmidt, S., Moeser, G., Munker, F.: The Karlsruhe SysKIT Approach - A Three-Step SysML Teaching Approach for Mechatronic Students. In: Proceedings of the 24th CIRP Design Conference, *Procedia CIRP*, vol. 21, pp. 385–390 (2014). https://doi.org/10.1016/j.procir.2014.03.136

[S373] Maurandy, J., Gill, E., Helm, A., Stalford, R.: Cost-Benefit Analysis of SysML Modelling for the Atomic Clock Ensemble in Space (ACES) Simulator. In: Proceedings of the 22nd Annual International Symposium of the International Council on Systems Engineering (INCOSE 2012) and the 8th Biennial European Systems Engineering Conference (EuSEC 2012), *INCOSE International Symposium*, vol. 22, pp. 1726–1745 (2012)

[S374] Mazzini, S., Stragapede, A.: SysML: A language for space system engineering. In: Proceedings of the Conference on DAta Systems In Aerospace (DASIA 2008), Noordwijk, Netherlands, no. 665 in ESA SP (2008)

[S375] McGinnis, L.F., Ustun, V.: A Simple Example of SysML-driven Simulation. In: Proceedings of the 2009 Winter Simulation Conference (WSC 2009), Hilton Austin Hotel, Austin, TX, USA, December 13–16, pp. 1703–1710. WSC (2009). https://doi.org/10.1109/WSC.2009.5429169

[S376] McKelvin Jr., M., Jimenez, A.: Specification and design of electrical flight system architectures with SysML. In: Proceedings of the AIAA Infotech at Aerospace Conference and Exhibit 2012, Garden Grove, CA, USA (2012)

[S377] Meacham, S., Gioulekas, F., Phalp, K.: SysML based design for variability enabling the reusability of legacy systems towards the support of diverse standard compliant implementations or standard updates: the case of IEEE-802.15.6 standard for e-Health applications. In: Proceedings of the 8th International Conference on Simulation Tools and Techniques, Athens, Greece, August 24–26, 2015, pp. 284–289. ICST/ACM (2015). https://doi.org/10.4108/eai.24-8-2015.2261108

[S378] Mehrpouyan, H., Tumer, I., Hoyle, C., Giannakopoulou, D., Brat, G.: Formal verification of complex systems based on SysML functional requirements. In: Proceedings of the Annual Conference of the Prognostics and Health Management Society 2014 (PHM), pp. 176–187 (2014)

[S379] Mendes, J., Carreira, A., Aleluia, M., Mendes, J.: Formulating strategic problems with Systems Modeling Language. Journal of Enterprise Transformation **6**(1), 23–38 (2016). https://doi.org/10.1080/19488289.2016.1210705

[S380] Mendieta, R., de la Vara, J.L., Morillo, J.L., Rodríguez, J.M.Á.: Towards Effective SysML Model Reuse. In: L.F. Pires, S. Hammoudi, B. Selic (eds.) Proceedings of the 5th International Conference on Model-Driven Engineering and Software Development, MODELSWARD 2017, Porto, Portugal, February 19–21, pp. 536–541. SciTePress (2017). https://doi.org/10.5220/0006267605360541

[S381] Meng, C., Kim, S., Son, Y., Kubota, C.: A SysML-based simulation model aggregation framework for seedling propagation system. In: Proceedings of the Winter Simulations Conference: Simulation Making Decisions in a Complex World, WSC 2013, Washington, DC, USA, December 8–11, pp. 2180–2191. IEEE (2013). https://doi.org/10.1109/WSC.2013.6721595

[S382] Mentré, D.: SysML2B: Automatic Tool for B Project Graphical Architecture Design Using SysML. In: Proceedings of the 5th International Conference on Abstract State Machines, Alloy, B, TLA, VDM, and Z (ABZ), Linz, Austria, May 23–27, *LNCS*, vol. 9675, pp. 308–311. Springer (2016). https://doi.org/10.1007/978-3-319-33600-8_26

[S383] Mhenni, F., Choley, J.Y., Nguyen, N.: Extended mechatronic systems architecture modeling with SysML for enhanced safety analysis. In: Proceedings of the IEEE International Systems Conference, SysCon 2014, Ottawa, ON, Canada, March 31–April 3, pp. 378–382. IEEE (2014). https://doi.org/10.1109/SysCon.2014.6819284

[S384] Mhenni, F., Choley, J.Y., Nguyen, N.: SysML safety profile for mechatronics. In: Proceedings of the 10th France-Japan / 8th Europe-Asia Congress on Mecatronics (MECATRONICS) 2014, pp. 29–34 (2014). https://doi.org/10.1109/MECATRONICS.2014.7018622

[S385] Mhenni, F., Choley, J.Y., Nguyen, N.: SysML extensions for safety-critical mechatronic systems design. In: Proceedings of the 1st IEEE International Symposium on Systems Engineering (ISSE), Sept., pp. 242–247 (2015). https://doi.org/10.1109/SysEng.2015.7302764

[S386] Mhenni, F., Choley, J.Y., Nguyen, N., Frazza, C.: Flight Control System Modeling with SysML to Support Validation, Qualification and Certification. IFAC-PapersOnLine: Proceedings of the 14th IFAC Symposium on Control in Transportation Systems (CTS), Istanbul, Turkey, 18–20 May **49**(3), 453–458 (2016). https://doi.org/10.1016/j.ifacol.2016.07.076

[S387] Mhenni, F., Choley, J.Y., Penas, O., Plateaux, R., Hammadi, M.: A SysML-based methodology for mechatronic systems architectural design. Advanced Engineering Informatics **28**(3), 218–231 (2014). https://doi.org/10.1016/j.aei.2014.03.006

[S388] Mhenni, F., Choley, J.Y., Riviere, A., Nguyen, N., Kadima, H.: SysML and safety analysis for mechatronic systems. In: Proceedings of the 9th France-Japan and 7th Europe-Asia Congress on Mechatronics (MECATRONICS 2012) / 13th International Workshop on Research and Education in Mechatronics (REM 2012), pp. 417–424. IEEE (2012). https://doi.org/10.1109/MECATRONICS.2012.6451042

[S389] Mhenni, F., Nguyen, N., Choley, J.Y.: Towards the Integration of Safety Analysis in a Model-Based System Engineering Approach with SysML. In: Proceedings of the 5th International Conference Design and Modeling of Mechanical Systems (CMSM 2013), Djerba, Tunisia, March 25–27, *Lecture Notes in Mechanical Engineering*, vol. 1, pp. 61–68 (2013). https://doi.org/10.1007/978-3-642-37143-1_8

[S390] Mhenni, F., Nguyen, N., Choley, J.Y.: Automatic fault tree generation from SysML system models. In: Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), pp. 715–720 (2014). https://doi.org/10.1109/AIM.2014.6878163

[S391] Mhenni, F., Nguyen, N., Kadima, H., Choley, J.: Safety analysis integration in a SysML-based complex system design process. In: Proceedings of the IEEE International Systems Conference, SysCon 2013, Orlando, FL, USA, April 15–18, pp. 70–75. IEEE (2013). https://doi.org/10.1109/SysCon.2013.6549861

[S392] Min, B., Kerzhner, A., Paredis, C.: Process Integration and Design Optimization for Model-Based Systems Engineering with SysML. In: Proceedings of the ASME Design Engineering Technical Conference and Computers and Information in Engineering Conference, vol. 2, pp. 1361–1369 (2011). https://doi.org/10.1115/DETC2011-48453

[S393] Miyazawa, A., Cavalcanti, A.: Formal Refinement in SysML. In: Proceedings of the 11th International Conference on Integrated Formal Methods, IFM 2014, Bertinoro, Italy, September 9–11, *Lecture Notes in Computer Science*, vol. 8739, pp. 155–170. Springer (2014). https://doi.org/10.1007/978-3-319-10181-1_10

[S394] Miyazawa, A., Lima, L., Cavalcanti, A.: Formal Models of SysML Blocks. In: Proceedings of the 15th International Conference on Formal Engineering Methods, ICFEM 2013, Queenstown, New Zealand, October 29–November 1, *Lecture Notes in Computer Science*, vol. 8144, pp. 249–264. Springer (2013). https://doi.org/10.1007/978-3-642-41202-8_17

[S395] Monica, F.D., Patalano, S., Choley, J.Y., Mhenni, F., Gerbino, S.: A hierarchical set of SysML Model-based objects for tolerance specification. In: Proceedings of the 2016 IEEE International Symposium on Systems Engineering (ISSE), Oct., pp. 1–7 (2016). https://doi.org/10.1109/SysEng.2016.7753143

[S396] Morelli, M.: Automated generation of robotics applications from simulink and SysML models. In: Proceedings of the 30th Annual ACM Symposium on Applied Computing, Salamanca, Spain, April 13–17, pp. 1948–1954. ACM (2015). https://doi.org/10.1145/2695664.2695882

[S397] Morgan, D., Waldock, A., Corne, D.: Efficient systems analysis by combining SysML and coevolution. In: Proceedings of the 7th International Conference on System of Systems Engineering (SoSE 2012), pp. 83–88. IEEE (2012). https://doi.org/10.1109/SYSoSE.2012.6384141

[S398] Mori, M., Ceccarelli, A., Lollini, P., Bondavalli, A., Frömel, B.: A Holistic Viewpoint-Based SysML Profile to Design Systems-of-Systems. In: Proceedings of the 17th IEEE International Symposium on High Assurance Systems Engineering HASE 2016, Orlando, FL, USA, January 7–9, pp. 276–283. IEEE Computer Society (2016). https://doi.org/10.1109/HASE.2016.21

[S399] Mori, M., Ceccarelli, A., Lollini, P., Frömel, B., Brancati, F., Bondavalli, A.: Systems-of-systems modeling using a comprehensive viewpoint-based SysML profile. Journal of Software: Evolution and Process p. e1878 (2017). https://doi.org/10.1002/smr.1878

[S400] Morkevicius, A., Jankevicius, N.: An approach: SysML-based automated requirements verification. In: Proceedings of the 1st IEEE International Symposium on Systems Engineering (ISSE), Sept., pp. 92–97 (2015). https://doi.org/10.1109/SysEng.2015.7302739

[S401] Müller, M., Roth, M., Lindemann, U.: The hazard analysis profile: Linking safety analysis and SysML. In: Proceedings of the 10th Annual International Systems Conference, SysCon 2016, Orlando, FL, USA, April 18–21, pp. 1–7. IEEE (2016). https://doi.org/10.1109/SYSCON.2016.7490532

[S402] Müller, W., He, D., Mischkalla, F., Wegele, A., Larkham, A., Whiston, P., Peñil, P., Villar, E., Mitas, N., Kritharidis, D., Azcarate, F., Carballeda, M.: The SATURN Approach to SysML-Based HW/SW Codesign. In: Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI 2010), Lixouri Kefalonia, Greece, July 5–7, *Lecture Notes in Electrical Engineering*, vol. 105, pp. 151–164. Springer (2011). https://doi.org/10.1007/978-94-007-1488-5_9

[S403] Mura, M., Murillo, L.G., Prevostini, M.: Model-based Design Space Exploration for RTES with SysML and MARTE. In: Proceedings of the Forum on specification and Design Languages (FDL 2008), Stuttgart, Germany, September 23–25, pp. 203–208. IEEE (2008). https://doi.org/10.1109/FDL.2008.4641446

[S404] Muşat, L., Hübl, M., Buzo, A., Pelz, G., Kandl, S., Puschner, P.: Semi-formal representation of requirements for automotive solutions using SysML, vol. 361, pp. 57–81. Springer International Publishing (2016). https://doi.org/10.1007/978-3-319-24457-0_4

[S405] Nakajima, S., Furukawa, S., Ueda, Y.: Co-analysis of SysML and Simulink Models for Cyber-Physical Systems Design. In: Proceedings of the 18th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 2012), Seoul, Korea (South), August 19–22, pp. 473–478. IEEE Computer Society (2012). https://doi.org/10.1109/RTCSA.2012.22

[S406] Nasraoui, K., Lakhoua, N., Amraoui, L.E.: Study and analysis of micro smart grid using the modeling language SysML. In: Proceedings of the International Conference on Green Energy Conversion Systems (GECS), pp. 1–8 (2017). https://doi.org/10.1109/GECS.2017.8066142

[S407] Nejati, S., Sabetzadeh, M., Arora, C., Briand, L., Mandoux, F.: Automated change impact analysis between SysML models of requirements and design. In: Proceedings of the 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, FSE 2016, Seattle, WA, USA, November 13–18, pp. 242–253. ACM (2016). https://doi.org/10.1145/2950290.2950293

[S408] Nejati, S., Sabetzadeh, M., Falessi, D., Briand, L., Coq, T.: A SysML-based approach to traceability management and design slicing in support of safety certification: Framework, tool support, and case studies. Information and Software Technology **54**(6), 569–590 (2012). https://doi.org/10.1016/j.infsof.2012.01.005

[S409] Ngo, V., Soriano, T.: A model transformation process to realize controllers of ship autopilot systems by the specialized MDA's features with UML/SysML. In: Proceeding of the 9th France-Japan and 7th Europe-Asia Congress on Mechatronics (MECATRONICS 2012) / 13th International Workshop on Research and Education in Mechatronics (REM 2012), pp. 20–26. IEEE (2012). https://doi.org/10.1109/MECATRONICS.2012.6450983

[S410] Nguyen, N., Kadima, H.: SysML Parametric Models for Complex System Performance Analysis - A Case Study. In: SIMULTECH 2012 - Proceedings of the 2nd International Conference on Simulation and Modeling Methodologies, Technologies and Applications, Rome, Italy, July 28–31, pp. 321–327. SciTePress (2012)

[S411] Nigischer, C., Gerhard, D.: Lightweight visualization of SysML models in PDM systems. In: Proceedings of the 21st International Conference on Engineering Design (ICED 17): Product, Services and Systems Design, Vancouver, Canada, 21–25.08, vol. 3, pp. 61–70 (2017)

[S412] Nikolaidou, M., Dalakas, V., Mitsi, L., Kapos, G., Anagnostopoulos, D.: A SysML Profile for Classical DEVS Simulators. In: Proceedings of the 3rd International Conference on Software Engineering Advances (ICSEA 2008), Sliema, Malta, October 26–31, pp. 445–450. IEEE Computer Society (2008). https://doi.org/10.1109/ICSEA.2008.24

[S413] Nikolaidou, M., Kapos, G., Dalakas, V., Anagnostopoulos, D.: Basic guidelines for simulating SysML models: An experience report. In: Proceedings of the 7th International Conference on System of Systems Engineering (SoSE 2012), Genova, Italy, July 16–19, pp. 95–100. IEEE (2012). https://doi.org/10.1109/SYSoSE.2012.6384172

[S414] Nikolaidou, M., Kapos, G.D., Tsadimas, A., Dalakas, V., Anagnostopoulos, D.: Simulating SysML models: Overview and challenges. In: Proceedings of the 10th System of Systems Engineering Conference, SoSE 2015, San Antonio, TX, USA, May 17–20, pp. 328–333. IEEE (2015). https://doi.org/10.1109/SYSOSE.2015.7151961

[S415] Nikolaidou, M., Michalakelis, C.: Techno-economic analysis of SysML models. In: Proceedings of the IEEE International Systems Engineering Symposium (ISSE), pp. 1–6 (2017). https://doi.org/10.1109/SysEng.2017.8088276

[S416] Nonsiri, S., Christophe, F., Coataneéa, E., Mokammel, F.: A combined design structure matrix (DSM) and discrete differential evolution (DDE) approach for scheduling and organizing system development tasks modelled using SysML. Journal of Integrated Design & Process Science **18**(3), 19–40 (2014). https://doi.org/10.3233/jid-2014-0013

[S417] Noten, J.V., Gadeyne, K., Witters, M.: Model-based Systems Engineering of Discrete Production Lines Using SysML: An Experience Report. In: Complex Systems Engineering and Development Proceedings of the 27th CIRP Design Conference Cranfield University, UK 10th – 12th May, *Procedia CIRP*, vol. 60, pp. 157 – 162 (2017). https://doi.org/10.1016/j.procir.2017.01.018

[S418] Nottage, D., Corns, S., Soylemezoglu, A., Kinnevan, K.: A SysML framework for modeling contingency basing. Systems Engineering **18**(2), 162–177 (2015). https://doi.org/10.1002/sys.21297

[S419] Nottage, D., Corns, S.M.: SysML profiling for handling army base camp planning. In: Proceedings of the Complex Adaptive Systems 2011 Conference, Chicago, Illinois, USA, October 31-November 2, *Procedia Computer Science*, vol. 6, pp. 63–68. Elsevier (2011). https://doi.org/10.1016/j.procs.2011.08.014

[S420] Oates, R.F., Thom, F., Herries, G.: Security-Aware, Model-Based Systems Engineering with SysML. In: Proceedings of the 1st International Symposium for ICS & SCADA Cyber Security Research 2013, ICS-CSR 2013, Leicester, UK, September 16–17, Workshops in Computing. BCS (2013)

[S421] Ober, I., Ober, I., Dragomir, I., Aboussoror, E.: UML/SysML semantic tunings. Innovations in Systems and Software Engineering **7**(4), 257–264 (2011). https://doi.org/10.1007/s11334-011-0163-2

[S422] Ohara, K., Iwane, K., Takubo, T., Mae, Y., Arai, T.: Component-based robot software design for pick-and-place task described by SysML. In: Proceedings of the 8th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI 2011), Incheon, Korea (South), November 23–26, pp. 124–127. IEEE (2011). https://doi.org/10.1109/URAI.2011.6145945

[S423] Ollinger, L., Zühlke, D., Theorin, A., Johnsson, C.: A reference architecture for service-oriented control procedures and its implementation with SysML and Grafchart. In: Proceedings of 2013 IEEE 18th Conference on Emerging Technologies & Factory Automation, ETFA 2013, Cagliari, Italy, September 10–13, pp. 1–8. IEEE (2013). https://doi.org/10.1109/ETFA.2013.6648044

[S424] Ono, K., Nakamura, H., Ishikawa, H.: A dynamic verification method of executable UML/SysML models with timed-functional constraints. Computer Software **27**(2), 33–49 (2010)

[S425] Orellana, D., Turso, J., McClure, C.: Architecting a Nuclear Power Plant with SysML and a Model Driven Engineering Environment. In: Proceedings of the ANS Annual Winter Meeting and Embedded Topical Meetings: 1st ANS SMR 2011 Conference and Young Professionals Congress, *Transactions-American Nuclear Society*, vol. 105, pp. 323–324 (2011)

[S426] Orr, K., Ramakrishnan, S., Dagli, C.: Can systems modeling language impact systems engineering? In: Proceedings of the 16th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2006), *INCOSE International Symposium*, vol. 16, pp. 1688–1699 (2006)

[S427] Ortega-Míguez, C., Hein, A.: A methodology for Rapid Preliminary Space Mission Design using SysML. In: Proceedings of the International Astronautical Congress (IAC), vol. 10, pp. 8225–8236 (2012)

[S428] Ouchani, S., Aït Mohamed, O., Debbabi, M.: A property-based abstraction framework for SysML activity diagrams. Knowledge-Based Systems **56**, 328–343 (2014). https://doi.org/10.1016/j.knosys.2013.11.016

[S429] Ouchani, S., Lenzini, G.: Generating attacks in SysML activity diagrams by detecting attack surfaces. Journal of Ambient Intelligence and Humanized Computing **6**(3), 361–373 (2015). https://doi.org/10.1007/s12652-015-0269-8

[S430] Ouchani, S., Mohamed, O., Debbabi, M.: A formal verification framework for SysML activity diagrams. Expert Systems with Applications **41**(6), 2713–2728 (2014). https://doi.org/10.1016/j.eswa.2013.10.064

[S431] Ouchani, S., Mohamed, O.A., Debbabi, M.: A Probabilistic Verification Framework for SysML Activity Diagrams. In: New Trends in Software Methodologies, Tools and Techniques - Proceedings of the 11th SoMeT '12, Genoa, Italy, September 26–28, *Frontiers in Artificial Intelligence and Applications*, vol. 246, pp. 108–123. IOS Press (2012). https://doi.org/10.3233/978-1-61499-125-0-108

[S432] Ouchani, S., Mohamed, O.A., Debbabi, M.: Efficient probabilistic abstraction for sysml activity diagrams. In: G. Eleftherakis, M. Hinchey, M. Holcombe (eds.) Software Engineering and Formal Methods - 10th International Conference, SEFM 2012, Thessaloniki, Greece, October 1–5, 2012. Proceedings, *Lecture Notes in Computer Science*, vol. 7504, pp. 263–277. Springer (2012). https://doi.org/10.1007/978-3-642-33826-7_18

[S433] Ouchani, S., Mohamed, O.A., Debbabi, M.: A probabilistic verification framework of SysML activity diagrams. In: Proceedings of the IEEE 12th

International Conference on Intelligent Software Methodologies, Tools and Techniques, SoMeT 2013, Budapest, Hungary, September 22–24, pp. 165–170. IEEE (2013). https://doi.org/10.1109/SoMeT.2013.6645657

[S434] Ouchani, S., Mohamed, O.A., Debbabi, M.: A Security Risk Assessment Framework for SysML Activity Diagrams. In: Proceedings of the IEEE 7th International Conference on Software Security and Reliability, SERE 2013, Gaithersburg, MD, USA, June 18–20, pp. 227–236. IEEE (2013). https://doi.org/10.1109/SERE.2013.11

[S435] Ouerdi, N., Azizi, M., Ziane, M., Azizi, A., Lanet, J., Savary, A.: Security vulnerabilities tests generation from SysML and event-B models for EMV cards. International Journal of Security and its Applications **8**(1), 373–388 (2014). https://doi.org/10.14257/ijsia.2014.8.1.35

[S436] Ouerdi, N., Ziane, M., Azizi, A., Azizi, M.: Modeling embedded systems with SysML. In: Proceedings of International Conference on Multimedia Computing and Systems (ICMCS 2012), pp. 905–910. IEEE (2012). https://doi.org/10.1109/ICMCS.2012.6320272

[S437] Ouerdi, N., Ziane, M., Azizi, A., Azizi, M., Lanet, J.L.: Abstract tests based on SysML models for EMV Card. In: Proceedings of the National Security Days (JNS3), pp. 1–6 (2013). https://doi.org/10.1109/JNS3.2013.6595461

[S438] Pais, R., Barros, J., Gomes, L.: From SysML State Machines to Petri Nets Using ATL Transformations. In: Proceedings of Technological Innovation for Collective Awareness Systems - 5th IFIP WG 5.5/SOCOLNET Doctoral Conference on Computing, Electrical and Industrial Systems, DoCEIS 2014, Costa de Caparica, Portugal, April 7–9, *IFIP Advances in Information and Communication Technology*, vol. 423, pp. 227–236. Springer (2014). https://doi.org/10.1007/978-3-642-54734-8_26

[S439] Palachi, E., Cohen, C., Takashi, S.: Simulation of cyber physical models using SysML and numerical solvers. In: Proceedings of the IEEE International Systems Conference, SysCon 2013, Orlando, FL, USA, April 15–18, pp. 671–675. IEEE (2013). https://doi.org/10.1109/SysCon.2013.6549954

[S440] Pantsar-Syväniemi, S., Ovaska, E.: Model based architecting with MARTE and SysML profiles. In: Proceedings of the IASTED International Conference on Software Engineering (SE 2010), pp. 1–8 (2010)

[S441] Papakonstantinou, N., Sierla, S.: Generating an Object Oriented IEC 61131-3 software product line architecture from SysML. In: C. Seatzu (ed.) Proceedings of 2013 IEEE 18th Conference on Emerging Technologies & Factory Automation, ETFA 2013, Cagliari, Italy, September 10–13, pp. 1–8. IEEE (2013). https://doi.org/10.1109/ETFA.2013.6648057

[S442] Paredis, C., Bernard, Y., Burkhart, R., de Koning, H.P., Friedenthal, S., Fritzson, P., Rouquette, N., Schamai, W.: An overview of the SysML-Modelica transformation specification. In: Proceedings of the 20th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2010), *INCOSE International Symposium*, vol. 20, pp. 709–722 (2010)

[S443] Paredis, C., Johnson, T.: Using OMG'S SysML to support simulation. In: Proceedings of the 2008 Winter Simulation Conference, Global Gateway to Discovery (WSC 2008), InterContinental Hotel, Miami, Florida, USA, December 7–10, pp. 2350–2352. WSC (2008). https://doi.org/10.1109/WSC.2008.4736341

[S444] Patel, V., McGregor, J., Goasguen, S.: SysML-based domain-specific executable workflows. In: Proceedings of the IEEE International Systems Conference (SysCon 2010), pp. 505–509 (2010). https://doi.org/10.1109/SYSTEMS.2010.5482335

[S445] Patwari, P., Chaudhuri, S.R., Banerjee, A., Natarajan, S., Pandey, S.: A complementary domain specific design environment aiding SysML. In: Proceedings of the 2016 IEEE International Symposium on Systems Engineering (ISSE), Oct., pp. 1–8 (2016). https://doi.org/10.1109/SysEng.2016.7753164

[S446] Peak, R., Burkhart, R., Friedenthal, S., Wilson, M., Bajaj, M., Kim, I.: Simulation-based design using SysML part 1: A parametrics primer. In: Proceedings of the 17th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2007), *INCOSE International Symposium*, vol. 17, pp. 1516–1535 (2007)

[S447] Peak, R., Burkhart, R., Friedenthal, S., Wilson, M., Bajaj, M., Kim, I.: Simulation-based design using SysML part 2: Celebrating diversity by example. In: Proceedings of the 17th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2007), *INCOSE International Symposium*, vol. 17, pp. 1536–1557 (2007)

[S448] Pedroza, G., Apvrille, L., Knorreck, D.: AVATAR: A SysML environment for the formal verification of safety and security properties. In: Proceedings of the 11th Annual International Conference on New Technologies of Distributed Systems (NOTERE 2011), pp. 1–10. IEEE (2011). https://doi.org/10.1109/NOTERE.2011.5957992

[S449] Petrinca, P., Gammaldi, M., Tirone, L.: A SysML-based approach for the specification of complex systems. In: Proceedings of the 22nd Annual International Symposium of the International Council on Systems Engineering (INCOSE 2012) and the 8th Biennial European Systems Engineering Conference (EuSEC 2012), vol. 2, pp. 817–830 (2012)

[S450] Peukert, A.: S/C behavior modeling using SysML for model-based systems engineering support. In: Proceedings of the AIAA Space 2008 Conference & Exposition, San Diego, California, USA (2008)

[S451] Phaoharuhansa, D., Shimada, A.: An approach to SysML and Simulink based motion controller design for inverted pendulum robots. In: Proceedings of the SICE Annual Conference, pp. 2190–2193. IEEE (2011)

[S452] Piétrac, L., Lelevé, A., Henry, S.: On the use of SysML for manufacturing execution system design. In: Proceedings of the 16th Conference on Emerging Technologies & Factory Automation (ETFA 2011), Toulouse, France, September 5–9, pp. 1–8. IEEE (2011). https://doi.org/10.1109/ETFA.2011.6058984

[S453] Pihlanko, P., Sierla, S., Thramboulidis, K., Viitasalo, M.: An industrial evaluation of SysML: The case of a nuclear automation modernization project. In: Proceedings of 2013 IEEE 18th Conference on Emerging Technologies & Factory Automation, ETFA 2013, Cagliari, Italy, September 10–13, pp. 1–8. IEEE (2013). https://doi.org/10.1109/ETFA.2013.6647945

[S454] Pires, A.F., Duprat, S., Besseyre, C.: Approche UML/SysML pour la spécification logicielle de systèmes embarqués aéronautiques. Travaux et retours d'expérience. Technique et Science Informatiques **31**(7), 897–916 (2012). https://doi.org/10.3166/tsi.31.897-916

[S455] Polit-Casillas, R., Howe, A.S.: Virtual construction of space habitats: Connecting Building Information Models (BIM) and SysML. In: Proceedings of the AIAA SPACE Conference and Exposition, San Diego, CA, United States, September 10–12 (2013)

[S456] Prevostini, M., Ganesan, S.: Bridging the Gap Between SysML and Design Space Exploration. In: Proceedings of the Forum on specification and Design Languages (FDL 2006), Darmstadt, Germany, September 19–22, pp. 389–395. ECSI (2006)

[S457] Qamar, A., During, C., Wikander, J.: Designing mechatronic systems, a model-based perspective, an attempt to achieve SysML-Matlab/Simulink model integration. In: Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2009), pp. 1306–1311 (2009). https://doi.org/10.1109/AIM.2009.5229869

[S458] Qiao, D., Liu, X., Li, H.: Research on SysML-based modeling for production management system. In: Proceedings of the 3rd International Conference on Advanced Engineering Materials and Technology (AEMT 2013), *Advanced Materials Research*, vol. 753–755, pp. 1868–1874 (2013). https://doi.org/10.4028/www.scientific.net/AMR.753-755.1868

[S459] Rabelo, L., Clark, T.: Modeling Space Operations Systems Using SysML as to Enable Anomaly Detection. SAE International Journal of Aerospace **8**(2) (2015). https://doi.org/10.4271/2015-01-2388

[S460] Raher, A., Pangaro, P.: Cybernetics and SysML: First steps. In: Proceedings of the 22nd Annual International Symposium of the International Council on Systems Engineering (INCOSE 2012) and the 8th Biennial European Systems Engineering Conference (EuSEC 2012), *INCOSE International Symposium*, vol. 22, pp. 1846–1855 (2012)

[S461] Rahim, M., Ahmed, H., Ioualalen, M.: Modular and Distributed Verification of SysML Activity Diagrams. In: Proceedings of the 1st International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2013), Barcelona, Spain, February 19–21, pp. 202–205. SciTePress (2013). https://doi.org/10.5220/0004320602020205

[S462] Rahim, M., Boukala-Ioualalen, M., Ahmed, H.: Petri Nets Based Approach for Modular Verification of SysML Requirements on Activity Diagrams. In: Proceedings of the International Workshop on Petri Nets and Software Engineering, co-located with 35th International Conference on Application and Theory of Petri Nets and Concurrency (PetriNets 2014) and 14th International Conference on Application of Concurrency to System Design (ACSD 2014), Tunis, Tunisia, June 23–24, *CEUR Workshop Proceedings*, vol. 1160, pp. 233–248. CEUR-WS.org (2014)

[S463] Rahim, M., Hammad, A., Boukala-Ioualalen, M.: Towards the formal verification of SysML specifications: Translation of activity diagrams into modular Petri nets. In: Proceedings of the 3rd International Conference on Applied Computing and Information Technology and 2nd International Conference on Computational Science and Intelligence, ACIT-CSI, pp. 509–516 (2015). https://doi.org/10.1109/ACIT-CSI.2015.97

[S464] Rahim, M., Hammad, A., Ioualalen, M.: A methodology for verifying SysML requirements using activity diagrams. Innovations in Systems and Software Engineering (ISSE) **13**(1), 19–33 (2017). https://doi.org/10.1007/s11334-016-0281-y

[S465] Rahim, M., Kheldoun, A., Boukala-Ioualalen, M., Hammad, A.: Recursive ECATNets-based approach for formally verifying system modelling language activity diagrams. IET Software **9**(5), 119–128 (2015). https://doi.org/10.1049/iet-sen.2014.0087

[S466] Rahman, M., Mizukawa, M.: Model-based development and simulation for robotic systems with SysML, Simulink and Simscape profiles. International Journal of Advanced Robotic Systems **10** (2013). https://doi.org/10.5772/55533

[S467] Rahman, M., Mizukawa, M.: Modeling and design of mechatronics system with SysML, Simscape and Simulink. In: Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics: Mechatronics for Human Wellbeing, AIM 2013, pp. 1767–1773 (2013). https://doi.org/10.1109/AIM.2013.6584353

[S468] Rahman, M., Mizukawa, M., Phaoharuhansa, D., Shimada, A.: Modelling and Simulation of Robotic Systems using SysML. International Journal of Modelling and Simulation **33**(3), 152–161 (2013). https://doi.org/10.2316/Journal.205.2013.3.205-5797

[S469] Rahman, M., Nor, N., Mizukawa, M.: Evaluation for SysML-based design and analysis models using PCE. In: Proceedings of the IEEE International Conference on Control System, Computing and Engineering (ICCSCE 2012), pp. 339–344. IEEE (2013). https://doi.org/10.1109/ICCSCE.2012.6487167

[S470] Ramirez, S.: SysML model of exoplanet archive functionality and activities. In: Proceedings of SPIE 9911, Modeling, Systems Engineering, and Project Management for Astronomy VII, 991129 (2016). https://doi.org/10.1117/12.2232862

[S471] Ramos, A., Ferreira, J.: SysML: The dialect for model-based systems engineering. In: Systems Engineering: Concepts, Tools and Applications, pp. 63–104. Nova Science Publishers (2016)

[S472] Rao, M.: SysML with ARTiSAN Studio. In: Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC 2005), Dallas, TX, USA, September 21–24, p. 15. IEEE Computer Society (2005). https://doi.org/10.1109/VLHCC.2005.59

[S473] Rao, M., Ramakrishnan, S., Dagli, C.: Modeling and simulation of net centric system of systems using systems modeling language and Colored Petri-nets: A demonstration using the Global Earth Observation System of Systems. Systems Engineering **11**(3), 203–220 (2008). https://doi.org/10.1002/sys.20095

[S474] Raslan, W., Sameh, A.: Accelerating SoC Design Using SysML and SystemC. In: Proceedings of the 5th International Industrial Simulation Conference (ISC 2007), Delft, Netherland, June 11–13, pp. 247–251 (2007)

[S475] Raslan, W., Sameh, A.: Mapping SysML to SystemC. In: Proceedings of the Forum on specification and Design Languages (FDL 2007), Barcelona, Spain, September 18–20, pp. 225–230. ECSI (2007)

[S476] Raslan, W., Sameh, A.: System-level modeling and design using SysML and SystemC. In: Proceedings of the International Symposium on Integrated Circuits, ISIC, pp. 504–507. IEEE (2007). https://doi.org/10.1109/ISICIR.2007.4441909

[S477] Raymond, C., Prun, D.: Extending MBSE methodology and SysML formalism to integrate human considerations. In: Proceedings of the International Conference on Human-Computer Interaction in Aerospace, HCI-Aero 2016, Paris, France, September 14–16, 2016, pp. 18:1–18:4. ACM (2016). https://doi.org/10.1145/2950112.2951852

[S478] Reichwein, A., Paredis, C., Canedo, A., Witschel, P., Stelzig, P.E., Votintseva, A., Wasgint, R.: Maintaining consistency between system architecture and dynamic system models with SysML4Modelica. In: Proceedings of the 6th International Workshop on Multi-Paradigm Modeling (MPM@MoDELS 2012), Innsbruck, Austria, October 1–5, pp. 43–48. ACM (2012). https://doi.org/10.1145/2508443.2508451

[S479] Ribeiro, F., Pereira, C., Rettberg, A., Soares, M.: Model-based requirements specification of real-time systems with UML, SysML and MARTE. Software & Systems Modeling pp. 1–19 (2016). https://doi.org/10.1007/s10270-016-0525-1

[S480] Ribeiro, F.G.C., Misra, S., Soares, M.S.: Application of an Extended SysML Requirements Diagram to Model Real-Time Control Systems. In: Proceedings of the 13th International Conference on Computational Science and Its Applications (ICCSA 2013), Ho Chi Minh City, Vietnam, June 24–27, Part III, *Lecture Notes in Computer Science*, vol. 7973, pp. 70–81. Springer (2013). https://doi.org/10.1007/978-3-642-39646-5_6

[S481] Ribeiro, F.G.C., Soares, M.S.: An Approach for Modeling Real-time Requirements with SysML and MARTE Stereotypes. In: Proceedings of the 15th International Conference on Enterprise Information Systems (ICEIS 2013), Volume 2, Angers, France, July 4–7, pp. 70–81. SciTePress (2013). https://doi.org/10.5220/0004449800700081

[S482] Ribeiro, Q.A.D.S., Ribeiro, F.G.C., Soares, M.S.: A Technique to Architect Real-time Embedded Systems with SysML and UML through Multiple Views. In: S. Hammoudi, M. Smialek, O. Camp, J. Filipe (eds.) ICEIS 2017 - Proceedings of the 19th International Conference on Enterprise Information

Systems, Volume 2, Porto, Portugal, April 26–29, 2017, pp. 287–294. SciTePress (2017). https://doi.org/10.5220/0006294802870294

[S483] Riccobene, E., Scandurra, P.: Integrating the SysML and the SystemC-UML profiles in a model-driven embedded system design flow. Design Automation for Embedded Systems **16**(3), 53–91 (2012). https://doi.org/10.1007/s10617-012-9097-7

[S484] Richards, D., Stuart, A., Hause, M.: Testing solutions through SysML/UML. In: Proceedings of the 19th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2009), *INCOSE International Symposium*, vol. 19, pp. 760–774 (2009)

[S485] Robol, M., Salnitri, M., Giorgini, P.: Toward GDPR-Compliant Socio-Technical Systems: Modeling Language and Reasoning Framework. In: G. Poels, F. Gailly, E.S. Asensio, M. Snoeck (eds.) The Practice of Enterprise Modeling - 10th IFIP WG 8.1. Working Conference, PoEM 2017, Leuven, Belgium, November 22–24, *Lecture Notes in Business Information Processing*, vol. 305, pp. 236–250. Springer (2017). https://doi.org/10.1007/978-3-319-70241-4_16

[S486] Romaniw, Y., Bras, B., Guldberg, T.: Sustainable Manufacturing Analysis Using Activity Based Costing in SysML. In: Proceedings of the ASME Design Engineering Technical Conferences and Computers and Information in Engineering Conference, vol. 2, pp. 1019–1028 (2011). https://doi.org/10.1115/DETC2011-48867

[S487] Roth, N.: An architectural assessment of bitcoin: Using the systems modeling language. In: Proceedings of the Conference on Systems Engineering Research, *Procedia Computer Science*, vol. 44, pp. 527–536 (2015). https://doi.org/10.1016/j.procs.2015.03.066

[S488] Roudier, Y., Apvrille, L.: SysML-Sec - A Model Driven Approach for Designing Safe and Secure Systems. In: Proceedings of the 3rd International Conference on Model-Driven Engineering and Software Development, MODELSWARD 2015, ESEO, Angers, Loire Valley, France, 9–11 February, pp. 655–664. SciTePress (2015). https://doi.org/10.5220/0005402006550664

[S489] Ruin, T., Levrat, E., Iung, B.: Modeling Framework based on SysML and AltaRica Data Flow languages for developing models to support complex maintenance program quantification. In: Proceedings of the 2nd IFAC Workshop on Advanced Maintenance Engineering, Services and Technology, *IFAC Proceedings Volumes (IFAC-PapersOnline)*, vol. 45, pp. 169–174 (2012). https://doi.org/10.3182/20121122-2-ES-4026.00018

[S490] Ruin, T., Levrat, E., Iung, B., Despujols, A.: Using SysML language for maintenance decision-making model development to support complex maintenance program quantification. In: Proceedings of the 11th International Probabilistic Safety Assessment and Management Conference and the Annual European Safety and Reliability Conference (PSAM11 ESREL 2012), vol. 2, pp. 1485–1494 (2012)

[S491] Sabetzadeh, M., Nejati, S., Briand, L.C., Mills, A.E.: Using SysML for Modeling of Safety-Critical Software-Hardware Interfaces: Guidelines and Industry Experience. In: Proceedings of the 13th IEEE International Symposium on High-Assurance Systems Engineering (HASE 2011), Boca Raton, FL, USA, November 10–12, pp. 193–201. IEEE Computer Society (2011). https://doi.org/10.1109/HASE.2011.23

[S492] Sadovykh, A., Bagnato, A., Quadri, I., Mady, A.E.D., Couto, L.D., Basagiannis, S., Hasanagic, M.: SysML As a Common Integration Platform for Co-Simulations: Example of a Cyber Physical System Design Methodology in Green Heating Ventilation and Air Conditioning Systems. In: Proceedings of the 12th Central and Eastern European Software Engineering Conference in Russia, CEE-SECR ’16, pp. 1:1–1:5. ACM, New York, NY, USA (2016). https://doi.org/10.1145/3022211.3022212

[S493] Sakairi, T., Palachi, E., Cohen, C., Hatsutori, Y., Shimizu, J., Miyashita, H.: Designing a control system using SysML and Simulink. In: Proceedings of the SICE Annual Conference, pp. 2011–2017. IEEE (2012)

[S494] Sakairi, T., Palachi, E., Cohen, C., Hatsutori, Y., Shimizu, J., Miyashita, H.: Model Based Control System Design Using SysML, Simulink, and Computer Algebra System. Journal of Control Science and Engineering **2013** (2013). https://doi.org/10.1155/2013/485380

[S495] Scanniello, G., Staron, M., Burden, H., Heldal, R.: On the effect of using SysML requirement diagrams to comprehend requirements: Results from two controlled experiments. In: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, EASE ’14, London, England, United Kingdom, May 13–14, pp. 49:1–49:10. ACM (2014). https://doi.org/10.1145/2601248.2601259

[S496] Schlecht, S., Alt, O.: Strategien zur Testfallgenerierung aus SysML Modellen. In: Software Engineering 2007 - Beiträge zu den Workshops, Fachtagung des GI-Fachbereichs Softwaretechnik, 27.-

30.3.2007 in Hamburg, *LNI*, vol. 106, pp. 101–106. GI (2007)

[S497] Schönherr, O., Moss, J., Rehm, M., Rose, O.: A free simulator for modeling production systems with SysML. In: Proceedings of the Winter Simulation Conference, pp. 1–12 (2012). https://doi.org/10.1109/WSC.2012.6465090

[S498] Schönherr, O., Pappert, F., Rose, O.: Domain specific simulation modeling with SysML and model-to-model transformation for discrete processes. In: Formal Languages for Computer Simulation: Transdisciplinary Models and Applications, pp. 267–304. IGI Global (2013). https://doi.org/10.4018/978-1-4666-4369-7.ch009

[S499] Schönherr, O., Rose, O.: First Steps Towards a General SysML Model for Discrete Processes in Production Systems. In: Proceedings of the 2009 Winter Simulation Conference (WSC 2009), Hilton Austin Hotel, Austin, TX, USA, December 13–16, pp. 1711–1718. WSC (2009). https://doi.org/10.1109/WSC.2009.5429164

[S500] Schönherr, O., Rose, O.: Important Components for Modeling Production Systems with SysML. In: Proceedings of the IIE Annual Conference and Expo, pp. 1–6 (2010)

[S501] Schütz, D., Aicher, T., Vogel-Heuser, B.: Automatic generation of shop floor gateway configurations from systems modeling language. In: Proceedings of the IEEE International Systems Engineering Symposium (ISSE), pp. 1–8 (2017). https://doi.org/10.1109/SysEng.2017.8088288

[S502] Schütz, D., Legat, C., Vogel-Heuser, B.: MDE of manufacturing automation software - Integrating SysML and standard development tools. In: Proceedings of the 12th IEEE International Conference on Industrial Informatics, INDIN 2014, Porto Alegre, RS, Brazil, July 27–30, pp. 267–273. IEEE (2014). https://doi.org/10.1109/INDIN.2014.6945519

[S503] Schütz, D., Obermeier, M., Vogel-Heuser, B.: SysML-Based Approach for Automation Software Development - Explorative Usability Evaluation of the Provided Notation. In: Proceedings of the 2nd International Conference on Design, User Experience, and Usability. Web, Mobile, and Product Design, DUXU 2013, Held as Part of HCI International 2013, Las Vegas, NV, USA, July 21–26, Part IV, *Lecture Notes in Computer Science*, vol. 8015, pp. 568–574. Springer (2013). https://doi.org/10.1007/978-3-642-39253-5_63

[S504] Scippacercola, F., Pietrantuono, R., Russo, S., Silva, N.: SysML-based and Prolog-supported FMEA. In: Proceedings of the IEEE International Symposium on Software Reliability Engineering Workshops, ISSRE Workshops, Gaithersburg, MD, USA, November 2–5, pp. 174–181. IEEE Computer Society (2015). https://doi.org/10.1109/ISSREW.2015.7392064

[S505] Seal, D.: Promise and progress of SysML and merit functions as applied to systems engineering at JPL. In: Proceedings of the 61st International Astronautical Congress (IAC 2010), vol. 8, pp. 6226–6232 (2010)

[S506] Seki, K., Muraoka, Y., Nishimura, H.: System level thermal design-process modeling for functional/structure design using SysML and MDM. In: Proceedings of the 17th International DSM Conference, Modeling and Managing Complex Systems, pp. 149–159. Carl Hanser Verlag (2015)

[S507] Selvy, B., Claver, C., Angeli, G.: Using SysML for verification and validation planning on the Large Synoptic Survey Telescope (LSST). In: Modeling, Systems Engineering, and Project Management for Astronomy VI, Montréal, Quebec, Canada, June 22, *Proceedings of SPIE*, vol. 9150, pp. 91,500N–91,500N–13 (2014). https://doi.org/10.1117/12.2056773

[S508] Sena Marques, M., Siegert, E., Brisolara, L.: Integrating UML, MARTE and SysML to improve requirements specification and traceability in the embedded domain. In: Proceedings of the 12th IEEE International Conference on Industrial Informatics, INDIN 2014, Porto Alegre, RS, Brazil, July 27–30, pp. 176–181. IEEE (2014). https://doi.org/10.1109/INDIN.2014.6945504

[S509] Sha, Z., Le, Q., Panchal, J.: Using SysML for Conceptual Representation of Agent-Based Models. In: Proceedings of the ASME Design Engineering Technical Conference and Computers and Information in Engineering Conference, Washington, DC, USA, August 28–31, vol. 2, pp. 39–50 (2011). https://doi.org/10.1115/DETC2011-47476

[S510] Shah, A., Kerzhner, A., Schaefer, D., Paredis, C.: Multi-view Modeling to Support Embedded Systems Engineering in SysML. In: Graph Transformations and Model-Driven Engineering - Essays Dedicated to Manfred Nagl on the Occasion of his 65th Birthday, *Lecture Notes in Computer Science*, vol. 5765, pp. 580–601. Springer (2010). https://doi.org/10.1007/978-3-642-17322-6_25

[S511] Shah, A., Paredis, C., Burkhart, R., Schaefer, D.: Combining Mathematical Programming and SysML for Component Sizing of Hydraulic Systems. In: Proceedings of the ASME Design Engineering Technical Conference and Computers and Information in Engineering Conference (IDETC/CIE2010), August

15–18, vol. 3, pp. 1231–1245 (2010). https://doi.org/10.1115/DETC2010-28960

[S512] Shah, A., Paredis, C., Burkhart, R., Schaefer, D.: Combining Mathematical Programming and SysML for Automated Component Sizing of Hydraulic Systems. Journal of Computing and Information Science in Engineering **12**(4), 041,006–1 – 041,006–14 (2012). https://doi.org/10.1115/1.4007764

[S513] Shames, P., Sarrel, M., Friedenthal, S.: Modeling systems-of-systems interfaces with SysML. In: Proceedings of the 14th International Conference on Space Operations (SpaceOps), Daejon, Korea, 16–20 May (2016)

[S514] Shinozaki, M., Mhenni, F., Choley, J., Ming, A.: Reuse of SysML model to support innovation in mechatronic systems design. In: Proceedings of the Annual IEEE International Systems Conference, SysCon 2017, Montreal, QC, Canada, April 24–27, 2017, pp. 1–6. IEEE (2017). https://doi.org/10.1109/SYSCON.2017.7934709

[S515] Siebold, U., Häring, I.: Semi-formal safety requirement specification using SysML state machine Diagrams. In: Proceedings of the 11th International Probabilistic Safety Assessment and Management Conference and the Annual European Safety and Reliability Conference (PSAM11 ESREL 2012), Helsinki, Finland, June 25–29, vol. 3, pp. 2102–2111 (2012)

[S516] Silva, R.F., Fragal, V.H., de Oliveira Junior, E.A., de Souza Gimenes, I.M., Oquendo, F.: SyMPLES - A SysML-based Approach for Developing Embedded Systems Software Product Lines. In: Proceedings of the 15th International Conference on Enterprise Information Systems (ICEIS 2013), Volume 2, Angers, France, July 4–7, pp. 257–264. SciTePress (2013). https://doi.org/10.5220/0004446802570264

[S517] Sindico, A., Natale, M.D., Panci, G.: Integrating SysML with Simulink using Open-source Model Transformations. In: Proceedings of 1st International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2011), Noordwijkerhout, The Netherlands, July 29 - 31, pp. 45–56. SciTePress (2011)

[S518] Sjöstedt, C., Chen, D., Cuenot, P., Frey, P., Johansson, R., Lönn, H., Servat, D., Törngren, M.: Developing Dependable Automotive Embedded Systems using the EAST-ADL; representing continuous time systems in SysML. In: Proceedings of the 1st International Workshop on Equation-Based Object-Oriented Languages and Tools (EOOLT 2007), Berlin, Germany, July 30, *Linköping Electronic Conference Proceedings*, vol. 24, pp. 25–36. Linköping University Electronic Press (2007)

[S519] Soares, M., Vrancken, J.: Requirements specification and modeling through SysML. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics 2007, Montréal, Canada, October 7–10 October, pp. 1735–1740. IEEE (2007). https://doi.org/10.1109/ICSMC.2007.4413936

[S520] Soares, M., Vrancken, J.: A Proposed Extension to the SysML Requirements Diagram. In: Proceedings of the IASTED International Conference on Software Engineering (SE 2008), Innsbruck, Austria, pp. 220–225. ACTA Press (2008)

[S521] Soares, M., Vrancken, J.: Model-driven user requirements specification using SysML. Journal of Software **3**(6), 57–68 (2008)

[S522] Soares, M.S., do Nascimento, R.P.C.: Evaluation of SysML diagrams to document requirements using TAM. In: Proceedings of the 7th Euro American Conference on Telematics and Information Systems, EATIS'14, Valparaiso, Chile, April 02 - 04, pp. 9:1–9:6. ACM (2014). https://doi.org/10.1145/2590651.2590661

[S523] Spyropoulos, D., Baras, J.S.: Extending Design Capabilities of SysML with Trade-off Analysis: Electrical Microgrid Case Study. In: Proceedings of the Conference on Systems Engineering Research, CSER 2013, Atlanta, Georgia, USA, March 19–22, *Procedia Computer Science*, vol. 16, pp. 108–117. Elsevier (2013). https://doi.org/10.1016/j.procs.2013.01.012

[S524] Stancescu, S., Neagoe, L., Marinescu, R., Enoiu, E.: A SysML model for code correction and detection systems. In: Proceedings of the 33rd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO 2010), pp. 189–191 (2010)

[S525] Steimer, C., Fischer, J., Aurich, J.C.: Model-based Design Process for the Early Phases of Manufacturing System Planning using SysML. In: Complex Systems Engineering and Development Proceedings of the 27th CIRP Design Conference Cranfield University, UK 10th – 12th May, *Procedia CIRP*, vol. 60, pp. 163–168 (2017)

[S526] Tadano, K., Xiang, J., Kawato, M., Maeno, Y.: Synthesizing SRN Models from System Operations with SysML Diagrams for Availability Analysis. In: Proceedings of the 5th International Conference on Secure Software Integration and Reliability Improvement (SSIRI 2011), Jeju Island, Korea, June 27–29, pp. 5–6. IEEE Computer Society (2011). https://doi.org/10.1109/SSIRI-C.2011.44

[S527] Tamura, Y., Nishigaki, H., Miyoshi, K., Huang, H., Kawata, S.: A proposal of home continuity plan service system, modeling by SysML and validating by discrete event simulation. In: Proceedings of the SICE Annual Conference, pp. 137–144. IEEE (2012)

[S528] Tarawneh, H.: SysML-based web engineering a successful way to design web applications. In: Proceedings of the 1st International Conference on Software and Data Technologies (ICSOFT 2006), Setúbal, Portugal, September 11–14, pp. 269–272. INSTICC Press (2006)

[S529] Tenbergen, B., Bohn, P., Weyer, T.: Ein strukturierter Ansatz zur Ableitung methodenspezifischer UML/SysML-Profile am Beispiel des SPES 2020 Requirements Viewpoints. In: Software Engineering 2013 - Workshopband (inkl. Doktorandensymposium), Fachtagung des GI-Fachbereichs Softwaretechnik, 26. Februar - 1. März 2013 in Aachen, *LNI*, vol. 215, pp. 235–244. GI (2013)

[S530] Thom, F.: Modelling distributed integrated modular systems using the UML™ and the SysML™. In: Proceedings of DASIA 2005 : Data Systems In Aerospace, Edinburgh, Scotland, 30 May-2 June, no. 602 in ESA SP, pp. 55–62. ESA Publications Division (2005)

[S531] Thoma, A., Kormann, B., Vogel-Heuser, B.: Fault-centric system modeling using SysML for reliability testing. In: Proceedings of 17th International Conference on Emerging Technologies & Factory Automation, ETFA 2012, Krakow, Poland, September 17–21, pp. 1–8. IEEE (2012). https://doi.org/10.1109/ETFA.2012.6489543

[S532] Thramboulidis, K.: The 3+1 SysML View-Model in Model Integrated Mechatronics. JSEA **3**(2), 109–118 (2010). https://doi.org/10.4236/jsea.2010.32014

[S533] Thramboulidis, K., Buda, A.: 3+1 SysML view model for IEC61499 Function Block control systems. In: Proceedings of the 8th IEEE International Conference on Industrial Informatics (INDIN), pp. 175–180. IEEE (2010). https://doi.org/10.1109/INDIN.2010.5549440

[S534] Thramboulidis, K., Scholz, S.: Integrating the 3+1 SysML view model with safety engineering. In: Proceedings of 15th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2010), Bilbao, Spain, September 13–16, pp. 1–8. IEEE (2010). https://doi.org/10.1109/ETFA.2010.5641353

[S535] Trancho, G., Wang, L., Herzig, S., Karban, R., Boyer, C., Herriot, G., Anderson, D., Ellerbroek, B.: Analyzing the operational behavior of NFIRAOS LGS MCAO acquisition on the thirty meter telescope using SysML. In: 5th Adaptive Optics for Extremely Large Telescopes, 2017 AO4ELT5, vol. 2017-June, pp. 1–9 (2017). https://doi.org/10.26698/AO4ELT5.0168

[S536] Trapp, S., Ramollari, E., Heintz, M., Weber, S., Dranidis, D., Börstler, J.: Collaborative Learning of UML and SysML. iJEP **1**(2), 6–12 (2011)

[S537] Tsadimas, A.: Model-based enterprise information system architectural design with SysML. In: Proceedings of the 9th IEEE International Conference on Research Challenges in Information Science, RCIS 2015, Athens, Greece, May 13–15, pp. 492–497. IEEE (2015). https://doi.org/10.1109/RCIS.2015.7128911

[S538] Tsadimas, A., Kapos, G.D., Dalakas, V., Nikolaidou, M., Anagnostopoulos, D.: Integrating simulation capabilities into SysML for enterprise information system design. In: Proceedings of the 9th International Conference on System of Systems Engineering, SoSE 2014, Glenelg, Australia, June 9–13, pp. 272–277. IEEE (2014)

[S539] Tsadimas, A., Kapos, G.D., Dalakas, V., Nikolaidou, M., Anagnostopoulos, D.: Simulating simulation-agnostic SysML models for enterprise information systems via DEVS. Simulation Modelling Practice and Theory **66**, 243–259 (2016). https://doi.org/10.1016/j.simpat.2016.04.001

[S540] Tsadimas, A., Nikolaidou, M., Anagnostopoulos, D.: Extending SysML to explore non-functional requirements: the case of information system design. In: Proceedings of the ACM Symposium on Applied Computing (SAC 2012), Riva, Trento, Italy, March 26–30, pp. 1057–1062. ACM (2012). https://doi.org/10.1145/2245276.2231941

[S541] Tsadimas, A., Nikolaidou, M., Anagnostopoulos, D.: Model-Based system design using SysML: The role of the evaluation diagram. In: Formal Languages for Computer Simulation: Transdisciplinary Models and Applications, pp. 236–266. IGI Global (2013). https://doi.org/10.4018/978-1-4666-4369-7.ch008

[S542] Tueno, S., Laleau, R., Mammar, A., Frappier, M.: Towards Using Ontologies for Domain Modeling within the SysML/KAOS Approach. In: Proceedings of the IEEE 25th International Requirements Engineering Conference Workshops, RE 2017 Workshops, Lisbon, Portugal, September 4–8, pp. 1–5. IEEE Computer Society (2017). https://doi.org/10.1109/REW.2017.22

[S543] Turki, S., Soriano, T., Sghaier, A.: Mechatronic systems modeling with SysML: A bond graphs addendum for energy analysis. WSEAS Transactions on Systems **4**(5), 617–624 (2005)

[S544] Väljaots, E., Sell, R.: Unmanned ground vehicle SysML navigation model conducted by energy efficiency. In: Proceedings of the 3rd International Conference on Applied Materials and Electronics Engineering, *Advanced Materials Research*, vol.

