Message from the Modellierung'22 Workshop Chairs

Judith Michael[1], Jérôme Pfeiffer[2], Andreas Wortmann[3]

This volume includes the proceedings of the Workshops of the 2022 Modellierung conference (Modellierung'22). As a forum for foundations, methods, techniques, tools as well as domains and applications of modeling, the Modellierung conference series, which has been organized by the Cross-Sectional Expert Committee on Modeling ("Querschnittsfachausschuss Modellierung") of the Gesellschaft für Informatik (GI) since 1998, has established itself as a central conference on modeling topcis for German-speaking researchers. The Modellierung conference series serves as a platform to exchange experiences and insights on modeling for which it addresses an audience from both practice and academia. The workshops were held on the 27th and 28th of June 2022 in Hamburg.

1 Workshops

The workshops were selected by the workshop chairs, considering the feasibility of the proposed workshop and the potential to attract an engaged audience. The following workshops were accepted for Modellierung'22:

- **Workshop on Modellierung in der Hochschullehre (MoHoL'22)**. So far, modeling in academics is taught in classic classroom-style teaching, e.g., lectures. However, in the last decades, scientific research and academic didactics demand a change of perspective from a lecturer-centric one to a student and competence-related perspective on teaching. This perspective requires the active participation of students in applying theoretical knowledge. At the same time, not least because of the Bologna Process, the question regarding the quality of academic degrees, and the corresponding capability to check the students' achievements in an objective, fair, and learning goal-oriented way. Teaching modeling has to face these challenges, too. The target audience for this workshop is all who are interested in and participate in modeling in the academic domain.
- **Workshop on Modelle und KI (MoKI'22)**. The increasing availability of a large amount of data in all application areas effects a growing interest in artificial intelligence

[1] Software Engineering, RWTH Aachen, Germany michael@se-rwth.de
[2] Institute for Control Engineering of Machine Tools and Manufacturing Units (ISW), University of Stuttgart, Germany jerome.pfeiffer@isw.uni-stuttgart.de
[3] Institute for Control Engineering of Machine Tools and Manufacturing Units (ISW), University of Stuttgart, Germany wortmann@isw.uni-stuttgart.de

("künstliche Intelligenz (KI)ïn German). The new paradigm of data-driven AI, i.e., learning (domain-) models and their update using data mining techniques is popular because it decreases the effort to create application systems. However, it has many disadvantages. This workshop is interested in any submission regarding interfaces between conceptual modeling and AI.

- **Workshop on Modeling in (and for) Production (MoPro'22)**. The production domain is permeated by heterogeneous data sources, a variety of IT systems, and complex industrial use cases - aspects that offer an exciting field for research. The MoPro Workshop aims to be a platform for researchers and practitioners within the production domain to exchange their modeling techniques, interesting use cases, and challenges. This workshop is interested in the use of models for development, production, and usage cycles, as well as model-based and model-driven approaches that span these domains across disciplinary boundaries.
- **Workshop on State of the Art Methods and Tools in Model-based Systems Engineering (SpesML'22)**. The transition from document-based to model-based systems engineering (MBSE) offers an approach to develop complex cyber-physical systems in a interdisciplinary context in a sustainable and comprehensible way. The capability of companies to efficiently develop these systems in a interdisciplinary context is a competitive factor already.
- **Workshop on Research Data Management in Modeling (RDM4MOD'22)**. The demand increases to substantiate the claims made in the scientific processes in the realm of modeling in all areas of computer science. Thus, publications, funding proposals etc. require more often that empirical data (if available) along with the related context of experiments and the artifacts in terms of descriptions, software and other tools will be part of the publication or proposed project as well. Infrastructure will be provided to store and make available this kind of research data according to the FAIR – principles (Findable, Accessible, Interoperable, Reusable) as part of the National Research Data Infrastructure. With this workshop, the consortium NFIDxCS aims at collecting requirements and existing approaches to build such an infrastructure with a special focus on modeling issues.
- **Workshop on Modellierung und Simulation im Engineering und zur virtuellen Inbetriebnahme im Maschinen- und Anlagenbau (VDI/VDE-GMA FA 6.11 Virtuelle Inbetriebnahme)**. In the past years, virtual commissioning ("virutelle Inbetriebnahme (VIBN)") evolved into an established tool for engineering machines and plants. However, there are still several challenges and many companies have not integrated virtual commissioning into their commissioning processes. The basic idea of virtual commissioning is testing an automation system for the production plant with a digital model before the physical plant is built. Ideally, this happens in real-time. For the underlying model, the term executable digital twin has proven suitable.

2 Acknowledgements and Thanks

We would like to thank all those who contributed to making the Modellierung’22 workshops possible. First of all