905, pp. 443–447 (2014). https://doi.org/10.4028/www.scientific.net/AMR.905.443
[S545] Valles-Barajas, F.: A Formal Model for the Requirements Diagrams of SysML. IEEE Latin America Transactions **8**(3), 259–268 (2010). https://doi.org/10.1109/TLA.2010.5538400
[S546] Vanderperren, Y., Dehaene, W.: The SysML profile for embedded system modelling. In: Proceedings of the Forum on specification and Design Languages (FDL 2005), Lausanne, Switzerland, September 27–30, pp. 589–598. ECSI (2005)
[S547] Vanderperren, Y., Dehaene, W.: UML 2 and SysML: An Approach to Deal with Complexity in SoC/NoC Design. In: Proceedings of the Design, Automation and Test in Europe Conference and Exposition (DATE 2005), Munich, Germany, March 7–11, pp. 716–717. IEEE Computer Society (2005). https://doi.org/10.1109/DATE.2005.319
[S548] Vernoosfaderani, M.: Generic satellite model libraries: Rapid Move to SysML. In: Proceedings of the 68th International Astronautical Congress: Unlocking Imagination, Fostering Innovation and Strengthening Security (IAC 2017), vol. 3, pp. 1373–1377 (2017)
[S549] Verries, J., Sahraoui, A.: Case study on SysML and VHDL-AMS for designing and validating systems. In: Proceedings of the World Congress on Engineering & Computer Science (WCECS 2013), London, UK, July 3 - 5, *Lecture Notes in Engineering and Computer Science*, vol. 1, pp. 90–95 (2013)
[S550] Viehl, A., Bringmann, O., Rosenstiel, W.: Virtual Prototyping und frühe Evaluierung von Systems-on-Chip mit UML2 und SysML. In: Methoden und Beschreibungssprachen zur Modellierung und Verifikation von Schaltungen und Systemen (MBMV 2006), Dresden, Germany, February 20–22, pp. 266–270. Fraunhofer Institut für Integrierte Schaltungen (2006)
[S551] Viehl, A., Schönwald, T., Bringmann, O., Rosenstiel, W.: Formal performance analysis and simulation of UML/SysML models for ESL design. In: Proceedings of the Conference on Design, Automation and Test in Europe (DATE 2006), Munich, Germany, March 6–10, pp. 242–247. European Design and Automation Association, Leuven, Belgium (2006). https://doi.org/10.1109/DATE.2006.244110
[S552] Vogel-Heuser, B., Schütz, D., Frank, T., Legat, C.: Model-driven engineering of Manufacturing Automation Software Projects - A SysML-based approach. Mechatronics **24**(7), 883–897 (2014). https://doi.org/10.1016/j.mechatronics.2014.05.003
[S553] Wagner, D., Bennett, M., Karban, R., Rouquette, N., Jenkins, S., Ingham, M.: An ontology for state analysis: Formalizing the mapping to SysML. In: Proceedings of the IEEE Aerospace Conference, pp. 1–16. IEEE (2012). https://doi.org/10.1109/AERO.2012.6187335
[S554] Walker, L., Thomas, D.: Integrated System Modeling in SysML for Small Satellites. In: Proceedings of the AIAA Modeling and Simulation Technologies Conference, Grapevine, Texas (2017)
[S555] Wan, W., Cheong, H., Li, W., Zeng, Y., Iorio, F.: Automated transformation of design text ROM diagram into SysML models. Advanced Engineering Informatics **30**(3), 585–603 (2016). https://doi.org/10.1016/j.aei.2016.07.003
[S556] Wang, R., Dagli, C.: An executable system architecture approach to discrete events system modeling using SysML in conjunction with Colored Petri Net. In: Proceedings of the 2nd Annual IEEE International Systems Conference (SysCon 2008), pp. 118–125 (2008). https://doi.org/10.1109/SYSTEMS.2008.4518997
[S557] Wang, R., Dagli, C.: Executable system architecting using systems modeling language in conjunction with colored Petri nets in a model-driven systems development process. Systems Engineering **14**(4), 383–409 (2011). https://doi.org/10.1002/sys.20184
[S558] Wang, W., Liu, X.: Research on MDS method based on DEVS and SysML. In: Proceedings of the International Conference on System Science, Engineering Design and Manufacturing Informatization (ICSEM 2011), vol. 1, pp. 180–182. IEEE (2011). https://doi.org/10.1109/ICSSEM.2011.6081177
[S559] Wang, W., Zhao, B.: Research on construction method of DoDAF view based on DEVS and SysML. In: Proceedings of the World Automation Congress, pp. 1–4. IEEE (2012)
[S560] Warniez, A., Penas, O., Plateaux, R., Soriano, T.: SysML geometrical profile for integration of mechatronic systems. In: Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM, pp. 709–714 (2014). https://doi.org/10.1109/AIM.2014.6878162
[S561] Watson, M.E., Rusnock, C.F., Colombi, J.M., Miller, M.E.: Human-centered design using system modeling language. Journal of Cognitive Engineering and Decision Making **11**(3), 252–269 (2017). https://doi.org/10.1177/1555343417705255
[S562] Wei, Q., Jiao, J., Zhou, S., Zhao, T.: Research on accident process meta-modeling based on SysML. In: Proceedings of the 1st International Conference on Reliability Systems Engineering (ICRSE), Oct., pp. 1–6 (2015). https://doi.org/10.1109/ICRSE.2015.7366487

[S563] Wells, W., Karwowski, W., Sala-Diakanda, S., Williams, K., Ahram, T., Pharmer, J.: Application of Systems Modeling Language (SysML) for Cognitive Work Analysis in Systems Engineering Design Process. Journal of Universal Computer Science **17**(9), 1261–1280 (2011). https://doi.org/10.3217/jucs-017-09-1261

[S564] Weyprecht, P., Rose, O.: Model-driven development of simulation solution based on SysML starting with the simulation core. In: Proceedings of the Spring Simulation Multi-conference, SpringSim '11, Boston, MA, USA, April 03–07, 2011. Volume 4: Proceedings of the 2011 Symposium on Theory of Modeling & Simulation: DEVS Integrative M&S Symposium (TMS-DEVS)., pp. 189–192. SCS/ACM (2011)

[S565] Wölkl, S., Shea, K.: A Computational Product Model for Conceptual Design Using SysML. In: Proceedings of the ASME Design Engineering Technical Conference and Computers and Information in Engineering Conference (DETC 2009), vol. 2, pp. 635–645 (2009). https://doi.org/10.1115/DETC2009-87239

[S566] Wrycza, S., Marcinkowski, B.: SysML Parametric Diagrams in Business Applications. In: Information Systems Development, Business Systems and Services: Modeling and Development [Proceedings of ISD 2010, Charles University in Prague, Czech Republic, August 25–27], pp. 791–798. Springer (2010). https://doi.org/10.1007/978-1-4419-9790-6_64

[S567] Wrycza, S., Marcinkowski, B.: SysML Requirement Diagrams: Banking Transactional Platform Case Study. In: Research in Systems Analysis and Design: Models and Methods - 4th SIGSAND/PLAIS EuroSymposium 2011, Gdańsk, Poland, September 29, Revised Selected Papers, *Lecture Notes in Business Information Processing*, vol. 93, pp. 15–22. Springer (2011). https://doi.org/10.1007/978-3-642-25676-9_2

[S568] Wrycza, S., Marcinkowski, B.: Systems requirements specification with SysML. In: Proceedings of the 8th International Conference on Perspectives in Business Informatics Research (BIR 2009) (2014)

[S569] Wu, D., Zhang, L., Jiao, R., Lu, R.: SysML-based design chain information modeling for variety management in production reconfiguration. Journal of Intelligent Manufacturing **24**(3), 575–596 (2013). https://doi.org/10.1007/s10845-011-0585-6

[S570] Wu, J., Wang, M.Z., Fang, H.J.: Product design of systems architecture using SysML. Xi Tong Gong Cheng Yu Dian Zi Ji Shu/Systems Engineering and Electronics **28**(4), 594–598 (2006)

[S571] Yamada, T.: A Generic Method for Defining Viewpoints in SysML. In: Proceedings of the 19th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2009), *INCOSE International Symposium*, vol. 19, pp. 844–852 (2009)

[S572] Yerushalmi, R., Felice, R.: Implementing AUTOSAR atomic software components using UML/SysML in C. In: Proceedings of the SAE 2010 World Congress & Exhibition, SAE Technical Papers (2010). https://doi.org/10.4271/2010-01-0265

[S573] Yin, S.Y., Yang, Y., Miao, X.W., Zhao, T.D.: SysML-based safety analysis of thrust reverser. Hangkong Dongli Xuebao/Journal of Aerospace Power **26**(3), 498–504 (2011)

[S574] Yin, Y., Xu, Y., Miao, W., Chen, Y.: An automated test case generation approach based on activity diagrams of SysML. International Journal of Performability Engineering **13**(6), 922–936 (2017). https://doi.org/10.23940/ijpe.17.06.p13.922936

[S575] Yue, G., Rao, A., Jones, R.: Architectural and functional modelling of an automotive driver information system using SysML. In: Proceedings of the IEEE/ASME International Conference on Mechatronics and Embedded Systems and Applications (MESA 2008), pp. 552–557. IEEE (2008). https://doi.org/10.1109/MESA.2008.4735654

[S576] Zdanis, L., Cloutier, R.: The use of behavioral diagrams in SysML. In: Proceedings of the IEEE Long Island Systems, Applications and Technology Conference (LISAT 2007), pp. 1–1 (2007). https://doi.org/10.1109/LISAT.2007.4312634

[S577] Zhang, W.Z., Wang, Z.X., Zhu, W.X., Chen, J.: Supporting study on modeling C4ISR systems based on SysML. Nanjing Li Gong Daxue Xuebao/Journal of Nanjing University of Science and Technology **35**(3), 386–391 (2011)

[S578] Zhou, S., Sun, Q., Jiao, J.: A safety modeling method based on SysML. In: Proceedings of the 10th International Conference on Reliability, Maintainability and Safety: More Reliable Products, More Secure Life (ICRMS), pp. 1180–1185 (2014). https://doi.org/10.1109/ICRMS.2014.7107390

[S579] Zingel, C., Albers, A., Matthiesen, S., Maletz, M.: Experiences and Advancements from One Year of Explorative Application of an Integrated Model-Based Development Technique Using C&C^2-A in SysML. IAENG International Journal of Computer Science **39**(2), 165–181 (2012)

B Books and theses (not used)

SysML-Books

[B1] Debbabi, M., Hassaïne, F., Jarraya, Y., Soeanu, A., Alawneh, L.: Verification and Validation in Systems Engineering: Assessing UML/SysML Design Models. Springer (2010). https://doi.org/10.1007/978-3-642-15228-3

[B2] Dori, D.: Model-Based Systems Engineering with OPM and SysML. Springer (2016). https://doi.org/10.1007/978-1-4939-3295-5

[B3] Friedenthal, S., Moore, A., Steiner, R.: A Practical Guide to SysML: Systems Modeling Language. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2008)

[B4] Friedenthal, S., Moore, A., Steiner, R.: A Practical Guide to SysML, second edition edn. Morgan Kaufmann Publishers Inc. (2012). https://doi.org/10.1016/C2010-0-66331-0

[B5] Holt, J., Perry, S.: SysML for Systems Engineering: A Model-Based Approach, second edition edn. Computing and Networks (2014)

[B6] Oquendo, F., Leite, J.C., Batista, T.: Software Architecture in Action - Designing and Executing Architectural Models with SysADL grounded on the OMG SysML Standard. Undergraduate Topics in Computer Science. Springer (2016). https://doi.org/10.1007/978-3-319-44339-3

[B7] Weilkiens, T.: Systems Engineering mit SysML / UML - Modellierung, Analyse, Design. dpunkt.verlag (2006)

[B8] Weilkiens, T.: Systems Engineering with SysML/UML. Morgan Kaufmann Publishers Inc. (2007)

SysML-Theses

[T1] Abdulhameed, A.: Combining SysML and SystemC to Simulate and Verify Complex Systems. (Utilisation conjointé de SysML et systemC pour simmuler et vérifier les systèmes complexes). Ph.D. thesis, University of Franche-Comté, Besançon, France (2016)

[T2] Bouaziz, H.: Adaptation of SysML Blocks and Verification of Temporal Properties. (Adptation des Blocs SysML et verification des propriétés temporelles). Ph.D. thesis, University of Franche-Comté, France (2016)

[T3] Fontan, B.: Méthodologie de conception de systèmes temps réel et distribués en contexte UML/SysML. Ph.D. thesis, Paul Sabatier University, Toulouse, France (2008)

[T4] Frank, T.: Entwicklung und Evaluation einer Modellierungssprache für den Architekturentwurf von verteilten Automatisierungsanlagen auf Basis der Systems Modeling Language (SysML). Ph.D. thesis, Technical University Munich (2014)

[T5] Gauthier, J.: Combining Discrete and Continuous Domains for SysML-Based Simulation and Test Generation. (Combinaison des domaines discret et continu pour la simulation et la génération de tests à partir de modèle SysML). Ph.D. thesis, University of Franche-Comté, Besançon, France (2015)

[T6] Huang, C.: Discrete event system modeling using SysML and model transformation. Ph.D. thesis, Georgia Institute of Technology, Atlanta, GA, USA (2011)

[T7] Lasalle, J.: Génération automatique de tests à partir de modèles SysML pour la validation fonctionnelle de systèmes embarqués. (Automatic tests generation from SysML models for the functionnal validation of embedded). Ph.D. thesis, University of Franche-Comté, Besançon, France (2012)

[T8] Rozo, O.C.: Formal and incremental verification of SysML for the design of component-based system. (Vérification formelle et incrémentale de spécifications SysML pour la conception de systèmes à base de composants). Ph.D. thesis, University of Franche-Comté, Besançon, France (2015)

[T9] Schönherr, O.: Modellierung, Simulation und Transformation von diskreten Prozessen in der Produktion und Logistik auf der Basis von SysML. Ph.D. thesis, Bundeswehr University Munich (2014)

[T10] Schütz, D.: Automatische Generierung von Softwareagenten für die industrielle Automatisierungstechnik der Steuerungsebene des Maschinen- und Anlagenbaus auf Basis der Systems Modeling Language. Ph.D. thesis, Technical University Munich (2015)

[T11] Turki, S.: Ingénierie système guidée par les modèles: Application du standard IEEE 15288, de l'architecture MDA et du langage SysML à la conception des systèmes mécatroniques. Ph.D. thesis, University of the South, Toulon-Var, France (2008)

References

1. Ameller, D., Burgués, X., Collell, O., Costal, D., Franch, X., Papazoglou, M.P.: Development of service-oriented architectures using model-driven development: a mapping study. Inf. Softw. Technol. **62**, 42–66 (2015)
2. Assar, S.: Model driven requirements engineering: mapping the field and beyond. In: Proceedings of the 4th International Model-Driven Requirements Engineering Workshop (MoDRE), pp. 1–6. IEEE (2014)
3. Badreddin, O., Abdelzad, V., Lethbridge, T., Elaasar, M.: fSysML: Foundational executable SysML for cyber-physical system mod-

eling. In: Proceedings of the 4th International Workshop on the Globalization Of Modeling Languages co-located with ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems (MODELS 2016), Saint Malo, France, October 4th, CEUR Workshop Proceedings, vol. 1731, pp. 38–51. CEUR-WS.org (2016)
4. Barbieri, G., Kernschmidt, K., Fantuzzi, C., Vogel-Heuser, B.: A SysML based design pattern for the high-level development of mechatronic systems to enhance re-usability. In: Proceedings of the 19th IFAC World Congress, IFAC Proceedings Volumes, vol. 47, pp. 3431–3437 (2014)
5. Barmi, Z.A., Ebrahimi, A.H., Feldt, R.: Alignment of requirements specification and testing: a systematic mapping study. In: Proceedings of the 4th International Conference on Software Testing, Verification and Validation Workshops (ICSTW), pp. 476–485. IEEE (2011)
6. Batarseh, O., McGinnis, L.: SysML to discrete-event simulation to analyze electronic assembly systems. In: Proceedings of the 2012 Symposium on Theory of Modeling and Simulation—DEVS Integrative M&S Symposium, Orlando, FL, USA, March 26–29, pp. 48:1–48:8. SCS/ACM (2012)
7. Baumgart, S., Fröberg, J.: Functional safety in product lines—a systematic mapping study. In: Proceedings of the 42th Euromicro Conference on the Software Engineering and Advanced Applications (SEAA), pp. 313–322. IEEE (2016)
8. Budgen, D., Turner, M., Brereton, P., Kitchenham, B.: Using mapping studies in software engineering. In: Proceedings of PPIG, vol. 8, pp. 195–204. Lancaster University (2008)
9. Budgen, D., Burn, A.J., Brereton, O.P., Kitchenham, B.A., Pretorius, R.: Empirical evidence about the UML: a systematic literature review. Softw. Pract. Exp. **41**(4), 363–392 (2011)
10. Bundesverband Informationswirtschaft, Telekomunikation und neue Medien e.V., Verband Deutscher Maschinen- und Anlagenbau e.V.: Umsetzungsstrategie Industrie 4.0 Ergebnisbericht der Plattform Industrie 4.0. Zentralverband Elektrotechnik und Elektronikindustrie e.V. (2015) (in German)
11. Da Silva, F.Q., Santos, A.L., Soares, S., França, A.C.C., Monteiro, C.V., Maciel, F.F.: Six years of systematic literature reviews in software engineering: an updated tertiary study. Inf. Softw. Technol. **53**(9), 899–913 (2011)
12. David, P., Idasiak, V., Kratz, F.: Reliability study of complex physical systems using SysML. Reliab. Eng. Syst. Saf. **95**(4), 431–450 (2010). https://doi.org/10.1016/j.ress.2009.11.015
13. do Nascimento, L.M., Viana, D.L., Neto, P.A.S., Martins, D.A., Garcia, V.C., Meira, S.R.: A systematic mapping study on domain-specific languages. In: Proceedings of the 7th International Conference Software Engineering Advances (ICSEA'12), pp. 179–187 (2012)
14. Doğan, S., Betin-Can, A., Garousi, V.: Web application testing: a systematic literature review. J. Syst. Softw. **91**, 174–201 (2014)
15. Engström, E., Runeson, P.: Software product line testing—a systematic mapping study. Inf. Softw. Technol. **53**(1), 2–13 (2011)
16. Essaadi, F., Maissa, Y.B., Dahchour, M.: MDE-based languages for wireless sensor networks modeling: a systematic mapping study. In: Advances in Ubiquitous Networking 2, pp. 331–346. Springer (2017)
17. Fernández-Sáez, A.M., Genero, M., Chaudron, M.R.: Empirical studies concerning the maintenance of UML diagrams and their use in the maintenance of code: a systematic mapping study. Inf. Softw. Technol. **55**(7), 1119–1142 (2013)
18. Garousi, V., Shahnewaz, S., Krishnamurthy, D.: UML-driven software performance engineering: a systematic mapping and trend analysis. Progressions and Innovations in Model-Driven Software Engineering (2013)
19. Gauthier, J.M., Bouquet, F., Hammad, A., Peureux, F.: A SysML formal framework to combine discrete and continuous simulation for testing. In: Proceedings of the 17th International Conference on Formal Engineering Methods, ICFEM 2015, Paris, France, November 3–5, Lecture Notes in Computer Science, vol. 9407, pp. 134–152. Springer (2015). https://doi.org/10.1007/978-3-319-25423-4_9
20. Guessi, M., Neto, V.V., Bianchi, T., Felizardo, K.R., Oquendo, F., Nakagawa, E.Y.: A systematic literature review on the description of software architectures for systems of systems. In: Proceedings of the 30th Annual ACM Symposium on Applied Computing, pp. 1433–1440. ACM (2015)
21. Huang, E., Ramamurthy, R., McGinnis, L.F.: System and simulation modeling using SysML. In: Proceedings of the Winter Simulation Conference (WSC), Washington, DC, USA, December 9–12, pp. 796–803. WSC (2007). https://doi.org/10.1109/WSC.2007.4419675
22. Ingibergsson, J.T.M., Schultz, U.P., Kuhrmann, M.: On the use of safety certification practices in autonomous field robot software development: a systematic mapping study. In: Proceedings of the International Conference on Product-Focused Software Process Improvement, pp. 335–352. Springer (2015)
23. Johnson, T., Kerzhner, A., Paredis, C., Burkhart, R.: Integrating models and simulations of continuous dynamics into sysml. J. Comput. Inf. Sci. Eng. **12**(1), 011002 (2012). https://doi.org/10.1115/1.4005452
24. Kernschmidt, K., Vogel-Heuser, B.: An interdisciplinary SysML based modeling approach for analyzing change influences in production plants to support the engineering. In: Proceedings of the 2013 IEEE International Conference on Automation Science and Engineering, CASE 2013, Madison, WI, USA, August 17–20, pp. 1113–1118. IEEE (2013). https://doi.org/10.1109/CoASE.2013.6654030
25. Kitchenham, B., Charters, S.: Guidelines for performing systematic literature reviews in software engineering, version 2.3. EBSE Technical Report EBSE-2007-01. Keele University and University of Durham (2007)
26. Kitchenham, B.: Procedure for undertaking systematic reviews. Computer Science Department, Keele University (TRISE-0401) and National ICT Australia Ltd (0400011T. 1), Joint Technical Report (2004)
27. Kitchenham, B., Brereton, O.P., Budgen, D., Turner, M., Bailey, J., Linkman, S.: Systematic literature reviews in software engineering—a systematic literature review. Inf. Softw. Technol. **51**(1), 7–15 (2009)
28. Kitchenham, B.A., Budgen, D., Brereton, O.P.: Using mapping studies as the basis for further research—a participant-observer case study. Inf. Softw. Technol. **53**(6), 638–651 (2011)
29. Korff, A.: Modellierung von eingebetteten Systemen mit UML und SysML. Springer Spektrum Akademischer Verlag, Berlin (2008). (in German)
30. Kosar, T., Bohra, S., Mernik, M.: A systematic mapping study driven by the margin of error. J. Syst. Softw. **144**, 439–449 (2018). https://doi.org/10.1016/j.jss.2018.06.078
31. Kuhrmann, M., Méndez Fernández, D., Tiessler, M.: A mapping study on the feasibility of method engineering. J. Softw. Evol. Process **26**(12), 1053–1073 (2014). https://doi.org/10.1002/smr.1642
32. Laguna, M.A., Crespo, Y.: A systematic mapping study on software product line evolution: from legacy system reengineering to product line refactoring. Sci. Comput. Program. **78**(8), 1010–1034 (2013)
33. Lucas, F.J., Molina, F., Toval, A.: A systematic review of UML model consistency management. Inf. Softw. Technol. **51**(12), 1631–1645 (2009)
34. Lúcio, L., Mustafiz, S., Denil, J., Meyers, B., Vangheluwe, H.: The Formalism Transformation Graph as a Guide to Model Driven Engineering. Tech. rep. McGill University (2012)

35. Majikes, J.J., Pandita, R., Xie, T.: Literature review of testing techniques for medical device software. In: Proceedings of the 4th Medical Cyber-Physical Systems Workshop (MCPS'13), Philadelphia, USA (2013)
36. Mazak, A., Wimmer, M.: Towards liquid models: an evolutionary modeling approach. In: Kornyshova, E., Poels, G., Huemer, C., Wattiau, I., Matthes, F., Sanz, J.L.C. (eds.) Proceedings of the 18th IEEE Conference on Business Informatics, CBI 2016, 29th August - 1st September 2016, Paris, France, Volume 1 - Conference Papers, pp. 104–112. IEEE (2016). https://doi.org/10.1109/CBI.2016.20
37. Nelson, M., Piattini, M.: A systematic literature review on the quality of UML models. Innovations in Database Design, Web Applications, and Information Systems Management pp. 310–334 (2012)
38. Neto, P.A.d.M.S., do Carmo Machado, I., McGregor, J.D., De Almeida, E.S., de Lemos Meira, S.R.: A systematic mapping study of software product lines testing. Inf. Softw. Technol. **53**(5), 407–423 (2011)
39. Nguyen, P.H., Kramer, M., Klein, J., Le Traon, Y.: An extensive systematic review on the model-driven development of secure systems. Inf. Softw. Technol. **68**, 62–81 (2015)
40. Nguyen, P.H., Ali, S., Yue, T.: Model-based security engineering for cyber-physical systems: a systematic mapping study. Inf. Softw. Technol. **83**, 116–135 (2017)
41. Ormeño, Y.I., Panach, J.I.: Mapping study about usability requirements elicitation. In: Proceedings of the International Conference on Advanced Information Systems Engineering, pp. 672–687. Springer (2013)
42. Paredis, C., Bernard, Y., Burkhart, R., de Koning, H.P., Friedenthal, S., Fritzson, P., Rouquette, N., Schamai, W.: An overview of the SysML-Modelica transformation specification. In: Proceedings of the 20th Annual International Symposium of the International Council on Systems Engineering (INCOSE 2010), INCOSE International Symposium, vol. 20, pp. 709–722 (2010)
43. Peak, R., Burkhart, R., Friedenthal, S., Wilson, M., Bajaj, M., Kim, I.: Simulation-based design using SysML part 1: A parametrics primer. In: Proceedings of the 17th Annual International Symposium of the International Council on Systems Engineering (INCOSE), INCOSE International Symposium, vol. 17, pp. 1516–1535 (2007)
44. Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M.: Systematic mapping studies in software engineering. In: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering, EASE'08, pp. 68–77. British Computer Society, Swinton, UK, UK (2008)
45. Petersen, K., Vakkalanka, S., Kuzniarz, L.: Guidelines for conducting systematic mapping studies in software engineering: an update. Inf. Softw. Technol. **64**, 1–18 (2015)
46. Petticrew, M., Roberts, H.: Systematic Reviews in the Social Sciences: A Practical Guide. Wiley, New York (2008)
47. Pretorius, R., Budgen, D.: A mapping study on empirical evidence related to the models and forms used in the UML. In: Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, pp. 342–344. ACM (2008)
48. Rashid, M., Anwar, M.W., Azam, F., Kashif, M.: Model-based requirements and properties specifications trends for early design verification of embedded systems. In: Proceedings of the 11th System of Systems Engineering Conference (SoSE), pp. 1–7. IEEE (2016)
49. Rashid, M., Anwar, M.W., Khan, A.M.: Toward the tools selection in model based system engineering for embedded systems—a systematic literature review. J. Syst. Softw. **106**, 150–163 (2015)
50. Reichwein, A., Paredis, C.J.J., Canedo, A., Witschel, P., Stelzig, P.E., Votintseva, A., Wasgint, R.: Maintaining consistency between system architecture and dynamic system models with SysML4Modelica. In: Proceedings of the 6th International Workshop on Multi-Paradigm Modeling (MPM@MoDELS 2012), Innsbruck, Austria, October 1–5, pp. 43–48. ACM (2012). https://doi.org/10.1145/2508443.2508451
51. Roques, P.: SysML vs. UML2: A Detailed Comparison (2011). http://ecs.victoria.ac.nz/foswiki/pub/Events/MODELS2011/Material/MODELS_2011_T2-Roques-SysML_UML2.pdf. Talk at 14th International Conference on Model Driven Engineering Languages and Systems (MODELS 2011), Wellington, New Zealand, October 16–21
52. Rusli, H.M., Ibrahim, S.: Testing web services composition: a mapping study. Communications of the IBIMA (2011)
53. Shaw, M.: Writing good software engineering research paper. In: Proceedings of the 25th International Conference on Software Engineering, Portland, Oregon, USA, May 3–10, pp. 726–737. IEEE Computer Society (2003). https://doi.org/10.1109/ICSE.2003.1201262
54. Siavashi, F., Truscan, D.: Environment modeling in model-based testing: concepts, prospects and research challenges: a systematic literature review. In: Proceedings of the 19th International Conference on Evaluation and Assessment in Software Engineering, p. 30. ACM (2015)
55. Souag, A., Mazo, R., Salinesi, C., Comyn-Wattiau, I.: Reusable knowledge in security requirements engineering: a systematic mapping study. Requir. Eng. **21**(2), 251–283 (2016)
56. Torre, D., Labiche, Y., Genero, M.: UML consistency rules: a systematic mapping study. In: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, p. 6. ACM (2014)
57. Valença, G., Alves, C., Alves, V., Niu, N.: A systematic mapping study on business process variability. Int. J. Comput. Sci. Inf. Technol. **5**(1), 1 (2013)
58. VDI Verein Deutscher Ingenieure: https://www.vdi.de/2206. Accessed: 2017-03-14
59. Wieringa, R., Maiden, N., Mead, N., Rolland, C.: Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. Requir. Eng. **11**(1), 102–107 (2005). https://doi.org/10.1007/s00766-005-0021-6
60. Wohlin, C.: Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering, p. 38. ACM (2014)
61. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B.: Experimentation in Software Engineering. Springer, Berlin (2012). https://doi.org/10.1007/978-3-642-29044-2
62. Wortmann, A., Combemale, B., Barais, O.: A systematic mapping study on modeling for industry 4.0. In: Proceedings of the 20th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems (MODELS), Austin, TX, USA, September 17–22, pp. 281–291. IEEE Computer Society (2017). https://doi.org/10.1109/MODELS.2017.14
63. Zhang, H., Babar, M.A., Tell, P.: Identifying relevant studies in software engineering. Inf. Softw. Technol. **53**(6), 625–637 (2011). https://doi.org/10.1016/j.infsof.2010.12.010

Sabine Wolny is a Ph.D. student working in the Christian Doppler Laboratory for Model-Integrated Smart Production (CDL-MINT) at the JKU Linz in the module Reactive Model Repositories. Before she was a doctoral student at the Doctoral College for Cyber-Physical Production Systems (CPPS) at TU Wien. Her topic of interest is SysML-based modeling and execution of systems. In addition to CDL-MINT, she is working since 2013 in the Research Unit of Building Physics at TU Wien with a focus on project management and developing software solutions. For more information, please visit https://www.se.jku.at/sabine-wolny/.

Alexandra Mazak is a senior researcher, heading the module Reactive Model Repositories in the Christian Doppler Laboratory for Model-Integrated Smart Production (CDL-MINT) at the Johannes Kepler University Linz and is also working as a scientific advisor for technology policy in the Austrian Council for Research and Technology Development. Her research interests comprise data integration, statistical modeling and forecast as well as model-driven systems and software engineering in the research field of Industry 4.0. She headed numerous national funded projects in this research field. For more information, please visit https://www.se.jku.at/alexandra-mazak-huemer/.

Christine Carpella was an industrial researcher at SCCH, working primarily in business process and systems modeling. She holds an M.Sc. and an M.A. in Computer Science from the University of Applied Sciences of Upper Austria and a Ph.D. from the Johannes-Kepler-University Linz. Since 2017, she is working at ENGEL AUSTRIA GmbH, a company producing injection molding machines.

Verena Geist holds a Diploma in Software Engineering from the University of Applied Sciences Upper Austria, Hagenberg, and a Ph.D. from the Johannes Kepler University Linz, Austria. Since 2005, she is a researcher at the Software Competence Center Hagenberg (SCCH). Her main research areas are business process modeling, information systems, formal methods, and model-based systems engineering.

Manuel Wimmer is full professor leading the Institute of Business Informatics–Software Engineering at the Johannes Kepler University Linz, and he is the head of the Christian Doppler Laboratory CDL-MINT. His research interests comprise foundations of model engineering techniques as well as their application in domains such as tool interoperability, legacy modeling tool modernization, model versioning and evolution, and industrial engineering. For more information, please visit https://www.se.jku.at/manuel-wimmer/.

8 Towards Liquid Models: An Evolutionary Modeling Approach

A. Mazak and M. Wimmer;
Proceedings of the 18th IEEE International Conference on Business Informatics (CBI),
IEEE Computer Society, (2016), pp. 104–112.
DOI: 10.1109/CBI.2016.20

2016 IEEE 18th Conference on Business Informatics

Towards Liquid Models: An Evolutionary Modeling Approach

Alexandra Mazak and Manuel Wimmer
Business Informatics Group
Institute of Software Technology and Interactive Systems
TU Wien
Vienna, Austria
Email: {mazak, wimmer}@big.tuwien.ac.at

***Abstract*—Today, we recognize a discrepancy between design time models concentrating on the desired behavior of a system and its real world correspondents reflecting deviations taking place at runtime. In order to close this gap, design time models must not be static, but evolutionary artifacts so called "liquid" models. Such liquid models are the cornerstone of our future research project "CDL-MINT: Model Integrated Smart Production". In this position paper, we present an early result of this project: the liquid models architecture for linking design models to runtime concerns, which are derived from distributed and heterogeneous systems during operation. We elaborate on the proposed technologies for the respective architecture layers and identify the research challenges ahead.**

I. INTRODUCTION

Forecasts show that in the upcoming years most of the things and devices we interact with will be linked to a global computing infrastructure [1]. In the automated manufacturing engineering domain, the *Internet of Things (IoT)* makes it possible to create networks incorporating the entire manufacturing process to convert production to *smart production* [2]. The establishment of IoT in manufacturing (also known as Industrial Internet) will have a disruptive impact on the IT industry. This impact will result in the convergence of the physical world and the virtual world in the form of *Cyber Physical Production Systems (CPPS)* [3]. This represents a new tendency, in which the physical environment is populated by interconnected and communicating objects (e.g., sensors, actuators and other smart devices) capable for autonomously interacting with each other and with the environment itself. As a result, both, the volume and the level of detail of the corporate data generated will highly increase.

Due to this ever growing importance of flexibility, also resulting from shorter innovation cycles, rapidly changing customer needs, etc., the role of software is becoming more and more important in the industrial automation domain. Especially with the transition in the next industrial revolution, in Germany and Austria known as Industrie 4.0[1], several new challenges arise. As a consequence systems may no longer be designed to stay for decades in certain settings, but they may be designed to evolve and exist in different variants adapted to certain contexts [4].

Particularly in Industrie 4.0 scenarios, there will be an extended usage of models also during operation time that allows runtime monitoring and design model enhancement [2], [5], [6], but still, models are completely isolated from the running systems which have been beforehand described by these models. Generally, the automated manufacturing engineering domain exploit the benefits of modeling for code generation [7]. Thereby, the dynamic extraction of runtime models to better link operation with engineering is often overlooked. However, current modeling foundations and practices are lacking behind this emerging requirement. Models are still considered as static entities, basically neglected in later lifecycle phases of systems.

One reason is that models are never complete and only created for a specific purpose, such as being a "blueprint" of a system to be developed [8]. Since, modelers or system engineers tend to concentrate on the desired behavior of the system during its design, they are not aware of the many deviations that may take place at runtime. These deviations may result in discrepancies between the design model and its real world correspondent. This gap can be seen as a discrepancy between models created at design time and real operations within systems. In software engineering this gap is referred to as the *DevOps* gap. DevOps is a trend ensuring a continuous feedback loop (i.e., "model in the loop") between development (Dev) and operations (Ops) [9]. DevOps aims at a better integration of all activities in software development and the operation of an application system lifecycle in order to continuously deploy stable versions of application systems [10].

If models should be fully integrated in the lifecycle of a system, having only an a-priori description of the system as a product of the initial engineering phases is not enough. From this perspective models are no longer static descriptions, but evolutionary artifacts so called "liquid models". The term "liquid" is used to stress that models as any kind of artifact should not be isolated and frozen, but reusable and evolutionary. The evolutionary aspect of engineering artifacts refers to the fact that they change over time. In engineering processes models are generally developed from initial ideas to first drafts. They are then continuously revised, often by taking

[1]Please note, that the approach introduced in this position paper is aligned with the German initiative "Industrie 4.0", and therefore, we do not translate "Industrie" to the English term "Industry".

2378-1971/16 $31.00 © 2016 IEEE
DOI 10.1109/CBI.2016.20

IEEE computer society

into account feedback from other resources, until they are finally released. However, also the feedback after the release from the operation should be reflected in the models to cover the complete lifecycle of a system.

Liquid models represent a corner stone of our research project *CDL-MINT: Model-Integrated Smart Production*. Thus, this project aims at linking design models to runtime concerns which are derived from distributed and heterogeneous systems during operation. As a first step in this direction we develop a liquid models architecture that forms the core of this paper. This architecture comprises three layers on top of heterogeneous data sources. We discuss the goals of each layer and the potential technologies to be used. Furthermore, we identify the research challenges ahead.

The remainder of this paper is structured as follows: In Section 2 we introduce related work we considered in the development of our architecture. We discuss the importance of model repositories. Emerging approaches in the field of *process mining* in the area of workflow systems and *runtime models* in the area of model-driven engineering (MDE) [11] are noticed as first approaches considering systems in operation. Data analytics techniques are evidently required to learn from systems in operation. Given the state-of-the-art, we identify key research issues towards liquid models in Section 3. Section 4 introduces our liquid models architecture and discusses its layers in detail. Finally, Section 5 summarizes this paper.

II. Related Work

Several lines of research are relevant to realize our goal to establish liquid models managing operational models that are reactive with respect to their origin (i.e., built from real world data acquired from system operations). In the following, we outline the state-of-the-art of the most relevant ones. First, we discuss emerging model repositories as well as modeling approaches considering runtime aspects in addition to design. Second, we survey techniques to deal with operational data and to turn it into abstracted model representations. In this context, we specifically highlight process mining, runtime models, and data analytics techniques.

A. Model Repositories

Research concerning *model repositories* comprises mainly two areas: concurrent modeling using a central repository to coordinate the editing of models [12], [13] and scalability in storing and retrieving models [14]. Currently, the general services offered by a model repository are twofold: (i) load a complete model from a repository, and (ii), store a complete model to a repository. Other services, such as more fine grained model loading or manipulation is currently missing in most repositories [15]. The scalability problems of loading large models represented by XML-based documents into memory has been already recognized several years ago. One of the first improved solutions for models is the *Connected Data Objects* (CDO)[2] model repository, which enables to store models in all kinds of database back-ends, such as traditional relational databases or emerging NoSQL databases. CDO supports the ability to store and access large-sized models due to the transparent loading of single objects on demand and caching them. If objects are no longer referenced, they are automatically garbage collected. There are also several projects for storing very large models, like MongoEMF[3] and Morsa [16]. Both approaches are built on top of MongoDB. Furthermore, graph-based databases as well as map-based databases are also exploited for model storage, such as done in Neo4EMF [17], [18] where also different unloading strategies for partial models are explored [19]. In [20], Clasen et al. elaborate on strategies for storing models in a distributed manner by horizontal and vertical partitioning in Cloud environments. A similar idea is explored in [21] where different automatic partitioning algorithms are discussed for graph-based models. What all the mentioned approaches have in common is that models are residing behind the walls of the model repository without a proper connection to the environment as is needed, for instance, to provide reactivity by observing runtime environments during operations.

This is one particular goal of our approach to integrate model streaming to model repositories in addition to full loading and storing of complete models in order to have means for observing systems during operation.

B. Runtime Models

There are several different approaches for *runtime modeling*. All of them aim on bridging the gap between design time modeling and runtime modeling to enable runtime analysis. Blair et al. [22] show the importance of supporting runtime adaptations to extend the use of model-driven engineering. They propose models that provide abstractions of systems during runtime. These operational models are an abstraction of runtime states. Due to this abstraction, different stakeholders can use the models in various ways, like dynamic state monitoring or observing runtime behavior. Hartmann et al. [23] go one step further. They combine the ideas of runtime models with reactive programming and peer-to-peer distribution. Reactive programming aims on enabling support for interactive applications, which react on events by focusing on streams. For this purpose a typical publish/subscribe pattern, well known as the observer pattern in software engineering [24], is used. Khare et al. show the application of such an approach in the IoT domain in [25].

Luckham [26] introduces Complex Event Processing (CEP) by defining complex events which are correlated among each others. Such an approach of CEP on stream data was described by Saleh et al. [27]. Hartmann et al. [23] define runtime models as a stream of model chunks, like it is common in reactive programming. The models are continuously updated during runtime, therefore they grow indefinitely. With their interpretation that every chunk has the data of one model element, they process them piecewise without looking at the total size.

[2]http://projects.eclipse.org/projects/modeling.emf.cdo

[3]http://code.google.com/a/eclipselabs.org/p/mongo-emf

In order to prevent the exchange of full runtime models, peer-to-peer distribution is used between nodes to exchange model chunks. In addition, automatic reloading mechanism are used to respond on events for enabling reactive modeling. As the models are distributed, operations like transformations have to be adapted. For this purpose transformations on streams as proposed by Cuadrado et al. [28] can be used.

In our approach, we will explore this path of research even further by closely combining model streaming reasoning, reactive programming, and operational data monitoring. For this purpose we have to integrate powerful reasoning algorithms which may be inspired from the fields of process mining in particular and of data analytics in general.

C. Process Mining

Process mining (PM) is a process-centric management technique bridging the gap between data mining and traditional model-driven Business Process Management (BPM) [29], [30]. In this field of research business processes are analyzed on the basis of *event logs*. Events are defined as process steps and event logs as sequential events recorded by an information system [31]. This demonstrates that unlike BPM approaches PM works on the basis of event data instead of designed models. The main challenge is to capture behavioral aspects. In [29], van der Aalst introduces specialized algorithms (e.g., the α-algorithm) to extract knowledge from event logs. Thereby, an algorithm produces a Petri net, which can easily be converted into a process model as, for instance, a BPMN model, or a UML activity diagram.

A process model is used in the (re)design, configuration and implementation phase, whereas, data provides insight on actual processes for monitoring and diagnosis purposes [32]. The main objective of PM is to extract valuable, process-related information from *event logs* for providing detailed information about actual processes, for instance, to identify bottlenecks, to anticipate problems, to record policy violations, to streamline processes, etc. [30]. PM is not limited to this *control-flow perspective*. There are further perspectives as introduced in [29], namely (i) the *organizational perspective* focusing on information about resources (e.g., people, systems, departments) hidden in event-logs, (ii) the *case perspective* focusing on properties of cases, and (iii) the *time perspective* concerned with the timing and frequency of events.

In [29], van der Aalst lists three basic PM goals, which are (i) discovery, (ii) conformance, and (iii) enhancement. *Discovery* means to take an event log as input and to produce a process model as output. When targeting for *conformance* an existing process model is compared with an event log of the same process. This means an event log and a model are used as input and a diagnostic information is produced as output. Thereby, a user can check whether information recorded in the log conforms to the intended model and vice versa. Conformance checking can be applied to any kind of models (e.g., business process models, declarative process models, etc.). The third type of PM is called *enhancement*. Its idea is to improve or extend an existing model. It takes an event log and a model as input and produces a new model as output.

Current event processing technologies usually monitor single streams of events at a time. Even if users monitor multiple streams, they often end up with multiple "silo" views. A more unified view is needed that correlates with events from multiple data streams of various sources and in different formats. Thereby, heterogeneity and incompleteness of data are major challenges [33]. Mostly, PM operates on the basis of events that belongs to cases that are already completed [34]. This off-line analysis is not suitable for cases which are still in the pipeline. In [29], the author mixes current data with historic data to support on-line and off-line analysis.

D. Data Analytics

Data analytics deals with the acquisition of information derived from a big amount of data. Its target is to recognize new models within these quantities of data, to recover existing patterns and to discover new patterns [35]. Therefore, complex analytical procedures and methods of various fields, e.g., machine learning, data mining, statistics and mathematics are being applied [36].

In [37], Fayyad et al. define data mining as the process of knowledge discovery in databases. This process consists of several steps, starting with the selection, the pre-processing, the transformation, the mining, the interpretation and the evaluation of data. Various methods can be applied that deal with the application of suitable models on observed data and the finding of interesting patterns within these observed data [38]. There are statistical models that permit non-deterministic and deterministic effects, and logic approaches [37]. These methods are commonly based on algorithms known from other areas, as for instance, machine learning.

Machine learning deals with the construction of systems that optimize performance criteria based on sample data or past experiences [39]. Usually these systems built on models that are "trained" on the behavior of existing data with the help of machine learning algorithms. These models can be differentiated into descriptive and predictive models. Descriptive models attempt to learn new information out of existing data, whereas predictive models provide predictions on the future. Additionally, machine learning can be subdivided into *supervised* and *unsupervised learning* techniques. Methods for supervised learning attempt to learn a hypotheses based on known data (target value and / or value for reward), whereas unsupervised learning applies algorithms that do not need to known target values or values for reward beforehand [40].

There are many existing research projects focusing on machine learning in the field of Big Data analytics [36]. For instance, the WEKA[4] toolkit provides various machine learning algorithms for pre-processing, classification, regression, clustering, and the visualization of data. The project Apache Mahoud[5] works on the efficient and parallelized

[4]http://www.cs.waikato.ac.nz/ml/weka
[5]http://mahout.apache.org

implementation of machine learning algorithms and provides these algorithms open source to the community. Additionally, the MLib machine learning library implemented in the course of the Apache Spark[6] project provides a broad set of machine learning algorithms, which can easily be applied on a large amount of data offering high-performance.

Additionally to this overview, there are various techniques that can be applied in order to learn new models or improve existing ones. These techniques primarily rely on statistical models and algorithms of the research fields of neuronal networks as well as genetic algorithms [35]. As concrete research lines for our approach, we combine these techniques with model repositories in order to provide scalable solutions for liquid models which may have to deal with huge amounts of operational data. One important open challenge, for linking design models to runtime models, is how to combine advanced data analytics technologies and modeling technologies, such as model transformation engines or model checkers, in order to get the best of both worlds. This is a particular challenge towards a moving target, since there are still several open challenges in these areas to be combined.

III. Open Research Issues

This section summarizes key research issues (RI) we have identified in the state-of-the-art to realize our research goals.

A. Real-time Analytic Correlation across Multiple Data Streams

Manufacturing machines and sensors continuously generate and transmit data (e.g., detailed logs) about their actions and current condition. As a result, there is an exponential growth of volumes of unstructured, semi-structured, and multi-structured data (e.g., machine data, sensor data, event streams, XML, CSV, SCADA). Data streams are ordered and potentially unbounded sequences of data points created by a typically non-stationary data generating process. However, traditional algorithms for data mining cannot mine even a fraction of these streams in real-time. The situation is aggravated by the fact that data is distributed over a variety of different sources in various latencies (from batch to real-time). Merging this data tend to be problematic, e.g., caused by heterogeneities (e.g., technical, syntactical, semantical), different levels of granularity, etc. A further challenge is to deal with incomplete and noisy data streams as well as with concept drifts (i.e., changes on data streams while being observed). For analytical purpose (i) each event has to be captured from a stream, (ii) events of interest have to be separated from noise, (iii) correlations with other streams and databases have to be established, (iv) it has to be reacted to the events of interest in real-time, (v) and events have to be stored in an appropriate model structure for on-line and off-line analytics. In order to tackle this research issue, we see two sub research issues that have to be addressed:

- **RI 1.1**: How to integrate distributed and heterogeneous data streams?
- **RI 1.2**: How to provide reactive model stream processing mechanisms within existing modeling technologies and accompanying languages?

[6]http://spark.apache.org

B. DevOps-centric Methodology to support Evolutionary Model Mining

DevOps is a trend ensuring a continuous feedback loop between development (Dev) and operations (Ops) [9] concerning technical as well as social aspects. DevOps aims at a better integration of all activities in software development (e.g., also in the context of MDE) and the operation of an application lifecycle in order to continuously deploy stable versions of applications [10]. The monitoring process is, among others, one of the key factors for a successful implementation of DevOps. In our research project, we transfer the DevOps concept to the industrial automation domain for a continuous end-to-end engineering in this ever-changing environment. In particular, by having integrated and unified model streams, model mining techniques can be applied to extract operational models which can be matched against the design models in order to enhance them with more knowledge from observations during operation time. In order to tackle this research issue, we see two sub research issues that have to be addressed:

- **RI 2.1**: How to realize scalable model mining techniques on top of model streams?
- **RI 2.2**: How to propagate back and capture the observed knowledge from operation to design models?

IV. Liquid Models Architecture

In this section, we lay out our envisioned architecture (cf. Fig. 1) to tackle the aforementioned research issues.

A. Overview

The full integration of design models in the systems' lifecycles requires a continuously acquisition of real-time data (e.g., machine data, sensor data, event streams) from various distributed and heterogeneous systems. As a first step in our approach, we acquire heterogeneous event logs from various data sources. This kind of data acquisition is taking place during operation time in the various runtime environments. Therefore, the data is not a finite set, but, a continuous stream of heterogeneous data. After resolving certain forms of heterogeneity (e.g., technical, syntactical, semantical), as discussed in [41], [42], the data is processed within distributed operational models. These operational models implement statistical methods as well as machine learning algorithms, targeting the identification of patterns and anomalies. Based on this runtime information, the models created during design time can constantly be improved. The alignment between operational models and design models is what we call the *DevOps bridge* in our architecture.

Fig. 1 gives an overview of the liquid models architecture. We describe this figure bottom-up by a 4-layered stack. Level 0 (L0) represents distributed, physical components, sending data streams during operation. Due to their heterogeneity, these data streams have to be aligned and synchronized. Traversing

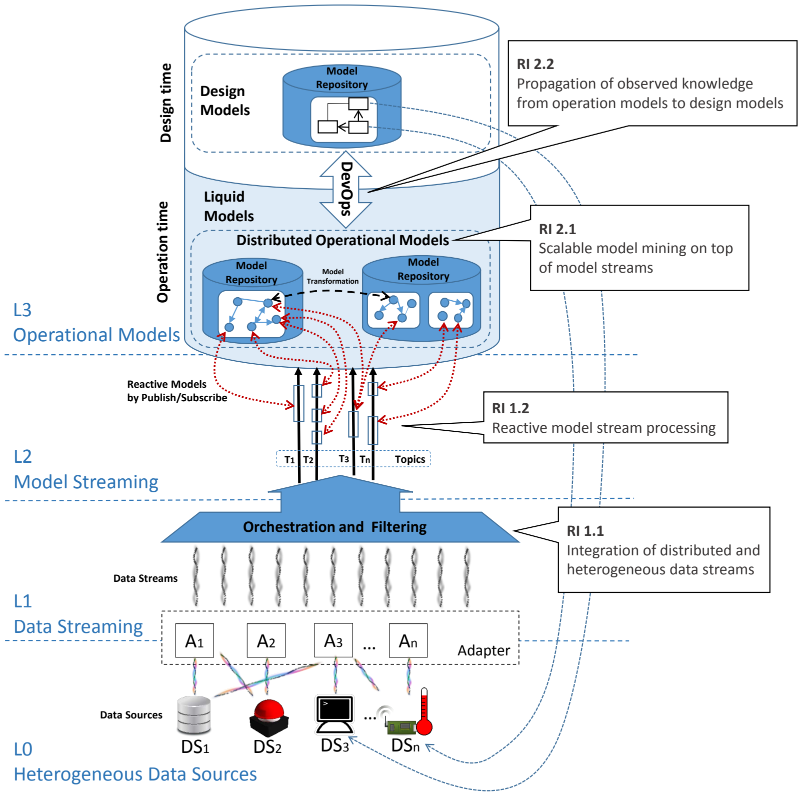


Fig. 1. Liquid Models Architecture.

the layers bottom-up (L0-L3), heterogeneities, distribution and timing inconsistencies are resolved. At the top layer (L3) the operational models are built, based on model mining from subscribed topics as for instance non-functional properties [43]. Hereby, a topic is an orchestration of data streams with the application of potential filters. Even if only filtered topics of interest are subscribed by operational models, the data volume can become very high. Therefore, we propose a distributed modeling approach in order to ensure scalability and availability. Please note that the model repositories embedded in the outlined architecture of Fig. 1 become *reactive model* repositories. First, the operational models are reactive with respect to event occurrences in the runtime environments. Second, the design models are reactive with respect to changes in the operational models. Finally, there are the design models at the top of Fig. 1, which we assume as given artifacts.

In the following subsections, we outline how we will realize this architecture, namely, how to integrate distributed and heterogeneous data streams, how to react on important events in unified model streams, how to realize distributed model repositories, and finally, how to apply model learning on operational data to bridge the DevOps gap mentioned before.

B. Level 0: Heterogeneous Data Sources

The starting point of our approach are data streams from various and physically distributed data sources (cf. Fig. 1, level L0 $DS_1, ..., DS_n$). This sources have to be processed in real-time, where there is no random access and which can be read only once a time (e.g., readings from sensor networks). These data streams are infinite sequences of ongoing events. They are received continuously and in real-time, either implicitly ordered by arrival time, or explicitly associated with timestamps. Nearly everything can be considered as a stream, e.g., sensor data, machine data, user inputs, calculation results (see Fig. 1, ground level). The data content is scattered over several sources, which are based on different technologies (e.g., CoAP, OPC UA, MTConnect, MQTT, OSLC, etc.). In order to enable real-time data handling, we provide appropriate adapters for further processing the data. These adapters are used to resolve technical and data model heterogeneities in order to create *isolated profiles* for each source stored in a repository. Based on our previous work introduced in [44], [45], we provide *mapping operators* to resolve structural heterogeneity and data *fusion operators* to deal with duplicates and conflicts. Additionally, we aim for *provenance recording* which automatically records all information about the integration task.

C. Level 1: Data Streaming

In a next step, the data streams are the input for the *Data Distributed Service (DDS)* [46] (cf. Fig. 1, orchestration and filtering on L1) which is a proven international standard from the Object Management Group (OMG). The DDS is a middleware protocol and API standard for data-centric connectivity to save data communication overhead. It provides a data-centric solution to understand the schema of shared data. We select DDS as a first potential candidate to support our approach. However, we will also critically evaluate whether there are better alternatives for certain scenarios. DDS allows to filter the data that is actually needed. For this purpose DDS offers communication by publishing and subscribing to *topics* for collaborative filtering (cf. Fig. 1, L2). Subscriptions can specify time and context filters to get specific subsets of the data being published on the topic. In our approach, we use DDS for filtering and smoothing the time component, to get the right data at the right place at the right time.

The filtering mechanisms help to reduce the volume of observed data streams. Nevertheless, the filtered streams are still a continuous and infinite sequence of data. There are various methods for accessing information provided by streams, as for instance, *Complex Event Processing (CEP)* [27] or *C-SPARQL* [47]. Compared to these emerging techniques, the classical DDS stream reasoning techniques seems limited and have to be extended in this respect. In a first step, we plan to investigate the appropriateness of different technologies for model streaming. For example, we may investigate C-SPARQL as continuous query language that enables stream reasoning over RDF data streams [47], [48]. An RDF data stream is not static, it is continuously produced and with a times-stamp annotated RDF triple identified by an *Internationalized Resource Identifier (IRI)* [48]. The *windows function* enables to extract certain criteria over these infinite sets. This extraction can be a given number of triples or a variable number of triples occurring in a given time interval. By using continuous query languages these triples can be queried in order to get unified data streams for further processing. In order to allow the reuse of existing model query and transformation languages, we plan to integrate concepts of C-SPARQL within the Object Query Language (OCL) to allow an easier adoption of stream reasoning for engineers compared to switching to the RDF level. Initial experiences for implementing approximate model transformations may be reused where we already reformulated the classical OCL operators for potentially infinite models, introduced in our work in [49].

D. Level 2: Model Streaming

The unified streams provide a "dynamic picture" based on various DDS-topics (e.g., non-functional properties). However, due to the still vast amount of data, these streams cannot be stored in memory. Thus, we use statistical methods to generate inductive-empirical models to extract *operational models* which can be matched against the design models (cf. Fig. 1, L3). Thereby, we aim for the development of a hybrid approach by combining techniques and methods of advanced process mining, multivariate statistics, stochastic and complex event processing. The challenge will be the efficient combination of these techniques to extract more insights from data streams. The idea of a hybrid approach is demonstrated by a short example: to mine unified data streams for classification, we combine Online Convex Programing (OCP) and Support Vector Machines (SVM). OCP is a general framework for online prediction algorithms [50] and SVM is a vector-based machine learning technique [40]. By following this approach not all data streams must be kept in memory. Thereby, optimal event-based feature vectors can still be calculated by means of optimizing a convex function on a convex set. This convex set may be generated by techniques adapted from CEP. This is one candidate method for classification, there may be others which have to be identified and investigated in the course of our future work.

In the context of data (stream) mining the "curse of dimensionality" is a critical challenge, too. In order to overcome this problem, we apply statistical methods for dimension reduction (e.g., Principal Component Analysis, or data stream clustering by micro-clustering, or the BIRCH algorithm, as well as canonical analysis techniques) combined with methods of CEP (e.g., sliding time window) and approaches for combinatorial optimization. Depending on certain scenarios, we apply multivariate statistical methods (exploratory and inductive), advanced time series analysis, machine learning techniques such as neural networks, genetic algorithms, and stochastic models (e.g., Hidden Markov Models) to generate appropriate operational models. For the fulfillment of *adequacy* and *appropriateness* of the outcomes, we develop a metric to evaluate which algorithms are best suited, since some algorithms may be more challenging to adapt than others.

E. Level 3: Operational Models

For persisting the operational models, we plan to implement distributed model repositories in the spirit of Hartmann et al. [23]. The models within these repositories subscribe on topics delivered by DDS. The events within the topics are captured, unified and saved in the reactive operational models. In this way the semantic heterogeneity is resolved. For transformations between different operational models (cf. Fig. 1, dotted arrow between the Distributed Operational Models), we plan to built on existing transformation languages like ATL and approximative transformations as presented in our work in [49].

The operational models build the starting point for the *DevOps Bridge*. Being in a "data-rich" situation we divide the model samples into three parts, a training set, a validation set, and a test set. For model (stream) learning, we use the training set to fit the models, the validation set to estimate prediction error for model selection, and the test set to assess the generalization error of the finally chosen model [40]. For training purpose, algorithms need to access the complete training set several times. Providing this access is not straight forward, since data streams deliver constantly new data. Additionally, it is important to evaluate how well the algorithm is able to adapt to changes during a learning phase, known as *concept drift* [30]. For instance, Aggarwal et al. [51] introduced an approach where data streams are splitted into consecutive horizons to overcome the concept drift problem during clustering. Another example could be to calculate the probability when changes had happened by time series analysis in combination with CEP techniques. Change detection may be used as filter to receive all changes within time slots. This approach requires an appropriate classifier. For example, the Kalman filter (limited to multi-dimensional Gaussian distribution) may serve as a suitable classifier in case of time series analysis with Hidden Markov Models [52]. In addition, clustering techniques applied on time series help to identify the internal and external triggers that might have caused changes.

The validation step can either be conducted analytically (Akaike information criterion, Bayesian information criterion), or by efficient sample reuse (cross-validation and bootstrap) [53]. The assessment of the *generalization performance* is extremely important in stream learning [54]. It shows the interplay between bias, variance, and model complexity. It guides users in *model selection* (i.e., when estimating the performance of different models in order to choose the approximate best one) and in *model assessment* (i.e., when estimating the prediction error of the chosen model) [40]. In a final step, we adapt the *conformance checking* technique, as introduced in [29], to align operational models and design models.

This alignment includes the extension of design models with information about derived information from operational models, as well as information about runtime characteristics unknown at design time. For the latter, standard languages will be used, such as for instance the OMG MARTE profile [55] for capturing real-time and embedded characteristics of the operational systems (e.g., resource allocations and performance characteristics) or existing languages to represent variants of a system and enrich the variant description with important operational information such as performance, reliability, and responsiveness. In the model learning task, we will enrich the conformance checking method introduced in [29] with techniques from supervised and unsupervised learning and concepts of non-functional property languages, to repair design models that are not aligned well with reality, which provides a basis to close the gap between DevOps.

V. Conclusion

In this position paper, we present several upcoming research challenges for stimulating a shift from isolated, one-shot, monolithic system descriptions to evolutionary, reusable predictions. We focus on how to connect runtime environments to model repositories to extract operational models and their connection to design models. Thus, specific techniques are needed to connect to runtime environments and to deal with model streams to efficiently react to events occurring in various highly physically distributed data sources at the ground level.

We present an architecture for this purpose that uses appropriate techniques on four dedicated levels. Between level L0 and L1, we adapt an idea from previous work [44], [45] applying adapters to resolve technical and data model heterogeneities in order to create isolated profiles for each source stored in a repository. In a next step, between L1 and L2, we introduce Data Distributed Services (DDS) to filter these profiles by publishing and subscribing to topics in order to reduce the volume of observed data streams. At this level, we present additional methods (CEP, C-SPARQL) for stream reasoning and for enhancing the DDS technique. Between L2 and L3, we outline some promising statistical methods to generate inductive empirical models. These kind of operational models are then matched to the corresponding design models in order to improve the latter ones. Thereby, design models are extended with information previously unknown at design time.

References

[1] M. Broy and A. Schmidt, "Challenges in engineering cyber-physical systems," *Computer*, vol. 47, no. 2, pp. 70–72, 2014.

[2] BITKOM, VDMA, and ZVEI, "Umsetzungsstrategie Industrie 4.0," Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (BITKOM), Verband Deutscher Maschinen- und Anlagenbau e. V. (VDMA), Zentralverband Elektrotechnik- und Elektronikindustrie e. V. (ZVEI), Ergebnisbericht der Plattform Industrie 4.0, 2015.

[3] H. Kagermann, W. Wahlster, and J. Helbig, "Recommendations for Implementing the Strategic Initiative INDUSTRIE 4.0 – Securing the Future of German Manufacturing Industry," Forschungsunion im Stifterverband für die Deutsche Wirtschaft e. V., Final Report of the Industrie 4.0 Working Group, 2013.

[4] M. Kowal, C. Legat, D. Lorefice, C. Prehofer, I. Schaefer, and B. Vogel-Heuser, "Delta modeling for variant-rich and evolving manufacturing systems," in *Proceedings of the 1st International Workshop on Modern Software Engineering Methods for Industrial Automation (MoSEMInA 2014)*. ACM, 2014, pp. 32–41.

[5] BITKOM, VDMA, and ZVEI, "Industrie 4.0 - Whitepaper FuE-Themen," *Plattform Industrie 4.0*, 2015, http://www.zvei.org/Downloads/Automation/Whitepaper-I40-FuE-Themen-2015-04.pdf.

[6] S. Schmitz, M. Schluetter, and U. Epple, "Automation of automation: Definition, components and challenges," in *Proceedings of the 14th IEEE International Conference on Emerging Technologies Factory Automation (ETFA 2009)*. IEEE, 2009, pp. 1–7.

[7] V. Vyatkin, "Software engineering in industrial automation: State-of-the-art review," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 3, pp. 1234–1249, 2013.

[8] H. Stachowiak, *Allgemeine Modelltheorie*. Springer, 1973.

[9] A. Brunnert, A. van Hoorn, F. Willnecker, A. Danciu, W. Hasselbring, C. Heger, N. Herbst, P. Jamshidi, R. Jung, J. von Kistowski, A. Koziolek, J. Kroß, S. Spinner, C. Vögele, J. Walter, and A. Wert, "Performance-oriented DevOps: A research agenda," SPEC Research Group — DevOps Performance Working Group, Standard Performance Evaluation Corporation (SPEC), Tech. Rep. SPEC-RG-2015-01, 2015.

[10] M. Hüttermann, *DevOps for Developers*, 1st ed. Apress, 2012.

[11] M. Brambilla, J. Cabot, and M. Wimmer, *Model-Driven Software Engineering in Practice*. Morgan & Claypool, 2012.

[12] P. Brosch, G. Kappel, M. Seidl, K. Wieland, M. Wimmer, H. Kargl, and P. Langer, "Adaptable model versioning in action," in *Proceedings of Modellierung 2010*. GI, 2010, pp. 221–236.

[13] M. Koegel and J. Helming, "EMFStore: A Model Repository for EMF Models," in *Proceedings of the 32nd International Conference on Software Engineering (ICSE)*. ACM, 2010, pp. 307–308.

[14] D. S. Kolovos, L. M. Rose, N. D. Matragkas, R. F. Paige, E. Guerra, J. S. Cuadrado, J. de Lara, I. Ráth, D. Varró, M. Tisi, and J. Cabot, "A research roadmap towards achieving scalability in model driven engineering," in *Proceedings of the 1st International Workshop on Scalability in Model Driven Engineering (BigMDE 2013)*, 2013, pp. 2:1–2:10.

[15] F. Basciani, J. D. Rocco, D. D. Ruscio, A. D. Salle, L. Iovino, and A. Pierantonio, "MDEForge: An Extensible Web-Based Modeling Platform," in *Proceedings of the 2nd International Workshop on Model-Driven Engineering on and for the Cloud (CloudMDE 2014) co-located with the 17th International Conference on Model Driven Engineering Languages and Systems (MoDELS)*. CEUR-WS.org, 2014, pp. 66–75.

[16] J. Espinazo Pagan and J. Garcia Molina, "Querying large models efficiently," *Information and Software Technology*, vol. 56, no. 6, pp. 586–622, 2014.

[17] A. Gómez, M. Tisi, G. Sunyé, and J. Cabot, "Map-based transparent persistence for very large models," in *Proceedings of the 18th International Conference on Fundamental Approaches to Software Engineering (FASE 2015)*. Springer, 2015, pp. 19–34.

[18] A. Benelallam, A. Gómez, G. Sunyé, M. Tisi, and D. Launay, "Neo4EMF, A Scalable Persistence Layer for EMF Models," in *Proceedings of the 10th European Conference on Modelling Foundations and Applications (ECMFA)*. Springer, 2014, pp. 230–241.

[19] G. Daniel, G. Sunyé, A. Benelallam, and M. Tisi, "Improving Memory Efficiency for Processing Large-Scale Models," in *Proceedings of the 2nd International Workshop on Scalability in Model Driven Engineering (BigMDE 2014)*. CEUR-WS.org, 2014, pp. 31–39.

[20] C. Clasen, M. Didonet Del Fabro, and M. Tisi, "Transforming Very Large Models in the Cloud: a Research Roadmap," in *Proceedings of the 1st International Workshop on Model-Driven Engineering on and for the Cloud (CloudMDE) co-located with the 8th European Conference on Modelling Foundations and Applications (ECMFA)*, 2012, pp. 1–10.

[21] L. Deak, G. Mezei, T. Vajk, and K. Fekete, "Graph partitioning algorithm for model transformation frameworks," in *Proceedings of the International Conference on Computer as a Tool (EUROCON)*. IEEE, 2013, pp. 475–481.

[22] G. Blair, N. Bencomo, and R. France, "Models@ run.time," *Computer*, vol. 42, no. 10, pp. 22–27, 2009.

[23] T. Hartmann, A. Moawad, F. Fouquet, G. Nain, J. Klein, and Y. L. Traon, "Stream my models: Reactive peer-to-peer distributed models@run.time," in *Proceedings of the 18th International Conference on Model Driven Engineering Languages and Systems (MoDELS 2015)*. ACM/IEEE, 2015.

[24] J. Vlissides, R. Helm, R. Johnson, and E. Gamma, *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

[25] S. Khare, K. An, A. S. Gokhale, S. Tambe, and A. Meena, "Reactive stream processing for data-centric publish/subscribe," in *Proceedings of the 9th International Conference on Distributed Event-Based Systems (DEBS 2015)*. ACM, 2015, pp. 234–245.

[26] D. C. Luckham, *The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems*. Addison-Wesley, 2001.

[27] O. Saleh, S. Hagedorn, and K. Sattler, "Complex event processing on linked stream data," *Datenbank-Spektrum*, vol. 15, no. 2, pp. 119–129, 2015.

[28] J. S. Cuadrado and J. de Lara, "Streaming model transformations: Scenarios, challenges and initial solutions," in *Proceedings of the 6th International Conference on Theory and Practice of Model Transformations (ICMT 2013)*. Springer, 2013, pp. 1–16.

[29] W. M. P. van der Aalst, "Process mining," *Commun. ACM*, vol. 55, no. 8, pp. 76–83, 2012.

[30] W. M. P. van der Aalst, A. Adriansyah, A. K. A. de Medeiros, F. Arcieri, T. Baier, T. Blickle, R. P. J. C. Bose, P. van den Brand, R. Brandtjen, J. C. A. M. Buijs, A. Burattin, J. Carmona, M. Castellanos, J. Claes, J. Cook, N. Costantini, F. Curbera, E. Damiani, M. de Leoni, P. Delias, B. F. van Dongen, M. Dumas, S. Dustdar, D. Fahland, D. R. Ferreira, W. Gaaloul, F. van Geffen, S. Goel, C. W. Günther, A. Guzzo, P. Harmon, A. H. M. ter Hofstede, J. Hoogland, J. E. Ingvaldsen, K. Kato, R. Kuhn, A. Kumar, M. La Rosa, F. M. Maggi, D. Malerba, R. S. Mans, A. Manuel, M. McCreesh, P. Mello, J. Mendling, M. Montali, H. R. M. Nezhad, M. zur Muehlen, J. Munoz-Gama, L. Pontieri, J. Ribeiro, A. Rozinat, H. S. Pérez, R. S. Pérez, M. Sepúlveda, J. Sinur, P. Soffer, M. Song, A. Sperduti, G. Stilo, C. Stoel, K. D. Swenson, M. Talamo, W. Tan, C. Turner, J. Vanthienen, G. Varvaressos, E. Verbeek, M. Verdonk, R. Vigo, J. Wang, B. Weber, M. Weidlich, T. Weijters, L. Wen, M. Westergaard, and M. T. Wynn, "Process mining manifesto," in *Proceedings of the Business Process Management Workshops (BPM)*. Springer, 2011, pp. 169–194.

[31] M. Dumas, W. M. P. van der Aalst, and A. H. M. ter Hofstede, *Process-Aware Information Systems: Bridging People and Software Through Process Technology*. Wiley, 2005.

[32] W. M. P. van der Aalst, T. Weijters, and L. Maruster, "Workflow mining: Discovering process models from event logs," *Trans. Knowl. Data Eng.*, vol. 16, no. 9, pp. 1128–1142, 2004.

[33] D. Agrawal, P. Bernstein, E. Bertino, S. Davidson, U. Dayal, M. Franklin, J. Gehrke, L. Haas, A. Halevy, J. Han *et al.*, "Challenges and opportunities with big data," Purdue University, Cyber Center Technical Reports, Tech. Rep., 2011.

[34] B. F. van Dongen and W. M. P. van der Aalst, "A meta model for process mining data," in *Proceedings of the International Workshop on Enterprise Modelling and Ontologies for Interoperability (EMOI 2005) co-located with the 17th Conference on Advanced Information Systems Engineering (CAiSE)*, 2005.

[35] P. M. Domingos and G. Hulten, "Catching up with the data: Research issues in mining data streams," in *Proceedings of the 6th International Workshop on Research Issues in Data Mining and Knowledge Discovery (DMKD)*, 2001.

[36] M. Meir-Huber and M. Köhler, "Big Data in Austria," Austrian Ministry for Transport, Innovation and Technology (BMVIT), Tech. Rep., 2014.

[37] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery: An overview," in *Advances in Knowledge Discovery and Data Mining*, U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Eds. AAAI, 1996, pp. 1–34.

[38] L. Kelly, S. Dungs, S. Kriewel, A. Hanbury, L. Goeuriot, G. J. F. Jones, G. Langs, and H. Müller, "Khresmoi professional: Multilingual, multimodal professional medical search," in *Proceedings of the 36th European Conference on Advances in Information Retrieval (ECIR 2014)*. Springer, 2014, pp. 754–758.

[39] E. Alpaydin, *Introduction to Machine Learning*, 2nd ed. The MIT Press, 2010.

[40] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer, 2001.

[41] U. Leser and F. Naumann, *Informationsintegration*. dpunkt.verlag, 2007.

[42] J. Euzenat and P. Shvaiko, *Ontology Matching*. Springer, 2007.

[43] D. Ameller, X. Franch, C. Gómez, J. Araújo, R. Berntsson-Svensson, S. Biffl, J. Cabot, V. Cortellessa, M. Daneva, D. M. Fernández, A. Moreira, H. Muccini, A. Vallecillo, M. Wimmer, V. Amaral, H. Brunelière, L. Burgueño, M. Goulão, B. Schätz, and S. Teufl, "Handling non-functional requirements in model-driven development: An ongoing industrial survey," in *Proceedings of the 23rd IEEE International Requirements Engineering Conference (RE 2015)*. IEEE Computer Society, 2015, pp. 208–213.

[44] G. Kappel, J. Schönböck, M. Wimmer, G. Kotsis, A. Kusel, B. Pröll, W. Retschitzegger, W. Schwinger, R. R. Wagner, and S. Lechner, "TheHiddenU - A Social Nexus for Privacy-assured Personalisation Brokerage," in *Proceedings of the 12th International Conference on Enterprise Information Systems (ICEIS 2010)*. SciTePress, 2010, pp. 158–162.

[45] G. Taentzer, C. Ermel, P. Langer, and M. Wimmer, "Conflict Detection for Model Versioning Based on Graph Modifications," in *Proceedings of the 5th International Conference on Graph Transformations (ICGT 2010)*. Springer, 2010, pp. 171–186.

[46] Object Management Group, *Data Distribution Service (DDS)*, Object Management Group Std., Rev. 1.4, 2015. [Online]. Available: \url{http://www.omg.org/spec/DDS/1.4/}
[47] D. F. Barbieri, D. Braga, S. Ceri, E. D. Valle, and M. Grossniklaus, "Querying RDF streams with C-SPARQL," *SIGMOD Record*, vol. 39, no. 1, pp. 20–26, 2010.
[48] ——, "C-SPARQL: SPARQL for continuous querying," in *Proceedings of the 18th International Conference on World Wide Web (WWW)*. IW3C2, 2009, pp. 1061–1062.
[49] J. Troya, M. Wimmer, L. Burgueño, and A. Vallecillo, "Towards approximate model transformations," in *Proceedings of the International Workshop on Analysis of Model Transformations (AMT 2014) co-located with 17th International Conference on Model Driven Engineering Languages & Systems (MoDELS)*. CEUR-WS.org, 2014, pp. 44–53.
[50] M. Zinkevich, "Online convex programming and generalized infinitesimal gradient ascent," in *Proceedings of the 20th International Conference on Machine Learning (ICML)*. Addison-Wesley Professional, 2003, pp. 928–936.
[51] C. C. Aggarwal, *Data Streams - Models and Algorithms*. Springer, 2007.
[52] G. A. Einicke, "Iterative frequency-weighted filtering and smoothing procedures," *Signal Processing Letters*, vol. 21, no. 12, pp. 1467–1470, 2014.
[53] W. N. Venables and B. D. Ripley, *Modern Applied Statistics with S-PLUS*, ser. Statistics and computing. Springer, 1999.
[54] J. Gama, R. Sebastião, and P. P. Rodrigues, "On evaluating stream learning algorithms," *Machine Learning*, vol. 90, no. 3, pp. 317–346, 2013.
[55] Object Management Group, *UML Profile for MARTE: Modeling and Analysis of Real-Time Embedded Systems*, Object Management Group Std., Rev. 1.1, 2011. [Online]. Available: \url{http://www.omg.org/spec/MARTE/}

9 Model-Driven Time-Series Analytics

S. Wolny, A. Mazak, M. Wimmer, R. Konlechner and G. Kappel;
Journal of Conceptual Modeling - Enterprise Modelling and Information Systems Architectures (EMISA), Springer, 13(Special), (2018), pp. 252–261.
DOI: 10.18417/emisa.si.hcm.19

International Journal of Conceptual Modeling
February 2018. DOI:10.18417/emisa.si.hcm
252 Sabine Wolny, Alexandra Mazak, Manuel Wimmer, Rafael Konlechner, Gerti Kappel
Special Issue on Conceptual Modelling in Honour of Heinrich C. Mayr

Model-Driven Time-Series Analytics

Sabine Wolny[*,a], Alexandra Mazak[a], Manuel Wimmer[a], Rafael Konlechner[a], Gerti Kappel[a]

[a] CDL-MINT, TU Wien, Austria

Abstract. *Tackling the challenge of managing the full life-cycle of systems requires a well-defined mix of approaches. While in the early phases model-driven approaches are frequently used to design systems, in the later phases data-driven approaches are used to reason on different key performance indicators of systems under operation. This immediately poses the question how operational data can be mapped back to design models to evaluate existing designs and to reason about future re-designs. In this paper, we present a novel approach for harmonizing model-driven and data-driven approaches. In particular, we introduce an architecture for time-series data management to analyse runtime properties of systems which is derived from design models. Having this systematic generation of time-series data management opens the door to analyse data through design models. We show how such data analytics is specified for modelling languages using standard metamodelling techniques and technologies.*

Keywords. Model-Driven Engineering • Time-Series • Data Analytics • Language Engineering

1 Introduction

In model-driven engineering (MDE), models are the central artefact and used as a main driver throughout the software development process, finally leading to an automated generation of software systems (Lara et al. 2015). In the current state-of-practice in MDE (Brambilla et al. 2017; Karagiannis et al. 2016), models are used as an abstraction and generalization of a system to be developed. By definition, a model never describes reality in its entirety, rather it describes a scope of reality for a certain purpose in a given context (Brambilla et al. 2017). Thus, models are mostly used as *prescriptive models* for creating a software system (Heldal et al. 2016). Such design models determine the scope and details of a domain of interest to be studied. For this purpose, different types of general modelling languages (e. g., state charts, class diagrams, etc.) may be used or domain-specific languages (DSLs) (Karagiannis et al. 2016) may be employed. It has to be emphasized that engineers typically have the desirable behaviour in mind when creating a system, since they are not aware in these early phases of many deviations that may take place at runtime (van der Aalst 2016).

According to Brambilla et al. (2017) the implementation phase deals with the mapping of prescriptive models to some executable systems and consists of three levels: (*i*) the *modelling level* where the models are defined, (*ii*) the *realization level* where the solutions are implemented through artefacts that are used in the running system, and (*iii*) the *automation level* where mappings from the modelling to the realization phase are made. However, these levels are currently only used for down-stream processes. The possibility of up-stream processes is mostly neglected in MDE (Mazak and Wimmer 2016). Especially, for later phases of the system lifecycle *descriptive*

* Corresponding author.
E-mail. wolny@big.tuwien.ac.at
This work has been supported by the National Foundation for Research, Technology and Development (CDG), the Austrian Federal Ministry of Science, Research, and Economy (BMWFW Austria), and the TU Wien research funds.

Enterprise Modelling and Information Systems Architectures
February 2018. DOI:10.18417/emisa.si.hcm
Model-Driven Time-Series Analytics 253
Special Issue on Conceptual Modelling in Honour of Heinrich C. Mayr

models may be employed to better understand how the system is actually realized and how it is operating in a certain environment (Mazak and Wimmer 2016). Compared to prescriptive models, those descriptive models are only marginal explored in the field of MDE, and if used at all, they are built manually.

In this paper, we move towards a well-defined mix of approaches to better manage the full lifecycle of systems by combining prescriptive and descriptive model types. In particular, we introduce a model-driven time-series data analytics architecture for harmonizing model-driven and data-driven approaches. Based on this architecture, we show how data analytics can be specified for modelling languages using standard metamodelling techniques. This means, design-oriented languages are equipped with extensions for representing runtime states as well as runtime histories, which in turn allow the formulation and computation of runtime properties with the Object Constraint Language (OCL). This approach has the advantage to directly interpret measurements and events within the design models without introducing an impedance mismatch.

The remainder of this paper is structured as follows. Section 2 provides the background for this paper by introducing a motivating example which is subsequently used as running example. Section 3 gives an overview of our architecture for unifying model-driven and data-driven approaches. In Section 4, we present in detail how time-series analytics can be integrated in metamodels. Section 5 discusses the related work. Finally, in Section 6, we conclude with an outlook on future work.

2 Motivating Example

In this section, we introduce a motivating example, which will subsequently become the running example of this paper. We first describe the example from the modelling perspective, then from the realization perspective with a focus on runtime data collection, and finally conclude with the challenges we aim to address with this paper.

Model-Driven Perspective

As our motivating example, we consider a grip-arm robot (gripper) with different position properties of axis angles: `BasePosition (BP)`, `MainArmPosition (MAP)`, and `GripperPosition (GP)`. From a device point of view (cf. Figure 1(a)), the structure of the gripper component and its behaviour are modelled at design time by a subset of a SysML-like language, i.e., blocks with associated state machines. The top of Figure 1(a) shows the specific properties (`BP`,`MAP`,`GP`) of the block, whereas the actual property values depend on the different states (e.g., `Idle`, `Pick Up`). The states are given at the bottom of Figure 1(a).

By the given state machine, property value changes are modelled. The gripper has certain positions at initialization, in state `Idle` and in state `Pick Up`. The assumption of the modelled state machine is that as soon these states are reached, the position values are set. However, such state machines are a kind of black box, where only the discrete values before entering the state and after leaving the state are known (cf. Figure 1(b)). While this may be sufficient for several design tasks and discrete systems, for continuous systems more information may be required. This is in particular true for our example case. The gripper represents a continuous system, since it does not immediately realize the next position, but needs time to move to the given place. Usually, such information is not directly given in a design model, but it may be important for several tasks such as optimization, validation, and verification. The ability to observe property value changes over time within states may contribute to capture the current capabilities and shortcomings of systems. Thus, the presented approach of this paper aims to transform the black box into a so-called "grey box" to make the effects of value changes visible (see Figure 1(c)). For instance, observation sequences of property value changes are an important base information of a system's operation to compute operating figures to check if the behaviour of each

International Journal of Conceptual Modeling
February 2018. DOI:10.18417/emisa.si.hcm
254 Sabine Wolny, Alexandra Mazak, Manuel Wimmer, Rafael Konlechner, Gerti Kappel
Special Issue on Conceptual Modelling in Honour of Heinrich C. Mayr

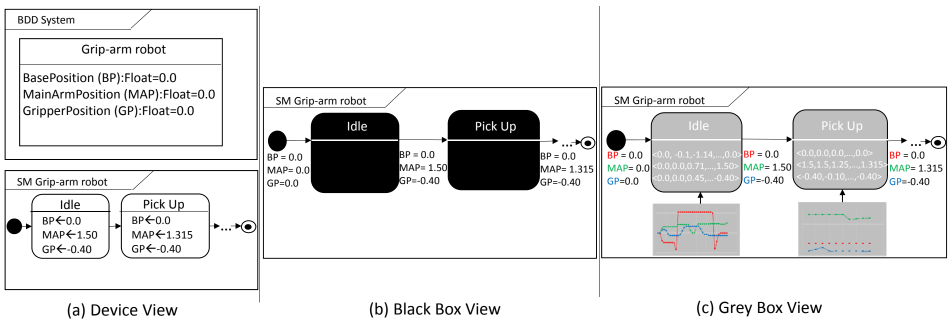


Figure 1: Different model-based views on a grip-arm robot.

gripper's axis corresponds to the defined one in the design model.

Data-Driven Perspective

For the technical realization of our example, we developed a simulation model of the gripper consisting of three angle sensors, which we executed by the open source tool Blender[1]. We deploy the scenario of a pick-and-place unit, where the gripper picks up different color-coded work pieces, place them on a test rig, picks the items up again and puts them down, depending on their red or green color, in two different storage boxes. The simulation environment receives its commands via Message Queue Telemetry Transport (MQTT) from a server controller implemented with Kotlin[2]. The simulation enables to acquire transient data streams in real-time from the angle axes of the gripper (BP, MAP, GP, unit is radian), which are equipped with sensors.

To react on events of interest provided by these data streams, we employ the publish/subscribe pattern. In our example, we subscribe to the sensor topic to receive in a temporal distance of 15 milliseconds the filtered data streams of the sensors of the gripper during simulation. Thereby, we are interested in property value changes (i. e., positions of the axes) in the simulation at given points in time. Messages from the sensor topic are defined in JSON[3] specifying the sending unit as well as the measured data. The following example shows such a message from the angle sensors of the gripper to the controller.

```
{"entity": "GripperArm",
"basePosition": 0.0,
"mainArmPosition": 0.0,
"gripperPosition": 0.0}
```

This example shows the positions of the angle axes at system initialization (see Figure 1). The angle position of each axis has the value 0.0. The default range of the angle values is $[-\pi,\pi]$. To analyze our scenario, it is important to save the measured data over time. For this purpose, we use the time series database InfluxDB[4]. InfluxDB allows us to store a large amount of time-stamped data. In addition, by the tool Grafana[5] we can visualize our stored sensor values.

Challenges

Our motivating example is discussed from two angles: (*i*) from the model-driven, i. e., how the intended system should work, and (*ii*) from the data-driven, i. e., how measurements can be taken from the running system to reason about the actual realization. While the first perspective is lacking concepts to define runtime data such as

[1] https://www.blender.org
[2] https://kotlinlang.org
[3] http://json.org/example.html
[4] https://www.influxdata.com
[5] https://grafana.com

Enterprise Modelling and Information Systems Architectures
February 2018. DOI:10.18417/emisa.si.hcm
Model-Driven Time-Series Analytics 255
Special Issue on Conceptual Modelling in Honour of Heinrich C. Mayr

time-series, the second perspective has to correctly interpret the collected measurements. The challenge is how to overcome the gap between those two perspectives (*i*) to monitor important data from operation, (*ii*) to align the measurements with the design model in order to provide a semantic anchoring of the data, and (*iii*) to provide meaningful analytics whereas the results of the analytics are interpretable for the given design models to reason about improvements or fulfilments of given requirements.

3 Unifying Architecture for Model-Driven and Data-Driven Approaches

In order to allow a smooth integration of model-driven and data-driven approaches, we present in this section an architecture, which builds on the classical model to system downstream in terms of code generators, but at the same time, supports an upstream in terms of mapping data back to design models. Figure 2 gives an overview of this architecture. In the following section, a more detailed description of the different parts will be presented based on our running example.

The proposed architecture consists of four main parts. First, the left hand side of Figure 2 captures the classical downstream MDE approach (cf. (a) in Figure 2). At the metamodel layer, the `design language` is defined with the help of a metamodelling language (in our setting Ecore). Conforming to the design language, the `design models` are defined at the model level describing the static (i. e., structure) and dynamic aspects (i. e., behaviour) of a system to be developed. For the vertical transition from the modelling to the realization level we assume the existence of model-to-text transformations for code generation. Thus, this part of our architecture describes how we can derive the executable system from the design model as is the state-of-the-art in MDE.

Second, we continue with defining the first part of the up-stream process of runtime data to the design model (cf. (b) in Figure 2). In addition to the actual systems, the `runtime observer` is generated out of the design model. The runtime observer collects important information from the running system to represent the current state of the system. Those observations should not only be recorded by observing the running system, but should be also representable at the model level. Thus, we extend the `design language` with a dedicated `runtime language`. This metamodel defines the syntax to represent snapshots of the running system connected to the design model elements. Those snapshots are represented in the `runtime state models` which extend the `design models` and may be directly updated by the `runtime observer` during runtime. In summary, this part of our architecture maps runtime data at the model level for one single point in time and may be used to monitor a system on the model level.

Third, we define the runtime history of a system (cf. (c) in Figure 2). For reasoning about, e.g., property value distributions, it is important to have the complete history of value changes as starting point as one snapshot is definitely not sufficient for such computations. Thus, in the `time series database` the observations of the running system are stored. Based on these collected observations, the `runtime history models` may be directly updated. These models conform to the `runtime history language`, which is an extension of the `runtime language`. In the `runtime history language`, the syntax is defined for representing histories of runtime phenomena of interest, e.g., property values, events, etc.

Finally, after defining those concepts for storing observation histories at the model level, it is also possible to analyse the stored observations (cf. (d) in Figure 2). For this purpose, we define `runtime properties` based on the Object Constraint Language (OCL) by introducing derived properties for the metamodel elements. These derived properties enable us, e.g., (*i*) to compute descriptive statistics, (*ii*) to evaluate monotony behaviour of value changes, or (*iii*) to compute lower and upper bounds of properties to mention just a few examples. Based on the `runtime properties`, the `runtime property values` are computed by analysing the collected time-series. Thus, runtime

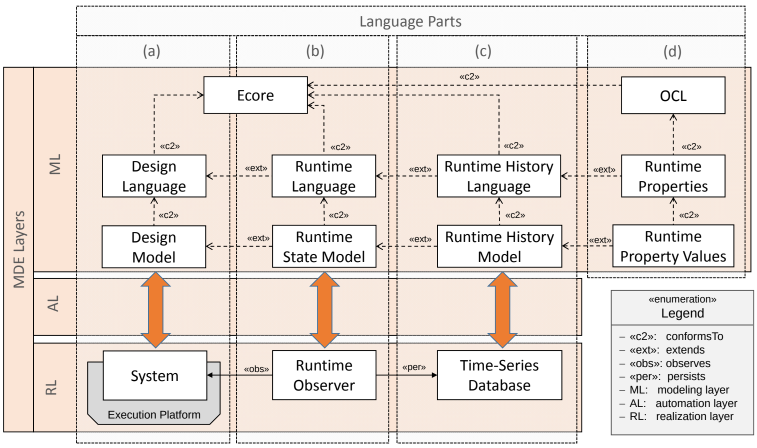


Figure 2: Unifying architecture for model-driven and data-driven perspectives.

data is back propagated to the design models and this mapping allows to interpret the data through the design model elements as there is a clear traceability guaranteed from design elements, runtime states, and runtime histories.

By this architecture, we are able to harmonize model-driven and data-driven approaches, where time-series data management of systems at runtime can be derived from initial design models and be used again at the model layer by importing the time-series to model structures. How this model structures are defined is the topic of the next section.

4 Metamodelling Blueprint for Enhancing Models with Time-Series Analysis

Based on our running example, we further detail in this section how the afore presented architecture can be realized for a given language. In particular, we show for the introduced design modelling language, how the extensions for runtime states, runtime histories, and runtime properties are defined as reusable metamodelling blueprints. The time-series analysis we are focusing on for demonstration purposes is about property value changes of the axis angles (i. e., BP, MAP, GP) of the gripper in our running example.

Design Elements
As already mentioned before, our starting point is the availability of a design modelling language expressed in Ecore. For our running example, we model the structure of the gripper with its properties as a kind of block diagram similar to what is known from SysML. A block has an associated state machine, where different states and transitions are defined. For states, assignments can be defined, which are executed when a state is activated. The assignments in our exemplary language are simple value assignments for the properties of a block. The resulting metamodel for the described design language is shown in Figure 3.

Runtime States
In order to express concrete runtime states on the model level, the metamodel has to be extended with runtime concepts. For this task, there are several existing approaches available, e. g., (Engels et al. 2000; Mayerhofer et al. 2013; Meyers et al. 2014). Most of them add additional metamodel elements to the design language to describe what

Enterprise Modelling and Information Systems Architectures
February 2018. DOI:10.18417/emisa.si.hcm

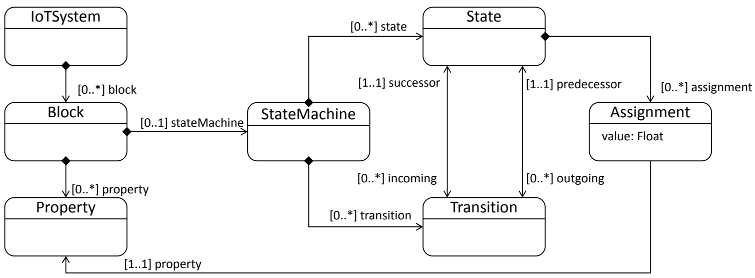


Figure 3: Design metamodel for the running example.

runtime phenomena are of interest and how they are connected to design concepts. For our running example, the `runtime language` is considered as an extension of the design metamodel to allow representing property values for a given point in time (i. e., for a snapshot of the running system). In addition, transitions may fire during runtime. Thus, the concept of transition firing is introduced. While values are considered by measurements during the operation phase, the firing of transitions are categorized as events. Please note that the relation to the design concepts has to be clearly stated by the runtime concepts, e. g., the value concept is related to the property concept. Figure 4 captures the concrete realization of the runtime extension for our design language.

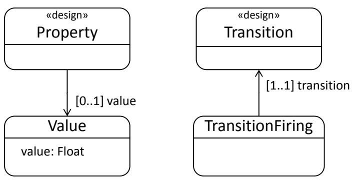


Figure 4: Runtime metamodel for the running example.

Runtime Histories

To reason about operation figures going beyond one snapshot in time such as distributions, upper and lower bounds, histories of property values and event sequences are necessary. Therefore, we need another extension which allows to represent the runtime history of a system. For this, we introduce a novel metamodelling blueprint which introduces the concept of history by providing a sequence of steps having a particular timestamp associated. Figure 5 illustrates the separation of steps into measurement snapshots and event snapshots. These specific steps are forming the event histories and measurement histories. The measurement history contains all measurement snapshots, which comprise values for given time steps. Event histories do the same for events. In our running example, the measurement snapshots refer to the value runtime concept introduced by the runtime extension and the event snapshots are referring to the transition firing concept.

Having this base structure introduced allows us to represent time-series data in design models by using runtime concepts as glue between models and data.

Runtime Properties

For analysing the time-series data represented in the aforementioned runtime history models, we introduce derived properties which actually represent runtime properties. Derived properties have been already used heavily in the past for deriving additional information from given structures and values. As we explicitly represent runtime histories as model structures, we can make use of derived properties to derive runtime information from the base time-series recorded during operation.

In the following we state three runtime properties for the given design language, namely for

International Journal of Conceptual Modeling
February 2018. DOI:10.18417/emisa.si.hcm
258 Sabine Wolny, Alexandra Mazak, Manuel Wimmer, Rafael Konlechner, Gerti Kappel
Special Issue on Conceptual Modelling in Honour of Heinrich C. Mayr

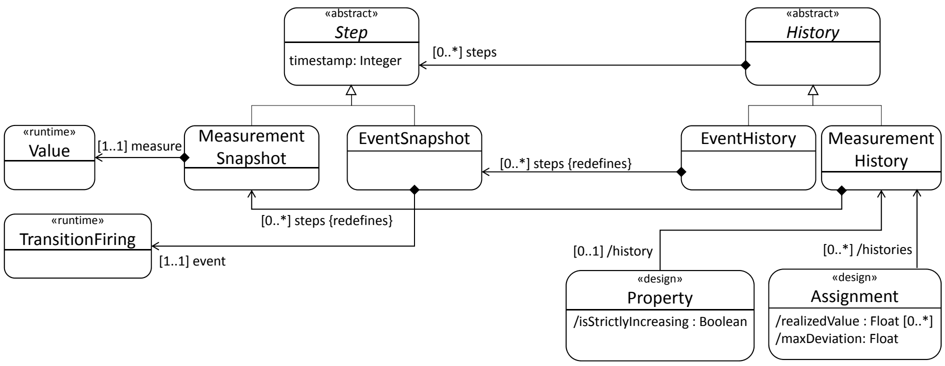


Figure 5: Runtime history metamodel for the running example.

the `Property` metaclass and the `Assignment` metaclass. We use standard OCL to derive the runtime properties.

For properties defined in blocks, it may be of interest if their values are strictly increasing over time or not. This can be expressed in OCL by providing a derived history reference for properties from the complete measurement history. The reference only contains the slice of the full history which concerns the given property. Using this reference, we can simply collect all values as a sequence (the ordering expresses the occurrence of the values). If the sorted sequence corresponds to the base sequence, then the property is strictly increasing.

```
context Property::isStrictlyIncreasing:
    ↪Boolean
derive: self.history.steps.measure.value->
    ↪flatten()->sortedBy(x|x) = self.
    ↪history.steps.measure.value->flatten()
```

Concerning the assignments within states, one may be interested if the stated value is actually realized by the system. For this, the realized values may be collected by taking the last snapshots of all assignment executions for a given assignment.

```
context Assignment::realizedValues:Set(Float
    ↪)
derive: self.histories -> collect(x|x.steps
    ↪-> last()) -> collect(x|x.measure.
    ↪value) -> asSet()
```

Having the set of realized values, the maximum deviation is calculated by introducing another derived property which builds on the previous one.

```
context Assignment::maxDeviation:Float
derive: self.realizedValues -> collect(x|(
    ↪self.value-x).abs()) -> sortedBy(x|x)
    ↪-> last()
```

5 Related Work

In this section, we discuss existing work with respect to the contribution of this paper, namely the combination of model-driven and data-driven approaches with a focus on time-series analytics. Therefore, we first discuss data-driven approaches for enhancing existing domain specific languages (DSLs), and subsequently, we enumerate existing work which proposes dedicated DSLs for time-series analytics.

Data-driven approaches for DSLs
An emerging field for data-enhanced modelling languages is Web engineering. For instance, Bernaschina et al. (2017) point to the fact that there is the need for merging Web site navigation statistics of user behaviour with the structure of

the Web application models. The authors show the advantages of combining user interaction models with user tracking information in form of user navigation logs, and details about the visualized content in the pages. Their approach interweaves design time information and runtime execution data of Web sites in order to significantly improve the analysis of user behaviour. In (Artner et al. 2017), we combined navigation models with Markov chains for representing navigation path probabilities, which are derived from execution logs. While these existing approaches for Web applications follow the general idea of combining data-driven and model-driven approaches, the approach of this paper is independent from the actual domain and may be also used in the future to reproduce these existing specific approaches.

Another very active research field is process mining (van der Aalst 2016) which aims to discover process models from workflow execution logs. A variety of process mining algorithms exists that allows the discovery of different process models in different formalisms. In (Wolny et al. 2017), we present an initial architecture how process mining may be related with time-series mining. By this, not only the dependencies between different process steps may be uncovered, but also dependencies between data and process steps are approachable.

Finally, in (Hartmann et al. 2017) the authors present a DSL which allows not only the specification of structural aspects of a systems, but also the definition of so-called learned properties. Such properties are computed from runtime data by using some kind of machine learning algorithms. Our approach directly allows to encode such properties as derived properties based on time-series data computed with OCL as we model the runtime history explicitly. In future work, it will be interesting to combine our time-series analysis with machine learning algorithms as proposed by Hartmann et al. (2017).

DSLs for Time-Series Analytics

The OMS3 modelling framework[6] introduces an extensible and lightweight layer for a simulation description expressed as Simulation DSL by using Groovy[7] as a framework for providing the code generator implementation. In (David et al. 2012), the authors present DSL variants in OMS3, e. g., the Ensemble Streamflow Prediction (ESP) DSL. This DSL uses time-series of historic meteorological data as model input to predict future conditions. In their approach, DSLs are employed for time-series unlike in our approach, where we use time-series for domain-specific modelling.

Gekko[8] is an open-source modelling approach for time-series data management and for solving and analysing large-scale time-series models. Gekko may be considered as a kind of DSL with a time-series domain focus. It provides interfaces to statistic packages such as R. In our approach, we use an open-source time-series database which offers besides high-availability storage and monitoring of time-series, application metrics and real-time analytics in addition. Nevertheless, in future work it is of interest to evaluate different possibilities to perform time-series analytics in addition to our current approach.

6 Conclusion and Future Work

In this paper, we have introduced an unifying architecture for combining model-driven and data-driven approaches for system engineering. By this architecture, we allow for specifying and computing runtime properties based on time-series data through design models. The extensions needed on the metamodel level are non-intrusive and connected to existing approaches for specifying the operational semantics of languages. The presented runtime history metamodel fragments are applicable for any design modelling language comprising features to be measured and events to be tracked as the current metamodelling languages Ecore and OCL are reused. We demonstrated our

[6] https://alm.engr.colostate.edu/cb/project/oms
[7] http://groovy-lang.org
[8] http://t-t.dk/gekko

International Journal of Conceptual Modeling
February 2018. DOI:10.18417/emisa.si.hcm

approach for a cyber-physical production system case. We have also realized a prototype in Eclipse supporting our approach which is available on our project website[9] .

While the presented approach opens the door for using time-series analytics in a model-driven engineering toolbox, there are still several challenges to be tackled in the future. In particular, we consider the following points on our roadmap: *scalability* (e. g., should the analysis be performed on the model level or directly in the time-series database?), *expressivity* (e. g., which extensions of OCL are necessary for statistical reasoning?), *understandability* (e. g., how to visualize time-series oriented information in diagrams?), and *predictability* (e. g., how to derive and use operations from time-series for predicting future runtime states?).

[9] https://cdl-mint.big.tuwien.ac.at/case-study-artefacts-for-emisa-2017

References

Artner J., Mazak A., Wimmer M. (2017) Towards Stochastic Performance Models for Web 2.0 Applications. In: Proceedings of the 17th International Conference on Web Engineering (ICWE). Springer, pp. 360–369

Bernaschina C., Brambilla M., Mauri A., Umuhoza E. (2017) A Big Data Analysis Framework for Model-Based Web User Behavior Analytics. In: Proceedings of the 17th International Conference on Web Engineering (ICWE). Springer, pp. 98–114

Brambilla M., Cabot J., Wimmer M. (2017) Model-Driven Software Engineering in Practice. Morgan & Claypool

David O., Lloyd W., II J. C. A., Green T. R., Olson K., Leavesley G. H., Carlson J. (2012) Domain Specific Languages for Modeling and Simulation: Use Case OMS3. In: Proceedings of the International Congress on Environmental Modelling and Software Managing Resources of a Limited Planet, pp. 1201–1207

Engels G., Hausmann J. H., Heckel R., Sauer S. (2000) Dynamic Meta Modeling: A Graphical Approach to the Operational Semantics of Behavioral Diagrams in UML. In: Proceedings of the 3rd International Conference on the Unified Modeling Language (UML). Springer, pp. 323–337

Hartmann T., Moawad A., Fouquet F., Le Traon Y. (2017) The next evolution of MDE: a seamless integration of machine learning into domain modeling. In: Software & Systems Modeling

Heldal R., Pelliccione P., Eliasson U., Lantz J., Derehag J., Whittle J. (2016) Descriptive vs prescriptive models in industry. In: Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems (MODELS). ACM, pp. 216–226

Karagiannis D., Mayr H. C., Mylopoulos J. (2016) Domain-Specific Conceptual Modeling, Concepts, Methods and Tools. Springer

de Lara J., Guerra E., Cuadrado J. S. (2015) Model-driven engineering with domain-specific meta-modelling languages. In: Software and System Modeling 14(1), pp. 429–459

Mayerhofer T., Langer P., Wimmer M., Kappel G. (2013) xMOF: Executable DSMLs Based on fUML. In: Proceedings of the 6th International Conference on Software Language Engineering (SLE), pp. 56–75

Mazak A., Wimmer M. (2016) Towards Liquid Models: An Evolutionary Modeling Approach. In: Proceedings of the 18th IEEE Conference on Business Informatics (CBI). IEEE, pp. 104–112

Meyers B., Deshayes R., Lucio L., Syriani E., Vangheluwe H., Wimmer M. (2014) ProMoBox: A Framework for Generating Domain-Specific Property Languages. In: Proceedings of the 7th International Conference on Software Language Engineering (SLE). Springer, pp. 1–20

van der Aalst W. M. P. (2016) Process Mining - Data Science in Action. Springer

Enterprise Modelling and Information Systems Architectures
February 2018. DOI:10.18417/emisa.si.hcm
Model-Driven Time-Series Analytics 261
Special Issue on Conceptual Modelling in Honour of Heinrich C. Mayr

Wolny S., Mazak A., Konlechner R., Wimmer M. (2017) Towards Continuous Behavior Mining. In: Proceedings of the 7th International Symposium on Data-driven Process Discovery and Analysis (SIMPDA), pp. 149–150

10 Temporal Models on Time Series Databases

A. Mazak, S. Wolny, A. Gómez, J. Cabot, M. Wimmer and G. Kappel;
Journal of Object Technology, Springer, 19(3), (2020), pp. 3:1–3:15.
DOI: 10.5381/jot.2020.19.3.a14

JOT JOURNAL OF OBJECT TECHNOLOGY

Journal of Object Technology | RESEARCH ARTICLE

Temporal Models on Time Series Databases

Alexandra Mazak*, **Sabine Wolny**†, **Abel Gómez**‡, **Jordi Cabot**§, **Manuel Wimmer**†, **and Gerti Kappel****

*Institute for Subsurface Engineering - Digital Transformation in Tunnelling, Montanuniversität Leoben, Austria
†CDL-MINT, Johanens Kepler University Linz, Austria
‡Internet Interdisciplinary Institute (IN3), Universitat Oberta de Catalunya (UOC), Spain
§ICREA, Spain
**Business Informatics Group, TU Wien, Austria

ABSTRACT With the emergence of Cyber-Physical Systems (CPS), several sophisticated runtime monitoring solutions have been proposed in order to deal with extensive execution logs. One promising development in this respect is the integration of time series databases that support the storage of massive amounts of historical data as well as to provide fast query capabilities to reason about runtime properties of such CPS.

In this paper, we discuss how conceptual modeling can benefit from time series databases, and vice versa. In particular, we present how metamodels and their instances, i.e., models, can be partially mapped to time series databases. Thus, the traceability between design and simulation/runtime activities can be ensured by retrieving and accessing runtime information, i.e., time series data, in design models. On this basis, the contribution of this paper is four-fold. First, a dedicated profile for annotating design models for time series databases is presented. Second, a mapping for integrating the metamodeling framework EMF with InfluxDB is introduced as a technology backbone enabling two distinct mapping strategies for model information. Third, we demonstrate how continuous time series queries can be combined with the Object Constraint Language (OCL) for navigation through models, now enriched with derived runtime properties. Finally, we also present an initial evaluation of the different mapping strategies with respect to data storage and query performance. Our initial results show the efficiency of applying derived runtime properties as time series queries also for large model histories.

KEYWORDS Runtime Models, Query Languages, Model-Based Analysis, Temporal Modeling, Time Series Databases.

1. Introduction

With the emergence of Cyber-Physical Systems (CPS) and sophisticated runtime monitoring infrastructures, time series databases (Bader et al. 2017) are nowadays frequently applied to store historical data of systems as well as to provide powerful analysis by dedicated query languages.

At the same time, Model-Driven Engineering (MDE) (Brambilla et al. 2017) approaches are a promising line for dealing with the complexity of designing CPS. However, in recent years the scope of MDE has been also extended to runtime aspects of CPS (Mazak & Wimmer 2016; Benelallam et al. 2017; Cruz, Sadovykh, Truscan, Brunelière, et al. 2020; Bencomo et al. 2019; Gogolla et al. 2019; Kästner et al. 2018).

Several approaches for dealing with runtime data in models have been proposed which are often referred to temporal models in analogy to temporal databases (Gómez et al. 2018; Wolny et al. 2018; Bill et al. 2017). Temporal models go beyond representing and processing the current state of systems. By this, they extend research done in the last decades where several dedicated mappings from design models to different database technologies following different data paradigms have been proposed, e.g., see (Gogolla 2005). However, currently there is a lack of approaches which deal with the explicit mapping of design models to time series databases which can be considered as a special type of temporal databases (Schmidt et

JOT reference format:
Alexandra Mazak, Sabine Wolny, Abel Gómez, Jordi Cabot, Manuel Wimmer, and Gerti Kappel. *Temporal Models on Time Series Databases.* Journal of Object Technology. Vol. 19, No. 3, 2020.
http://dx.doi.org/10.5381/jot.2020.19.3.a14

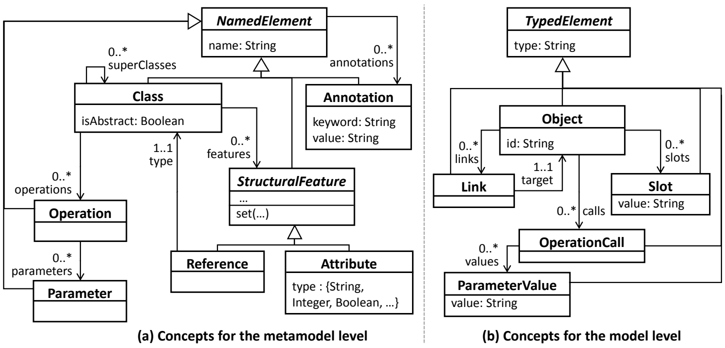


Figure 1 Excerpt of Ecore: (a) concepts for defining metamodels and (b) concepts for defining models.

al. 1995; Böhlen et al. 2018). Such mappings are required to further close the gap between design time modeling activities and simulation/runtime monitoring activities (Gogolla et al. 2019), which employ time series analytics. For instance, time series representations and analytics are foreseen in the development of SysML v2 (Wolny et al. 2020) in order to deal with additional activities in engineering technical systems such as computing different key performance indicators for running systems by applying aggregation functions such as mean, max, mode, etc.

To tackle this limitation, we propose in this paper a novel partial mapping from metamodels and their instances, i.e., the models, to time series databases. The partial mapping deals with the fact that most often not all model elements contribute to a time series, and thus, only those elements which have a runtime history are explicitly mapped to time series structures. Therefore, we propose a dedicated profile for extending the metamodels with appropriate annotations to drive and optimize the generation of model-based time series database connectors. In addition, we propose a Model-to-Time Series (M2TS) mapper that allows to inject data to time series databases from model changes as well as to extract data from the time series databases by model-based queries in OCL (Cabot & Gogolla 2012). By providing these features, we allow for model simulation runs which may be analysed by time series analytics as well as allow for model-based runtime monitoring of systems reporting their changes and states to time series databases. We demonstrate both scenarios by a production system case study and evaluate in particular two mapping strategies with respect to the required data storage as well as query answering performance.

The reminder of this paper is structured as follows. The foundations of this work are introduced in Section 2, namely MDE, in particular, metamodeling, and time series databases. A model to time series mapping approach is proposed in Section 3 incorporating two mapping strategies, which are subsequently evaluated in Section 4 by a case study based on a productions system demonstrator project. Section 5 presents research work marrying MDE-based approaches with temporal aspects concerning linking, versioning, languages, analytics, etc., before we conclude the paper in Section 6 with some directions for future research.

2. Background

In this section, we describe the background of this work, i.e., (meta)modeling and time series databases.

2.1. Metamodeling

Model-driven Engineering (MDE) considers models as first class citizens (Bézivin et al. 2014). A model is used to describe an abstraction of reality for a specific purpose. The basis of such models are modeling languages which are defined by their metamodels. Metamodels are used to describe the abstract syntax of modeling languages. Models created by using a modeling language are instances of the metamodel, and thus, conform to it (Bézivin et al. 2014). One of the best known modeling languages (amongst others) is the Unified Modeling Language[1] (UML) which bases on the Meta Object Facility[2] (MOF) standard. The advantages of UML are platform independence as well as adaption and extension capabilities for users to meet their own requirements for a specific purpose. UML offers a wide range of views and different types of diagrams to represent the structure and behavior of a system to be modeled. One example of a metamodeling language which is based on a core subset of UML and MOF is Ecore from the Eclipse Modeling Framework[3] (EMF). Since Ecore supports the key concepts of using models as input to development and integration tools, it is one of the most widely used languages for code generation and model serialization for data interchange.

In our approach, we focus on Ecore. For illustrating metamodels and models we employ as concrete syntax UML class diagrams and UML object diagrams, respectively. Figure 1 shows excerpts of (a) Ecore's concepts for defining metamodels and (b) Ecore's concepts for representing instances of the metamodels, i.e., models. Models are represented by object graphs and consist of `objects` (instances of `classes`), `slots`

[1] https://www.uml.org
[2] https://www.omg.org/mof
[3] https://www.eclipse.org/modeling/emf

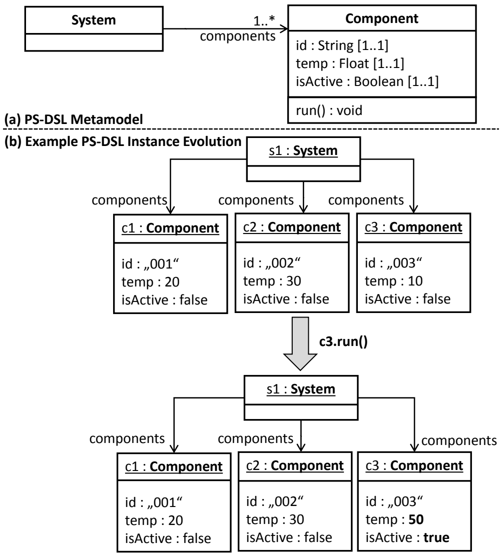


Figure 2 Example: (a) PS-DSL metamodel and (b) model instance evolution.

for storing values (instances of `attributes`), `calls` for executing operations (instances of `operations`) with particular values (instances of `parameters`), and `links` between objects (instances of `references`).

UML object graphs have to conform to the given UML class diagrams. For instance, this means that if an object is existing in the object graph, a corresponding concrete class must exist in the metamodel which act as type for the object.

Additionally, with Ecore, metamodel elements might be annotated with further information (so-called `annotations`), e.g., for tagging elements for particular platforms or purposes as we will see also later in the context of this work.

On the basis of these two metamodels, Figure 2 shows an example of an excerpt of a Production System Domain Specific Language (PS DSL) and an example instantiation of it. The DSL in Figure 2(a) shows that a `system` consists of various `components`. Each component has an unique `id`, a temperature value (`temp`), a property for showing if the component is active or not (`isActive`) and a method `run()` which is active when the component is busy processing an order.

Based on this discussed metamodel, Figure 2(b) shows an instance of a particular system. This system `s1` consists of three different components (`c1`-`c3`) with different property values. If c3 is starting to work (`c3.run()`), the property values of `c3`, more specific the temperature value (`temp`) and the `isActive` value, are changing. Thus, the system state is changing over time and in the shown example only snapshots of the system at a specific point in time are represented (Gogolla et al. 2014).

2.2. Time Series Database

A time series (TS) is a sequence of data points acquired by repeatedly measuring certain parameters (e.g., temperature) over time. The measured values are stored together with the timestamps at which the measurements are taken (Jensen et al. 2017). Although the measurements are usually performed at regular intervals (default in milliseconds), regularity is not a mandatory requirement. The increased interest on this data is in particular the result of the ongoing development in the CPS domain with its IoT technologies as described in the introduction, in which the number of sensors that regularly measure defined conditions is constantly increasing, e.g., for an efficient runtime monitoring.

Time series databases are used for storing, processing, querying as well as analyzing this data generated over time (Bader et al. 2017). Such data consists of timestamps, corresponding values, and optional tags which can consist of names and values (both mostly alphanumeric). Queries can be executed for timestamps or intervals without having to model the data into another structure (Bader et al. 2017). Since the TSDB is not only used for simply collecting data, the term “Time Series Database” (TSDB) is synonymous to the term “Time Series Database Management System” as a kind of software with specialized functions such as compressing or aggregating time series data (Kholod et al. 2017). As mentioned above such time series data is metering from a lot of different sensors. For storing these large amount of data with sufficiently high performance, TSDBs provide the relevant scalability (Jensen et al. 2019).

The level of granularity depends on the type of time series data and the requirements for data analysis, especially since not every time series has to be measured at the same level of detail in order to gain valuable insights of the monitored system (Bader et al. 2017). As an example, the half-hourly measurement of temperature in several rooms of an office building can be mentioned. In this example the granularity is 30 minutes. The values of a tag called “room” can then further specify to which room of the house the measured temperature (value of the time series) belongs.

Time series data differs from other data sets in that it is usually added as a new entry in a TSDB, and therefore, already stored entries are not overwritten (Kholod et al. 2017). Exceptions may only caused by the correction of faulty data, e.g., due to delayed measurements or a failure of sensors. Therefore TSDBs allow the recording and analysis of massive historical data, e.g., for anomaly detection or predictive analytics (Mazak et al. 2018). Thus, any changes over time can be traced nearly in a seamless manner. The storage of time series data, the analysis, and the monitoring of any changes over time provide a great deal of informative added value compared to other types of data, which can only represent a current status (Kholod et al. 2017). In our approach, we use InfluxDB[4] an open source TSDB by which we can continuously store and query data independently of another DBMS (Bader et al. 2017). For querying, it provides a SQL-like language, and for storing it provides rules for (long-

[4] https://www.influxdata.com/products

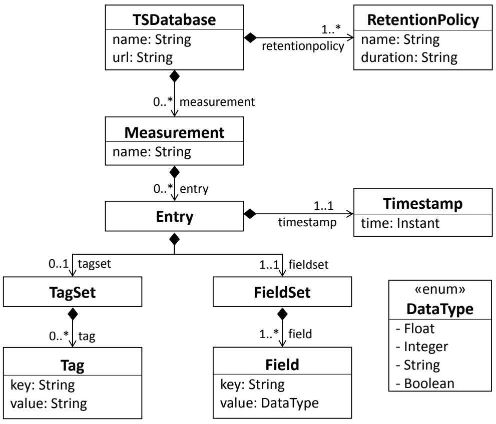


Figure 3 TSDB metamodel.

term) data storage. For instance, InfluxDB enables flexible data aggregations based on the timing factor and running calculations of functions (e.g., average temperature per hour).

Figure 3 shows a metamodel of the TSDB. The `TSDatabase` has a specific `name` and consists of various `Measurements`. Each measurement must consist of a `Timestamp` and a `FieldSet` where the different time series values are stored. Optionally, the measurement can have some additional meta-information stored in a `TagSet`. For instance, InfluxDB implements this metamodel and its line protocol informs the database of the measurement, tag set, field set, and timestamp. Listing 1 shows the structure of the line protocol, with first its `measurement`, followed by a optional `TagSet`, followed by a `FieldSet` with at least one field and optionally a `timestamp`. If no timestamp is specified, the current system time is taken by default.

```
<measurement>{,<tag_key>=<tag_value>}␣<field_key>
=<field_value>{,<field_key>=<field_value>}␣[
<timestamp>]
```

Listing 1 Example of the line protocol of InfluxDB.

3. Mapping Models to Time Series Representations

In order to allow an integration of time series storage and analysis in a model-based manner, in this section we present the design rationale for our approach before we outline two mapping strategies from object-oriented models (as described in Section 2) to TSDB.

3.1. A Polyglot for Combining Models with Time Series Databases

For combining models, especially EMF-based models, with TSDB, we aim for a polyglot solution where the static information resides in the model as it is already available, e.g., by XMI or other model persistence mechanisms, and only the time-sensitive information is stored in the TSDB (see Fig. 4).

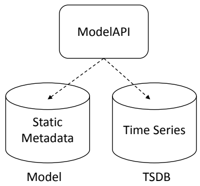


Figure 4 Polyglot solution for models on TSDB.

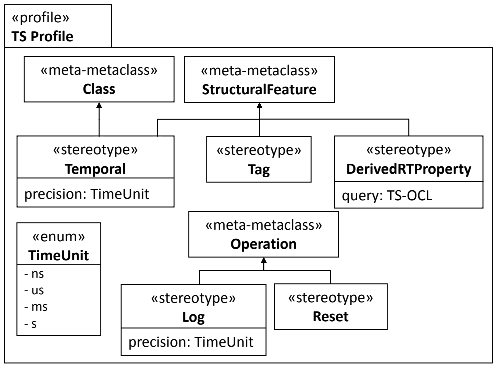


Figure 5 Time Series Profile based on TemporalEMF (Gómez et al. 2018)

These two storage parts are combined by a common `ModelAPI`, which then can be accessed and used by various applications. This unifying API abstracts implementation details and allow for a similar way of working with models as it is provided by EMF out-of-the-box. In particular, we reuse as much as possible and only extend those parts which are really required. As a result, the model is applied as close as possible to the EMF standard and the required information for the TSDB can be attached in a light-weight manner. In order to achieve such unifying API with a polyglot there are various requirements that must be fulfilled. First of all, there has to be a built-in mechanism that determines which information from the model should be transferred to the TSDB and stored there. Second, there should be as well a procedure that extracts data from the TSDB by querying and displaying it back in the model. Third, our temporal extensions should not hinder or pollute the use of models and they should be still manipulated as before. The main goal is to embed this process into a conceptual schema to avoid hard coding the functionalities again and again for different cases. In the following subsections, we describe the design choices of the polyglot.

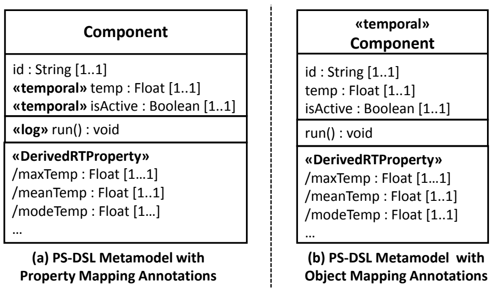


Figure 6 Annotated example metamodel: (a) single property mappings vs. (b) complete object mapping.

3.2. Time Series Profile

In a first step, we propose a profile (realized with EMF Annotations) for extending existing metamodels by time series aspects (cf. Figure 5). The profile defines different kinds of stereotypes for `Classes`, `StructuralFeatures` as well as `Operations`. The stereotype `Temporal` indicates that these elements (classes or structural features such as attributes and references) are temporal features that should be recorded as time series in a TSDB. With the use of `precision`, the accuracy of the recordings can be defined from nanoseconds (`ns`) to seconds (`s`). The stereotype `Tag` should be used for features that represent important metadata of time series features. For instance, the `id` of the room where the temperature is measured. The stereotype `DerivedRTProperty` marks features as derived runtime properties. Derived properties are features where the feature value is computed based on other feature values. Our stereotype is used for defining properties that get their values during runtime based on runtime data stored in the TSDB. For instance, the average temperature of a specific room over a whole day. We introduce this custom stereotype for our derived runtime properties since we use an OCL dialect that need further processing before execution (cf. Section 3.4). Additionally, there are two stereotypes for operations: stereotype `Log` is for logging the start and the end of an operation and stereotype `Reset` is used to refresh the system state, e.g., for a new simulation run.

In Figure 6, we show an example application for the previously presented metamodel in Section 2. In particular, we show on the left hand side the usage of the profile to map on a fine grained level two properties as temporal while on the right hand side we configure the whole class as temporal. These two usages are reflecting the two mapping strategies which are explained next.

3.3. Mapping Strategies

Based on the afore presented profile, we now establish the mapping between models and the TSDB. By this mapping, (i) the traceability between design and runtime activities should be enabled, (ii) and runtime information (i.e., time series data) should be retrieved and should be accessible through models. For this purpose, it must be decided how objects, slots, operation calls, and links from the models, i.e., object graphs (conform to classes, attributes, operations and references, respectively) are mapped to the TSDB elements such as measurements, tags, and fields.

In this paper, we consider two conceptual strategies for the M2TS mapper: (i) a strategy where it is possible to store each temporal property individually (cf. Section 3.3.1), and (ii), a strategy by which the whole object with all its associated information is stored (cf. Section 3.3.2). Of course, there exist various other combinations of strategies additionally to these two discussed ones. However, selecting a suitable strategy depends on performance, memory size, and general feasibility. In this paper, we focus on the presented ones, since they give already several configuration opportunities for modelers and consider the capabilities of the deployed TSDB functions. The developed TS profile is designed to cover both strategies, but is not limited to them. It can be extended as well as the mappings may be adapted to specific strategy changes.

3.3.1. Single Property Mappings The first strategy is to map single properties, e.g., the temperature of a room. The goal is to continuously log the progression of such property values in a TSDB and to query these values in terms of the models if necessary. This means that the property becomes a "temporal feature" with its own measurement. Only such time relevant data is stored in the TSDB. The remaining information, such as static metadata, is stored in the model, since such information is constant and does never change over time. However, to ensure that properties of different objects can be distinguished and that the relationship between them is not lost, the information to which object the temporal feature belongs must also be stored in the measurement. For example, if the average temperature of a specific room is required, it should be avoided that the average temperature of all rooms is analyzed. Therefore, the room `id` is important to be related to the measurement.

The table in Appendix A shows the mapping of the object diagram elements to the specific elements of the TSDB based on the TS profile. It has to be mentioned that tags are optional additional information and fields are mandatory for value recording. Similarly, each measurement in the TSDB contains a mandatory timestamp, where precision can be used to determine the accuracy. For `DerivedRTProperties` a query is executed on the TSDB which returns a `series` as a result (cf. Section 3.4).

```
ON
object.set(feature, value)
IF
feature.isTemporal
THEN
time = UnixTimestamp(precision = <<feature.temporal.
    precision>>) //default:ns
db.insert(measure=<<feature.name>>, tag1=['object',
    <<object.id>>] field1=['value',<<value>>] time)
```

Listing 2 Template for single property mapping.

A simple pseudo code line protocol template for the mapping of a temporal feature embedded in an ECA rule is shown in Listing 2. If a temporal property is set, then the new value is stored as a field in the TSDB, the object ID as tag, and the timestamp as Unix timestamp with the defined precision.

Additional to this example, annotated tags will be added to temporal features of the same object.

Based on the example of Figure 2(b), the following listings show the entries of the different measurements in the database. Listing 3 shows the stored values of the three different components `c1-c3`, before executing c3.run(). For the two temporal properties `isActive` and `temp` the initial values are stored with their timestamp.

Listing 4 shows the values of the different measurements after starting c3.run(). There is a new measurement for method `run`. Additionally, `isActive` is changed to true for component c3 and also the `temp` value changes, and therefore, in both measurements a new entry for c3 is added.

```
database: s1
measurements: isActive, temp

measurement isActive:
time                 object     value
----                 ------     -----
1589406897116594100  40170008   false
1589406897196737400  1443055846 false
1589406897207745200  502838712  false

measurement temp:
time                 object     value
----                 ------     -----
1589406897184730200  40170008   20
1589406897201751200  1443055846 30
1589406897213821300  502838712  10
```

Listing 3 TSDB entries for the single property strategy before c3.run() is executed.

```
c3.run()
measurements: isActive, temp, run

measurement isActive:
time                 object     value
----                 ------     -----
1589407171078884600  40170008   false
1589407171166080300  1443055846 false
1589407171178340200  502838712  false
1589407171200396800  502838712  true

measurement temp:
time                 object     value
----                 ------     -----
1589407171155071700  40170008   20
1589407171172226700  1443055846 30
1589407171184344500  502838712  10
1589407171206409400  502838712  50

measurement run:
time                 object     value
----                 ------     -----
1589407171190348700  502838712  start
```

Listing 4 TSDB entries for the single property strategy after c3.run() is executed.

3.3.2. Complete Object Mappings The second strategy does not map single properties of objects in isolation but rather the entire object at once. This means that individual properties do not have to be annotated as temporal features, but the containing classes, and thus, the associated objects with their properties are stored in the database as measurements.

The table in Appendix B gives an overview of the complete object mapping strategy, how the individual annotated elements from the model are stored in the TSDB. Based on this, Listing 5 shows a pseudo code line protocol template, again embedded in an ECA rule, for storing complete objects as measurements and their features as fields. In addition, if structural features are annotated as tags, then they would be saved as tags in the object measurement.

```
ON
object.set(feature, value)
IF
object.class.isTemporal
THEN
time = UnixTimestamp(precision = <<object.class.
    temporal.precision>>) //default:ns
db.insert(measure=<<object.id>> FOREACH(f in
    <<object.features>>){fieldN=[<<f.name>>,
    <<f.value>>]} time)
```

Listing 5 Template of complete object mapping.

In comparison to the presented single property mapping (cf. Section 3.3.1), Listing 6 and Listing 7 show the set-up of the database and its entries for the complete object mapping. The structure of the information has changed and therefore also the structure of the TSDB queries (cf. Section 3.4) depends on the corresponding mapping.

```
database: s1
measurements: obi40170008, obj1443055846,
    obj502838712

measurement obj40170008:
time                 isActive   temp
----                 --------   ----
1589406897116594100  false      20

measurement obj1443055846:
time                 isActive   temp
----                 --------   ----
1589406897196737400  false      30

measurement obj502838712:
time                 isActive   temp
----                 --------   ----
1589406897207745200  false      10
```

Listing 6 TSDB entries for the complete object strategy before c3.run() is executed.

```
c3.run()
measurements: obj40170008, obj1443055846,
    obj502838712, run
measurement obj40170008:
time                 isActive   temp
----                 --------   ----
1589406897116594100  false      20

measurement obj1443055846:
time                 isActive   temp
----                 --------   ----
1589406897196737400  false      30

measurement obj502838712:
time                 isActive   temp
----                 --------   ----
1589406897207745200  false      10
1589407171200396800  true       10
```

```
1589407171206409400   true        50

measurement run:
time                  object          value
----                  ------          -----
1589407171190348700  obj502838712    start
```

Listing 7 TSDB entries for the complete object strategy after c3.run() is executed.

3.4. Query Capabilities

On the basis of the TS profile and the applied mapping strategies, we now present the query capabilities of our approach. In a first step, we offer four basic operations for temporal properties, the first two are adapted from previous work (Gómez et al. 2018), and the last two are extensions:

(1) *getValueAt(Instant t)*
Result: DataType value
(2) *getValueBetween(Instant t1, Instant t2)*
Result: Map(Instant time, DataType value)
(3) *getTimePointsforValue(DataType value)*
Result: List(Instant time)
(4) *getTimePointsforValueBetween(DataType value1, DataType value2)*
Result: Map(Instant time, DataType value)

Based on the used mapping strategy, the query implementation in the background, i.e., in the TSDB, differs, since there is a different data structure used in the TSDB. For instance, the following Listing 8 shows the difference for the basic operation (1).

```
Single property mapping:
getValueAt(Instant t) {
    db.exeQuery(SELECT value FROM <<feature.name>>
    WHERE object = <<object.id>> and time = t)
}

Complete object mapping:
getValueAt(Instant t) {
    db.exeQuery(SELECT <<feature.name>> FROM
    <<object.id>> WHERE time = t)
}
```

Listing 8 TSDB query for getValueAt(Instant t) based on single property mapping and complete object mapping.

On the basis of the four defined operations, it is now possible, e.g., to calculate the utilization of a component within a defined period of time directly using Java. Listing 9 shows a pseudo code snippet for such a metric calculation.

```
Map<Instant, Boolean> map = c1.isActiveT.
    getValueBetween(t1, t2);
Duration total = Duration.between(t1, t2);
Duration active = 0;
Instant[] keys = map.keySet().toArray();
for(int i=0; i < keys.length-1; i++){
  if(map.get(keys[i]))
    active+= Duration.between(keys[i], keys[i+1]);
}
float utilization = active / total;
```

Listing 9 Pseudo code for the calculation of the utilization time of a component.

Additionally to these four basic operations, derived properties can be annotated with `DerivedRTProperty(query:TS-OCL)`. As a first realization, the `TS-OCL` query must be expressed in the syntax of the query language of the TSDB (in the InfluxDB case it is Influx QL), or in combination with standard OCL[5] for navigation through the model (i.e., using self, naviation operatores, etc.) The combination of OCL with Influx QL is preformed as a pre-processor approach. OCL is used to query the model elements which are injected into the InfluxQL query. The M2TS mapper is resolving the model elements to database entries and thus completes the InfluxQL query.

As an example, we consider as a derived property the maximum temperature of a component. Listing 10 shows the `TS-OCL` query for this example and the respective conversion to a `TSQuery` based on the two different mapping strategies.

```
DerivedRTProperty: MaxTemperature
    TS-OCL=SELECT max(<<self.temp>>) FROM <<self.
    temp>>

Single property mapping:
    TSQuery=SELECT max(value) FROM temp WHERE object
    =<<object.id>>

Complete object mapping:
    TSQuery=SELECT max(temp) FROM <<object.id>>
```

Listing 10 Example query code of a derived runtime property.

These query capabilities enable the M2TS mapper not only to inject data to the TSDB from model changes, but also to extract data from the TSDB by model-based queries. As the derived runtime properties are in essence standard derived properties, they can be simply reused in standard OCL queries. Finally, the combination of OCL with Influx QL allows to write model-based queries without having to deal with the concrete mapping approach in use.

4. Evaluation

In this section, we present and discuss the performance and scalability of our approach using a case study based on the PS-DSL metamodel (cf. Section 2, Figure 2 (a)). From a methodological view, we follow the guidelines for conducting case studies by Runeson and Höst (Runeson & Höst 2009) for performing the evaluation. The implementation of our approach and evaluation results can be found at our project website[6].

4.1. Research Questions

Our general evaluation interest is the comparison of the two presented mapping strategies for our M2TS mapper on basis of performance and scalability. Therefore, we aim to answer the following research questions (RQs):

RQ1—Scalability of the database size with single property mapping vs. complete object mapping: How does the database size develop regarding different number of model changes and number of entries? Does the database size grow linear to model

[5] https://www.omg.org/spec/OCL
[6] https://cdl-mint.se.jku.at/case-study-artefacts-jot-2020/

changes? Is there a significant difference between the two mapping strategies?

RQ2—Performance of the runtime queries for single property mapping vs. complete object mapping: How long do the queries take for (i) values at a given timestamp, (ii) timestamps for specific values, and (iii) aggregate calculations such as average, maximum, and modal values? Is there a significant difference observable for the two mapping strategies?

4.2. Case Study Design

Requirements: As an appropriate input for our case study, we first require a system based on an Ecore model, which is annotated by our TS profile. The corresponding models must be executable and contribute to time series when executed. In addition, we require InfluxDB as running TSDB to store value records.

Setup: For our evaluation, as already mentioned, we use instances based on the PS-DSL metamodel. Our execution system consists of different numbers of components and the run method of each component is executed for various numbers of time. Table 1 gives an overview of the different evaluation settings regarding number of components, number of runs, and number of entries in the TSDB. For instance, one setting consists of 100 components, 100 runs are executed for each component, and finally 80000 entries are stored in the TSDB. During simulation, the values of the properties `isActive` and `temp` are changing over time and logged in the TSDB based on the respective mapping strategy.

No.	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6
Comp.	100	1 000	2 000	3 000	4 000	5 000
Run()	100	1 000	2 000	3 000	4 000	5 000
Entries	80 K	8 M	32 M	72 M	128 M	200 M

Table 1 Number of (i) components in the model, (ii) run() executions for different settings, and (iii) entries in the TSDB.

For these settings, the query performance is evaluated as follows. On the one hand, for the different derived runtime properties, i.e., the maximum, mean, and mode values of the temp attribute for a selected component is calculated, and on the other hand, the general methods provided by our approach *getTimePointsForValue* and *getValueAt* are executed for particular values and time points.

For answering our RQs, we calculate the different durations for storing data, and for each query by System.getNanoTime() in Java based on nanoseconds (ns). The performance is measured on an Acer Aspire VN7-791 with an Intel(R) Core(TM) i7-4720 HQ CPU@2.60 GHz 2.60 GHz, with 16 GB of physical memory, and running Windows 8.1. 64 bits operating system. Please note that we measured the CPU time by executing each mapping five times for all different settings and calculated the arithmetic mean of these runs. We use EMF, JDK 13 (important for precision accuracy of nanoseconds), and InfluxDB 1.8.0 to execute our approach.

Prototype: In a first prototypical implementation, we realized our M2TS mapper for EMF. In particular, we provide annotations for the metamodeling language Ecore with respect to utilizing the TSDB InfluxDB. The different stereotypes of our TS profile are implemented as EAnnotations on the Ecore model. For connecting the database, we make use of the open source Java client for InfluxDB[7] and provide our own InfluxDBConnector which provides the glue between EMF models and InfluxDB. For automation purposes, we adapt the existing Java Emitter Templates (JET) for the EMF code generation. Thus, by the extended code generator we are able to provide an enriched API for EMF models to deal with temporal information, i.e., storage and query capabilities.

4.3. Results

In this subsection, we present the measurements for answering our research questions.

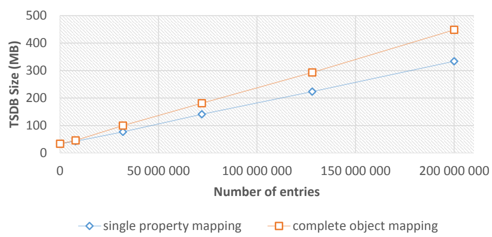


Figure 7 Database size in relation to number of entries for both mapping strategies (in MB).

Answering RQ1 - Scalability of database sizes: Our investigations regarding the TSDB size on the basis of the model changes show that both strategies show a linear increase (cf. Figure 7). It can be recognized that the size of the database for complete object mapping strategy increases slightly faster than for single property mapping strategy. However, this can be explained by the fact that whenever a value of a property is changed, the entire object is stored with a new timestamp.

Answering RQ2 - Performance of runtime queries: Figure 8 shows the measurements of the query duration for both mapping strategies. In general, the queries are fast, as they take only from 1ms to about 7ms, depending on the entries in the TSDB. However, as the size of the database increases, the queries for `MeanMaxMode` and `GetValueAt` for the single property mapping become slightly slower than in the case of the complete object mapping. This can be explained by the fact that starting from a certain number of entries, it plays a role whether the possible results have to be selected first (for single mapping using `object.id`), or are already selected and only need to be screened (for the complete mapping strategy, the object has its own measurement). However, based on a hypothesis testing (i.e., Wilcoxon rank-sum testing (Venables & Ripley 2002)), there is no significance regarding the difference between the

[7] https://github.com/influxdata/influxdb-java

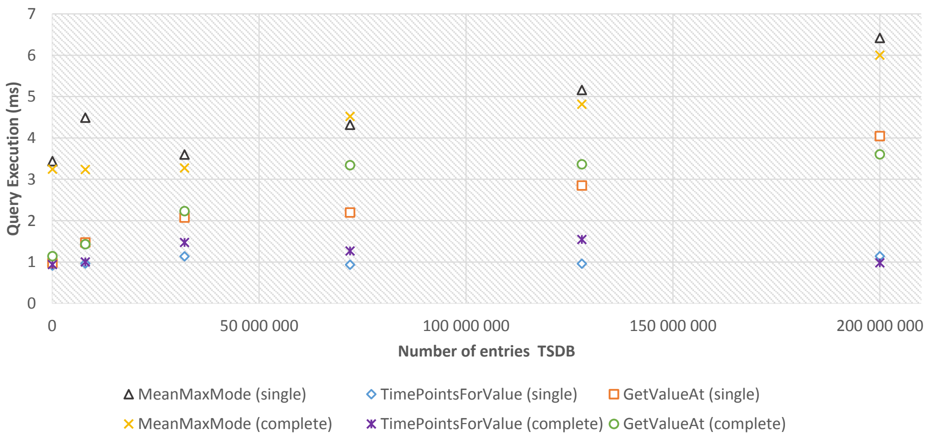


Figure 8 Query execution time in relation to number of entries for both mapping strategies (in ms).

two mapping strategies: p-value =0.4848, H0: single property = complete object; p-value>0.05. Therefore, H0 is not rejected.

4.4. Critical Discussion

In summary, the introduced mapping strategies both have advantages as well as limitations. The right choice of strategy mainly depends on the task to be accomplished, i.e., which queries are subsequently evaluated. Imagine you aim to query all instances for a given type. This would be straightforward for the complete object mapping strategy. Imagine you would like to query the max temperature for all components. This would be faster for the single property mapping.

Overall, the evaluation demonstrated the feasibility of both strategies concerning the data storage and query performance. However, we cannot generalize our results beyond our initial case study. First, we have to mention that there may be other cases where larger objects, i.e., objects having many slot values and links, have to be stored. Consequently, a higher size of the databases may be expected, and this may call for new strategies of mapping objects with only a partial subset of their slots and links. In future studies, we plan to evaluate settings in a larger context such as building a monitoring systems for an IoT network or building a runtime-based verification tool as well. Such studies will allow a more practice-oriented evaluation of the different strategies which may be further collected in a particular benchmark for temporal models.

Another feature to exploit from the TSDB may be the down sampling capability for data for given time frames, i.e., aggregating data from millisec to sec to minute ranges and so on. Again, this feature has to be evaluated in future work. Moreover, the explicit usage of tags instead of fields has to be evaluated in future work as it may have impacts on both: the data storage size and the query performance.

Finally, we would like to mention that our presented case study with all the corresponding artefacts is provided online and may be used by the research community as experimental test bed for future studies concerning finding appropriate mappings from models to time series databases.

5. Related Work

With respect to the contribution of this paper, namely, the mapping and connection of conceptual models to TSDBs, we discuss various threads of related work. First, we discuss temporal modeling approaches for validation and verification of models. Second, we present approaches for linking design and runtime models. Third, we explore approaches for versioning of models in temporal repositories. Finally, we discuss approaches that combine modeling languages with time series analytics.

5.1. Temporal Modeling Languages

There is abundant research on temporal extensions for modeling languages to specify the temporal characteristics of the system data (e.g., consider (Gregersen & Jensen 1999) for a survey), but not regarding the temporal dimensions of models themselves.

Further works advance these first attempts by extending also the query languages with temporal properties, mainly to enable the validation and verification of temporal properties on the data. Temporal OCL (TOCL) (Ziemann & Gogolla 2003) and Temporal UML (Cabot et al. 2003) are two examples of OCL extensions for the evaluation of temporal constraints.

Temporal extensions have also been applied to specific types of systems (e.g., adaptive systems (Mouline et al. 2018)) and DSLs (e.g. timed Petri nets (Bender et al. 2008)). Even TOCL, which can be seen as a generic language, can also be used as an component in other DSLs as described in (Meyers et al. 2014). In this line, (Bousse et al. 2019) discuss and apply a pattern to extend modeling languages with events, traces, and further runtime concepts to represent the state of a model's execution and to use TOCL for defining properties that are verified by mapping the models as well as the properties expressed in TOCL to formal domains that provide verification support. Efficiency of these types of temporal inspection queries is also the focus of (García-Domínguez et al. 2018) and (García-Domínguez et al. 2019).

Nevertheless, all these approaches (including our own previous TemporalEMF proposal (Gómez et al. 2018)) are mostly oriented towards the retrieval of specific past states of the model/data, elaborating on the concepts of *valid time* and *transaction time* of (bi)temporal models. Instead, in this work we explicitly focus on the support for complete time series storage and analysis, which opens the door to more powerful and rich possibilities, like the computation of different KPIs for models as part of design exploration and simulation scenarios.

5.2. Linking Design-time and Runtime Models

In this subsection, we discuss approaches using traceability between design and runtime models. The evolutionary aspect of engineering artifacts refers to the fact that they change over time. Models in engineering processes, e.g., usually develop from initial ideas to first drafts. They are then continuously revised, often by taking into account feedback from other resources, until they are finally released. However, also the feedback after the release from the operation should be reflected in those models to make traceability between design and operation feasible (Mazak & Wimmer 2016).

For this purpose, the authors of (Wolny et al. 2018) present an architecture to map runtime data back to the model level by using standard metamodeling techniques. Thereby, they do not only develop a unifying architecture for creating model snapshots on-the-fly, but to map the history of operation concerning certain properties. This allows to specify and compute runtime properties based on time series data through design models. This means, design-oriented languages are equipped with extensions for representing runtime states as well as runtime histories, which in turn allow the formulation and computation of runtime properties with OCL. This makes it feasibly to directly interpret measurements within design models without introducing an impedance mismatch. The challenge with using OCL for this purpose is that even simple mathematical calculations (e.g., computing upper bounds or averages) may quickly become complex with respect to their definition and evaluation. For better scalability such calculations should be directly performed in the TSDB as we allow by the presented work of this paper.

If the design model is not yet coupled with its runtime counterpart, i.e., no annotations are made at model level, the authors of (Wolny et al. 2019) present an approach to transform raw sensor log data to UML sequence diagrams for graphical representation. Therefore, they provide a text-to-model transformation to transform text-based traces of a running system to UML sequence diagrams. As a basis for reconstructing such UML sequence diagrams, they develop a metamodel for representing system logs in an object-oriented manner. This makes it feasible to express system logs explicitly as models. However, they only use the time aspect to trace the correct order of the performed operations, but not to store the execution time, e.g., to be able to annotate information about average duration. This could be complemented by the approach presented in this paper.

Another project that also deals with the connection of design and runtime is the project MegaM@Rt2[8]. In this scalable model-based framework for continuous development and runtime validation of complex systems trace links between design models and runtime are established based on bidirectional transformations (Cruz, Sadovykh, Truscan, Bruneliere, et al. 2020). Temporal aspects as we discuss in the context of this paper are not explicitly considered. However, the MegaM@Rt2 approach is applicable for already existing systems which may be combined with our approach to enrich existing systems with TS collection and analysis.

5.3. Temporal Model Repositories

In (Bill et al. 2017), the authors discuss the need for temporal model repositories and the explicit representation of time in models. They discuss the gap of traditional Version Control Systems (VCS) such as SVN and Git, where each version of an evolving model is stored with a timestamp for the whole model (Altmanninger et al. 2009). While versioning the whole model is suitable for many development tasks, it makes it challenging to trace the evolution of specific model elements over time. Furthermore, the authors discuss several challenges when moving towards temporal model repositories such as (i) model storage, (ii) model access, (iii) model consistency, (iv) model manipulation, and (v) model visualization. In this paper, we have mostly focused on the first two points.

In the work presented in (Hartmann et al. 2014), the authors present an approach for versioning on the model element level. They discuss the lack of native mechanisms in MDE as well as Models@run.time to handle the history of data. They state that especially for the Models@run.time paradigm (Blair et al. 2009), which propagates the use of models to support runtime reasoning, an efficient mechanism is needed to store and navigate the history of model element values. Therefore, model elements have to be versioned independently from each other. Furthermore, they simplify and improve the performance of navigating between model elements coming from different versions by defining a navigation context for navigating in two dimensions (space and version). However, the versions have to be explicitly introduced and managed as in the aforementioned versioning systems. In our approach, we store individual model element or even individual properties with their associated timing aspects.

To tackle the discussed challenges in (Bill et al. 2017), in (Gómez et al. 2018) we present a temporal model infrastructure built on top of EMF—*TemporalEMF*. In summary, we showed how TemporalEMF enables to treat conceptual schemas as temporal models. On these models, temporal queries can be performed to retrieve model contents at different time points, e.g., to compare model content and to trace model states in the past. The TemporalEMF approach bases on concepts from temporal languages. The history of a model is transparently stored in a NoSQL database (i.e., HBase[9]). In our newly presented approach no dependence to other DBMS, such as NoSQL ones, is needed, since we use a TSDB to reason about the history of property values accessible in the model.

[8] https://megamart2-ecsel.eu/
[9] http://hbase.apache.org/

In (Haeusler et al. 2019), the authors discuss the need for tool support in the area of IT Landscape documentation. Therefore, they present a solution for storing, versioning, and querying of such IT Landscape models by means of an open source graph-based EMF model repository. In addition, the modular architecture allows to consider those models still as standalone components outside the repository context. A limitation is that *ChronoSphere* operates in local deployments, and therefore, is currently not distributed across several machines for greater scalability. In our approach, models and the TSDB can run separately on different machines. Since, our polyglot approach provides a ModelAPI, it could be considered in the ChronoSphere repository as well, which is generic, and therefore, not limited to the domain of IT Landscape documentation. This allows other applications to use the provided functionalities of our API for various use cases.

5.4. Modeling Languages for Time Series Analytics

In (David et al. 2012), the authors present the OMS3[10] modeling framework which provides an extensible and lightweight layer for simulation description expressed as so-called "Simulation DSL" based on Groovy[11]. The authors propagate DSLs for completing General Purpose Languages (GPLs) for specific simulation purposes. In their work, they present DSL variants in OMS3 such as a DSL for Ensemble Streamflow Prediction (ESP) based on meteorological time series data for predicting future conditions. Instead of creating a DSL for a specific purpose, in our approach, we propose a dedicated profile for extending metamodels with appropriate annotations to extend existing metamodels (e.g., of GPLs) by time series aspects.

Gekko[12] is an open source modeling approach for time series data management and for solving as well as analyzing large-scale time series models. It could be considered as a kind of DSL with a strong time series domain focus. It provides interfaces to statistical computing and graphics packages such as R[13]. In our approach, we use InfluxDB which offers besides high-availability storage and monitoring of time series data, application metrics as well as real-time analytics.

In this context, we also mention TimescaleDB[14] which is an extension of PostgreSQL[15]. TimescaleDB is specially optimized for time series data in order to automatically partition data by time. Like PostgreSQL, TimescaleDB stores the data in a RDBMS and supports SQL as query language. Furthermore, it provides additional features for analyzing and manipulating time series data. Similar to the InfluxDB, TimescaleDB offers the possibility of a continuous calculation of functions. In particular such functions are queries that are executed continuously and in real time on the incoming data. The results of these regular queries are also stored in the TSDB as specified metrics (e.g., average room temperature every half hour with the applied metric). External tools such as Grafana[16] or Tableau[17] may also be used to visualize and analyze time series data. In addition to Grafana, the open source statistics software R for analyzing time series data should also be mentioned. However, the probably most extensive functionalities for querying data, setting warnings, and visualizing time series data is offered by InfluxDB, respectively by the InfluxData platform. Moreover, long-term storage of data is only provided by InfluxDB, and only to a limited extent by TimescaleDB. Additionally, the data scripting and query language Flux[18] can be used in combination with InfluxDB. This standalone tool is optimized, e.g., for monitoring and provides built-in functions as well as importable packages to retrieve, transform, process, and output time series data. In contrast, our approach looks at TSDBs from a model-driven perspective and how conceptual modeling and TSDBs can benefit from each other.

[10] https://alm.engr.colostate.edu/cb/wiki/16961
[11] https://groovy-lang.org
[12] http://t-t.dk/gekko
[13] https://www.r-project.org
[14] https://www.timescale.com
[15] https://www.postgresql.org

6. Conclusion and Future Work

In this paper, we have presented a novel set of partial mappings from models to TSDB. In particular, we presented a profile to annotate metamodels in order to automatically generate wrappers to time series databases that enable storing model updates as well as querying historical model information. Two different mapping strategies are proposed and evaluated in terms of their feasibility and scalability. While the current work presents interesting insights how modeling technologies may be combined with TSDB, we foresee several additional lines of research worth to investigate in addition to the ones mentioned in the evaluation section.

On the modeling side, we need to deal with co-evolution issues given that the TSDB is schema-less. For usability reasons, we would also like to be able to express complex time-related queries in OCL (e.g., by pre-defining a set of time-series operators, similar to what we did in (Cabot et al. 2010) for multidimensional models).

On the mapping side, we will investigate how to run approximate queries to deal with a variety of uncertainty scenarios (Burgueño et al. 2019) and study the potential of combining both temporal and time-series information. This would enable even more complex analysis where we could, for instance, evaluate whether a new design model behaves better than one we used in the past by comparing their respective associated time-series data. It even allows to forecast the expected behavior of future designs. Finally, we are interested in mapping and storing not only the models themselves but also all modeling operations on them (e.g., by storing the trace information automatically created by some transformation engines such as ATL).

Acknowledgments

The work has been supported by the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development (CDG) and by the Austrian Federal Ministry for Education, Science and Research

[16] https://grafana.com
[17] https://www.tableau.com
[18] https://www.influxdata.com/products/flux

in the TransIT Project (BMBWF-11.102/0033-IV/8/2019) as well as by the FWF under the Grant Numbers P28519-N31 and P30525-N31, and from the Spanish government under project *Open Data for All* (RETOS TIN2016-75944-R).

References

Altmanninger, K., Seidl, M., & Wimmer, M. (2009). A survey on model versioning approaches. *IJWIS*, *5*(3), 271–304. Retrieved from https://doi.org/10.1108/17440080910983556 doi: 10.1108/17440080910983556

Bader, A., Kopp, O., & Falkenthal, M. (2017). Survey and Comparison of Open Source Time Series Databases. In *Workshopband der 17. fachtagung des gi-fachbereichs datenbanken und informationssysteme (dbis) datenbanksysteme für business, technologie und web (BTW 2017)* (Vol. P-266, pp. 249–268). GI. Retrieved from https://dl.gi.de/20.500.12116/922

Bencomo, N., Götz, S., & Song, H. (2019). Models@run.time: a guided tour of the state of the art and research challenges. *Software and Systems Modeling*, *18*(5), 3049–3082. Retrieved from https://doi.org/10.1007/s10270-018-00712-x doi: 10.1007/s10270-018-00712-x

Bender, D. F., Combemale, B., Crégut, X., Farines, J., Berthomieu, B., & Vernadat, F. (2008). Ladder Metamodeling and PLC Program Validation through Time Petri Nets. In *Proc. of the 4th european conference on model driven architecture - foundations and applications, ECMDA-FA 2008* (Vol. 5095, pp. 121–136). Springer. Retrieved from https://doi.org/10.1007/978-3-540-69100-6_9 doi: 10.1007/978-3-540-69100-6_9

Benelallam, A., Hartmann, T., Mouline, L., Fouquet, F., Bourcier, J., Barais, O., & Traon, Y. L. (2017). Raising Time Awareness in Model-Driven Engineering: Vision Paper. In *Proc. of the 20th ACM/IEEE international conference on model driven engineering languages and systems, MODELS 2017* (pp. 181–188). IEEE Computer Society. Retrieved from https://doi.org/10.1109/MODELS.2017.11 doi: 10.1109/MODELS.2017.11

Bézivin, J., Paige, R. F., Aßmann, U., Rumpe, B., & Schmidt, D. C. (2014). Manifesto - Model Engineering for Complex Systems. *CoRR*, *abs/1409.6591*. Retrieved from http://arxiv.org/abs/1409.6591

Bill, R., Mazak, A., Wimmer, M., & Vogel-Heuser, B. (2017). On the Need for Temporal Model Repositories. In *Software technologies: Applications and foundations - STAF 2017 collocated workshops, revised selected papers* (Vol. 10748, pp. 136–145). Springer. Retrieved from https://doi.org/10.1007/978-3-319-74730-9_11 doi: 10.1007/978-3-319-74730-9_11

Blair, G. S., Bencomo, N., & France, R. B. (2009). Models@run.time. *IEEE Computer*, *42*(10), 22–27. Retrieved from https://doi.org/10.1109/MC.2009.326 doi: 10.1109/MC.2009.326

Böhlen, M. H., Dignös, A., Gamper, J., & Jensen, C. S. (2018). Database Technology for Processing Temporal Data. In *Proc. of the 25th international symposium on temporal representation and reasoning, TIME 2018* (Vol. 120, pp. 2:1–2:7). Schloss Dagstuhl - Leibniz-Zentrum für Informatik. Retrieved from https://doi.org/10.4230/LIPIcs.TIME.2018.2 doi: 10.4230/LIPIcs.TIME.2018.2

Bousse, E., Mayerhofer, T., Combemale, B., & Baudry, B. (2019). Advanced and efficient execution trace management for executable domain-specific modeling languages. *Software and Systems Modeling*, *18*(1), 385–421. Retrieved from https://doi.org/10.1007/s10270-017-0598-5 doi: 10.1007/s10270-017-0598-5

Brambilla, M., Cabot, J., & Wimmer, M. (2017). *Model-Driven Software Engineering in Practice, Second Edition*. Morgan & Claypool Publishers. Retrieved from https://doi.org/10.2200/S00751ED2V01Y201701SWE004 doi: 10.2200/S00751ED2V01Y201701SWE004

Burgueño, L., Mayerhofer, T., Wimmer, M., & Vallecillo, A. (2019). Specifying quantities in software models. *Inf. Softw. Technol.*, *113*, 82–97. Retrieved from https://doi.org/10.1016/j.infsof.2019.05.006 doi: 10.1016/j.infsof.2019.05.006

Cabot, J., & Gogolla, M. (2012). Object Constraint Language (OCL): A Definitive Guide. In *Formal methods for model-driven engineering - 12th international school on formal methods for the design of computer, communication, and software systems, SFM 2012* (Vol. 7320, pp. 58–90). Springer. Retrieved from https://doi.org/10.1007/978-3-642-30982-3_3 doi: 10.1007/978-3-642-30982-3_3

Cabot, J., Mazón, J., Pardillo, J., & Trujillo, J. (2010). Specifying Aggregation Functions in Multidimensional Models with OCL. In *Proc. of the 29th international conference on conceptual modeling - ER 2010* (Vol. 6412, pp. 419–432). Springer. Retrieved from https://doi.org/10.1007/978-3-642-16373-9_30 doi: 10.1007/978-3-642-16373-9_30

Cabot, J., Olivé, A., & Teniente, E. (2003). Representing Temporal Information in UML. In *Proc. of the 6th international conference on modeling languages and applications, «uml» 2003 - the unified modeling language* (Vol. 2863, pp. 44–59). Springer. Retrieved from https://doi.org/10.1007/978-3-540-45221-8_5 doi: 10.1007/978-3-540-45221-8_5

Cruz, J. G., Sadovykh, A., Truscan, D., Brunelière, H., Pierini, P., & Muniz, L. L. (2020). MegaM@Rt2 EU Project: Open Source Tools for Mega-Modelling at Runtime of CPSs. In *Proc. of the 16th IFIP WG 2.13 international conference on open source systems, OSS 2020* (Vol. 582, pp. 183–189). Springer. Retrieved from https://doi.org/10.1007/978-3-030-47240-5_18 doi: 10.1007/978-3-030-47240-5_18

Cruz, J. G., Sadovykh, A., Truscan, D., Bruneliere, H., Pierini, P., & Muñiz, L. L. (2020). MegaM@Rt2 EU Project: Open Source Tools for Mega-Modelling at Runtime of CPSs. In *Open source systems* (pp. 183–189). Springer.

David, O., Lloyd, W., II, J. C. A., Green, T. R., Olson, K., Leavesley, G. H., & Carlson, J. (2012). Domain Specific Languages for Modeling and Simulation: Use Case OMS3. In *Proc. of the international congress on environmental modelling and software managing resources of a limited planet* (p. 1201-1207).

García-Domínguez, A., Bencomo, N., & Paucar, L. H. G. (2018). Reflecting on the past and the present with temporal graph-based models. In *Proc. of MODELS 2018 workshops:*

Modcomp, mrt, ocl, flexmde, exe, commitmde, mdetools, gemoc, morse, mde4iot, mdebug, modevva, me, multi, hufamo, ammore, PAINS co-located with ACM/IEEE 21st international conference on model driven engineering languages and systems (MODELS 2018) (Vol. 2245, pp. 46–55). CEUR-WS.org. Retrieved from http://ceur-ws.org/Vol-2245/mrt_paper_1.pdf

García-Domínguez, A., Bencomo, N., Ullauri, J. M. P., & Paucar, L. H. G. (2019). Querying and Annotating Model Histories with Time-Aware Patterns. In *Proc. of the 22nd ACM/IEEE international conference on model driven engineering languages and systems, MODELS 2019* (pp. 194–204). IEEE. Retrieved from https://doi.org/10.1109/MODELS.2019.000-2 doi: 10.1109/MODELS.2019.000-2

Gogolla, M. (2005). Tales of ER and RE Syntax and Semantics. In *Transformation techniques in software engineering* (Vol. 05161). Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany. Retrieved from http://drops.dagstuhl.de/opus/volltexte/2006/425

Gogolla, M., Desai, N., & Doan, K. (2019). Developing User and Recording Interfaces for Design Time and Runtime Models. In *STAF 2019 co-located events joint proceedings: 1st junior researcher community event, 2nd international workshop on model-driven engineering for design-runtime interaction in complex systems, and 1st research project showcase workshop co-located with software technologies: Applications and foundations (STAF 2019)* (Vol. 2405, pp. 39–48). CEUR-WS.org. Retrieved from http://ceur-ws.org/Vol-2405/08_paper.pdf

Gogolla, M., Hamann, L., Hilken, F., Kuhlmann, M., & France, R. B. (2014). From application models to filmstrip models: An approach to automatic validation of model dynamics. In *Tagungsband der modellierung 2014* (Vol. P-225, pp. 273–288). GI. Retrieved from https://dl.gi.de/20.500.12116/17056

Gómez, A., Cabot, J., & Wimmer, M. (2018). TemporalEMF: A Temporal Metamodeling Framework. In *Proc. of the 37th international conference on conceptual modeling, ER 2018* (Vol. 11157, pp. 365–381). Springer. Retrieved from https://doi.org/10.1007/978-3-030-00847-5_26 doi: 10.1007/978-3-030-00847-5_26

Gregersen, H., & Jensen, C. S. (1999). Temporal Entity-Relationship Models - A Survey. *IEEE Trans. Knowl. Data Eng.*, *11*(3), 464–497. Retrieved from https://doi.org/10.1109/69.774104 doi: 10.1109/69.774104

Haeusler, M., Trojer, T., Kessler, J., Farwick, M., Nowakowski, E., & Breu, R. (2019). Chronosphere: a graph-based EMF model repository for IT landscape models. *Software and Systems Modeling*, *18*(6), 3487–3526. Retrieved from https://doi.org/10.1007/s10270-019-00725-0 doi: 10.1007/s10270-019-00725-0

Hartmann, T., Fouquet, F., Nain, G., Morin, B., Klein, J., Barais, O., & Traon, Y. L. (2014). A Native Versioning Concept to Support Historized Models at Runtime. In *Proc. of the 17th international conference on model-driven engineering languages and systems, MODELS 2014* (Vol. 8767, pp. 252–268). Springer. Retrieved from https://doi.org/10.1007/978-3-319-11653-2_16 doi: 10.1007/978-3-319-11653-2_16

Jensen, S. K., Pedersen, T. B., & Thomsen, C. (2017). Time series management systems: A survey. *IEEE Trans. Knowl. Data Eng.*, *29*(11), 2581–2600. Retrieved from https://doi.org/10.1109/TKDE.2017.2740932 doi: 10.1109/TKDE.2017.2740932

Jensen, S. K., Pedersen, T. B., & Thomsen, C. (2019). Scalable Model-Based Management of Correlated Dimensional Time Series in ModelarDB. *CoRR*, *abs/1903.10269*. Retrieved from http://arxiv.org/abs/1903.10269

Kästner, A., Gogolla, M., Doan, K., & Desai, N. (2018). Sketching a Model-Based Technique for Integrated Design and Run Time Description - Short Paper - Tool Demonstration. In *Software technologies: Applications and foundations - STAF 2018 collocated workshops, revised selected papers* (Vol. 11176, pp. 529–535). Springer. Retrieved from https://doi.org/10.1007/978-3-030-04771-9_39 doi: 10.1007/978-3-030-04771-9_39

Kholod, I., Efimova, M., Rukavitsyn, A., & Shorov, A. (2017). Time Series Distributed Analysis in IoT with ETL and Data Mining Technologies. In *Proc. of the 17th international conference on internet of things, smart spaces, and next generation networks and systems* (Vol. 10531, pp. 97–108). Springer. Retrieved from https://doi.org/10.1007/978-3-319-67380-6_9 doi: 10.1007/978-3-319-67380-6_9

Mazak, A., Lüder, A., Wolny, S., Wimmer, M., Winkler, D., Kirchheim, K., … Biffl, S. (2018). Model-based generation of run-time data collection systems exploiting AutomationML. *Automatisierungstechnik*, *66*(10), 819–833. Retrieved from https://doi.org/10.1515/auto-2018-0022 doi: 10.1515/auto-2018-0022

Mazak, A., & Wimmer, M. (2016). Towards Liquid Models: An Evolutionary Modeling Approach. In *Proc. of the 18th IEEE conference on business informatics, CBI 2016* (pp. 104–112). IEEE. Retrieved from https://doi.org/10.1109/CBI.2016.20 doi: 10.1109/CBI.2016.20

Meyers, B., Deshayes, R., Lucio, L., Syriani, E., Vangheluwe, H., & Wimmer, M. (2014). ProMoBox: A Framework for Generating Domain-Specific Property Languages. In *Proc. of the 7th international conference on software language engineering, SLE 2014* (Vol. 8706, pp. 1–20). Springer. Retrieved from https://doi.org/10.1007/978-3-319-11245-9_1 doi: 10.1007/978-3-319-11245-9_1

Mouline, L., Benelallam, A., Fouquet, F., Bourcier, J., & Barais, O. (2018). A Temporal Model for Interactive Diagnosis of Adaptive Systems. In *Proc. of the IEEE international conference on autonomic computing, ICAC 2018* (pp. 175–180). IEEE Computer Society. Retrieved from https://doi.org/10.1109/ICAC.2018.00029 doi: 10.1109/ICAC.2018.00029

Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, *14*(2), 131–164. Retrieved from https://doi.org/10.1007/s10664-008-9102-8 doi: 10.1007/s10664-008-9102-8

Schmidt, D., Dittrich, A. K., Dreyer, W., & Marti, R. W. (1995). Time Series, A Neglected Issue in Temporal Database Research? In *Proc. of the international workshop on temporal* *databases, recent advances in temporal databases* (pp. 214–232). Springer. Retrieved from https://doi.org/10.1007/978-1-4471-3033-8_12 doi: 10.1007/978-1-4471-3033-8_12

Venables, W. N., & Ripley, B. D. (2002). *Modern applied statistics with s, 4th edition*. Springer. Retrieved from http://www.worldcat.org/oclc/49312402

Wolny, S., Mazak, A., Carpella, C., Geist, V., & Wimmer, M. (2020). Thirteen years of SysML: a systematic mapping study. *Software and Systems Modeling*, *19*(1), 111–169. Retrieved from https://doi.org/10.1007/s10270-019-00735-y doi: 10.1007/s10270-019-00735-y

Wolny, S., Mazak, A., & Wimmer, M. (2019). Automatic Reverse Engineering of Interaction Models from System Logs. In *Proc. of the 24th IEEE international conference on emerging technologies and factory automation, ETFA 2019* (pp. 57–64). IEEE. Retrieved from https://doi.org/10.1109/ETFA.2019.8869502 doi: 10.1109/ETFA.2019.8869502

Wolny, S., Mazak, A., Wimmer, M., Konlechner, R., & Kappel, G. (2018). Model-Driven Time-Series Analytics. *Enterp. Model. Inf. Syst. Archit. Int. J. Concept. Model.*, *13*(Special), 252–261. Retrieved from https://doi.org/10.18417/emisa.si.hcm.19 doi: 10.18417/emisa.si.hcm.19

Ziemann, P., & Gogolla, M. (2003). OCL Extended with Temporal Logic. In *Proc. of the 5th international andrei ershov memorial conference on perspectives of systems informatics, PSI 2003, revised papers* (Vol. 2890, pp. 351–357). Springer. Retrieved from https://doi.org/10.1007/978-3-540-39866-0_35 doi: 10.1007/978-3-540-39866-0_35

A. Single Property to TSDB Mapping Table

Model	**TS Profile**	T			
		measurement	**tag key**	**tag value**	**fi**
Slot	Temporal (precsion: TimeUnit)	attribute.name	“object”	object.id	“v
	Tag	-	attribute. name	slot.value	-
Derived Slot	DerivedRTProperty (query: TS-OCL)	-	-	-	-
Link	Temporal (precsion: TimeUnit)	reference.name	“object”	object.id	re op na
	Tag	-	reference. name	target.id	-
Derived Link	DerivedRTProperty (query: TS-OCL)	-	-	-	-
Method	Log	method.name	“object”	object.id	“v

B. Complete Object to TSDB Mapping Table

Model	**TS Profile**				
		measurement	**tag key**	**tag value**	**fie**
Object	Temporal (precsion: TimeUnit)	object.id	-	-	for att for ref
Slot	Tag	-	attribute. name	slot.value	-
Derived Slot	DerivedRTProperty (query: TS-OCL)	-	-	-	-
Link	Tag	-	reference. name	target.id	-
Derived Link	DerivedRTProperty (query: TS-OCL)	-	-	-	-
Method	Log	method.name	“object”	object.id	“v

About the authors

Alexandra Mazak is a Professor at Montanuniversität Leoben in the research unit Subsurface Engineering. Before she headed Module 3 in the Christian Doppler Laboratory for Model-Integrated Smart Production (CDL-MINT) at JKU Linz. Her research field focuses on digital transformation in tunnel information modeling as well as in modeling of production systems. Her research interests are on information integration, model-driven engineering and model-based data analytics. She headed numerous interdisciplinary national funded projects in the field of digital transformation. You can contact the author at alexandra.mazak-huemer@unileoben.ac.at or visit https://www.tunnellinghub.at/.

Sabine Wolny is currently working as PhD student at the JKU Linz in the module Reactive Model Repositories of the CDL-MINT and in the Research Unit of Building Physics at TU Wien with focus on project management and software development. Her research interests include SysML-based modeling, execution/simulation of production systems, reverse engineering, data integration and back-propagation of runtime information. You can contact the author at sabine.wolny@jku.at or visit https://www.se.jku.at/sabine-wolny/.

Abel Gómez is a senior researcher of the Internet Interdisciplinary Institute, and an Assistant Professor at the Faculty of Computer Science, Multimedia and Telecommunications, both belonging to the Universitat Oberta de Catalunya in Spain. His research interests fall in the broad field of model-driven engineering (MDE), and his research lines have evolved in two complementary directions: the development of core technologies to support MDE activities, and the application of MDE techniques to solve Software Engineering problems. You can contact the author at agomezlla@uoc.edu or visit https://abel.gomez.llana.me.

Jordi Cabot is currently an ICREA Research Professor with the Internet Interdisciplinary Institute. His research interests include software and systems modeling, formal verification, and the role of AI can play in software development (and vice versa). He has published over 200 peer-reviewed conference and journal articles on these topics. Apart from his scientific publications, he writes and blogs about all these topics in several sites. You can contact the author at jordi.cabot@icrea.cat or visit https://jordicabot.com/.

Manuel Wimmer is Full Professor leading the Institute of Business Informatics - Software Engineering at the Johannes Kepler University Linz and he is the head of the Christian Doppler Laboratory CDL-MINT. His research interests comprise foundations of model engineering techniques as well as their application in domains such as tool interoperability, legacy modeling tool modernization, model versioning and evolution, and industrial engineering. You can contact the author at manuel.wimmer@jku.at or visit https://www.se.jku.at/manuel-wimmer/.

Gerti Kappel is Full Professor at the Institute of Information Systems Engineering at TU Wien, chairing the Business Informatics Group. Since the beginning of 2020 she acts as the dean of the Faculty of Informatics at TU Wien. Her current research interests include Model Engineering, Web Engineering, and Process Engineering, with a special emphasis on cyber-physical production systems. You can contact the author at kappel@big.tuwien.ac.at or visit https://www.big.tuwien.ac.at/people/gkappel/.

11 Execution-Based Model Profiling

A. Mazak, P. Patsuk-Bösch and M. Wimmer;
Data-Driven Process Discovery and Analysis- 6th IFIP WG 2.6 International Symposium (SIMPDA), Revised Selected Papers, Vol. 307, LNBIP, Springer, (2016), pp. 37–52.
DOI: 10.1007/978-3-319-74161-1_3

Execution-Based Model Profiling

Alexandra Mazak(✉), Manuel Wimmer, and Polina Patsuk-Bösch

Christian Doppler Laboratory for Model-Integrated Smart Production (CDL-MINT),
Institute of Software Technology and Interactive Systems, TU Wien,
Favoritenstrasse 9-11, 1040 Vienna, Austria
{mazak,wimmer,patsuk}@big.tuwien.ac.at
https://cdl-mint.big.tuwien.ac.at

Abstract. In model-driven engineering (MDE), models are mostly used in prescriptive ways for system engineering. While prescriptive models are indeed an important ingredient to realize a system, for later phases in the systems' lifecycles additional model types are beneficial to use. Unfortunately, current MDE approaches mostly neglect the information upstream in terms of descriptive models from operations to (re)design phases. To tackle this limitation, we propose execution-based model profiling as a continuous process to improve prescriptive models at design-time through runtime information. This approach incorporates knowledge in terms of model profiles from execution logs of the running system. To accomplish this, we combine techniques of process mining with runtime models of MDE. In the course of a case study, we make use of a traffic light system example to demonstrate the feasibility and benefits of the introduced execution-based model profiling approach.

1 Introduction

In *model-driven engineering (MDE)*, models are put in the center and used as a driver throughout the software development process, finally leading to an automated generation of the software systems [14]. In the current state-of-practice in MDE [3], models are used as an abstraction and generalization of a system to be developed. By definition, a model never describes reality in its entirety, rather it describes a scope of reality for a certain purpose in a given context [3]. Thus, models are used as *prescriptive models* for creating a software system [11]. Such models@design.time determine the scope and details of a domain of interest to be studied. Thereby, different aspects of the domain or of its solution can be taken into account. For this purpose different types of modeling languages (e.g., state charts, class diagrams, etc.) may be used. It has to be emphasized that engineers typically have the desirable behavior in mind when creating a system, since they are not aware in these early phases of the many deviations that may take place at runtime [23].

According to Brambilla et al. [3] the implementation phase deals with the mapping of prescriptive models to some executable systems and consists of three levels: (*i*) the *modeling level* where the models are defined, (*ii*) the *realization*

P. Ceravolo et al. (Eds.): SIMPDA 2016, LNBIP 307, pp. 37–52, 2018.
https://doi.org/10.1007/978-3-319-74161-1_3

level where the solutions are implemented through artifacts that are used in the running system, and (*iii*) the *automation level* where mappings from the modeling to the realization phase are made. Thus, the flow is from models down to the running realization through model transformations.

While prescriptive or design models are indeed a very important ingredient to realize a system, for later phases in the system's lifecycle additional model types are needed. Therefore, *descriptive models* may be employed to better understand how the system is actually realized and how it is operating in a certain environment. Compared to prescriptive models, these other mentioned types of models are only marginal explored in the field of MDE, and if used at all, they are built manually. Unfortunately, MDE approaches have mostly neglected the possibility to describe an existing and operating system which may act as feedback for improving design models. As theoretically outlined in [16], we propose *model profiling* as a continuous process (*i*) to improve the quality of design models through runtime information by incorporating knowledge in form of *profiled metadata* from the system's operation, (*ii*) to deal with the evolution of these models, and (*iii*) to better anticipate the unforeseen. However, our aim is not to "re-invent the wheel" when we aim to close the loop between downstream information derived from prescriptive models and upstream information in terms of descriptive models. There exist already promising techniques to focus on runtime phenomena, especially in the research field of Process Mining (PM) [23]. Thus, our model profiling approach in its first version follows the main idea of combining MDE and PM. The contribution of this paper is to present a unifying architecture for a combined but loosely-coupled usage of MDE approaches and PM techniques.

The remainder of this paper is structured as follows. In the next section, we present a unified conceptual architecture for combining MDE with PM frameworks. In Sect. 3, we present a case study of execution-based model profiling conducted on a traffic light system example and present the results. In Sect. 4, we present recent work related to our approach and discuss its differences. Finally, we conclude this paper by an outlook on our next steps in Sect. 5.

2 Marrying Model-Driven Engineering and Process Mining

In this section, we briefly describe the main building blocks of both, MDE as well as PM, necessary for the context of this paper, before we present a unifying architecture for their combined but loosely-coupled usage.

2.1 Prerequisites

Model-Driven Engineering (MDE). In each phase of a MDE-based development process "models" (e.g., analysis models, design models) are (semi-) automatically generated by *model-to-model transformations (M2M)* that take as input models that were obtained in one of the previous phases. In the last step of this process the final code is generated using *model-to-text transformation (M2T)* from the initial model [3]. These transformation engineering aspects are based on the metamodels of the used modeling language, which provide the abstract syntax of that language. This syntax guarantees that models follow a clearly defined structure. In addition, it forms the basis for applying operations on models (e.g., storing, querying, transforming, checking, etc.).

As described in [3], the semantics of a modeling language can be formalized by giving (*i*) *denotational semantics* by defining a mapping from the modeling language to a formal language, (*ii*) *operational semantics* by defining a model simulator (i.e., implementing a model execution engine), or (*iii*) giving *translational semantics* by defining, e.g., a code generator for producing executable code. In order to generate a running system from models, they must be *executable*. This means that a model is executable when its *operational semantics* is fully specified [3]. However, executability depends more on the used execution engine than on the model itself. The main goal of MDE is to get running systems out of models.

In our approach, we consider executable modeling languages which explicitly state "what" the runtime state of a model is as well as all possible events that can occur during execution [17]. These executable modeling languages not only provide operational semantics for interpreters, but also translational semantics in form of code generators to produce code for a concrete platform to realize the system.

Process Mining (PM). PM combines techniques from data mining and model-driven Business Process Management (BPM) [23]. In PM, business processes are analyzed on the basis of *event logs*. Events are defined as process steps and event logs as sequential ordered events recorded by an information system [8]. This means that PM works on the basis of event data instead of prescriptive models. The main challenge of PM is to capture behavioral aspects. Thereby, specialized algorithms (e.g., the α-algorithm) produce a Petri net which can be easily converted into a descriptive model in form of a process model. To put it in a nutshell, there is a concrete, running system which is producing logs and there are algorithms used to compute derived information from these logs. Generally in PM, event logs are analyzed from a process-oriented perspective using general modeling languages (e.g., UML, Petri nets) [24].

There are three main techniques in PM: (*i*) the *discovery technique* by which a process model can be automatically extracted from log data [23], (*ii*) the *conformance checking technique*, which is used to connect an existing process model with an event log containing data related to activities (e.g., business activities) of this process [18], and (*iii*) the *enhancement technique* which is used to change or extend a process model by modifying it, or by adding a new perspective to this model [23].

Orthogonal to the dimension of these techniques, there exists a dimension of different perspectives [23]: (*i*) the *control-flow perspective* reflects the ordering of activities, (*ii*) the *organizational perspective* focuses on resources, organisational units and their interrelations, (*iii*) the *case perspective* deals with properties of individual cases, or process instances, and (*iv*) the *time perspective* focuses on execution time analysis and the frequency of events. These perspectives give a complete picture of the aspects that process mining intends to analyze. In [19], van der Aalst suggests to combine perspectives in order to create simulation models of business processes based on runtime information.

In recent work, van der Aalst already brings together PM with the domain of software engineering. For instance in [25], the authors present a novel reverse engineering technique to obtain real-life event logs from distributed software systems. Thereby, PM techniques are applied to obtain precise and formal models, as well as to monitor and improve processes by performance analysis and conformance checking. In the context of this paper we focus on the control-flow and time perspectives of PM.

2.2 Unifying Conceptual Architecture

In this section, we combine MDE with PM by presenting a unifying conceptual architecture. The alignment of these two different research fields may help us, e.g., to verify if the mapping feature of design models is really fulfilled, or if important information generated at runtime is actually missing in the design (i.e., prescriptive) model.

Figure 1 presents an overview of this architecture. On the left-hand side there is the *prescriptive perspective*, where we use models for creating a system, whereas on the right-hand side there is the *descriptive perspective*, where models are extracted from running systems (i.e., executed models). In the following, we describe Fig. 1 from left to right.

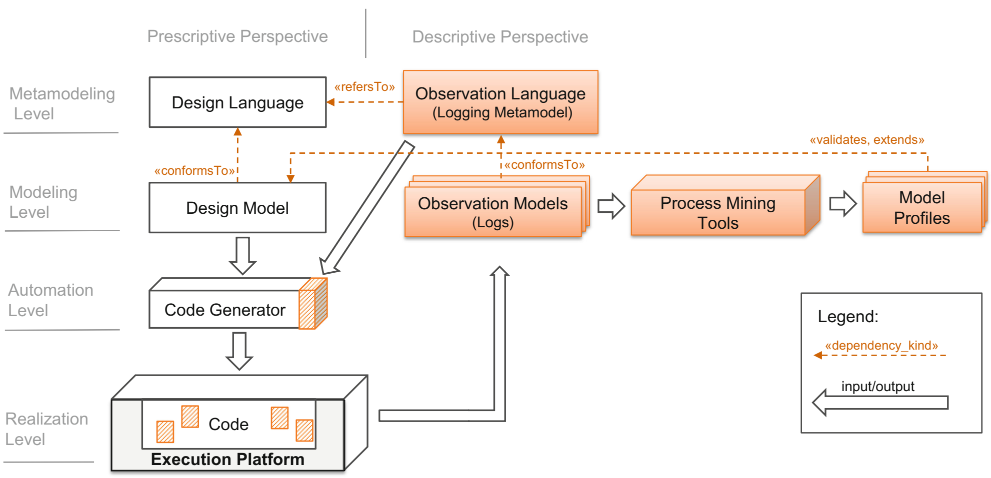


Fig. 1. Unifying conceptual architecture for MDE and PM.

The starting point is the *design language specification* at the metamodeling level which defines the syntax as well as semantics of a language like UML, SysML, or a certain domain specific language (DSML). The *design model* at the modeling level describes a certain system for a specific purpose and has to conform to the chosen design language (see Fig. 1, «conformsTo»). In our approach, such a model describes two different aspects of the system: (*i*) the *static aspect* which describes the main ingredients of the domain to be modeled, i.e., its entities and their relationships, and (*ii*) the *dynamic aspect* which describes the behavior of these ingredients in terms of events and interactions that may occur among them. For the vertical transition from the modeling level to the realization level (i.e., the process of transforming models into source code), we use *code generation* at the automation level as introduced in [3]. Finally, at the realization level, the running software relies on a specific platform for its execution (e.g., a Raspberry Pi as presented in our case study in Sect. 3).

At the right-hand side of Fig. 1 (at the top right), we present a logging metamodel—the so-called *observation language*. This metamodel defines the syntax and semantics of the logs we want to observe from the running system. In particular, we derive this metamodel from the *operational semantics* of the design language. This means that the observation metamodel can be derived from any modeling language that can be equipped with operational semantics. Figure 1 indicates this dependency at the metamodel level by the dashed arrow and the keyword «refersTo». The observation language has an influence on the code generator, which produces not only the code for the system to run, but also logging information (see Fig. 1, arrow from the observation language (input) to the code generator (output)). This means that the observation language determines which runtime changes should be logged and the code generator provides the appropriate logging code after every change (e.g., state change, attribute value change). Finally, these execution logs are stored as so-called *observation models* (see Fig. 1, arrow from the execution platform to the observation models). These observation models, which conform to the observation language, thumb the logs at runtime and provide these logs as input for any kind of tools used for checking purposes, e.g., for checking non-functional properties like performance, correctness, appropriateness. For instance, we transform the design language-specific observation model to a workflow representation which can be read by PM analysis tool as presented in our case study.

3 Case Study: Execution-Based Model Profiling

In this section, we perform an exploratory case study based on the guidelines introduced in [20]. The main goal is to evaluate if current approaches for MDE and PM may be combined in a loosely-coupled way, i.e., both can stay as they are initially developed, but provide interfaces to each other to exchange the necessary information to perform automated tasks. In particular, we report on our results concerning a fully model-driven engineered traffic light system which is enhanced with execution-based model profiling capabilities. All artifacts of the case study can be found on our project website[1].

3.1 Research Questions

As mentioned above, we performed this study to evaluate the feasibility and benefits of combining MDE and PM approaches. More specifically, we aimed to answer the following explanatory research questions (RQ) composed of two requirement satisfaction questions (Transformability, Interoperability), an effect question (Usefulness), and a trade-off question (Timeliness):

1. *RQ1—Transformability:* Is the operational semantics of the modeling language rich enough to automatically derive observation metamodels?
2. *RQ2—Interoperability:* Do observation metamodels satisfy interoperability by fulfilling the requirements of existing process mining formats?
3. *RQ3—Verifiability:* Are the generated model profiles resulting from the observation model sufficient for runtime verification?
4. *RQ4—Timeliness:* Are there significant differences between timing of transitions on the specification level and the implementation level?

3.2 Case Study Design

Requirements. As an appropriate input to this case study, we require a system which is generated by a MDE approach and equipped with an executable modeling language. This means that its syntax and operational semantics are clearly defined and accessible. Furthermore, the approach has to provide translational semantics based on a code generator which may be extended by additional concerns such as logging. Finally, the execution platform hosting the generated code must provide some means to deal with execution logs.

Setup. To fulfill these case study requirements, we selected an existing MDE project concerning the automation controller of a traffic light system. We modeled this example by using a small sub-set of UML which we named *Class/State Charts (CSC)* language. CSC stands for UML class diagram and UML state machine diagram, both shown in Fig. 2. The class diagram represents the static aspect of the system, whereas the state machine diagram describes the dynamic one. Generally, UML class diagrams consist of *classes* with *attributes*, and state charts containing *state machines* with *states* and *transitions* between them [21]. In a state chart diagram *transitions* can be triggered by different types of *events* like signal event, time event, call event, or change event [21]. Both, states and transitions can call *actions*.

Figure 2 presents the class diagram and state machine diagram of the traffic light system modeled in CSC. This system consists of several components such as lights (green, yellow, red) for cars and pedestrians, a controller as well as

[1] http://www.sysml4industry.org/?page_id=722.

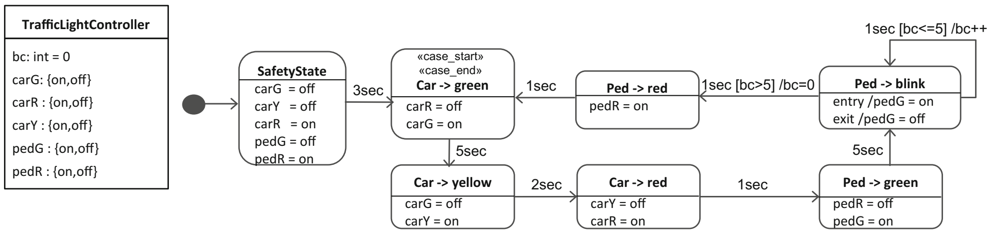


Fig. 2. CSC class diagram and state machine diagram of the traffic light system.

a blink counter for the pedestrian light. While the CSC state machine diagram (see Fig. 2, on the right-hand side) shows all possible and valid transitions/states within this example, the CSC class `TrafficLightController` (see Fig. 2, on the left-hand side) specifies the blink counter `bc:int=0` and the different lights which can be on or off.

We employed the Enterprise Architect[2] (EA) tool to model the CSC class and state machine diagram. Additionally, we used and extended the Vanilla Source plug-in of EA to generate Python code from the executed CSC (design) models. The code can be executed on a single-board computer. For this purpose we used Raspberry Pi (see Fig. 3, at the bottom left) as specific execution platform. It has to be noted that we aimed for full code generation by exploiting a model library which allows to directly delegate to the GPIO module (i.e., input/output module) of the Raspberry Pi.

3.3 Results

In this subsection, we present the results of applying the approach presented in Sect. 2.2 for the given case study setup. Firstly, we describe the technical realization of the example. Subsequently, we present the appropriate observation metamodel referring to the CSC design language and its conforming observation model. Finally, we generate different model profiles on the basis of PM techniques for checking purposes.

Technical Realization at a Glance. The execution logs of the running code on the Raspberry Pi form the basis for the experimental frame of our approach. Figure 3 gives an overview of its implementation. We extend the code generator to produce Python code (`CSC2Python`) which enables us to report logs to a log recording service implemented as `MicroService`, provided by an observation model repository. For data exchange between the running system and the log recording service we used JSON. This means that the JSON data transferred to the `MicroService` is parsed into log entry elements in the repository. We used the NoSQL database Neo4EMF[3] to store the execution logs for further

[2] http://www.lieberlieber.com.
[3] http://www.neoemf.com.

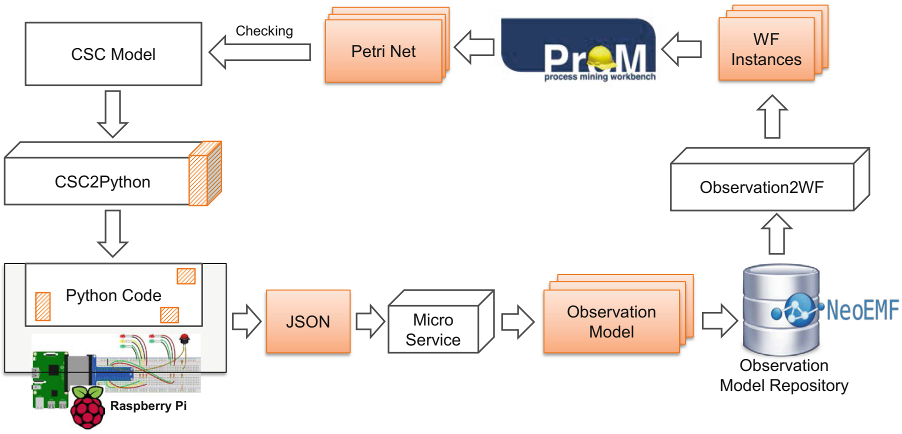


Fig. 3. Technical realization of the traffic light system example.

analysis. To be able to use established PM tools, we generated XML files from the recorded execution logs (i.e., the observation models).

For the case study of our approach we used ProM Lite 1.1[4] which is an open source PM tool. Files that this tool takes as input have to correspond to the XSD-schema of the workflow log language MXML[5]. To accomplish this we used the *ATLAS transformation language (ATL)* [12] for transforming the observation models to MXML-conform XML files (`Observation2WF`). In particular, we reverse-engineered the XML Schema of the MXML language into a metamodel. This step enabled us to translate the language-specific observation model into workflow instances (`WF Instances`) to directly import these instances in ProM Lite. For our case study example the used MXML format was sufficient. Nevertheless XES is the current standard, therefore, we will build on the XES format in future work.

The CSC Observation Metamodel. According to PM techniques, we consider an *observation model* as an event log with a start and end time registered as a *sequences of transactions* that having already taken place. However, we do not receive event logs from an executed process model (i.e., the activities of a business process in an ordered manner), rather we receive the traces from transformed log messages of an embedded system. Figure 4 shows the *observation metamodel* derived from the operational semantics of the CSC design language used in the context of this case study. The figure illustrates that changes at runtime are basically value updates for attributes of the CSC class diagram as well as updates concerning the current active state and current fired transition of the CSC state machine diagram.

[4] http://www.promtools.org/doku.php?id=promlite.
[5] http://www.processmining.org/WorkflowLog.xsd.

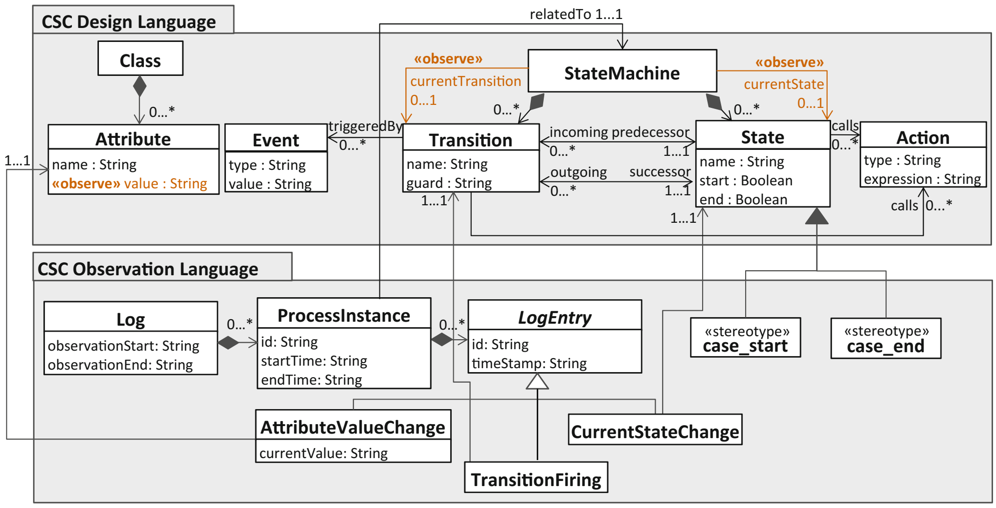


Fig. 4. Observation language for the CSC class diagram and CSC state machine diagram of the traffic light example.

As shown in the upper section of Fig. 4, these elements are marked with the «observe» stereotype. The CSC dependent observation metamodel is shown in the lower section of Fig. 4. The class `Log` represents a logging session of a certain running software system with a registered `observationStart` and an `observationEnd`. The class `Log` consists of process instances related to the CSC `StateMachine`. Every `ProcessInstance` has a unique `id`, `startTime`, and `endTime` attributes and consists of log entries with the attributes `id` and `timeStamp` for ordering purpose (i.e., indicating when the entry was recorded).

Additionally, we defined a subset of a state machine by indicating the stereotypes «case_start» and «case_end». These stereotypes have to be annotated in the design model whenever objects may execute more than one case. The reason for such a stereotype annotation is that, in contrast to business processes, state machines do not necessarily have a clearly defined start- and end point, like in the case of our traffic light system example. This is due to the fact that state machines are often defined for long-life (persistent) objects. This means that only values of objects change over time, but not the objects themselves. Therefore, we defined these stereotypes in our metamodel which enables us to capture single cycles (like cases in PM) of the state machine to be profiled. In our case study example, the start point and end point coincide. When the example starts, their is a safety state only entered once. Each further cycle starts and ends with the state `Car→green` (see Fig. 2).

The `LogEntry` either registers an `AttributeValueChange`, a `CurrentStateChange`, or a `TransitionFiring`. `CurrentStateChange` and `TransitionFiring` are associated with the state and the transition of the CSC design language. `AttributeValueChange` has an association with the changing attribute of a class and includes its `currentValue`.

Generated Model Profiles. We used ProM Lite for generating different *model profiles* from the observation model of the running code. For this purpose we employed ATL model transformations to import the CSC language-specific observation model as input into ProM Lite. By doing so, we focused on two PM-perspectives, (*i*) the control-flow perspective and (*ii*) the time perspective (cf. Sect. 2), as well as a (*iii*) data manipulation one. In the control-flow perspective, we employed the *α++-algorithm* of ProM Lite to generate Petri nets for reflecting all attribute value changes as well as state changes and their structure. For profiling the time perspective, we mined the sequence of fired transitions among all states with the *inductive miner* of ProM Lite and replayed the logs on the discovered Petri net by using a special performance plug-in of this tool.

In a first step of our case study, we implemented a model transformation in ATL which considered the state occurrences (`CurrentStateChange`) of the running system. By this, we checked on the one hand if the CSC state machine diagram is realized by the code generator as intended (see Fig. 5), and on the other hand, if the state machine executes the specified control-flow on the realization level. This enables, both, a semantically as well as syntactically "equivalence" checking of the prescriptive (design) model and the descriptive (operational) model. In particular, for semantically checking we compared the state space of the state machine with the state space of the profiled Petri net. As shown in Fig. 5 (see the dashed arrows) places with the same targets were merged. The dashed arrow at the bottom right symbolizes a manually interruption of a case. The figure shows that the places and transitions of the Petri net are equivalent to the states and transitions of the CSC state machine diagram presented in Fig. 2. For syntactically checking purpose we may define bi-directional transformation rules to check the consistency [5].

In a second step, we implemented a Python component in order to simulate random system failures which were not reflected in the initial design model presented in Fig. 2. We observed the control-flow perspective of this extended system and found out that the randomly simulated failure states were correctly detected by ProM Lite (compare the Petri net shown in Fig. 6 with that one

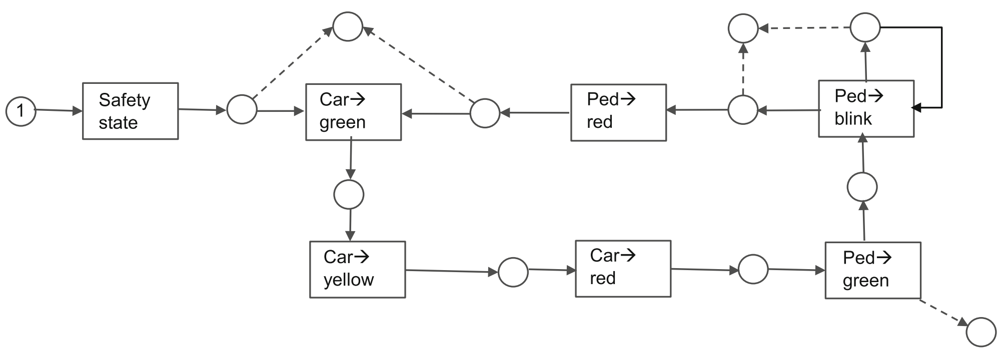


Fig. 5. Model profile of state changes.

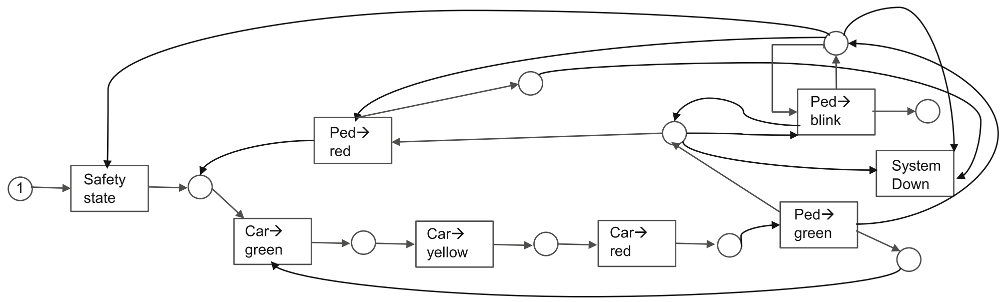


Fig. 6. Model profile of state changes including a failure state.

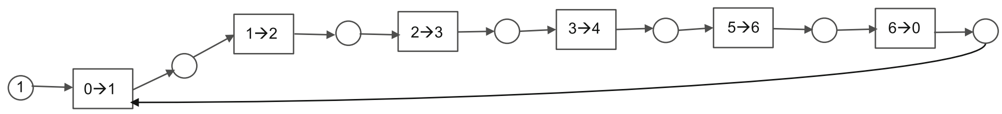


Fig. 7. Model profile of the attribute value changes for the blink counter (`bc`).

of Fig. 5). Thereby, we proof the usefulness of the approach for runtime verification. It shows that failures which may happened in the implementation phase would be correctly detected and visualized. For instance, this provides useful insights in the running system for validating the code generator and manual code changes.

In a next step, we developed another ATL transformation to extract for each attribute a workflow instance that contains the sequence of `AttributeValueChanges`s. By this, we extracted the shape of the values stored in the attribute to enrich the model with this kind of information and to check if certain value constraints were fulfilled during execution. For instance for the blink counter attribute, we derived a profile which explicitly shows a loop counting from zero to six as depicted in Fig. 7. These logged value changes conform to the attribute (`bc`) of the class `TrafficLightController` as shown at the left hand sight of Fig. 2.

In the CSC state machine diagram the timing component is explicitly assigned to transitions (see Fig. 2, «case_start» and «case_end»). In a last step of our case study, we observe the time perspective. Therefore, we needed an additional ATL transformation for filtering the sequence of `TransitionFiring`s (see Fig. 4 from the upper section to the lower section). This sequence includes several iterations of the traffic light system and is used as an input for the performance plug-in of ProM Lite. Our simulation covered 78 cycles, which took 22.26 min, and computed descriptive statistical values for performance evaluation like minimum, maximum and average transition time and sojourn time (i.e., waiting time), as well as the throughput which is the maximum rate at which a system can be processed. Table 1 presents the outcome of this descriptive analysis. To count several cycles (i.e., cases), we annotated the state `Car→green`

Table 1. Outcome of the performance evaluation based on transition firings.

Selected elements: Car→yellow to Car→red					
Timing_property	Min	Max	Avg	Std.Dev	Freq
Throughput_time	0.00 ms	0.00 ms	0.00 ms	0.00 ms	78
Waiting_time	2.02 s	2.12 s	2.04 s	19.24 ms	78
Sojourn_time	2.02 s	2.12 s	2.04 s	19.24 ms	78
Observation_period	22.26 min				

with the stereotypes «case_start» and «case_end» as introduced in the CSC metamodel. On average the transition from car yellow to car red is 2, 04 s, which is very close to the timing of transition (2 s) of the CSC state machine presented in Fig. 2.

3.4 Interpretation of Results

Answering RQ1. The operational semantics could be transferred into an observational viewpoint. By generating a change class for every element in the CSC design metamodel which is annotated with the «`observe`» stereotype, we are able to provide a language to represent observations of the system execution. This language can be also employed to instrument the code generator in order to produce the necessary logging statements as well as to parse the logs into observation model elements.

Answering RQ2. By developing ATL transformations from the language-specific observation metamodels to the general workflow-oriented formats of existing PM tools, we could reuse existing PM analysis methods for MDE approaches in a flexible manner. Not only the state/transition system resulting from the state machine can be checked between implementation and design, but also other mining tasks may be achieved such as computing value shapes for the given attributes of the CSC class diagram. Thus, we conclude that it is possible to reuse existing formats for translating the observations, however, different transformations may be preferred based on the given scenario.

Answering RQ3. For runtime verification, we took as input transformed event logs (i.e., selected state changes as a workflow file) and employed the α++-algorithm of ProM Lite to derive a Petri net. This generated Petri net, as shown in Fig. 5, exactly corresponds to the state machine, as shown in Fig. 2 on the right hand side. We are therefore convinced that the state machine is realized by the code generator as intended. Similarly, we have done this for attribute value changes. As output we extracted a value shape [0..6] stored in the attribute blink counter (see Fig. 7). Thus, we are also able to enrich the initial CSC class diagram presented in Fig. 2 with runtime information in terms of model profiles. Finally, we manually implemented random failure states in the Python code (not in the design model) in order to show that these system down states are reflected in the generated Petri net. By applying bi-directional transformations, these additional states may be also propagated to the initial CSC state machine diagram (i.e., prescriptive model) for completing the specification for error-handling states that are often neglected in design models [6].

Answering RQ4. For the detection of timing inconsistencies we filtered the sequence of transitions using an ATL transformation and analyzed it with the performance plug-in of ProM Lite. The inconsistencies between the specification and implementation levels are within the range of milliseconds. The average values of the delays can be propagated back to the design model in order to make the timing more precise during the system execution. The information about timing inconsistencies is especially relevant for time critical and safety critical systems, since this information may mitigate potential consequences of delays. However, it is important to observe a system for a sufficiently long period of time to have enough runtime information for reliable statistical values.

3.5 Threats to Validity

To critically reflect our results, we discuss several threats to validity of our study. First, in the current realization of our approach we do not consider the instrumentation overhead which may increase the execution time of the instrumented application. Of course, this may be critical for timed systems and has to be validated further in the future. Second, the current system is running as a single thread which means we are not dealing with concurrency. Extensions for supporting concurrency may result in transforming the strict sequences in partially ordered ones. Third, we assume to have a platform which has network access to send the logs to the micro service. This requirement may be critical in restricted environments and measurements of network traffic have to be done. Finally, concerning the generalizability of the results, we have to emphasize that we currently only investigated a single modeling language and a single execution platform. Therefore, more experiments are needed to verify if the results can be reproduced for a variety of modeling languages and execution platforms.

4 Related Work

We consider model profiling as a very promising field in MDE and as the natural continuation and unification of different already existing or emerging techniques, e.g., data profiling [1], process mining [23], complex event processing [15], specification mining [6], finite state automata learning [2], as well as knowledge discovery and data mining [9]. All these techniques aim at better understanding the concrete data and events used in or by a system and by focusing on particular aspects of it. For instance, data profiling and mining consider the information stored in databases, while process mining, FSA learning and specification mining focus on chronologically ordered events. Not to forget models@run.time, where runtime information is propagated back to engineering. There are several approaches for runtime monitoring. Blair et al. [4] show the importance of supporting runtime adaptations to extend the use of MDE. The authors propose models that provide abstractions of systems during runtime. Hartmann et al. [10] go one step further. The authors combine the ideas of runtime models with reactive programming and peer-to-peer distribution. They define runtime models as a stream of model chunks, like it is common in reactive programming.

Currently, there is emerging research work focusing on runtime phenomena, runtime monitoring as well as discussing the differences between descriptive and prescriptive models. For instance, Das et al. [7] combine the use of MDE, run-time monitoring, and animation for the development and analysis of components in real-time embedded systems. The authors envision a unified infrastructure to address specific challenges of real-time embedded systems' design and development. Thereby, they focus on integrated debugging, monitoring, verification, and continuous development activities. Their approach is highly customizable through a context configuration model for supporting these different tasks. Szvetits and Zdun [22] discuss the question if information provided by models can also improve the analysis capabilities of human users. In this context, they conduct a controlled experiment. Van der Aalst et al. [19] show the possibility to use runtime information and automatically construct simulation models based on event logs. These simulation models can be used, e.g., to evaluate performance of different alternative designs prior to roll-out. Heldal et al. [11] report lessons learned from collaborations with three large companies. The authors conclude that it is important to distinguish between descriptive models (used for documentation) and prescriptive models (used for development) to better understand the adoption of modeling in industry. Last but not least, Kühne [13] highlights the differences between explanatory and constructive modeling, which give rise to two almost disjoint modeling universes, each of it based on different, mutually incompatible assumptions, concepts, techniques, and tools.

5 Conclusion and Future Work

In this paper, we pointed to the gap between design time and runtime in current MDE approaches. We stressed that there are already well-established techniques considering runtime aspects in the area of PM and that it is beneficial to combine these approaches. Therefore, we presented a unifying conceptual architecture for execution-based model profiling, where we combined MDE and PM. We built the approach upon traditional activities of MDE such as design modeling, code generation, and code execution. In the conducted case study, we demonstrated and evaluated this approach on the basis of a traffic light system example. While the first results seem promising, there are still several open challenges, which we discussed in the threats to validity in the case study section. As next steps, we will focus on the observation of further PM perspectives (e.g., the organisational perspective) that can be used for software component communication discovery and on the reproduction of our current results by conduction additional case studies, in this respect, domain-specific modeling languages (DSMLs) would be of special interest.

Acknowledgment. The authors are affiliated with the Christian Doppler Laboratory for Model-Integrated Smart Production (CDL-MINT) at TU Wien, funded by the Austrian Federal Ministry of Science, Research, and Economy (BMWFW) and the National Foundation of Research, Technology and Development (CDG). Furthermore, the authors would thank LieberLieber Software GmbH for the provisioning of the traffic light example.

References

1. Abedjan, Z., Golab, L., Naumann, F.: Profiling relational data: a survey. VLDB J. **24**, 557–584 (2015)
2. Giles, C.L., Miller, C.B., Dong, C., Hsing-Hen, C., Guo-Zeng, S., Yee-Chun, L.: Learning and extracting finite state automata with second-order recurrent neural networks. Neural Comput. **4**(3), 393–405 (1992)
3. Brambilla, M., Cabot, J., Wimmer, M.: Model-Driven Software Engineering in Practice. Morgan & Claypool, San Rafael (2012)
4. Blair, G., Bencomo, N., France, R.B.: Models@run.time. IEEE Comput. **42**, 22–27 (2009)
5. Czarnecki, K., Foster, J.N., Hu, Z., Lämmel, R., Schürr, A., Terwilliger, J.F.: Bidirectional transformations: a cross-discipline perspective. In: Paige, R.F. (ed.) ICMT 2009. LNCS, vol. 5563, pp. 260–283. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02408-5_19
6. Dallmeier, V., Knopp, N., Mallon, C., Fraser, G., Hack, S., Zeller, A.: Automatically generating test cases for specification mining. IEEE TSE **38**, 243–257 (2012)
7. Das, N., Ganesan, S., Bagherzadeh, J.M., Hili, N., Dingel, J.: Supporting the model-driven development of real-time embedded systems with run-time monitoring and animation via highly customizable code generation. In: MoDELS (2016)
8. Dumas, M., van der Aalst, W.M.P., ter Hofstede, A.H.M.: Process-Aware Information Systems: Bridging People and Software Through Process Technology. Wiley, New York (2005)
9. Fayyad, U.M., Piatetsky-Shapiro, G., Smyth, P.: From data mining to knowledge discovery: an overview. In: Advances in Knowledge Discovery and Data Mining, pp. 1–34 (1996)
10. Hartmann, T., Moawad, A., Fouquet, F., Nain, G., Klein, J., Le Traon, Y.: Stream my models: Reactive peer-to-peer distributed models@run.time. In: MoDELS (2015)
11. Heldal, R., Pelliccione, P., Eliasson, U., Lantz, J., Derehag, J., Whittle, J.: Descriptive vs prescriptive models in industry. In: MoDELS (2016)
12. Jouault, F., Allilaire, F., Bézivin, J., Kurtev, I.: ATL: A model transformation tool. Sci. Comput. Program. **72**, 31–39 (2008)
13. Kühne, T.: Unifying explanatory and constructive modeling. In: MoDELS (2016)
14. de Lara, J., Guerra, E., Cuadrado, J.S.: Model-driven engineering with domain-specific meta-modelling languages. Softw. Syst. Model. **14**, 429–459 (2015)
15. Luckham, D.: The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Addison-Wesley, Boston (2005)

16. Mazak, A., Wimmer, M.: Towards liquid models: an evolutionary modeling approach. In: CBI (2016)
17. Meyers, B., Deshayes, R., Lucio, L., Syriani, E., Vangheluwe, H., Wimmer, M.: ProMoBox: A framework for generating domain-specific property languages. In: Combemale, B., Pearce, D.J., Barais, O., Vinju, J.J. (eds.) SLE 2014. LNCS, vol. 8706, pp. 1–20. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11245-9_1
18. Rozinat, A., van der Aalst, W.M.P.: Conformance checking of processes based on monitoring real behavior. Inf. Syst. **33**(1), 64–95 (2007)
19. Rozinat, A., Mans, R.S., Song, M., van der Aalst, W.M.P.: Discovering simulation models. Inf. Syst. **34**(3), 305–327 (2009)
20. Runeson, P., Höst, M., Sjoberg, D.: Guidelines for conducting and reporting case study research in software engineering. Empirical Softw. Eng. **14**, 131–164 (2009)
21. Seidl, M., Scholz, M., Huemer, C., Kappel, G.: UML Classroom: An Introduction to Object-Oriented Modeling. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-319-12742-2
22. Szvetits, M., Zdun, U.: Controlled experiment on the comprehension of runtime phenomena using models created at design time. In: MoDELS (2016)
23. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19345-3
24. van der Aalst, W.M.P.: Process mining. Commun. ACM **55**, 76–83 (2012)
25. van der Aalst, W.M.P., Leemans, M.: Process mining in software systems: Discovering real-life business transactions and process models from distributed systems. In: MoDELS (2014)

12 Reverse Engineering of Production Processes based on Markov Chains

A. Mazak, P. Patsuk-Bösch and M. Wimmer;
Proceedings of the 13th IEEE International Conference on Automation Science and Engineering (CASE), IEEE, (2017), pp. 680–686.
DOI: 10.1109/COASE.2017.8256182

2017 13th IEEE Conference on Automation Science and Engineering (CASE)
Xi'an, China, August 20-23, 2017

Reverse Engineering of Production Processes based on Markov Chains

Alexandra Mazak[1], Manuel Wimmer[1] and Polina Patsuk-Boesch[1]

***Abstract*— Understanding and providing knowledge of production processes is crucial for flexible production systems as many decisions are postponed to the operation time. Furthermore, dealing with process improvements requires to have a clear picture about the status of the currently employed process. This becomes even more challenging with the emergence of Cyber-Physical Production Systems (CPPS). However, CPPS also provide the opportunity to observe the running processes by using concepts from IoT to producing logs for reflecting the events happening in the system during its execution.**

Therefore, we propose in this paper a fully automated approach for representing operational logs as models which additionally allows analytical means. In particular, we provide a transformation chain which allows the reverse engineering of Markov chains from event logs. The reverse engineered Markov chains allow to abstract the complexity of run-time information as well as to enable what-if analysis whenever improvements are needed by employing current model-based as well as measurement-based technologies. We demonstrate the approach based on a lab-sized transportation line system.

I. INTRODUCTION

Production systems are becoming more and more software-intensive, thus, turning into *Cyber-Physical Production Systems (CPPS)*. In the manufacturing environment, these CPPS comprise smart machines, storage systems and production facilities capable of autonomously exchanging information and triggering actions [1]. As a consequence, the complexity of CPPS is continuously increasing. To deal with this increased complexity, *modeling* is a promising approach in this context. However, current modeling foundations and practices are still lacking behind the emerging requirements of Industrie 4.0. Amongst others, models are considered as static entities, mostly used as blueprints in the early design phase, but they are basically neglected in later life-cycle phases, which drastically limits their value during the production systems' operation [2]. This gap may result in discrepancies between design time models and its real world correspondent [3]. Therefore, one of the major milestones is the backpropagation of run-time information (i.e., measured data) derived from operations to engineering artifacts by exploiting *Internet of Things (IoT)* concepts, like smart sensors and actuators [1]. These requirements are emerging in industrial automation systems engineering, and thus, the software engineering community has been confronted with them [4].

In recent years, there emerged modeling technologies (e.g., multi-viewpoint modeling, multi-paradigm modeling) which allow users to apply different techniques such as simulation, model checking, etc., by using very promising modeling formalisms. However, modeling is mostly exploited for code generation approaches [5]. The other benefits of modeling such as dynamically extracting run-time models to better link operations with engineering are mostly overlooked. To counteract, *Model-driven Engineering (MDE)* has been proposed by which models are developed on a higher level of abstraction [6], [7], [8]. MDE has already found its way into the systems engineering domain, but not necessarily into the domain of industrial automation systems engineering [5].

In MDE, models are mostly used in *prescriptive* ways for system engineering. Although, models are an important ingredient to realize a system, for later phases in the systems' life-cycle additional model types are beneficial to use [9]. Therefore, *descriptive* models may be employed to better understand how a system is actually realized and how it is operating in certain environments [10]. Compared to prescriptive models, these other types of models are only marginal explored in the field of MDE, and if used at all, they are built manually. Thus, we aim for an automated reverse engineering approach that combines *model-based* downstream information derived from prescriptive models and *measurement-based* upstream information of a running production for managing the complexity of CPPS.

The remainder of this paper is structured as follows. In Section II, we outline the state of the art and main building blocks in the context of our approach. The case study used for motivating and demonstrating this approach is presented in Section IV. The reverse engineering framework is described in Section III. The application of our approach for the case study and an evaluation with a critical discussion are presented in Section V. Concluding the paper, in Section VI, remaining steps are discussed and an outlook on future research work is given.

II. STATE OF THE ART

In this section, we give a brief overview of related work in the field of industrial engineering, and we briefly describe the theoretic background and main building blocks, necessary for the context of the introduced approach which is influenced by these various different research fields. It is our goal to merge these topics as introduced in Section III.

A. Model-driven Engineering Techniques

There are two major MDE techniques: (i) *metamodeling* for specifying modeling languages, i.e., the structure and content of valid models, and (ii) *model transformations* to systematically manipulate models [7]. In our approach,

[1]A. Mazak, M. Wimmer, and P. Patsuk-Boesch are affiliated with the CDL-MINT and TU Wien at the Business Informatics Group (BIG), a research group of the Institute of Software Technology and Interactive Systems (e-mail: lastname@big.tuwien.ac.at).

978-1-5090-6781-7/17/$31.00

we use *metamodels* to specify language concepts and their relationships (i.e., abstract syntax), as well as concrete syntax (i.e., model notation), and semantics [7]. Metamodeling environments allow to generate modeling environments and are providing generic tool support, which can be employed for all the modeling languages defined with a metamodeling environment.

Generally, a *model transformation* is a program executed by a transformation engine which takes one or more models as input to produce one or more models as output. Model transformations are used to solve different tasks like code generation, model refactoring, or reverse engineering to name just a few examples. An important aspect is that model transformations are developed on the metamodel level, and thus, are reusable for all valid model instances [7]. For instance, we use *Model-driven Reverse Engineering (MDRE)* to create a set of models that represent a system. These models can then be used for different purposes, e.g., metrics and quality assurance computation, tailored system viewpoints, etc. For more insights in model-driven engineering in practice, we refer the interested reader to the work of Brambilla et al. [7].

B. Process Mining

Process mining (PM) is a process-centric management technique bridging the gap between data mining and traditional model-driven *Business Process Management (BPM)* [11], [12]. The main objective of PM is to extract valuable, process-related information from *event logs* for providing detailed information about actual processes, for instance, to identify bottlenecks, to anticipate problems, to record policy violations, to streamline processes, etc. [12]. In PM, *events* are defined as process steps and *event logs* as sequential events recorded by an information system [13]. This demonstrates that unlike BPM approaches PM works on the basis of event data instead of design models (i.e., prescriptive models).

In [11], van der Aalst lists three basic PM goals, which are (i) discovery, (ii) conformance, and (iii) enhancement. *Discovery* means to take an event log as input and to produce a process model as output. When targeting for *conformance* an existing process model is compared with an event log of the same process. This means an event log and a model are used as input and diagnostic information is produced as output. Thereby, a user can check whether information recorded in the log conforms to the intended model and vice versa. The third type of PM is called *enhancement*. Its idea is to improve or extend an existing model. It takes an event log and a model as input and produces a new model as output.

Current event processing technologies usually monitor single streams of events at a time. Even if users monitor multiple streams, they often end up with multiple "silo" views. A more unified view is needed that correlates with events from multiple data streams of various sources and in different formats. Thereby, heterogeneity and incompleteness of data are major challenges [14]. Mostly, PM operates on the basis of events that belongs to cases that are already completed [15]. This off-line analysis is not suitable for cases which are still in the pipeline. In [11], the author mixes current data with historic data to support on-line and off-line analysis.

C. Run-time Models

There are several different approaches for run-time modeling. All of them aim on bridging the gap between design time modeling and *run-time modeling* to enable run-time analysis. Blair et al. [16] show the importance of supporting run-time adaptations to extend the use of MDE. They propose models that provide abstractions of systems during run-time. These operational models are an abstraction of run-time states. Due to this abstraction, different stakeholders can use the models in various ways, like dynamic state monitoring or observing run-time behavior.

Hartmann et al. [17] go one step further. They combine the ideas of run-time models with reactive programming and peer-to-peer distribution. The authors define run-time models as a stream of model chunks, like it is common in reactive programming. The models are continuously updated during run-time, therefore, they grow indefinitely. With their interpretation that every chunk has the data of one model element, they process them piecewise without looking at the total size. In order to prevent the exchange of full run-time models, peer-to-peer distribution is used between nodes to exchange model chunks. In addition, automatic reloading mechanism are used to respond on events for enabling reactive modeling. As the models are distributed, operations like transformations have to be adapted. For this purpose transformations on streams as proposed by Cuadrado et al. can be used [18]. Reactive programming aims on enabling support for interactive applications, which react on events by focusing on streams. For this purpose a typical publish/subscribe pattern, well known as the observer pattern in software engineering [19], is used. Khare et al. show the application of such an approach in the IoT domain in [20].

D. Industrial Engineering

Folmer et al. [21] present in their work a *valve diagnosis system (VDS)* using data aggregated from multiple sources across company borders. The authors introduce a usual model for valve behavior and an adapted model enriched by process data and detailed design data of a plant's equipment. In their approach, they combine model-based *fault detection and isolation (FDI)* for valves with measurement-based techniques based on industry standards. The goal is to identify differences between both models for detecting gradual increasing valves' faults for plug wear and plug contamination.

Danar et al. [22] present approaches for process analysis and organizational mining in the domain of production automation engineering. The main goal of their mining approach is to align the run-time system process model with the design process model for conformance checking like in PM, as well as, to analyze the structure and interactions of system components during run-time, e.g., for improving maintenance planning. The authors evaluate their approach by using the SAW simulator [23], based on the *Simulator for Assembly Workhops (SAW) project*, as a running use case for a production automation system.

Vogel-Heuser et al. [24] focus in their work on software evolution in the domain of *automated production systems (aPS)*. They investigate the evolution and co-evolution of engineering models and code, quality assurance, as well as, variant and version management. In their work, the authors point to the fact that only focusing on challenges regarding the evolution of long-living automated production systems from a software perspective is not sufficient. Therefore, they (i) determine "why" this is not sufficient, (ii) present approaches to address the challenges in the aPS domain, (iii) define research goals, and (iv) identify "how" the presented approaches can lead to synergetic research goals and results focusing on the evolution of long-living aPS. In [25], Vogel-Heuser et al. provide an open case study for studying the evolution of automation systems by a bench-scale manufacturing system called *Pick and Place Unit (PPU)*. They present various scenarios to study the evolution in industrial plant automation and to document it. In [26], the authors present two different case studies. One from the domain of information systems, and the other one from the domain of automated production systems in order to validate their introduced approaches for analyzing the maintainability of software intensive systems.

E. Summary

One may argue that the before mentioned research fields often treated in isolation. Generally, the focus in MDE is on prescriptive models for code generation, whereas that one in PM is on descriptive models. However, Section II-D illustrates that both model types are required for the backpropagation of run-time information to design models in order to keep them up-to-date over their whole life-cycle. The models@run.time approach goes in a similar direction (cf. Section II-C). However, this research field focuses more on the creation of run-time models and their interaction with the environment as well as their continuously update during run-time based on changes within the environment, and not necessarily to combine design time with run-time for real-time tracking and tracing of models for their improvement, like we present in our stochastic-based reverse engineering approach.

III. Reverse Engineering of Markov Chains from Event Logs

A. Overview

In this section, we present a model-driven as well as data-driven reverse engineering approach to compute behavioral models from timely observations of system components. For this purpose we combine the prescriptive perspective of MDE with the descriptive one of PM (cf. Section I). By this, we generate descriptive models from execution logs which reflect the de-facto process characteristics and important performance characteristics of a system during run-time. In particular, we observe and analyze activities happening in a system during run-time for providing reasoning mechanisms for future adaptions. In doing so, we observe (i) *resources* that offer computing capabilities, (ii) *workload* that describes how these resources are being used, and (iii) *workload intensity* in terms of arrival times.

This approach bases on two metamodels, namely *CETO* and *MUPOM*. CETO extends existing concepts of PM (cf. Section II-B) like discovery process models from event logs. In MUPOM, we employ Markov processes of probability theory to describe resource-specific behaviors. Both metmodels are needed to combine the model-based perspective taking during systems engineering with the measurement-based one taking during the system's operation. CETO and MUPOM enable us to observe and to analyze the main characteristics of a system at run-time.

B. CETO Metamodel

The *Components Emission and Timely Observations (CETO)* model captures measurement data of the running system under observation. These tracked observations (i.e., resource-specific operation calls) base on the emissions generated by resources (e.g., machines, specific components) when being used, e.g., in a production process (cf. Section V). These emissions are observed by *operational logs* which reflect the activities happening in a system (e.g., IAF plant) during run-time. In this respect, these operational logs are handled very similar to event logs of PM (cf. Section II).

In the CETO model, we consider the duration time of operations applied on items. In a running system, these operations result in computing time on resources which is measureable. The time interval for an operation may vary for one item to another (e.g., based on the size of an item), or if an item processes an operation multiple times. The observation of many operation durations allows to build a probability distribution function over these durations. To put it in a nutshell, the CETO model tracks, measures and stores the emissions of the resources when being used during the system's operation.

C. MUPOM Metamodel

The *Markov Usage Process and Operation Measurements (MUPOM)* model describes resource-specific behaviors by using the Markov model formalism. We implemented the MUPOM model with the goal to model stochastic processes in order to extract the (probabilistic) *workload* of a production system. A workload is the primary stochastic element, since it changes and occurs in a non-deterministic way. It is used in a wide range of performance engineering literature in, both, model-based and measurement-based approaches (e.g., [27], [28], [29]).

The workload is the amount of work that requests some sort of service in a specific time interval. We describe the workload of a system by using simple Markov models like Markov chains. For this purpose we construct a model λ given the observing sequence of emissions in a specific time interval $\mathcal{O}_T$, where the probability $Pr(\mathcal{O}_T \mid \lambda)$ is maximized [30]. To achieve this, we have to assume that the system under observation is *ergodic*. This means that the system is *irreducible*, *aperiodic*, and *positive recurrent*. We briefly summarize these preconditions as follows [31]:

A system is *irreducible*, if it is possible to reach each state from any other state. The probability of being in state *j* after *n-steps* starting from *i* must therefore be greater than zero.

$$Pr(X_n = j \mid X_0 = i) > 0 \quad (1)$$

A system is *aperiodic*, if the system state is not systematically connected to time. And a system is *recurrent* if all states of the system are recurrent. A state is *recurrent*, if the summed probability of returning to that state for an infinite number of steps *n* is finite. To be *positive recurrent*, a system must be *irreducible* and *aperiodic*. This means that the system periodically restarts itself in finite time and every state *i* is visited infinitely often. This equates a expected return of [31]:

$$E(min \geq 1 : X_n = i \mid X_0 = i) < \infty \quad (2)$$

There are multiple ways to use a system. Thus, the workload may be substituted by so-called *operational profiles*. A single operational profile is expressed by sequences of component activations within the system. In our approach, the time between two component activations is described as "think times", which we express by probability distributions. *Workload intensities* are on top of operational profiles and describe the amount of items that are processed by the system in a certain time interval. Similar to the *Probabilistic Grammar model (HPG)* [32], we take all components of a system (e.g., turntables, conveyors, machines) as *states* and the relations between them as *transitions*.

The MUPOM metamodel defines only *open workloads* with external arrivals and departures. These arrival rates can be modeled by using three possible distributions: *Poisson*, *Exponential*, and *Normal distribution*. Therefore, duration and think times follow one of these distributions.

D. Transformation Chain

On the basis of the previously discussed MDE techniques (cf. Section II), we present a *modular transformation chain* from observed logs to Markov chains to abstract complexity of run-time information and for analytical purposes.

Markov chains have a discrete time space that can be finite or infinite. They can occur at any point in time. If the state transition probabilities do not change over time, a Markov chain is *stationary*. In the case of a discrete state space, the transition probabilities between states can be simply encoded in a *transition matrix*, whereas in continuous time a *transition rate matrix* is used [33]. In order to get a *n-steps* transition probability starting from a *state i* to an other *state j*, the transition matrix *P* is multiplied with itself *n*-times, denoted as P^n_{ij}. For $n \to \infty$ the resulting matrix may converge to a certain distribution which is called *limiting probability*, formally defined in [33] as: $\pi_j = \lim_{n\to\infty} P^n_{ij}$

Following the criterion of ergodicity, an irreducible aperiodic Markov chain is ergodic if and only if it has a positive stationary distribution in the form of [34]:

$$\pi_j = \frac{E_i\left[\sum_{n=0}^{\pi_i - 1} 1(X_n = j)\right]}{\mu_i},\ j \in S \quad (3)$$

where $\pi_i = min\ \{n \geq 1 : X_n = i\}$ and $\mu_i = E_i\,[\pi_i]$.

In order to combine CETO and MUPOM models and to transfer MUPOM to a Markov chain, we apply model-to-model transformations [35]. We apply these transformations to transform (i) an operation topology to CETO, (ii) CETO to MUPOM, and (iii) MUPOM to a Markov chain. This process, which we call *transformation chain*, enables the reverse engineering of Markov chains from event logs.

In a first step, we implement the CETO model by the *OperationsAndTraceMonitor* tool. This instrumentation tool enables (i) to track an item's navigation during run-time, (ii) to measure the duration of defined operations, and (iii) to write the observations to a `CSV` file for analysis purposes. The *OperationsAndTraceMonitor* produces two files: (i) `operationDuration.csv` and (ii) `itemTrace.csv`.

In the next step, the MUPOM model approach is implemented by the *UserTrace2Markov* tool. We use the output files (e.g., item traces) of the *OperationsAndTraceMonitor* tool as input of the *UserTrace2Markov* tool for computing Markov chains. In its first version, this tool constructs a Markov chain by calculating a *transition matrix*. Therefore, we calculate the average processing time for items based on the transition probability between two states for a concrete example (cf. Section V, Table I). Furthermore, think times of every state are calculated and described by an expected mean value and a standard deviation. Finally, the total aggregated time spent in each state by every item is summed up.

IV. Case Study

In this section, we provide the description of a reference case study, a lab-sized production system hosted at IAF of the Otto-v.-Guericke University Magdeburg [36] which is subsequently used as a demonstration example to exemplify our process reverse engineering approach. The IAF production system (cf. Figure 1) consists of a transportation line made up of sets of turntables, conveyors, and multi-purpose machines. Each turntable is equipped with an inductive proximity sensor for material detection and a motor for table rotation. The transportation line is wired to a modular fieldbus I/O system, which, in turn, communicates with Raspberry Pi based controllers by Ethernet. The Raspberry Pi based controller is running a Programmable Logic Controller (PLC) program governing the transportation line. Such programs logically divide the transportation line in three different areas as depicted in Figure 1. The production plant is supposed to continuously processes items by its multi-purpose machines located in one of the three areas. Turntables and conveyors are in charge of moving such items to these machines[1]. In the given IAF production system different events are observable such as

[1]Models realized for this case study can be downloaded from our CDL-MINT web site at the following address `https://cdl-mint.big.tuwien.ac.at/case-study-artefacts-for-case-2017/`

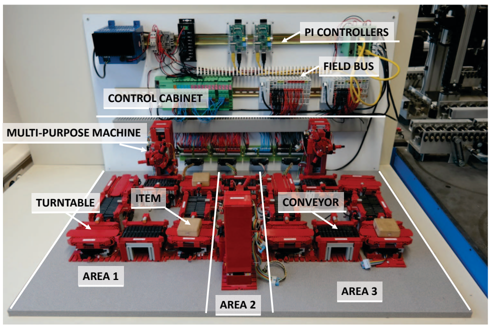


Fig. 1. Lab-sized production system hosted at IAF of the Otto-v.-Guericke University Magdeburg [36].

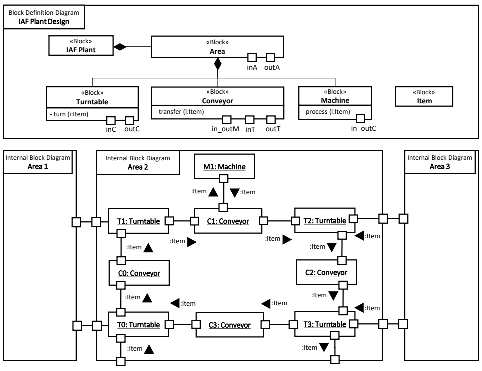


Fig. 2. Structural model of the IAF production system.

starting and ending different kinds of activities, for instance, turning an item by a turntable, transferring an item by a conveyor, and processing an item by a machine. As we have several turntables, conveyors, and machines, we assume that each of these components has a *unique identifier (ID)*. The presentation of the plant topology is achieved by Figure 2 which uses a SysML block diagram as notation. In the upper part of this figure, the different component types used for the production system are described by the block definition diagram including the operations which are provided by these components. We assume that the start and end events of calling these operations are logged by the controller. In the bottom part of Figure 2, we show the instances of the component types and how they are connected to represent the topology of the system by illustrating these aspects as an internal block diagram. In particular, we present `Area 2` as white-box to exemplify the structure of the production system. Even though, some aspects of the system are already explicitly modeled by SysML, a more *operational-oriented view* is needed to understand the process quality as well as the impact of changes on the behavioral aspects of the system. Of course, one could now start with modeling the intended process which should be supported by the production system with the risk of misinterpreting the system execution. A promising alternative is to observe the events of the running system and to derive process models automatically by applying performance metrics.

V. Evaluation Study

A. Setting and Outcome

For the purpose of validating our approach, we developed a prototypical implementation of the introduced modular transformation chain in the context of the IAF plant case study (cf. Section IV). All artifacts of this evaluation can be found on our CDL-MINT project website[2].

In particular, we created a SysML-based simulator for the IAF plant case study which is able to produce log files using the `OperationsAndTraceMonitor` (cf. Section III-D). An excerpt of the output is shown in Listing 1. The log entries are structured as follows: for each component or resource (e.g., turntable, machine, conveyors), identified by path expressions (e.g., `systemID/areaID/componentID`), we note the time stamp when the resource is requested by a certain `item`. Thus, this time stamp is considered as the start of a particular resources-specific operation, e.g., `Conveyor.transfer()`. The end of an operation is not explicitly represented in the log file since it is derived by searching for the next log entry (having the time stamp as close to the given time stamp for the start operation) for the given item. The textual log file format is inspired by existing log formats as used, e.g., for Web servers[3]. However, please note that this format can be automatically parsed in a model structure to allow compatibility, for instance, with different process mining tools, like ProMLite[4].

Listing 1. Exemplary `itemTrace` log file excerpt for the IAF plant case study

```
#Fields: component, timestamp, item
entered -/IAF/a2/t1,2017-02-08-23-28-51,923b4ff191d5
entered -/IAF/a2/c1,2017-02-08-23-28-54,923b4ff191d5
entered -/IAF/a2/t1,2017-02-08-23-28-57,83e5f507cff2
entered -/IAF/a2/t2,2017-02-08-23-28-61,923b4ff191d5
entered -/IAF/a2/c1,2017-02-08-23-28-63,83e5f507cff2
entered -/IAF/a2/m1,2017-02-08-23-28-69,923b4ff191d5
entered -/IAF/a2/t1,2017-02-08-23-28-74,5b73647866d4
...
```

The produced log files are the input for the `UserTrace2Markov` tool (cf. Section III-D), which produces a *transition probability matrix* of the underlying Markov chain. We illustrate this step for `Area 2` of the IAF plant example in Table I. The transition probability matrix has been generated based on the observed item traces from one state to another as described in Section III-D.

[2] https://cdl-mint.big.tuwien.ac.at/case-study-artefacts-for-case-2017/
[3] https://www.w3.org/Daemon/User/Config/Logging.html#common-logfile-format, https://www.w3.org/TR/WD-logfile.html
[4] http://www.promtools.org/doku.php?id=promlite

P_{ij}	t1	c1	t2	m1	t3	c2	t4	c3	in	out
t1	0	0.8	0	0	0	0	0	0	0	0.2
c1	0.1	0	0.9	0	0	0	0	0	0	0
t2	0	0.1	0	0.9	0	0	0	0	0	0
m1	0	0	0.1	0	0.9	0	0	0	0	0
t3	0	0	0	0.05	0	0.95	0	0	0	0
c2	0	0	0	0	0	0	1	0	0	0
t4	0	0	0	0	0	0	0	0.2	0	0.8
c3	0.98	0	0	0	0	0	0.02	0	0	0
in	0.95	0	0.05	0	0	0	0	0	0	0
out	0	0	0	0	0	0	0	0	0	1

TABLE I

TRANSITION PROBABILITY MATRIX FOR THE COMPONENTS OF AREA 2 OF THE IAF PLANT

In order to visualize the results of this matrix, as well as, the statistical performance information in a graphical modeling language, we additionally developed a specific modeling editor for MUPOM. This editor supports a *domain-specific modeling language (DSML)* for representing the resulting information to engineers. We present the resulting model for the given case study in Figure 3. This figures shows that there is some backwards routing which seems not beneficial. Especially, the path from T3-M1-T2-C1 seems unnecessary. Since C1 is no longer backwards routing, and consequently, is routing again the items forward. The reason for this is that the items are moved in one iteration from the entry point of the transportation line onwards which may cause backwards routing as the successor components are still occupied. In contrast, if a different routing strategy is used, such as moving items first from the components close to the exit point of the transportation line to make space for new items, the workload of the system can be positively improved.

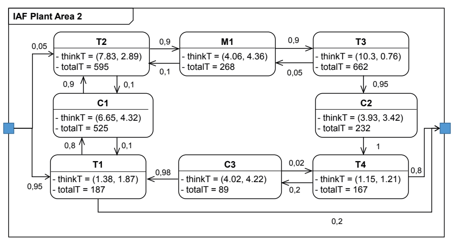


Fig. 3. MUPOM model of Area 2 of the IAF production system based on the observed traces.

By applying the proposed approach and its prototypical implementation, production systems provide a means for "self-modeling". Since structural models are mostly build in the engineering phases to guide the actual construction, performance models may be also employed already in these early phases. Often the experience is missing to approximate concrete numbers about workload characteristics. By using the presented measurement-based part of our approach, we are able to provide concrete information about the production system which may be used for checking workload assump-

Fig. 4. Analysis for estimating the required buffer size for the IAF plant.

tions made in the engineering phases, as well as, to find improvement possibilities after launching the first version of a system.

B. Critical Discussion

We only present a particular formalism which may be exploited for providing reasoning of the extracted observations. By using the modular transformation chain, we may also employ other formalisms such as, Petri nets which are automatically producible from the observed logs in the ProMLite tool. By this, interactive visualization may provide further insights on the currently employed production processes. We consider this line as a promising future research line which may complement the presented reverse engineering approach. Furthermore, other types of analysis can be performed. For instance, by utilizing the JMT tool[5], we are able to provide estimations for the given systems such as the required buffer length. By estimating the arrival rates of items, we are able to compute continuous time Markov chains which describe the probability of a certain buffer usage. For instance, the Markov chain shown in Figure 4 is computed for our given example. Based on this model, we can assume the maximum requirement for the buffer length as 6 given a arrival rate of 0.1 items per minute and a service time of 4 minutes per item.

VI. CONCLUSIONS AND OUTLOOK

In this paper, we presented an approach to automatically combine model-based downstream information derived from design (prescriptive) models with measurement-based upstream information derived from a running system. This reverse engineering approach computes behavioral models from timely observations based on the system components emission in form of operational logs (cf. Section III). These logs reflect the activities happening in a system during runtime. In the measuring part of our approach, we consider the duration time of operations applied on items (e.g., resource-specific operation calls). By tracing these logs, we provide the basis to abstract Markov chains by means of the presented modular transformation chain (cf. Section III-D). By this, we are able to deal with the complexity of run-time information and to provide reasoning mechanisms about future adaptations as presented in our evaluation (cf. Section V).

For future lines of research, we consider the following three topics: (i) big data analytics such as clustering of execution logs, e.g., to separate log entries into groups representing the behavior of different processes for testing purposes, (ii) computing complex events of higher abstraction based

[5] `http://jmt.sourceforge.net/`

on filtered execution logs, e.g., to analyze the reliability and availability of a system, (iii) the back-annotation of results in existing behavioral (prescriptive) models by means of a language specific model profiling approach.

VII. Acknowledgments

This work has been funded by the Austrian Federal Ministry of Science, Research and Economy (BMWFW) and the National Foundation for Research, Technology and Development (CDG). We are grateful for the help and support of Johannes Artner. Many thanks also to Prof. Dr.-Ing. Arndt Lüder of the Faculty for Mechanical Engineering at Otto-v.-Guericke University in Magdeburg, Germany for providing information on the demonstrator case.

References

[1] BITKOM, VDMA, and ZVEI, "Umsetzungsstrategie Industrie 4.0," Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e. V. (BITKOM), Verband Deutscher Maschinen- und Anlagenbau e. V. (VDMA), Zentralverband Elektrotechnik- und Elektronikindustrie e. V. (ZVEI), Ergebnisbericht der Plattform Industrie 4.0, 2015. [Online]. Available: http://www.bmwi.de/BMWi/Redaktion/PDF/I/industrie-40-verbaendeplattform-bericht

[2] M. Kowal, C. Legat, D. Lorefice, C. Prehofer, I. Schaefer, and B. Vogel-Heuser, "Delta modeling for variant-rich and evolving manufacturing systems," in *Proceedings of the 1st International Workshop on Modern Software Engineering Methods for Industrial Automation (MoSEMInA)*. ACM, 2014, pp. 32–41.

[3] M. L. van Eck, N. Sidorova, and W. M. P. van der Aalst, "Discovering and exploring state-based models for multi-perspective processes," in *Proceedings of the 14th International Conference on Business Process Management (BPM)*, 2016, pp. 142–157.

[4] D. Schmidt, "Guest Editor's Introduction: Model-Driven Engineering," *Computer*, vol. 39, no. 2, pp. 25–31, 2006.

[5] V. Vyatkin, "Software engineering in industrial automation: State-of-the-art review," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 3, pp. 1234–1249, 2013.

[6] J. Bézivin, "On the unification power of models," *Software and System Modeling*, vol. 4, no. 2, pp. 171–188, 2005.

[7] M. Brambilla, J. Cabot, and M. Wimmer, *Model-driven software engineering in practice*. Morgan & Claypool, 2012.

[8] J. Bézivin, R. F. Paige, U. Aßmann, B. Rumpe, and D. C. Schmidt, "Manifesto - model engineering for complex systems," *CoRR*, vol. abs/1409.6591, 2014.

[9] A. Mazak and M. Wimmer, "On marrying model-driven engineering and process mining: A case study in execution-based model profiling," in *Proceedings of the 6th International Symposium on Data-driven Process Discovery and Analysis (SIMPDA)*, 2016, pp. 78–88.

[10] R. Heldal, P. Pelliccione, U. Eliasson, J. Lantz, J. Derehag, and J. Whittle, "Descriptive vs prescriptive models in industry," in *Proceedings of the ACM/IEEE 19th International Conference on Model Driven Engineering Languages and Systems*, 2016, pp. 216–226.

[11] W. M. P. van der Aalst, "Process mining: making knowledge discovery process centric," *SIGKDD Explorations*, vol. 13, no. 2, pp. 45–49, 2011.

[12] ——, *Basics of Applied Stochastic Processes*, ser. Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer, 2011.

[13] M. Dumas, W. M. P. van der Aalst, and A. H. M. ter Hofstede, *Process-Aware Information Systems: Bridging People and Software Through Process Technology*. Wiley, 2005.

[14] D. Agrawal, P. Bernstein, E. Bertino, S. Davidson, U. Dayal, M. Franklin, J. Gehrke, L. Haas, A. Halevy, J. Han *et al.*, "Challenges and opportunities with big data," Purdue University, Cyber Center Technical Reports, Tech. Rep., 2011.

[15] B. F. van Dongen and W. M. P. van der Aalst, "A meta model for process mining data," in *Proceedings of the International Workshop on Enterprise Modelling and Ontologies for Interoperability (EMOI) co-located with the 17th Conference on Advanced Information Systems Engineering (CAiSE)*, 2005.

[16] G. Blair, N. Bencomo, and R. France, "Models@ run.time," *Computer*, vol. 42, no. 10, pp. 22–27, 2009.

[17] T. Hartmann, A. Moawad, F. Fouquet, G. Nain, J. Klein, and Y. L. Traon, "Stream my models: Reactive peer-to-peer distributed models@run.time," in *Proceedings of the 18th International Conference on Model Driven Engineering Languages and Systems (MoDELS)*. ACM/IEEE, 2015.

[18] J. S. Cuadrado and J. de Lara, "Streaming model transformations: Scenarios, challenges and initial solutions," in *Proceedings of the 6th International Conference on Theory and Practice of Model Transformations (ICMT)*. Springer, 2013, pp. 1–16.

[19] J. Vlissides, R. Helm, R. Johnson, and E. Gamma, *Design Patterns: Elements of Reusable Object-Oriented Software*, 1995.

[20] S. Khare, K. An, A. S. Gokhale, S. Tambe, and A. Meena, "Reactive stream processing for data-centric publish/subscribe," in *Proceedings of the 9th International Conference on Distributed Event-Based Systems (DEBS)*. ACM, 2015, pp. 234–245.

[21] J. Folmer, C. Schrufer, J. Fuchs, C. Vermum, and B. Vogel-Heuser, "Data-driven valve diagnosis to increase the overall equipment effectiveness in process industry," in *Proceedings of the 14th IEEE International Conference on Industrial Informatics, (INDIN)*, 2016, pp. 1082–1087.

[22] W. D. Sunindyo, S. Biffl, R. Mordinyi, T. Moser, A. Schatten, M. T. Irani, D. Wahyudin, E. R. Weippl, and D. Winkler, "An event-based empirical process analysis framework," in *Proceedings of the International Symposium on Empirical Software Engineering and Measurement, (ESEM)*, 2010.

[23] M. Merdan, T. Moser, D. Wahyudin, S. Biffl, and P. Vrba, "Simulation of workflow scheduling strategies using the MAST test management system," in *Proceedings of the 10th International Conference on Control, Automation, Robotics and Vision, (ICARCV)*, 2008, pp. 1172–1177.

[24] B. Vogel-Heuser, A. Fay, I. Schaefer, and M. Tichy, "Evolution of Software in Automated Production Systems: Challenges and Research Directions," *Systems and Software*, vol. 110, pp. 54–84, 2015.

[25] B. Vogel-Heuser, C. Legat, J. Folmer, and S. Feldmann, "Researching Evolution in Industrial Plant Automation: Scenarios and Documentation of the Pick and Place Unit," Technical University of Munich, Tech. Rep. TUM-AIS-TR-01-14-02, 2014.

[26] B. Vogel-Heuser, S. Feldmann, J. Folmer, S. Rösch, R. Heinrich, K. Rostami, and R. H. Reussner, "Architecture-based assessment and planning of software changes in information and automated production systems state of the art and open issues," in *Proceedings of the 2015 IEEE International Conference on Systems, Man, and Cybernetics*, 2015, pp. 687–694.

[27] "The UML for MARTE Specification," http://www.omg.org/spec/MARTE/1.1/, accessed: 2016-06-30.

[28] D. B. Petriu and M. Woodside, "An intermediate metamodel with scenarios and resources for generating performance models from UML designs," *Software & Systems Modeling*, vol. 6, no. 2, pp. 163–184, 2007.

[29] A. Van Hoorn, M. Rohr, and W. Hasselbring, "Generating probabilistic and intensity-varying workload for web-based software systems," in *Proceedings of the SPEC International Performance Evaluation Workshop*, 2008, pp. 124–143.

[30] L. Rabiner and B.-H. Juang, "An introduction to hidden Markov models," *ASSP Magazine, IEEE*, vol. 3, pp. 4–16, 1986.

[31] P. Walters, *An Introduction to Ergodic Theory*, ser. Graduate Texts in Mathematics. Springer, 1982, vol. 79, no. 1.

[32] J. Borges and M. Levene, "Data mining of user navigation patterns," in *Revised Papers from the International Workshop on Web Usage Analysis and User Profiling*, ser. WEBKDD '99. Springer, 2000, pp. 92–111.

[33] K.-H. Waldmann and U. M. Stocker, *Stochastische Modelle*. Springer, 2004.

[34] R. Serfozo, *Basics of Applied Stochastic Processes*, ser. Probability and Its Applications. Springer, 2009.

[35] L. Berardinelli, E. Maetzler, T. Mayerhofer, and M. Wimmer, "Integrating Performance Modeling in Industrial Automation through AutomationML and PMIF," in *Proceedings of the International Conference on Industrial Informatics (INDIN)*, 2016, pp. 1–6.

[36] "Equipment Center for Distributed Systems," http://www.iaf-bg.ovgu.de/en/technische_ausstattung_cvs.html, Institute of Ergonomics, Manufacturing Systems and Automation at Otto-v.-Guericke University Magdeburg, 2016.

13 Automatic Reverse Engineering of Interaction Models from System Logs

A. Mazak, S. Wolny and M. Wimmer;
Proceedings of the 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), IEEE, (2019), pp. 57–64.
DOI: 10.1109/ETFA.2019.8869502

Automatic Reverse Engineering of Interaction Models from System Logs

Sabine Wolny*, Alexandra Mazak*, Manuel Wimmer*
* Christian Doppler Labratory for Model-Integrated Smart Production (CDL-MINT)
Institute for Business Informatics - Software Engineering
Johannes Kepler University Linz, Science Park 3, 4020 Linz, Austria
{firstname}.{lastname}@jku.at

***Abstract*—Nowadays, software- as well as hardware systems produce log files that enable a continuous monitoring of the system during its execution. Unfortunately, such text-based log traces are very long and difficult to read, and therefore, reasoning and analyzing runtime behavior is not straightforward. However, dealing with log traces is especially needed in cases, where (i) the execution of the system did not perform as intended, (ii) the process flow is unknown because there are no records, and/or (iii) the design models do not correspond to its real-world counterpart. These facts cause that log data has to be prepared in a more user-friendly way (e.g., in form of graphical representations) and algorithms are needed for automatically monitoring the system's operation, and for tracking the system components interaction patterns. For this purpose we present an approach for transforming raw sensor data logs to a UML or SysML sequence diagram in order to provide a graphical representation for tracking log traces in a time-ordered manner. Based on this sequence diagram, we automatically identify interaction models in order to analyze the runtime behavior of system components. We implement this approach as prototypical plug-in in the modeling tool Enterprise Architect and evaluate it by an example of a self-driving car.**

***Index Terms*—Log traces, model transformation, sequence diagram, interaction model**

I. Introduction

Nowadays, in Model-driven Engineering (MDE) the use of object-oriented modeling languages and code generators for generating code is an already established approach for developing complex systems [1]. However, even if systems are described by means of such modeling languages and code generators are used to transform model elements to corresponding code statements, the execution of these statements is typically not represented in the same structure as the languages' metamodel. Based on this fact it is difficult to prove whether the design model corresponds to its runtime counterpart, meaning that the discovery of discrepancies is not straightforward. Therefore, engineers would benefit if they could treat runtime data in the same way as design models, i.e. operate with them like standard UML or SysML models. This would help to discover discrepancies more easily and, if appropriate, propagate this information back to the initial model. Thus, an automatic control and improvement of design decisions is enabled by establishing the well-known PDCA (plan-do-check-act) management method in the field of MDE.

During the execution of software or the operation of systems, the behavior, communication, as well as executed operations can only be traced based on sensor value streams or logging code. Typically such log traces have the form of huge text-based files, which are difficult to read and process. Therefore, it is not straightforward to fully understand and track the interaction and communication between system components. A scalable reverse engineering approach is needed that automatically transforms log traces to an appropriate graphical representation allowing an object-oriented view on executed operations as well as the back propagation of runtime information to design models.

In this paper we overcome these obstacles (i) by providing an automatically performed text-to-model transformation from text-based log traces to a graphical representation in form of UML sequence diagrams as object-oriented interaction models of executed operations, (ii) by aligning these models with their initial counterpart (i.e., design models) for creating so-called trace links, and (iii) by automatically creating runtime profiles and displaying these profiles in the design model.

The remainder of this paper is structured as follows. Section II shows the background as well as a motivating example to underline the challenges. In Section III, we present our automatically executable reverse approach by presenting a metamodel for object-oriented logs and by describing the architecture for the approach, and its prototypical implementation. Section IV demonstrates the evaluation of the introduced approach based on an example of log traces generated by a self-driving car. Section V discusses related work. In Section VI, we conclude this paper by an outlook on our next steps.

II. Background and Motivating Example

After briefly discussing the main background which forms the basis of our approach, we present a motivating example and discuss open issues to address.

In MDE, the abstraction power of models is used to tackle the complexity when representing systems [1]. From an abstracted point of view, MDE follows the principle of "everything is a model" [2]. This means, MDE supports system as well as software engineers by providing formal models like a tool box to achieve simplicity, generality, and integration

978-1-7281-0303-7/19/$31.00 ©2019 IEEE

in the design of systems. In this early phase of development such models are used to create and describe the scope and detail of interest. These so-called design models are then used to realize a system by automatically transforming model elements to code statements which can be then executed on a platform, as already mentioned in Section I. For this purpose MDE provides model transformations such as Model-to-Model (M2M) or Model-to-Text (M2T) transformations [2]. Based on this method, we implement a Text-to-Model (T2M) transformation to transform text-based log trace files to a user-friendly representation for analyzing the execution process of a system (cf. Section III). This means that we use established methods and techniques from MDE as background to establish an end-to-end traceability from design to runtime and back.

A. Motivating Example

Consider a simple autonomous car consisting of sensors, motor and servo controls for driving itself (i.e., without human control) in different environmental settings. This car should be able to (i) recognize barriers, (ii) change direction if necessary, (iii) drive forward and backward or stop. In order to fulfill these requirements the car needs an autonomous acting controller that controls the motor as well as the servo control based on continuously gathered sensor value streams. At design time, the structural and behavioral aspects of such a car are modeled by using a modeling language like UML[1]. The components with their properties and operations are modeled in a class diagram (CD) whereas the intended behavior of the system is modeled by a state machine (SM).

As usual in MDE, the formal model of a SM is used to automatically generate code statements for the car controller. This means that the modeled behavior at design time is used for running the controller. However, during operation the car is driving in different environmental settings. In this execution phase, it is important for the engineers to know whether the system behaves as intended or if there are occurring any unexpected transitions or error states.

In addition, it would be of interest if every operation is executed and, if so, how often, and if there are any specific interaction patterns between controller, motor and servo. To obtain such information based on the runtime behavior of the car, message flows between these components has to be logged and analyzed. However, such textual-based log files are huge, and therefore, difficult to interpret for system engineers. The challenges are: (i) providing a method to visualize those logs in a readable format, (ii) querying communication messages between the controller and the actuators (i.e., servo and motor), and (iii) extracting information of the observed system for improving the design after each execution, so to say, for holding the "model-in-the-loop" according to the PDCA cycle (cf. Section I).

III. APPROACH

In this section, we present our model-driven approach for automatically reverse engineering interaction models in terms of sequence diagrams from system logs for enhancing design models with runtime views. We start this section by presenting the requirements for our architecture and the required structure of system logs in terms of a metamodel. Subsequently, we present the conceptual architecture, and finally, the prototypical implementation of our approach.

A. System Requirements

First of all, we have a number of prerequisites to be met: (i) the structure of the system and its behavior has to be expressible by means of a family of object-oriented modeling languages such as UML class diagrams and UML state machines, (ii) the executed operations have to be logged and should represent inter-object communication, and (iii) the different executed operations have to be uniquely identifiable to ensure that the system logs can be clearly mapped to classes and their assigned operations.

B. System Logs - The Object-Oriented View

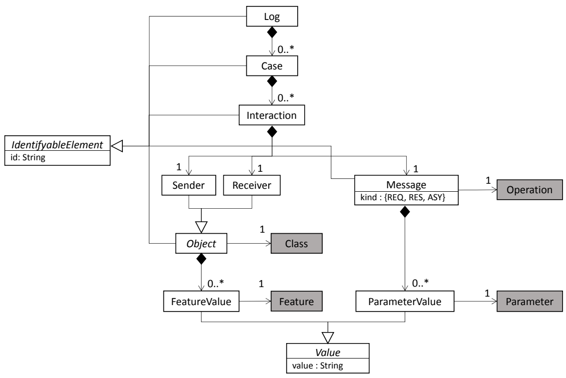


Fig. 1. Metamodel for an object-oriented system log representation

On the basis of the observed and recorded system logs, we automatically reverse engineer an object-oriented interaction model making the communication among system components read- and traceable, as well as enabling the back propagation of profiled information to the design models. With the term object-oriented interaction model we refer to UML interaction models. Such models follow the object-oriented paradigm: objects communicate with other objects via messages. A sequence of different messages results in an interaction between a set of objects. One kind of displaying such interaction models are UML sequence diagrams[2]. Amongst others, sequence diagrams focus on the representation of different interaction participants with their lifelines and messages between them, which can be either synchronous or asynchronous, based on their temporal order. For reconstructing such interaction models we develop a metamodel for representing system logs in an object-oriented

[1]http://www.uml.org/

[2]https://www.uml-diagrams.org/sequence-diagrams.html

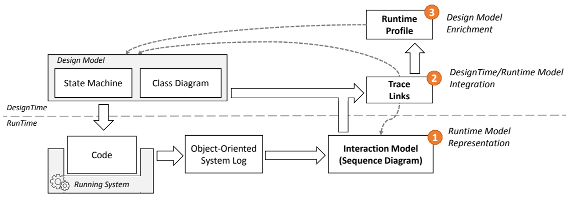


Fig. 2. Architecture for reverse engineering of interaction models from system logs

manner. This enables us to analyze logs from an object-oriented viewpoint as an explicit model. System logs are explicitly expressed as models and thus the unification power of models [2] can be fully applied. For example, to compare two system logs, model-oriented approaches such as model comparisons and model diffing processes can be used without taking further action.

Fig. 1 shows the metamodel we employ for capturing object-oriented system logs. A `Log` consists of any number of `Cases` which in turn consist of any number of interactions. Such an `Interaction` has to be composed of a `Sender`, a `Receiver`, and a `Message`. `Sender` and `Receiver` refer to an `Object` and can have additional optional `Feature Values`. A `Message` can comprise of any number of `Parameter Values`. All elements of this object-oriented log (except values) are indirect instances of `IdentifyableElement`, and therefore, have an `id` assigned. In addition, this object-oriented view can be mapped to a class view where `Objects` are referencing `Classes`, `Feature Values` are referencing `Features`, `Messages` are referencing `Operations`, and finally, `Parameter Values` are referencing `Parameters`. The presented metamodel enables the construction of object-oriented interaction models with all relevant information for reflecting and improving design decisions.

C. Architecture

Based on the defined requirements and the use of the object-oriented log metamodel, we develop the architecture for our model-driven reverse engineering approach. Fig. 2 shows the architecture and describes the interplay of design time and runtime artefacts. The architecture can be divided into three parts: First, the creation of object-oriented interaction models in form of sequence diagrams from executed operations (Fig. 2, Circle 1); second, the alignment between these sequence diagrams and their corresponding design models for creating so-called trace links (Fig. 2, Circle 2); and third, the creation of a runtime profile and display of that profile in the design models (Fig. 2, Circle 3).

For the creation of sequence diagrams from system logs (Fig. 2, Circle 1), we need the executed operations at runtime transformed to object-oriented logs as described in the metamodel shown in Fig. 1. There are two options to create such logs. On the one hand, such log files can be created manually, based on the implementation (i.e., coding). On the other hand, it is feasible that such a specific logging is already considered at design time by annotating a defined stereotype to certain model elements as presented and evaluated in previous work [3]. For this purpose we extended a code generator that recognizes certain annotated model elements. For these, code is generated that automatically logs changes at runtime. Thereby we use a `Class Diagram (CD)` to describe the structure of the system and its properties and operations, and a `State Machine (SM)` to model the behavior by states and transitions (see Fig. 2, System@DesignTime, and Section II).

In a further step, the system log files (no matter if generated manually or automatically by the extended code generator) are used as input for a Text-to-Model (T2M) transformation for automatically deriving an object-oriented interaction model in form of a `Sequence Diagram (SD)` (cf. Section III-D). Based on the metamodel described by Fig. 1, we perform an alignment between runtime and design time models (Fig. 2, Circle 2). Based on the defined classes and operations during design time we analyze the lifelines and messages to generate trace links between corresponding elements. This enables us to consider runtime and design time together. We query the elements of the SD in combination with the original CD and SM by an Application Programming Interface (API) (cf. Section III-D).

Based on these trace links we can go a step further by extracting some additional runtime information from the SD. This information is displayed as profile over runtime in the design model (Fig. 2, Circle 3). For this purpose we query our traces by the API to analyze if the message exchange works as intended and to count, e.g., the frequency of operation calls. This obtained runtime information is then saved as tagged values by a stereotype in the original design model (cf. Section III-D). This means that we learn from the runtime (actual) in order to constantly improve design models (target), as intended by the PDCA-cycle. But most importantly this reverse engineering approach enables an end-to-end traceability from design to runtime and back again. The elements can be examined on different levels, which enables to navigate from instance level to type level or the other way around.

D. Prototypical Implementation

For a prototypical implementation in a first version, we employ the modeling analysis and design tool Enterprise Architect[3] (EA). By using this tool, we model the CD and SM and use the extended code generator, which we presented in [3], to automatically generate execution code from these design models. During operation the execution is recorded in form of object-oriented system logs as presented in the metamodel (see Fig. 1). For generating the interaction model as SD, we developed the EA Add-In “EA Sequence Miner”. It is written in C# with the EA Automation API ActiveX COM library[4]. This Add-In enables to automatically transform

[3]https://www.sparxsystems.de/uml/neweditions/
[4]http://www.sparxsystems.com/enterprise_architect_user_guide/13.5/

system logs into a SD by a T2M-transformation.

The generating process consists of the following steps:

1) A new SD is created.
2) A new lifeline is created for each of the different values of `Sender` and `Receiver`.
3) For every `Message` entry in the log file a message with the corresponding name, parameter and message type is created in the same time order as in the log trace from sender lifeline to receiver lifeline.
4) Saving the generated SD elements in the integrated database of the EA file.

In our prototypical implementation the system log is stored in a csv-File with the following structure:

```
case;timestamp;Sender;Receiver;
    Message;ParameterValue;kind
1;2019-02-27 17:38:13.991;Car;Car;
    SteerStraight;none;REQ
1;2019-02-27 17:38:13.992;Car;ServoControl;
    SteerTo;direction=7;REQ
1;2019-02-27 17:38:15.145;ServoControl;Car;
    SteerTo;direction=7;RES
...
```

Such system log is used as input for the EA Sequence Miner to generate appropriate SD. Fig. 3 (not bold parts) shows an excerpt of an automatically generated SD of our motivating example of a self-driving car (cf. Section II). The SD shows the different instances of the car components, their lifelines, and the exchange of messages among them. Thus, the workflow of the car in a certain setting is easier to track and understand than per mining a text-based file.

For the alignment of runtime and design time information as well as for analysis purpose, we have developed an analysis tool in C# with the UniqueMint API[5] developed by LieberLieber Software GmbH[6] for EA. By using this library we are able to navigate through and to query UML models. After loading the generated SD by the UniqueMint API, we query the SD by using Language Integrated Query[7] (LINQ). For example, to get the instance classifiers (`represents`) of our lifelines we checked the class names and compare them with the lifelines:

```
public void SettingRepresentLifeline(IMyLifeline lifeline)
{
  lifeline.Represents = FindClass(lifeline.Name);
}
private IMyClass FindClass(String lifeline)
{
 var classProp = ClassesList.AsParallel().
   Where(g => g.Name.Contains(lifeline)).FirstOrDefault();
 return classProp;
}
```

The same principle is used for identifying if there exist belonging operations for the messages or not. In addition, parameters have to be considered. Return messages of calls are not linked to the original models. In Fig. 3 the established traces are shown in bold.

[5]https://demo.lieberlieber.com/EnArWeb/enarhtmlexport/MyUml/index.html
[6]https://www.lieberlieber.com/
[7]https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/linq/

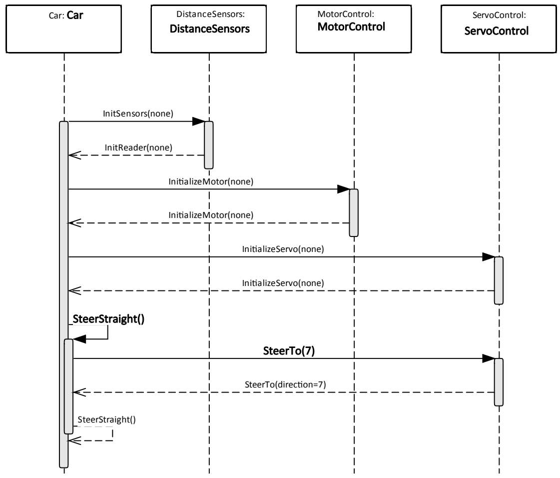


Fig. 3. Interaction model (SD) with established traces to CD (excerpt)

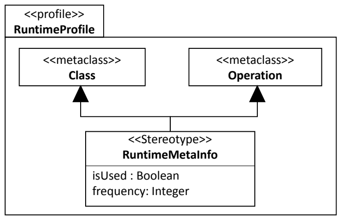


Fig. 4. Stereotype and tagged values for runtime information

Based on these established trace links we now extend our CD by a runtime profile consisting of a stereotype for classes and operations. Fig. 4 shows the `RuntimeProfile` with the `RuntimeMetaInfo` stereotype and its tagged values. On the one hand, we store the information whether an operation or class is used during execution (`isUsed`) as boolean value, and on the other hand, we calculate how often instances or operations are used in the interaction (`frequency`) as integer value. For this purpose we use the already established trace links and again the UniqueMint API. Based on our LINQ queries, we extract the runtime information and saved it as tagged values for the specific operations and classes in the CD.

For instance to get frequency of a specific operation («operation») of a specific instance of a class («class») in the SD the following query searches through all messages (`MessageList`). According to the UniqueMint API `m.Signature` specifies the corresponding operation.

```
List<IMyMessage> specificMessages =
    MessageList.AsParallel().
    Where(m => m.Signature==«operation»).
    ToList();
int frequency = specificMessages.Count;
```

The full implementation of our approach can be found at our project website[8].

IV. EVALUATION

In this section, we present and discuss the accuracy of our approach using a case study of a self-driving car based on the motivating example (see Section II). We follow the guidelines for conducting empirical explanatory case studies by Roneson and Höst [4].

A. Research Questions

Our general evaluation interest is the scalability of our presented approach. Therefore, in our study we aim to answer the following research questions (RQs):

RQ1—Scalability of interaction model generation: How long does it take to generate an interaction model from a system log?

RQ2—Scalability of interaction diagram generation: How long does it take to generate a diagram for an interaction model? Which system log sizes can be displayed as sequence diagrams?

RQ3—Scalability of interaction model integration: How long does it take to integrate the interaction model, i.e., runtime model, with the design model?

RQ4—Scalability of interaction profiling: How long does it take to compute the profiling information?

RQ5—Overall performance: How long does it take in total to (a) show a interaction model to an engineer, and (b) to present the profiling information to an engineer?

B. Case Study Design

a) Requirements: As an appropriate input for our case study, we require an automated system such as a self-driving car where at runtime system logs can be observed as object-oriented logs. In addition, we require that the structure of the system is modeled with UML class diagrams (CD) and instances of the system logs can be uniquely assigned to classes of the CD.

b) Setup: For our evaluation, we use different system runs resulting in different object-oriented system logs regarding the number of messages between the individual components of the self-driving car (see Table I). In our self-driving car case with four stationary components, the number of messages therefore varies from 10 to 100,000. The upper limit can be explained by the fact that with 100,000 messages, a broad spectrum of different interaction patterns can occur and thus can be analyzed by our approach.

TABLE I
DIFFERENT SIZES OF THE INVESTIGATED SYSTEM LOGS

#Lifelines	*#Messages*						
4	10	100	1,000	6,000	10,000	50,000	100,000

For answering our RQs, we calculate the duration between the start of SD generation and the finished process and the duration for each query by System.DateTime in C# in milliseconds (ms). The performance is measured on an Acer Aspire VN7-791 with an Intel(R) Core(TM) i7-4720 HQ CPU@2.60 GHz 2.60 GHz, with 16 GB of physical memory, and running Windows 8.1. 64 bits operating system. Please note that we measured the CPU time by executing each query and SD generation three times for all different settings and calculated the arithmetic mean of these runs in milliseconds (ms). To create our models we use Enterprise Architect version 13 with the integrated Microsoft Access database to store the models.

[8]https://cdl-mint.se.jku.at/case-study-artefacts-for-etfa-2019/

C. Results

We now present the results of applying our approach to the different settings of our self-driving car. Table II shows the execution times for generating the interaction models without visualization in a diagram (only saving the components in the database) and with visualization. In addition, the required disk space of the persisted models in the database is shown. It is noticeable that the difference between generation with and without visualization is marginal. However, the duration of the generation of the diagram rises sharply from a size of 50,000 messages. Fig. 6 (Interaction Model with and without Visualization) shows that the process is not linear, but polynomial (2nd degree). On the other hand, database sizes behave linearly as shown in Fig. 5.

For analysis of the creation of trace links and runtime profiles, we assume that the models are already loaded in memory. Therefore, we only analyze the concrete duration for the execution of the different queries. Table III shows the execution times for establishing the trace links and generating the runtime profile.

As before for the generation of sequence diagrams, the distribution of the data for establishing trace links and for generation the profile correspond in each case to a quadratic function.

Based on these execution times it is now possible to calculate average of the overall performance of the approach (see Table IV). It is significant that the process of generating and loading the model in memory is the most consuming task of the approach. The influence of generating profiling

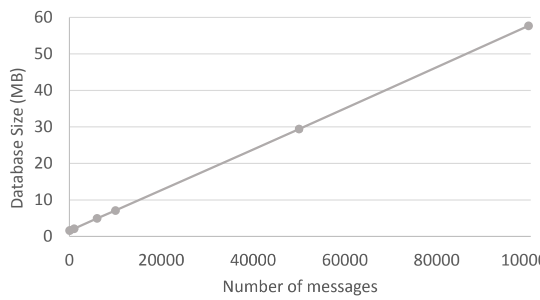


Fig. 5. Database size in relation to number of messages in the SD (in MB)

TABLE II
DATABASE SIZES AND EXECUTION TIMES FOR GENERATING INTERACTION MODELS

	Database	*Generation without Visualization (s)*				*Generation with Visualization (s)*			
#Messages	(MB)	Run 1	Run 2	Run 3	Average	Run 1	Run 2	Run 3	Average
10	1.61	0.43	0.43	0.45	0.44	0.47	0.44	0.4392426	0.45
100	1.66	3.42	3.44	3.45	3.44	3.44	3.45	3.4309164	3.44
1,000	2.11	34.19	34.33	34.36	34.29	34.75	34.37	34.8427815	34.65
6,000	4.94	245.15	246.03	246.42	245.87	245.31	248.43	252.24	248.66
10,000	7.11	454.44	461.04	459.45	458.31	442.99	463.44	465.27	457.23
50,000	29.40	6502.54	6552.92	6555.93	6537.13	6502.70	6554.64	6560.93	6539.42
100,000	57.70	24735.66	23667.25	24202.46	24201.79	23665.25	24734.25	24202.70	24200.73

TABLE III
EXECUTION TIMES FOR TRACE LINKS AND GENERATING RUNTIME PROFILE

	Trace Links (s)				*Runtime Profile (s)*			
#Messages	Run 1	Run 2	Run 3	Average	Run 1	Run 2	Run 3	Average
10	0.016	0.016	0.016	0.016	0.024	0.025	0.025	0.025
100	0.016	0.031	0.016	0.021	0.028	0.030	0.028	0.029
1,000	0.078	0.078	0.078	0.078	0.030	0.037	0.047	0.038
6,000	0.578	0.578	0.596	0.584	0.097	0.108	0.096	0.100
10,000	1.130	1.138	1.119	1.129	0.136	0.146	0.139	0.140
50,000	13.510	15.517	11.507	13.512	0.773	2.025	0.528	1.109
100,000	47.010	56.017	38.007	47.012	2.023	6.025	2.028	3.359

TABLE IV
AVERAGE EXECUTION TIME OF THE OVERALL PERFORMANCE OF THE APPROACH

#Messages	*Interaction Model SD (s)*	*Profiling Information (s)*
10	0.449	0.040
100	3.441	0.050
1,000	34.655	0.116
6,000	248.661	0.684
10,000	457.234	1.270
50,000	6539.424	14.620
100,000	24200.734	50.370

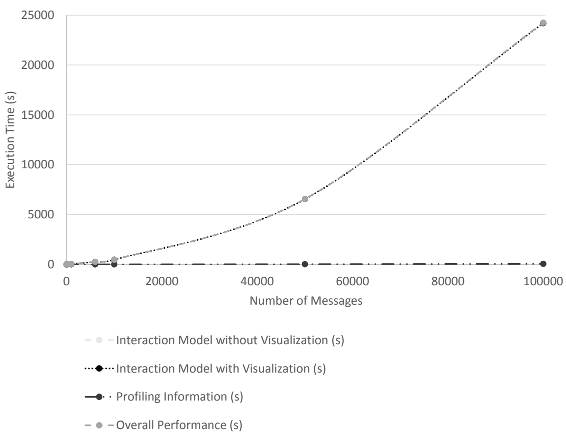


Fig. 6. Overall average execution time of the different tasks of the approach

information is low in our evaluation settings like shown in Fig. 6.

a) Interpretation of results: Answering RQ1: Our investigations of the execution time for generation interaction models from system logs already show for this use case the influence of number of messages in the system log. At a certain size (around 50,000 messages), the execution time no longer increases linearly. This means for larger system logs, a different storage technique may have to be found. The main reason for that seems to be that the used API writes each record individually in the database whereas bulk operations may be more applicable here. However, further investigations to substantiate this statement are needed in the future.

Answering RQ2: The effort to generate the graphical visualization from interaction models is very low in the generation process. The main effort consists in creating the specific elements in the database. The SD could be created for all investigated test cases.

Answering RQ3: Once the model has been loaded into the memory, the trace links are set up quickly. However, from 50,000 messages onwards, the analysis and assignments slows down and the execution time increases according to a quadratic function. However, the trace links could still be established for all settings.

Answering RQ4: The calculation of the profile information takes as long as the establishment of the trace links regarding a number of messages up to 6,000. With a larger number, the distribution also increases in a polynomial way, but less strongly.

Answering RQ5: The overall execution shows that the generation of the models takes the most time. Once the models are loaded into memory, optimized queries can be performed. However, the influence of the number of messages should not be underestimated as the evaluation of the execution times shows.

Our evaluation shows that scaling of modeling tools in MDE is still a crucial issue as described in [5].

D. Threats to Validity

Internal Validity - Are there factors that can influence the results of the case study? In our evaluation we only vary the number of messages, the scaling of the lifelines is not examined. Also, our evaluation is limited to a maximum of 100,000 messages. For execution with a larger number of messages, further studies are needed.

External Validity - Is it possible to generalize the results? Our investigations are limited to the type of use case described in the setup of the case study. In order to generalize the approach, the evaluation should be carried out with other use cases by persons without knowledge of the internal realization of the presented approach.

V. Related Work

In this section, we present and discuss related research works.

Process mining (PM) serves as a bridge between design time and runtime, by combining data mining and business process modeling to a new reserach field [6]. PM enables to extract process-related information from so-called event logs [6], [7]. PM defines events as process steps and event logs as sequential ordered events recorded by any information system during operation [8]. Such actual workflow information is then used to discover process models for enabling a target/actual comparison, e.g., to identify bottlenecks [6]. PM is also applied in software engineering. Ladiges et al. [9] show an approach where they demonstrate a learning algorithm for automatic generation of material flow Petri nets based on recorded PLC I/O data. Their approach allows a documentation of the processes, where anomalies in the material flow are detected and process properties can be determined. In [10], the authors present a novel reverse engineering technique to obtain real-life event logs from distributed software systems. Thereby, PM techniques are applied for obtaining precise and formal models, and to monitor and improve processes by performance analysis as well as conformance checking.

Similar to PM, our approach aims for visualizing runtime information as interaction model in a sequence diagram. However, we go a step further and use the provided information to automatically identify sequence patterns in the model to get a deeper insight about interaction frequencies, occurrences of properties and to back propagate this information in the system design model.

The research field of *Reverse Engineering (RE)* deals with reversing information from runtime to design time. This means, RE takes running code and elevates it on a higher level of abstraction (e.g., model level) for analyzing it. Raibulet et al. [11] conduct a systematic literature review on RE in the application field of MDE. Thereby, the authors analyze different used model languages, transformations, as well as the degree of automation the used tools. The authors come up that most of the investigated approaches deal with source code such as Java or C or web application code to get a representation on object-oriented model level. In contrast, we analyze runtime logs of a system in order to annotate an existing design model.

In [12], the authors analyze the requirements and conditions that allow partial automation of model-based reverse engineering. In their experiments, the authors use the power of combining metaprogramming and metamodeling techniques. Our approach is different in so far as we are back propagating metainformation from runtime information into design models instead of only creating a runtime model. In [13], the authors analyze the difficulty of automatically finding errors in object-oriented code. For this purpose they use unit tests to get traces of execution paths. From this they create then sequence diagrams. In order to minimize the size of the diagrams, the authors aggregate between error-free and error-prone runs in order to find the incorrect code line. Their approach serves to find faults in a runtime code by trying to extract information from runtime for improving the design. This means that the authors perform a comparison and then display faulty parts. In our approach, we generate additional information from runtime to improve the design by annotating this metainformation to the design model.

Briand et al. [14] propose the generation of sequence diagrams for specific use cases by performing dynamic analysis to help people understanding system behavior. The authors formally describe the RE-process by using metamodels and transformation rules in OCL (Object Constraint Language). In contrast to our approach, the authors are focusing on executed source code instead of observed system logs. In [15], the authors use static information from source code and dynamic information from executing the code. Both information are then represented as models. Based on these models they develop a model transformation to generate a sequence diagram out of the code. In our approach, we automatically generated runtime models from executed system logs and automatically generate the corresponding sequence diagram from those logs, e.g., for validation purposes.

Younis and Frey [16] show an approach where the generate an UML Class Diagram and State Diagram out of PLC code. Their approach deals with the reverse engineering of detailed design information of the PLC code that was not previously available. Davydova and Shershakov [17] describe an approach to generate sequence diagrams from execution traces of SOA information systems. In contrast to the other approaches and similar to ours, they do not use executed source code for generating sequence diagrams, but using event logs like in PM. However, they do not further examine automatically generated sequence diagrams in order to annotate design models.

As discussed, there are similar approaches but they are focusing either on executed source code or on creating an interaction model from it. Our approach aims for analyzing logs from an operating system (e.g., controller of a machine), generating sequence diagrams and annotating properties in form of metainformation to existing design models.

VI. Conclusion and Future Work

In this paper we presented an approach for automatically reverse engineering of interaction models from system logs and back propagation of runtime information into design models. Our case study about a self-driving car shows that the approach is feasible for annotating and tracing from runtime to design time. We are able to generate interaction models from system logs and then based on our developed queries, the relationships between runtime models and design models are established. Our evaluation shows that scaling of modeling tools is still a crucial issue. The strength of our approach is that we can keep the relevant information in a unified modeling language, namely UML. Thus, design tools can be reused with their integrated tooling and there is no need to learn new technologies or languages for analyzing runtime information. At the same time, the approach is depending on existing UML technologies, and thus, it may be challenging to use the approach in Big Data settings without any pre-processing. For such settings, the logs have to be decomposed before analysis.

Regarding our presented model-driven approach, we foresee the following next steps. First of all, we plan to vary the number of used lifelines by applying and validating our approach in another case studies such as distributed production systems, where multiple systems with different object-oriented system logs exist. We want to check the generalization of our approach. For instance, if IDs are for example hash values, then methods such as heuristic matching will have to be applied additionally. Second, we strive to adapt our approach to use the timing aspect not only to get the right order in the SD, but also to explicitly store how long messages take time. This allows, for example, to annotate information about minimum, maximum, average durations as tagged values of the stereotype for operations in the design model. Third, we plan to extend our approach to use the generated SD as input for the simulation of the SM in order to experiment even better with runtime data. For instance, in this simulation break points could be set to analyze certain processes more precisely or to manually trigger events and observe system reactions. Finally, we want to adapt the presented approach on the technology level to reach our main goal of linear scaling.

Acknowledgment

This work has been supported by the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development (CDG). We also thank Maximilian Medetz and Horst Kargl for their work and fruitful support.

References

[1] M. Brambilla, J. Cabot, and M. Wimmer, *Model-Driven Software Engineering in Practice, Second Edition*, ser. Synthesis Lectures on Software Engineering. Morgan & Claypool Publishers, 2017.

[2] J. Bézivin, "On the unification power of models," *Software and System Modeling*, vol. 4, no. 2, pp. 171–188, 2005.

[3] A. Mazak, M. Wimmer, and P. Patsuk-Bösch, "Execution-Based Model Profiling," in *Proceedings of the 6th IFIP WG 2.6 International Symposium Data-Driven Process Discovery and Analysis (SIMPDA), Revised Selected Papers*, ser. Lecture Notes in Business Information Processing, vol. 307. Springer, 2016, pp. 37–52.

[4] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, no. 2, pp. 131–164, 2009.

[5] D. S. Kolovos, L. M. Rose, N. D. Matragkas, R. F. Paige, E. Guerra, J. S. Cuadrado, J. de Lara, I. Ráth, D. Varró, M. Tisi, and J. Cabot, "A research roadmap towards achieving scalability in model driven engineering," in *Proceedings of the Workshop on Scalability in Model Driven Engineering*. ACM, 2013, p. 2.

[6] W. M. P. van der Aalst, *Basics of Applied Stochastic Processes*, ser. Process Mining: Discovery, Conformance and Enhancement of Business Processes. Springer, 2011.

[7] ——, "Process mining: making knowledge discovery process centric," *SIGKDD Explorations*, vol. 13, no. 2, pp. 45–49, 2011.

[8] M. Dumas, W. M. van der Aalst, and A. H. ter Hofstede, *Process-aware Information Systems: Bridging People and Software Through Process Technology*. New York, NY, USA: John Wiley & Sons, Inc., 2005.

[9] J. Ladiges, A. Fulber, E. Arroyo, A. Fay, C. Haubeck, and W. Lamersdorf, "Learning material flow models for manufacturing plants from data traces," in *Proceedings of 13th IEEE International Conference on Industrial Informatics (INDIN)*. IEEE, 2015, pp. 294–301.

[10] M. Leemans and W. M. P. van der Aalst, "Process mining in software systems: Discovering real-life business transactions and process models from distributed systems," in *Proceedings of the 18th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems (MoDELS)*. IEEE Computer Society, 2015, pp. 44–53.

[11] C. Raibulet, F. A. Fontana, and M. Zanoni, "Model-Driven Reverse Engineering Approaches: A Systematic Literature Review," *IEEE Access*, vol. 5, pp. 14 516–14 542, 2017.

[12] F. Jouault, J. Bézivin, R. Chevrel, and J. Gray, "Experiments in Run-Time Model Extraction," in *Proceedings of the 1st International Workshop on Models@runtime*, 2006.

[13] O. Pilskalns, S. Wallace, and F. Ilas, "Runtime Debugging Using Reverse-Engineered UML," in *Proceedings of the 10th International Conference on Model Driven Engineering Languages and Systems (MoDELS)*, ser. Lecture Notes in Computer Science, vol. 4735. Springer, 2007, pp. 605–619.

[14] L. C. Briand, Y. Labiche, and J. Leduc, "Toward the Reverse Engineering of UML Sequence Diagrams for Distributed Java Software," *IEEE Trans. Software Eng.*, vol. 32, no. 9, pp. 642–663, 2006.

[15] Y. Labiche, B. Kolbah, and H. Mehrfard, "Combining Static and Dynamic Analyses to Reverse-Engineer Scenario Diagrams," in *Proceedings of the IEEE International Conference on Software Maintenance*. IEEE Computer Society, 2013, pp. 130–139.

[16] M. B. Younis and G. Frey, "UML-based Approach for the Re-Engineering of PLC Programs," in *Proceedings of 32nd Annual Conference on IEEE Industrial Electronics (IECON)*, 2006, pp. 3691–3696.

[17] K. Davydova and S. Shershakov, "Mining Hierarchical UML Sequence Diagrams from Event Logs of SOA systems while Balancing between Abstracted and Detailed Models," in *Proceedings of the Spring/Summer Young Researchers' Colloquium on Software Engineering (SYRCoSE)*, 2016, pp. 85–102.

14 Model-driven Runtime State Identification

S. Wolny, A. Mazak, M. Wimmer and C. Huemer;
40 Years SIG EMISA: Digital Ecosystems of the Future: Methods, Techniques and Applications, 39(1), De Gruyter, (2019), pp. 29–44.
Corpus ID: 220055432

Heinrich C. Mayr, Stefanie Rinderle-MA, Stefan Strecker (Hrsg.): 40 Years EMISA
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 29

Model-driven Runtime State Identification

Sabine Wolny[1], Alexandra Mazak[2], Manuel Wimmer[3], Christian Huemer[4]

Abstract: With new advances such as Cyber-Physical Systems (CPS) and Internet of Things (IoT), more and more discrete software systems interact with continuous physical systems. State machines are a classical approach to specify the intended behavior of discrete systems during development. However, the actual realized behavior may deviate from those specified models due to environmental impacts, or measurement inaccuracies. Accordingly, data gathered at runtime should be validated against the specified model. A first step in this direction is to identify the individual system states of each execution of a system at runtime. This is a particular challenge for continuous systems where system states may be only identified by listening to sensor value streams. A further challenge is to raise these raw value streams on a model level for checking purposes. To tackle these challenges, we introduce a model-driven runtime state identification approach. In particular, we automatically derive corresponding time-series database queries from state machines in order to identify system runtime states based on the sensor value streams of running systems. We demonstrate our approach for a subset of SysML and evaluate it based on a case study of a simulated environment of a five-axes grip-arm robot within a working station.

Keywords: Model-driven Engineering; Time-Series Database; State Identification; Runtime Queries; Process Mining

1 Introduction

Forecasts show that in the upcoming years most of the devices we interact with will be linked to a global computing infrastructure [BS14]. This tendency represents an infrastructure in which the physical environment is populated by interconnected and communicating objects (e.g., sensors, actuators and other smart devices) capable for autonomously interacting with each other and with the environment itself. In order to deal with the increasing complexity of cyber-physical systems (CPS), models are used in many research fields as abstract descriptions of reality. This means that a model serves as an abstraction for a specific purpose, as a kind of "blueprint" of a system, describing the system's structure as well as desired behavior. However, often we recognize a discrepancy between these models and their real world correspondents [MW16b]. In other words, we experience deviations between design-time models and runtime models discovered from real data.

[1] JKU Linz, CDL-MINT, Linz, Austria sabine.wolny@jku.at
[2] JKU Linz, CDL-MINT, Linz, Austria alexandra.mazak-huemer@jku.at
[3] JKU Linz, CDL-MINT, Linz, Austria manuel.wimmer@jku.at
[4] TU Wien, BIG, Vienna, Austria huemer@big.tuwien.ac.at

This development raises new challenges for *Model-Driven Engineering (MDE)* approaches [MWP18]. While design models help in the engineering process by providing appropriate abstractions, data-driven approaches such as process mining [Aa16] may help to uncover some under-specified or unintended parts of these design models at runtime. For instance, on a high level of abstraction, behavioral modeling languages (e.g., state-machine-based languages) are used to describe the behavior of a physical asset by means of states and transitions. Such models define discrete states, which are represented by defined variable values. A system has to achieve/go through these states during its execution. However in reality, systems do not switch in a time discrete manner between states, but the values of the variables are continuously evolving to the intended values of the next state. This means, each variable undergoes a continuous series of changes that need to be continuously monitored, e.g., to be able to react immediately to a time delay in safety critical systems. The challenge is to continuously listening to value streams in order to determine whether a state has indeed occurred, i.e., if the specific combinations of variable values have occurred over all streams at the same time. In particular, the realization precision of systems as well as measuring inaccuracies complicate this process as false positives and false negatives may occur when matching state templates to data streams.

Based on first ideas presented in [Wo17], we address this challenge by introducing a novel approach where we automatically generate state realization event queries derived from state machines for an appropriate state identification at runtime. This approach enables us to continuously observe multiple data streams of distributed sensor devices for identifying a system's entire state at runtime. By applying the so-called *Model-driven Runtime State IdEntification (MD-RISE)* approach, we automatically transform behavioral models, i.e., state machines, into time-series queries to be able to match sensor value streams with pre-defined variable values of the design model to report identified states from execution. First evaluation results derived from a case study of a 5-axes grip-arm robot show the potential of the approach in terms of precision and recall of finding system states in sensor value streams. By this, state based monitoring is possible for instance, even if the systems are not able to provide a explicit state-based trace.

The remainder of this paper is structured as follows. In the next section, we present a motivating example for our approach. Section 3 presents the MD-RISE approach by describing the MD-RISE architecture and its prototypical implementation. Section 4 demonstrates the evaluation of MD-RISE based on a case study of a 5-axes grip-arm robot which is interacting with other components within a working station, like a pick-and-place unit. In Section 5, we discuss related work. Finally, we conclude this paper by an outlook on our next steps in Section 6.

2 Motivating Example

As motivating example for this paper, we consider a simple continuous automated system around a 3-axes grip-arm robot (gripper). This gripper is modeled by using by the *Systems*

Modeling Language (SysML) [FMS12], in particular by using the *block definition diagram (BDD)* and *the state machine diagram (SM)*. The BDD is used to define the structure of the gripper with its properties: `BasePosition (BP)`, `MainArmPosition (MAP)`, and `GripperPosition (GP)` (see Fig. 1(a) Design Models, `BDD System`). These properties describe the angle positions of the three axes of the gripper. Based on the machine operator's knowledge, these angle positions can be defined for different settings (e.g, drive down, pick-up) with pre-defined tolerance ranges. These ranges fix the accepted margin of deviation (e.g., ± 0.1) for the variable values of each property (`BP, MAP, GP`). The desired behavior of the gripper is described by various states and state transitions modeled by using the SM (see Fig. 1(a) Design Models, `SM Grip-arm robot`). These states are `DriveDown` and `PickUp` with assigned variable values specifying the respective angle position in these states. During operation (i.e., execution at runtime), the gripper as a continuous system moves in its environment (e.g., pick-and-place unit) on the basis of a workflow described by the SM. These movements are recorded by various axis sensors and returned as continuous sensor value streams on a log recording system. In our motivating example, we record three sensor value streams `BP, MAP, GP` (see Fig. 1(b) Runtime Data). These records show that the gripper does not "jump" from one discrete state into another as modeled in the SM, but is–of course–continuously moving. Thus, the challenge is to identify possible discrete states by analyzing the sensor value streams. For this purpose we have to raise raw sensor value streams on a higher level of abstraction. This enables, e.g., to better compare an initial model (e.g., SM) with its realization.

The state identification is done by matching the different raw sensor value streams to the pre-defined variable values defined in the SM (see Fig. 1(b) Runtime Data). It should be

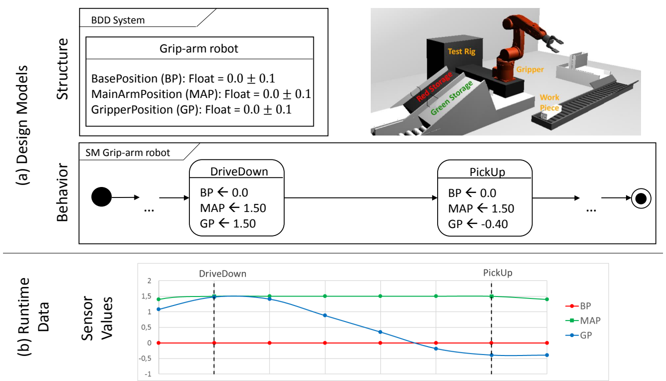


Fig. 1: Design time and runtime perspective of the motivating example

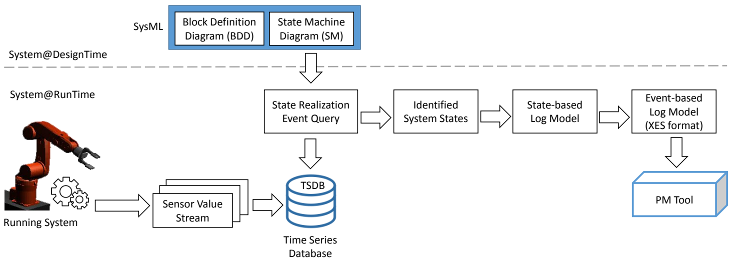


Fig. 2: Architecture for model-driven runtime state identification

considered that the pre-defined absolute variable values in the SM are not necessarily precisely measured in the real world because of, e.g., measuring inaccuracies. Such inaccuracies has to be taken into account by dealing with numerical values of objects of the physical world [MWV16]. Thus, in order to perform the state identification successfully, it is important to define appropriate tolerance ranges (see Section 4). For instance, the sequence of identified states can be used as input for further analysis (see Section 3).

3 Model-driven Runtime State Identification

In this section, we present our *Model-driven Runtime State IdEntification (MD-RISE)* approach which combines MDE-techniques with a *Time-Series Database (TSDB)* and *Process Mining (PM)*, for states identification, recording, abstraction, and analyses. Fig. 2 shows the architecture of MD-RISE as well as the interplay of design time and runtime artefacts.

3.1 MD-RISE Prerequisites

For prototypically realizing the approach, we have a number of prerequisites that must be met: (*i*) the system's workflow must be expressible by means of a state machine, (*ii*) the different states of the system must be unique in order that values describing a state are not identical for two different states, (*iii*) numeric values must be returned by sensors at runtime and must be storable in a TSDB, and (*iv*) it must be ensured that the time stamps are accessible.

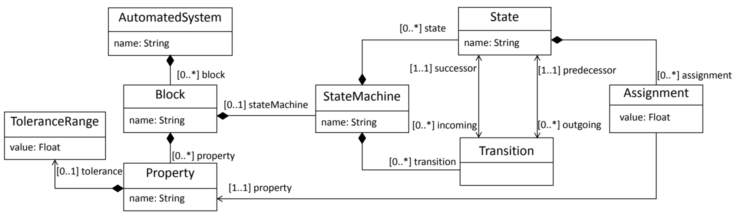


Fig. 3: Metamodel for describing a simple automated system

3.2 MD-RISE Architecture

Based on the motivating example of the gripper (see Section 2) and the mentioned prerequisites, we consider an automated system consisting of a controller, sensors, and actuators. At design time, we model the structure and behavior of this system by using a subset of SysML (see Figure 2: System@DesignTime, `BDD` and `SM`). Fig. 3 shows the simplified graphical metamodel used for modeling BDD and SM of the system. Every component of the system (`Block`) contains properties (`Property`) and can have a SM (`StateMachine`), which describes the behavior of this component. Each Property can have a specified tolerance range (`ToleranceRange`) that defines an acceptable deviation of the assigned property values, e.g., based on measurement inaccurancies. The SM consists of states (`State`) and transitions (`Transition`). Generally, a state can have multiple incoming and outgoing transitions. A transition must have a predecessor and successor state (see Fig. 3). Additionally, different values can be assigned to a state (`Assignment`). In this paper, we just focus on Float property values, since we are interested in value changes during execution (see Section 4).

Based on this metamodel, we automatically derive a query on the basis of the SM, a so-called "state realization event query" (see Fig. 2, System@RunTime). This query helps for identifying states based on the recorded sensor value streams in a TSDB. For this purpose we use a *Model-to-Text (M2T) transformation* to automatically transform model elements to query statements (i.e., text strings) (see Subsection 3.3). During runtime, the sensors of the running system continuously send data over a messaging system middleware. These sensor value streams (e.g., values of the angle positions of the gripper) are recorded in a TSDB (see Figure 2). A single log of the stream contains the following information: *timestamp* (the actual time in the granularity of seconds), *sensor* (the name of the specific sensor), *value* (the measured value). The number of log entries for one component varies depending on the number of sensors. The challenge is to continuously listening to value streams in order to determine whether a state has indeed occurred, i.e., if the specific combinations of variable values have occurred over all streams at the same time (see Fig. 1(b) Runtime Data). For this purpose we apply the aforementioned state realization event query for identifying states containing the following information: *timestamp* (the actual time in the granularity of seconds), *state* (the recognized state based on measured values).

However, the absolute values assigned in the SM at design time (see Fig. 1(a) Design Models) are necessarily not precisely identified as such during runtime due to measurement inaccuracies. For instance, we define for a certain state (e.g, `DriveDown`) a value of 1.50 for a certain angle position (e.g., `MAP`) at design time, but at runtime we measure a value for this position of 1.492. For this purpose we implement a tolerance range, assigned to the initial model (e.g., the SM), to define in which range such inaccuracies are still acceptable (see Figure 3, `ToleranceRange`). The definition of such a range is crucial. If the range is selected too small, the inaccuracies may result in too few or even no identified states. Otherwise, if the range is too large, too many states are identified. We examine this challenge in our case study presented in Section 4.

In a next step, we generate a state-based log model that consists of the information of all identified states and, in addition, a case ID for identifying the corresponding process instance (see Fig. 2: System@RunTime, `State-based Log Model`). Such a case ID is required when using PM tools in order to be able to distinguish different executions of the same process. We employ this case ID in our approach to identify single runs of the SM during runtime. In a further step, the state-based log model is transformed to an event-based log model (see Fig. 2, `Event-based Log Model`) by applying a *Model-to-Model (M2M) transformation*, like presented in previous research work [MW16a]. Since, we use a PM tool for analyzing this model, the structure must be based on *eXtensible Event Stream (XES)* schema. This is a supported input format of ProM Lite 1.1. For instance, by using this PM tool, the event-based log model can be analyzed, e.g., to uncover some under-specified or unintended events that were not considered in the SM.

In summary, by applying MD-RISE it is now possible to raise raw sensor value streams on a higher level of abstraction, namely the state level. MD-RISE bases on queries, so-called state realization event queries, which are automatically derived from an initial design model for the purpose of state identification at runtime. The identified system states can be automatically transformed into a state-based log model to make the outcome useable, e.g., for PM tools like ProMLite for further analyses.

3.3 MD-RISE Prototypical Realization

For a first prototypical realization, we use the defined metamodel (see Fig. 3) and implement it by using Ecore in the Eclipse Modeling Framework . Based on this metamodel, we develop a M2T transformation by using Xtend in order to automatically generate state realization event queries out of the SM for different states. The structure of this M2T

http://www.promtools.org/doku.php?id=promlite11
https://www.eclipse.org/modeling/emf
http://www.eclipse.org/xtend

transformation depends on the used TSDB. In our implementation, we use InfluxDB as TSDB. Therefore, the structure of our state realization event queries are similar to a SQL syntax, as shown in the following pseudo code example based on our metamodel:

```
«FOR s IN Block.stateMachine.state»
SELECT «FOR a IN s.assignment» «a.property.name», «ENDFOR» time
FROM «Block.name»
WHERE «FOR a IN s.assignment»
«a.property.name»>=«a.value-a.property.tolerance.value»
and «a.property.name»<=«a.value-a.property.tolerance.value»«ENDFOR»
«ENDFOR»
```

Based on the raw sensor value streams collected at runtime and stored in the TSDB, the queries are executed and the results are the identified states with their timestamps. In our prototypical implementation, we store the outcome as csv-file, which is then used as input for the state-based log model. This model is a Ecore model representation of the csv-file. In a next step, we use the *Atlas Transformation Language (ATL)* as transformation tool to transform the state-based log model to an event-based log model for importing it into ProM [MW16a]. The full implementation of MD-RISE can be found at our project website .

4 Case Study based on a CPPS-Simulation Environment

In this section, we present as well as discuss the accuracy and limitations of MDE-RISE on the basis of a case study of a CPPS-simulation environment around a 5-axes grip-arm. In doing so, we follow the guidelines for conducting empirical explanatory case studies by Roneson and Hörst [RH09]. In particular, we report on applying our approach to detect states at runtime based on stored value streams in a TSDB.

4.1 Research Questions

The study was performed to quantitatively assess the completeness, correctness, and performance of MDE-RISE. More specifically, we aimed to answer the following research questions (RQs):

https://www.influxdata.com
https://www.eclipse.org/atl
http://promtools.org/doku.php
https://cdl-mint.big.tuwien.ac.at/case-study-artefacts-for-emisa-2019/

RQ1—Correctness: Are the identified states at runtime correct in the sense that all identified states are representing real states? If our approach identifies incorrect states, what is the reason for this?

RQ2—Completeness: Are the identified states complete in the sense that all expected states are correctly identified? If the set of identified states is incomplete, what is the reason for missed identifications?

RQ3—Performance: How strongly is the performance of the query execution influenced by the number of sensor value streams or the number of stored values per sensor?

4.2 Case Study Design

Requirements. As an appropriate input for our case study, we require an automated system such as a gripper integrated in a simulated environment where we are able to observe the behavior of the gripper during operation. We require access to multiple sensors of the gripper for log acquisition and a method to automatically identify states based on sensor value streams from simulation runs.

Setup. To fulfill these requirements, we implemented a CPPS-simulation of an autonomous acting production unit executed by using the open source tool Blender . The simulation scenario considers a working station, like a pick-and-place unit, where a gripper takes work pieces from a conveyor belt, put them down on a test rig, and finally release them in a red or green storage box based on the information coded on each work piece by a QR-code. Each component communicates via a messaging system middleware with InfluxDB. This TSDB provides us to acquire raw sensor value streams. During simulation, the gripper enters several different states for processing the work pieces. To verify the correctness of our approach, we have chosen two very similar states (differ only in one sensor value stream) to determine if the detection works: `DriveDown` and `PickUp`. The assigned values of the axes `Base Position (BP), Main Arm Position (MAP), Second Arm Position (SAP), Wrist Position (WP), and Gripper Position (GP)` of the two states in the SM are shown in Tab. 1. Furthermore we need to define an acceptable tolerance range to determine when the state identification is as accurate and complete as possible. We use a tolerance range from a deviation of 0 to a deviation of 0.4 (in 0.01 steps). The upper bound is only set for evaluation purposes to show the distribution of precision and recall. In reality, a deviation of 0.4 may be already too large. The deviation values are added or subtracted to the respective SM values (see Tab. 1). We use the same tolerance ranges for all properties and do not vary them.

For our evaluation we use two different database settings in combination with different numbers of sensor value streams that are used for the states identification. We use a dataset

`https://www.blender.org`

Tab. 1: Expected values for the gripper's axes for the states DriveDown and PickUp.

State / Gripper Axis	DriveDown	PickUp
Base Position (BP)	0.0	0.0
Main Arm Position (MAP)	1.50	1.50
Second Arm Position (SAP)	-0.12	-0.12
Wrist Position (WP)	0.0	0.0
Gripper Position (GP)	1.5	-0.40

with 156 rows and a dataset with 1,560 rows stored in the database. For the state identification we use a single sensor value stream (GP), three sensor value streams (GP, BP, MAP), and all five sensor value streams (GP, BP, MAP, SAP, WP). For the performance check we also extend our dataset up to 15,600, 156,000, and 1,560,000 rows.

For performance purpose of the state realization queries, we calculate the duration between start of the query execution and result return by System.nanoTime() in Java. The performance figures have been measured on an Acer Aspire VN7-791 with an Intel(R) Core(TM) i7-4720 HQ CPU @ 2.60 GHz 2.60 GHz, with 16 GB of physical memory, and running the Windows 8.1. 64 bits operating system. Please note that we measured the CPU time by executing each query 40 times for all different settings and calculated the arithmetic mean of these runs in milliseconds (ms).

Measures. In order to assess the accuracy of our approach, we calculate *precision* and *recall* as defined in [MRS08]. In the context of our case study, precision denotes the fraction of *correctly identified* states among the set of *all identified* states. Recall indicates the fraction of *correctly identified* states among the set of *all actually occurring* states. Precision denotes the probability that a identified state is correct and the recall is the probability that an actually occurring state is identified. Both values range from 0 to 1.

Precision is used to answer RQ1 and recall to answer RQ2. Furthermore, we calculate the so-called *f-measure* to avoid having only isolated views on precision and recall [MRS08]. To answer RQ3, we compute the duration of the query execution.

To check if our approach is accurate for a given scenario to identify system states, we have manually obtained the gold standard of state identifications for our given case study (156 rows: 3 expected states for DriveDown and PickUp, 1560 rows: 30 exepected states for DriveDown and PickUp). For computing precision and recall, we extract the true-positive values (TPs), false-positive values (FPs) and false-negative values (FNs), with the help of the expected state identifications. From the TP, FP and FN values we then compute *precision*, *recall* and *f-measure* metrics as defined by Olson and Delen [OD08, p. 138].

Tab. 2: Precision, recall and f-measure for a single sensor value stream (GP). Bold line marks the best fit.

	DriveDown			PickUp		
tolerance range	**precision**	**recall**	**f-measure**	**precision**	**recall**	**f-measure**
0	NaN	0	NaN	NaN	0	NaN
0.01	NaN	0	NaN	0.08	1	0.14
0.02	**1**	**1**	**1**	**0.08**	**1**	**0.14**
0.03-0.05	1	1	1	0.07	1	0.14
0.06-0.08	1	1	1	0.07	1	0.13
0.09-0.11	0.75	1	0.86	0.07	1	0.13
0.12-0.19	0.75	1	0.86	0.07	1	0.12
0.20-0.30	0.6	1	0.75	0.05	1	0.10
0.31-0.37	0.5	1	0.67	0.05	1	0.10
0.38-0.39	0.5	1	0.67	0.05	1	0.09

4.3 Results

We now present the results of applying our approach to the different settings of our gripper simulation. Tab. 2–Tab. 4 show the results for precision, recall and f-measure for the two different states in the different value stream settings. The values are valid for both database settings (156 rows, 1560 rows), since there were no differences with regard to precision, recall and f-measure. This can be explained by the fact that the queries are independent of the number of values in the database. As soon as the sensor value streams are in the accepted tolerance range, the state is returned.

Tab. 3: Precision, recall and f-measure for three sensor value streams (GP, BP, MAP). Bold line marks the best fit.

	DriveDown			PickUp		
tolerance range	**precision**	**recall**	**f-measure**	**precision**	**recall**	**f-measure**
0	NaN	0	NaN	NaN	0	NaN
0.01	NaN	0	NaN	1	1	1
0.02-0.08	**1**	**1**	**1**	**1**	**1**	**1**
0.09-0.10	0.75	1	0.86	1	1	1
0.11-0.12	0.75	1	0.86	0.6	1	0.75
0.13-0.16	0.75	1	0.86	0.5	1	0.67
0.17-0.18	0.75	1	0.86	0.43	1	0.6
0.19	0.75	1	0.86	0.25	1	0.4
0.20-0.21	0.6	1	0.75	0.25	1	0.4
0.22-0.30	0.6	1	0.75	0.23	1	0.375
0.31-0.39	0.5	1	0.67	0.23	1	0.375

It is noticeable that the states identification fails and no states are found if the tolerance range is too small. The larger the range, the more false states are detected and the precision decreases as expected. In Tab. 2 for the state `PickUp` it could be recognized that the precision value is really small (highest value 0.08), because of wrong states identification based on a single sensor value stream. This can be explained by the fact that the gripper moves during the simulation and opens and closes the gripper arm in various locations (e.g., conveyor, test rig). These states do not differ in the value of GP but have a different BP. Thus, this one axis GP is not enough to identify the state `PickUp`. Furthermore, it is interesting that the use of all gripper's axes for state identification `PickUp` leads to a lower recall for the tolerance range 0.01 (see Tab. 4).

Tab. 4: Precision, recall and f-measure for five sensor value streams (GP, BP, MAP, SAP, WP). Bold line marks the best fit.

	DriveDown			PickUp		
tolerance range	**precision**	**recall**	**f-measure**	**precision**	**recall**	**f-measure**
0	NaN	0	NaN	NaN	0	NaN
0.01	NaN	0	NaN	1	0.33	0.5
0.02-0.08	**1**	**1**	**1**	**1**	**1**	**1**
0.09-0.10	0.75	1	0.86	1	1	1
0.11-0.12	0.75	1	0.86	0.6	1	0.75
0.13-0.16	0.75	1	0.86	0.5	1	0.67
0.17-0.18	0.75	1	0.86	0.43	1	0.6
0.19	0.75	1	0.86	0.25	1	0.4
0.20-0.21	0.6	1	0.75	0.25	1	0.4
0.22-0.3	0.6	1	0.75	0.23	1	0.375
0.31-0.39	0.5	1	0.67	0.23	1	0.375

Figure 4 shows the results of our performance check. It could be determined that the number of sensor value streams and the number of rows in the database both have an influence on the execution time.

Interpretation of results. *Answering RQ1*: The recognition of correct states depends on the defined tolerance range and the differentiability of states. A precondition for our

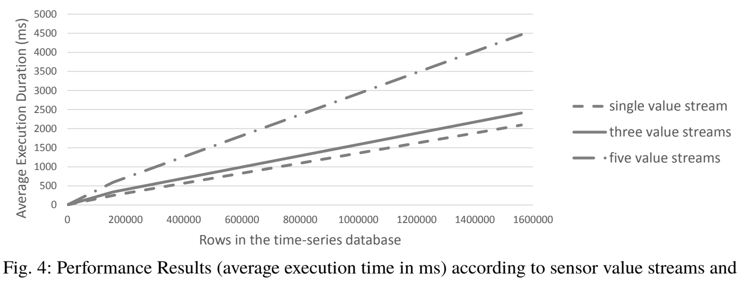


Fig. 4: Performance Results (average execution time in ms) according to sensor value streams and rows in the TSDB.

approach is the uniqueness of states. However, if the various states differ only slightly, the number of sensor value streams used for states identification is relevant for correctness.

Answering RQ2: The selected tolerance range and the number of sensor value streams are also decisive for the completeness of the states identification. The more sensor value streams are used, the more important individual sensor values become for the identification. In addition, the completeness of the identified states is better the larger the selected tolerance range is. In our evaluation we quickly achieve a good completeness. As soon as this is reached, the tolerance range should not be further increased, otherwise the correctness of the identified states suffers.

Answering RQ3: Our investigations of the execution time already show in this simple setting the influence of the number of data records in the database and the used number of different sensor value streams. However, the performance seems still promising for large cases as we experience a linear increase of execution time for all tested settings.

4.4 Threats to Validity

Internal Validity - Are there factors that can influence the results of the case study? At design time, values for our axis positions are assumed on the basis of, e.g., calibration values. At runtime, the same exact values are not always measured, but with a certain fluctuation range. Thus, a certain tolerance range must be defined at design time in which the values are accepted. In our case, we knew exactly which values to expect and were therefore able to keep our tolerance range small. However, this might not work with other settings.

External Validity - Is it possible to generalize the results? Our approach is based on queries automatically created by state machines. We focus on creating queries that are understood by the TSDB InfluxDB. Thus, the queries are currently in SQL syntax. If a different database query language is needed, only the Xtend code has to be adapted regarding syntax without changing the model in the background. At the moment our evaluation is based on a single case of a gripper simulation. For further and more detailed results the study has to be extended to other scenarios. Raw data from sensors are often noisy, incomplete and can contain erroneous records. This is not considered in our case study. In addition, the datasets for performance analysis are relatively small in relation to databases. Larger sets would be needed for further more detailed results.

5 Related Work

Discovering the behavior of running software. In [Li16], the authors utilize process mining (PM) techniques to discover and analyze the real behavior of software. By doing so, they discover behavioral models for each software component by considering hierarchies. In a first step of their approach, they identify component instances and construct event logs for each component from raw software execution data. In a second step, they recursively transform the logs to a hierarchical event log for each component by considering calling relations among method calls. Based on these hierarchical event logs, the authors discover a hierarchical process model to understand how the software is behaving at runtime. The authors' software component behavior discovery builds on the inter-disciplinary research field of *Software Process Mining (SPM)*, firstly introduced by Rubin et al. [Ru07]. Both approaches base their grounding on the well-established techniques and methods of the research field of PM [Aa16].

Applying reverse engineering for obtaining event logs. In [LA15], the authors present a reverse engineering technique based on PM for obtaining real event logs from distributed systems. Similar to [Li16], the authors present an inter-disciplinary approach based on PM techniques and reverse engineering. The aim of their approach is to analyze the operational processes of software systems when running. The formal definition, implementation, and instrumentation strategy of the approach bases on a joinpoint-pointcut model (JPM) known from the area of aspect-oriented programming [EFB01]. This JPM helps (*i*) by defining the parts of a system that are to be included, (*ii*) enables to quickly gain insight into the end-to-end process, and (*iii*) detects the main bottlenecks. The authors demonstrate the feasibility of their approach by two case studies.

Query-based process analytics. A query approach enabling business intelligence through query-based process analytics is presented by Polyvyanyy et al. [Po17]. In contrast to our approach they are focusing on PM techniques for the automated management of model repositories of designed and executed processes, and on the relationships among these processes. For this purpose the authors introduce a framework for specifying generic functionalities that can be configured and specialized to address process querying problems, such as filtering or manipulation of observed processes.

Finally, we would like to highlight two research works that underline our approach and discuss the differences. Mayr et al. [Ma17] critically note that models are mainly used as prescriptive documents. Therefore, the authors aim for a model-centered architecture paradigm to keep models and developed artefacts synchronized in all phases of software development as well as in the running system. In this context, our approach helps to lift raw sensor data through automated states identification during operation at a model level for enabling a comparison between prescriptive and descriptive models. Senderovich et al. [Se16] apply PM techniques for real-time locating systems. They solve the problem of mapping sensor data to event logs based on process knowledge since location data recordings do not relate to the process directly. Therefore, they provide interactions as an intermediate knowledge layer between the sensor data and the event log [Se16]. Contrary to our approach, their raw sensor log consists already of different business entities and they have to map interactions to activity instances, while the sensor logs in our approach consist only of numerical values which we first have to aggregate to events.

6 Conclusion and Future Work

In this paper, we presented an approach that automatically derives state realization event queries from the design model to identify system states of a continuous system based on sensor value streams at runtime. This enables to raise raw sensor data from the data layer on a higher model layer. At this model level, runtime processes can be analysed more quickly and possible unintended parts within the realized system may be identified more easily and time-saving. Since inaccuracies has to be taken into account by dealing with numerical values of objects of the physical world, additionally we implemented a tolerance range for defining in which range such inaccuracies are still acceptable for an identified state at runtime.

First results of our case study indicate that a high precision and recall of system state identification may be achieved if an appropriate tolerance range for the runtime values was defined. Nevertheless, the uniqueness and distinctiveness of the individual states determine whether the state identification works well or not. If states are very similar, enough different sensor value streams must be used for state identification to obtain a good precision and recall. The approach is a step towards a better integration of model-driven software development to all the operations within a system's life cycle in order to continuously deploy stable versions of application systems.

There are several lines for future work we are going to explore in more detail. First, we plan to apply and validate our approach in a real-world setting, instead of a simulation. Second, we want to extend our approach to monitor different components with a larger set of sensor value streams. Third, we only used identically tolerance ranges for the properties. In a further investigation, we want to find out if there are automated techniques possible to estimate good guesses for the tolerance ranges of different properties. Finally, we want to find out if we could extend our approach for state estimation and detection of possible hidden states.

Acknowledgment

This work has been supported by the Austrian Federal Ministry for Digital and Economic Affairs, the National Foundation for Research, Technology and Development, and by the FWF in the Project TETRABox under the grant number P28519-N31.

References

[Aa16] van der Aalst, W. M. P.: Process Mining-Data Science in Action. Springer, 2016.

[BS14] Broy, M.; Schmidt, A.: Challenges in Engineering Cyber-Physical Systems. Computer 47/2, pp. 70–72, 2014.

[EFB01] Elrad, T.; Filman, R. E.; Bader, A.: Aspect-oriented Programming: Introduction. Commun. ACM 44/10, pp. 29–32, Oct. 2001.

[FMS12] Friedenthal, S.; Moore, A.; Steiner, R.: A Practical Guide to SysML. Morgan Kaufmann, 2012, ISBN: 9780123852069.

[LA15] Leemans, M.; van der Aalst, W. M. P.: Process mining in software systems: Discovering real-life business transactions and process models from distributed systems. In: MODELS. Pp. 44–53, 2015.

[Li16] Liu, C.; van Dongen, B.; Assy, N.; van der Aalst, W. M. P.: Component behavior discovery from software execution data. In: SSCI. Pp. 1–8, 2016.

[Ma17] Mayr, H. C.; Michael, J.; Ranasinghe, S.; Shekhovtsov, V. A.; Steinberger, C.: Model Centered Architecture. In: Conceptual Modeling Perspectives. Springer, pp. 85–104, 2017.

[MRS08] Manning, C. D.; Raghavan, P.; Schütze, H.: Introduction to information retrieval. Cambridge University Press, 2008.

[MW16a] Mazak, A.; Wimmer, M.: On Marrying Model-driven Engineering and Process Mining: A Case Study in Execution-based Model Profiling. In: SIMPDA. Pp. 78–88, 2016.

[MW16b] Mazak, A.; Wimmer, M.: Towards Liquid Models: An Evolutionary Modeling Approach. In: CBI. Pp. 104–112, 2016.

[MWP18] Mazak, A.; Wimmer, M.; Patsuk-Bösch, P.: Execution-Based Model Profiling. In: Data-Driven Process Discovery and Analysis. Springer, pp. 37–52, 2018.

[MWV16] Mayerhofer, T.; Wimmer, M.; Vallecillo, A.: Adding uncertainty and units to quantity types in software models. In: SLE. Pp. 118–131, 2016.

[OD08] Olson, D. L.; Delen, D.: Advanced Data Mining Techniques. Springer, 2008, ISBN: 978-3-540-76916-3.

[Po17] Polyvyanyy, A.; Ouyang, C.; Barros, A.; van der Aalst, W. M.: Process querying: Enabling business intelligence through query-based process analytics. Decision Support Systems 100/, pp. 41–56, 2017.

[RH09] Runeson, P.; Höst, M.: Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering 14/2, pp. 131–164, 2009.

[Ru07] Rubin, V.; Günther, C. W.; van der Aalst, W. M. P.; Kindler, E.; van Dongen, B. F.; Schäfer, W.: Process Mining Framework for Software Processes. In: Software Process Dynamics and Agility. Springer, pp. 169–181, 2007.

[Se16] Senderovich, A.; Rogge-Solti, A.; Gal, A.; Mendling, J.; Mandelbaum, A.: The ROAD from Sensor Data to Process Instances via Interaction Mining. In: CAiSE. Pp. 257–273, 2016.

[Wo17] Wolny, S.; Mazak, A.; Konlechner, R.; Wimmer, M.: Towards Continuous Behavior Mining. In: SIMPDA. Pp. 149–150, 2017.

15 From AutomationML to AutomationQL: A By-Example Query Language for CPPS Engineering Models

M. Wimmer and A. Mazak;
Proceedings of the 14th IEEE International Conference on Automation Science and Engineering (CASE), IEEE, (2018), pp. 1394–1399.
DOI: 10.1109/COASE.2018.8560448

2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)
Munich, Germany, August 20-24, 2018

From AutomationML to AutomationQL: A By-Example Query Language for CPPS Engineering Models

Manuel Wimmer[1] and Alexandra Mazak[1]

***Abstract*— Model-based engineering is an emerging paradigm to deal with the complexity of multi-disciplinary engineering in CPPS projects. In such projects, different kinds of models are created during the lifecycle of a production system. AutomationML is a promising standard to provide a unifying format to represent and connect the different engineering models. Dedicated tool support has been developed for AutomationML in the last years to create and evolve models. However, when it comes to querying AutomationML models, implementation-related query languages have to be currently used. These languages have a certain complexity as they are not directly based on the concepts of AutomationML but on the underlying technological concepts and encodings of AutomationML. This often hinders the formulation of automatically executable queries by domain experts.**

In this paper, we propose a dedicated query language for AutomationML called Automation Query Language (AutomationQL) which is directly derived from AutomationML. Using this query language, queries can be defined in a by-example manner which allows engineers to formulate queries in terms of AutomationML concepts instead of switching to an implementation-oriented query language. We illustrate how AutomationQL is defined, how queries can be formulated as well as how tool support is provided to automatically evaluate the queries and represent their results. Finally, we contrast our solution with existing query languages and derive a roadmap for future research on AutomationQL.

I. INTRODUCTION

Industrial production systems engineering is a multi-disciplinary process involving activities of different engineering disciplines starting with the overall system design on a production function level, moving to the detailed mechanical, electrical, and software engineering, and finally leading to the installation, commissioning, and operation of the production system [1]. Faltinsky et. al [2] identify four main aspects of modern engineering processes: (i) tool-supported information exchange, (ii) model reuse and adaptability, (iii) executable models for early verification and validation, and (iv) a system-wide planning of the distributed system. In support of such a seamless development process, the *AutomationML* (AML) standard [3] has been proposed and continuously developed.

AML is designed as a flexible language able to represent a large spectrum of different engineering artifacts as well as for harmonizing engineering data exchange within a heterogeneous tool network [4], [5]. AML is also utilized to build *reusable libraries* containing reusable hardware and software component types, such as sensors, vendor-specific production equipment, and PLC controllers, to build up production systems. Engineering artefacts which are uniformly represented as a collection of AML models may be analyzed with dedicated tool support such as employing query engines to select elements of interest in large and distributed models [6], [7]. We focus in this paper on the core part of AML [2], [3] which is the Computer Aided Engineering Exchange (CAEX) [8], [9], an IEC standard providing a neutral and open data format for the storage of hierarchical object information. Thus, CAEX is used to represent fundamental characteristics of a production system which should allow the exchange of CAEX-compliant artifacts throughout the complete system life cycle.

To further utilize the benefits offered by AML, we show in this paper how AML may be used as a basis for deriving an AML-specific query language called AutomationQL (AQL). In particular, we propose to merge a query language for finding graph patterns in AML models into the AML language. The resulting query language allows to formulate queries in AML structures instead of switching to specific technology-specific encodings of AML to formulate queries with technology-specific query languages. Furthermore, the resulting query language allows to formulate queries in a similar manner as existing by-example query approaches [10], [11]. This means, the query is defined by stating how the results should look like by giving an example instead of defining how they are computed. By following this by-example approach and reusing the AML syntax to formulate the queries as much as possible, domain experts with AML know-how are empowered to model queries in a similar way as they define AML models. For automatically executing the queries on AML models, we provide an interpreter. The query results computed by the interpreter are explicitly represented as proxy models which reference the found graph pattern occurrences to the base AML models. This allows to process query results in any direction, e.g., to generate a documentation of the query results or to provide transformations for the found graph pattern occurrences. AQL is implemented on basis of our Eclipse-based AML workbench which is publicly available [7].

The remainder of this paper is as follows: Section 2 briefly introduces AML and discusses existing query support. Section 3 explains our proposal by describing the definition of AQL and accompanying tool support. For a better understanding of the approach, several examples are presented in Section 4. Section 5 critically discusses our approach, and finally, Section 6 concludes and sketches future work.

[1]Manuel Wimmer and Alexandra Mazak are affiliated with the Christian Doppler Laboratory for Model-Integrated Smart Production (CDL-MINT), TU Wien, Favoritenstrasse 9-11, 1040 Vienna, Austria lastname@big.tuwien.ac.at

978-1-5386-3593-3/18/$31.00 ©2018 IEEE

II. Background

In this section, we shortly introduce AML and discuss CAEX as we mainly focus on this part of AML. Furthermore, we discuss the current querying support for AML models as well as the general ideas behind query by-example approaches.

A. AutomationML

AutomationML (AML) [12] is a neutral XML-based data format for representing engineering knowledge in the area of process automation and control. AutomationML is becoming widely accepted in industry. It is standardized as IEC 62714. For more information about this format we refer the interested reader to the website of the AutomationML Office[1].

AML may be considered as data integration format for the following standardized data representations: CAEX for plant topology information, COLLADA for geometry and kinematic information, and PLCopen XML for logic information. The topology description is captured in the CAEX-related part of AML. The entire plant topology model is represented as an "instance hierarchy" in AML. Devices of the plant are represented as so-called "internal elements" of the aforementioned instance hierarchy or nested in other internal elements. The type of a device is represented as "system unit class", since an internal element is in this case an instance of a system unit class. The interconnections between devices are represented as "internal links". For expressing the meaning of captured information, AML defines "role class libraries", which should be shared among various projects. Interfaces of artifacts are modeled with "interface class libraries". The remaining parts of AML, i.e., PLCopen and COLLADA are not used in this paper and therefore not further introduced.

To utilize the benefits offered by modern model-driven frameworks [13] for AML, we have developed a model-driven engineering workbench for AML [7] based on the Eclipse Modeling Framework (EMF) [14]. In particular, we have formalized the CAEX language in a metamodel using EMF's metamodeling language Ecore, which enables the utilization of EMF's rich ecosystem of model-driven tools for AML. The developed AML workbench is publicly available in our source code repository [7]. Of course, query languages available for EMF such as the Object Constraint Language (OCL) may be directly employed for querying AML models. Although such query languages offer powerful query concepts and mechanisms, as we will discuss later, for domain experts it is challenging to use such general query languages to formulate queries. In this paper, we use our previous work as a basis for implementing the presented AML query approach AQL. In particular, we reuse the AML metamodel to derive the query definition language as well as the query result language.

B. Current Query Support for AutomationML

As AML comes with XML-based standards, one possible way to define queries is to use XML-based query languages such as XQuery, XPath, or XSLT. However, these languages heavily operate on the abstract tree structure behind the AML models and require extensive knowledge in XML processing models such as path expressions, axis navigation, etc.

Another approach is to use the APIs which are currently available in AML engines[2] to query the models by using general purpose programming languages such as C# and Java. However, this approach also requires knowledge on the structure of AML models behind the APIs as well as knowledge about the APIs and the used programming languages. Moreover, the queries have to be mostly formulated in an imperative way instead of using a more declarative query language.

In the AML Analyzer approach [6], AML models are represented as RDF knowledge graphs. This allows to apply query languages for RDF knowledge graphs such as SPARQL. Again, in this query approach knowledge is required about the encoding of AML models in RDF knowledge graphs before the queries can be formulated given also some knowledge on using the SPARQL query language. Similar possibilities and limitations arise when hosting AML models in databases, e.g., SQL or NoSQL based ones. For instance, a first proposal for hosting AML models in NoSQL databases is discussed in [15].

C. Query By-Example

Our proposed AML query approach follows the main principles of the query by-example (QBE) approach introduced in [10], [11]. Initially, the aim of QBE was to have a language for querying and manipulating relational data. This is achieved by so-called skeleton tables, which consist of example rows filled out with constants, constraints, and variables, combined with commands. Commands describe what to do with the selected tuples that match the defined queries, such as deletion or selection of tuples. In order to operate on relational data stored in DBMS, technology-specific queries (e.g., SQL scripts) are derived from the skeleton tables and can be executed on relational models.

The main motivation behind QBE is to provide a more natural interface for end-users to formulate queries. The main assumption is that stating as an example the result of a query is easier to define compared to writing the query in a more computation oriented language.

If we assume to host AML models in relational databases for which QBE is available, we would still require knowledge about how AML models are persisted in relational databases which may come with a huge impedance mismatch. Therefore, we aim in this paper for a dedicated by-example query language which does not require knowledge on how AML models are encoded in specific technologies but directly operates on the AML concepts and structures. The resulting language and dedicated tool support is presented in the following Section 3.

Other by-example approaches related to our proposed query approach are often summarized under the term programming by-example (PBE) [16]. The objective of these

[1] https://www.automationml.org

[2] https://www.automationml.org/o.red.c/tools.html

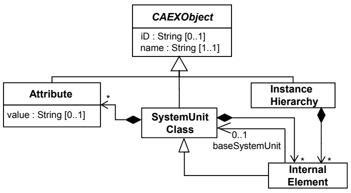


Fig. 1. AML metamodel excerpt.

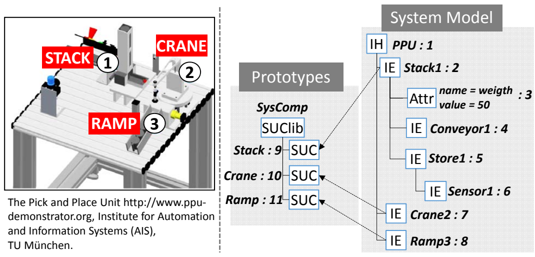


Fig. 2. Example model of an excerpt of the PPU demonstrator [17], [18].

approaches is to facilitate the end user to be able to perform tasks which normally need more knowledge, e.g., knowledge about programming languages. The way PBE tries to to achieve this objective is to record the users actions (e.g., using traces) maybe in more than one iteration, and generate a program from the traces to automatically perform the afore manually performed task by the computer. We consider this line as interesting future work not tackled in this paper. For instance, PBE may be used to derive the query specifications presented in this paper by recording selection actions of the users in the model editor.

III. Approach

In this section, we introduce our approach for extending the AML metamodel with query capabilities in order to model graph pattern queries as AML model fragments. Therefore, we shortly describe an excerpt of the core part of the AML metamodel. Then we introduce our proposal for AQL consisting of two main parts: (i) the AML Query Definition Language (AQDL) and (ii) the AML Query Result Language (AQRL). Finally, we describe the prototypical implementation of AQL.

A. AutomationML Metamodel

For explaining our approach, we use a small fragment of the AML metamodel which covers some core concepts such as the instance hierarchy, internal elements, and system unit classes. Thus, with this excerpt we can model the structure of a system and type the system elements with system unit classes. Fig. 1 depicts this excerpt in UML class diagram notation. Fig. 2 illustrates a small example which is using these concepts on the model level. The figure models just a small part of the Pick and Place Unit (PPU) [17], [18] hosted at the Institute of Automation and Information Systems at the Technical University of Munich.

B. A Graph Pattern Query Metamodel

In order to perform graph pattern queries, some dedicated concepts are needed to define query models which can be evaluated on base models. As we are aiming for a by-example query language, we do not define a query language on its own, but some query concepts which can be combined with AML. In particular, graph pattern queries can be seen as model fragments which mostly comply to the base modeling language. In other words, we embed some query concepts to the base modeling language AML. It may be seen as the inversion of general query languages which constitute new languages and embed the base modeling languages as part of their type systems.

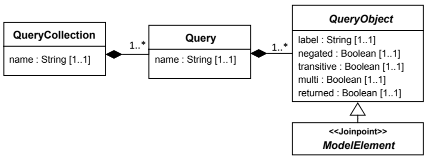


Fig. 3. Graph pattern query metamodel.

Our proposed graph pattern query modeling language is shown in Figure 3. Its central concept is the *QueryObject* which provides attributes needed for defining graph queries such as matching for non-existence or existence (cf. attribute *negated*) or specifying which elements should be returned by the query (cf. attribute *returned*). Please note that this class is abstract, and thus, only provides additional attributes for the modeling concepts of the base modeling languages. Therefore, it is intended to let all base language concepts inherit directly or indirectly from this class. By this mechanism, the modeling elements become query elements as we will now show for AML.

C. Augmenting AutomationML with Query Capabilities

Having the AML metamodel and the graph pattern query metamodel at hand, we are now discussing how the resulting AQL, in particular, the AML Query Definition Language (AQDL) part, looks like. The AML Query Result Language (AQRL) is discussed in the following subsection.

Fig. 4 shows the outcome of merging the AML metamodel with the query metamodel. As we can see in this figure, it is now possible to define queries with AML concepts such as internal elements and system unit classes as they are now query objects. We can even query for CAEXObjects which are the unification of all other AML concepts shown in this metamodel excerpt as we will see later in Section 4.

Please note that in addition to the merging of the two metamodels, we may make abstract classes as concrete ones. By this, we can formulate more general queries. Furthermore, we may have to relax some multiplicities as we are not modeling complete models, but only model fragments for defining the queries. This means, if we have lower bound multiplicities greater than zero for properties, such as for the

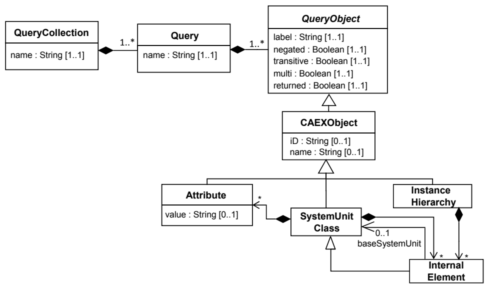


Fig. 4. AutomationML Query Definition Language (AQDL).

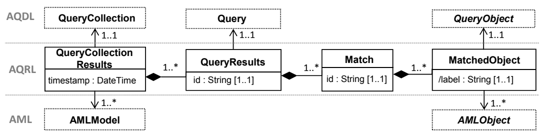


Fig. 5. AutomationML Query Result Language (AQRL).

name attribute of the CAEXObject class, we have to reset them to zero. Otherwise, we would have to introduce more information as needed for defining a query.

D. Explicitly Representing Query Results

Having AQDL, a language to explicitly model queries, at hand, an interpreter is able to execute the queries for particular AML models. For representing the results of the query execution, we provide an additional sublanguage of AQL, named AML Query Result Language (AQRL) which allows to represent the query results as proxy elements to the input model elements.

Fig. 5 shows the proposed language AQRL. By means of this language, we can document for which input models the query models have been executed. Furthermore, the main content of the query result models are the matches which are computed for each query. A match defines for each QueryObject which should be reported the actual binding, i.e., the matched objects in the AML model.

E. Prototypical Implementation in Eclipse

Based on our previous work for providing tool support for AML in Eclipse [7], we implemented a prototype of AQL. In particular, we specified AQDL and AQRL as Ecore-based metamodels. Using the standard EMF capabilities, we generated tree-based modeling editors for both languages. For executing AQDL queries on AML models, we implemented a prototypical interpreter in Java. The interpreter reads the AML models as well as the AQDL models and produces AQRL models as output. We provide an open source implementation of our prototype with further description and examples on our project website[3].

[3]`https://cdl-mint.big.tuwien.ac.at/prototype-artefacts-for-case-2018`

In Fig. 6, we provide a screenshot of our prototype showing from left to right: an AML model, an AQDL example query, and the result of the query as an AQRL model. This example is specifically showing a simple query, namely retrieve all internal elements. This query is defined in the AQDL model by stating a QueryObject that should be matched for internal elements. Thus, the QueryObject is represented by an internal element without any further constraints which acts as a template for the query. The computed AQRL model is providing three matches for the given query and the given model. In the right properties window, the result object for the first match is shown, i.e., the Stack internal element which is the first internal element in our example instance hierarchy (cf. left modeling editor).

In the following section, we describe several examples of queries which can be defined, executed, and the results represented in our prototypical implementation.

IV. AUTOMATIONQL BY-EXAMPLE

After introducing the language definition of AQL, we now demonstrate the language for the example model shown in Fig. 2. In particular, we instantiate each language feature by defining a specific query (Q) requiring this feature. Fig. 7 illustrates the queries and their results for the example model in pseudo notation. We use the identifiers of the base model elements (1-11) to mark if the element is contained in a query result or not.

Q1: This query is just containing a single query object, symbolizing that we want to find all CAEXObjects. Since we are interested in indirect instances as well, we mark this query object as deep to match for indirect instances. The result of the query is the set of all elements of the example model, since they are all indirect instances of the class CAEXObject.

Q2: This query searches for internal elements which have the weight attribute set with a value greater than 50. We mark both query objects to be reported and add constraints for the name attribute of the attribute as well as for the value attribute. The result of executing the query is a tuple which represents the internal element and its contained attribute, since only one internal element has a weight attribute defined, but it also fulfills the value constraint.

Q3: This query searches for all internal elements which are direct child nodes of the PPU instance hierarchy, i.e., more precisely, all instance hierarchies with the name “PPU”. The result of executing this query for our example is a triple which represents these internal elements: Stack1, Crane1, and Ramp1.

Q4: This query selects all internal elements of the PPU instance hierarchy which have at least another internal element as a child node. The result includes two entries: the internal element Stack1 with its internal element Conveyor1 and the internal element Stack1 with its internal element Store1. Please note that in this case, the entries are tuples as we return for a match more than one element.

Q5: This query searches for all internal elements of the PPU instance hierarchy which have another internal element

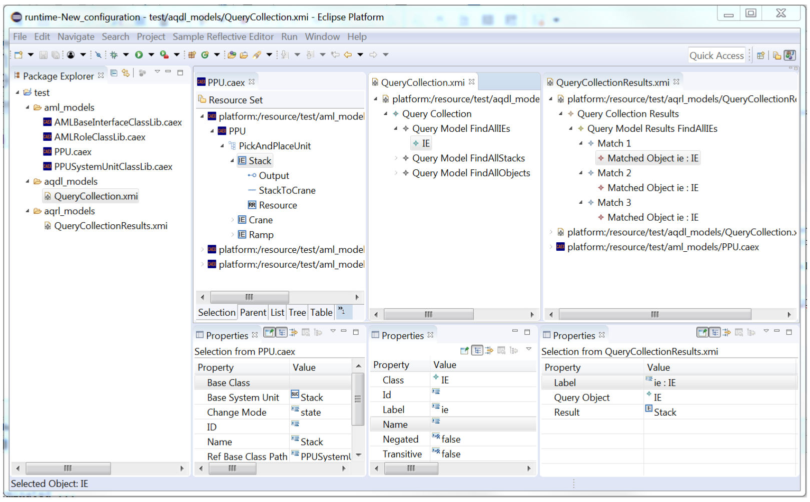


Fig. 6. Screenshot of the prototypical tool support for AutomationQL in Eclipse.

as child and if so all further internal elements down the hierarchy are included in the result. For this, we use the transitive operator which computes the transitive closure of the reflexive containment reference of internal elements. The result of this query contains two tuples. The first tuple contains the internal element Stack1 with its internal element Conveyor and the second tuple contains the internal element Stack1 with the internal elements Store1 and it's internal element Sensor1.

Q6: This query searches for all internal elements which have no further internal elements as a child node, i.e., we want to retrieve all leaf nodes. The result set of this query shows the internal elements Conveyor1, Sensor1, Crane2, and Ramp3.

Q7: This query selects all internal elements that refer to a system unit class (SUC) named "Stack". The result is the internal element Stack1 of the system model which references to the SUC Stack of the prototypes library.

Q8: Finally, this query searches for all internal elements which have more than two internal element as direct child nodes. For this, we use the multi object feature of our query language which directly matches for a set of elements at once. The result of this query is the empty set for our example.

V. Critical Discussion

The shown examples in the previous section illustrate that there is a class of queries which can be defined by augmented AML model fragments, i.e., by visualizing the inherited attributes from the QueryObject class for AML model elements.

Following this approach, we can mostly reuse the modeling style of AML for defining queries. This means, we instantiate elements from the AML language to represent what should be reported by a query. However, we also have to define conditions by adding expressions instead of concrete values to the properties of AML. Although this is still within the structures of AML, the expressions have to be defined with an expression language instead of "just" providing values. However, our hypothesis is that adding an expression language for this purpose, results in shorter code fragments as directly formulating the full query with expression languages.

Our current query definition and query result editors are not providing an optimized syntax for modeling the queries as we have done with our pseudo notation, e.g., cf. Fig. 6. Especially for representing the query results to end users, directly highlighting model elements in the AML editors may be preferred and interactive techniques should be provided to explore the matches which have been computed. Our current structured representation of the query results may be interpreted by dedicated user interface components as we provide a dedicated model structure. Another alternative is to provide model transformations to generate summaries of the query results which can be customized by the users.

Finally, we provide a first version of the query interpreter which is directly executing the query models. However, due to performance and scalability requirements, the query models may be translated to existing query engines which provide support for optimizing queries out-of-the-box. While we currently see the interpreter approach more suited to experiment with query language concepts, for industrialization concerns, a generator approach to target powerful query engines is preferable and subject to our future work.

VI. Conclusion and Future Work

In this paper, we have presented a novel approach for querying AML models by-example. We presented AQL, a graph pattern based query language which is based on AML. AQL supports positive and negative graph patterns,

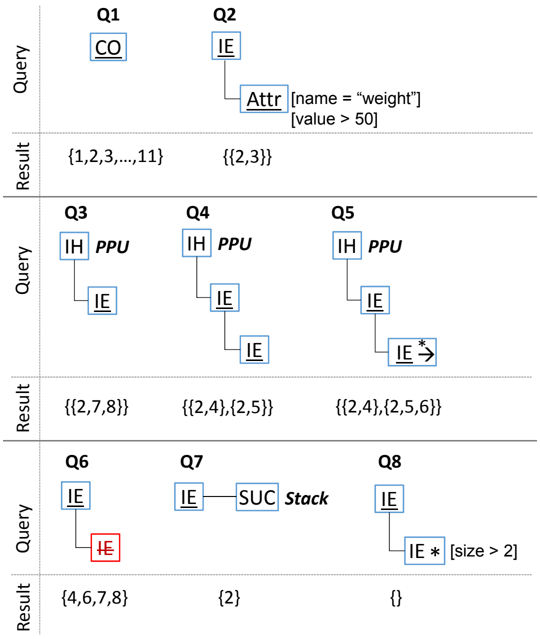


Fig. 7. Example queries and their results for the example of Figure 2. For visualization purposes, we use an informal pseudo notation: elements which should be reported have an underlined label, conditions are shown in brackets, negated elements are shown in red with crossed out label, transitive elements are annotated with $\xrightarrow{*}$, and multi elements with $\star$. For reporting the retrieved elements, we use the identifiers of the base model elements as a shorthand notation.

computing transitive closures to investigate recursive tree structures, and to match for element sets. The query results are explicitly represented in a result model which acts as a proxy to the AML base model elements.

While already several interesting queries can be formulated in AQL, further work is required to fully exploit the potential of a by-example query language for AML. In the future, we plan to explore further application cases of the developed query language as well as how to integrate other query concepts such as ordering and aggregation. In particular, we are interested in the evaluation of existing query (by-example) languages with respect to expressivity. Moreover, for providing a scalable evaluation of the queries, transformations to existing query engines such as EMF IncQuery[4] may be of interest to exploit sophisticated techniques such as query optimization and incremental evaluation. Finally, expanding the query by-example approach to a transformation by-example approach [19] for AML is another research line. We are already able to query model information which may be used in the future by transformation templates to introduce new elements, or to delete and modify existing ones.

ACKNOWLEDGMENT

This work has been funded by the Austrian Federal Ministry of Science, Research and Economy and the National Foundation for Research, Technology and Development. We thank Prof. Birgit Vogel-Heuser and her team for providing detailed documentation about the Pick and Place Unit (PPU).

REFERENCES

[1] A. Lüder, M. Foehr, L. Hundt, M. Hoffmann, Y. Langer, and S. Frank, “Aggregation of engineering processes regarding the mechatronic approach,” in *Proc. of ETFA*, 2011, pp. 1–8.
[2] S. Faltinski, O. Niggemann, N. Moriz, and A. Mankowski, “AutomationML: From data exchange to system planning and simulation,” in *Proc. of ICIT*, 2012, pp. 378–383.
[3] “IEC 62714 - Engineering data exchange format for use in industrial automation systems engineering - AutomationML,” www.iec.ch, 2014.
[4] N. Schmidt, A. Lüder, H. Steininger, and S. Biffl, “Analyzing requirements on software tools according to the functional engineering phase in the technical systems engineering process,” in *Proc. of ETFA*, 2014, pp. 1–8.
[5] S. Biffl, A. Lüder, E. Mätzler, N. Schmidt, and M. Wimmer, “Linking and Versioning Support for AutomationML: A Model-Driven Engineering Perspective,” in *Proc. of INDIN*, 2015, pp. 1–8.
[6] M. Sabou, F. J. Ekaputra, and S. Biffl, “Semantic web technologies for data integration in multi-disciplinary engineering,” in *Multi-Disciplinary Engineering for Cyber-Physical Production Systems*, 2017, pp. 301–329.
[7] T. Mayerhofer, M. Wimmer, L. Berardinelli, and R. Drath, “A Model-Driven Engineering Workbench for CAEX Supporting Language Customization and Evolution,” *IEEE Transactions on Industrial Informatics*, vol. PP, no. 99, pp. 1–11, 2017.
[8] M. Schleipen, R. Drath, and O. Sauer, “The system-independent data exchange format CAEX for supporting an automatic configuration of a production monitoring and control system,” in *Proc. of ISIE*, 2008, pp. 1786–1791.
[9] *IEC 62424:2016 - Representation of process control engineering*, http://www.iec.ch, IEC Std., 2016.
[10] M. M. Zloof, “Query by example,” in *Proc. of NCC*, 1975, pp. 431–438.
[11] ——, “Query-by-example: the invocation and definition of tables and forms,” in *Proc. of VLDB*, 1975, pp. 1–24.
[12] R. Drath, A. Lüder, J. Peschke, and L. Hundt, “AutomationML - the glue for seamless automation engineering,” in *Proc. of ETFA*, 2008, pp. 616–623.
[13] M. Brambilla, J. Cabot, and M. Wimmer, *Model-Driven Software Engineering in Practice*. Morgan & Claypool Publishers, 2017.
[14] D. Steinberg, F. Budinsky, M. Paternostro, and E. Merks, *EMF: Eclipse Modeling Framework 2.0*, 2nd ed. Addison-Wesley, 2009.
[15] R. Mordinyi, P. Schindler, and S. Biffl, “Evaluation of NoSQL graph databases for querying and versioning of engineering data in multi-disciplinary engineering environments,” in *Proc. of ETFA*, 2015, pp. 1–8.
[16] H. Lieberman, *Your Wish is My Command: Programming by Example*. Morgan Kaufmann Publishers, 2001.
[17] “The Pick and Place Unit,” http://www.ppu-demonstrator.org, Institute for Automation and Information Systems (AIS), TU Munich, 2013.
[18] B. Vogel-Heuser, C. Legat, J. Folmer, and S. Feldmann, “Researching Evolution in Industrial Plant Automation: Scenarios and Documentation of the Pick and Place Unit,” Technical University of Munich, Tech. Rep. TUM-AIS-TR-01-14-02, 2014.
[19] G. Kappel, P. Langer, W. Retschitzegger, W. Schwinger, and M. Wimmer, “Model transformation by-example: A survey of the first wave,” in *Conceptual Modelling and Its Theoretical Foundations*, 2012, pp. 197–215.

[4] https://www.eclipse.org/viatra/query.php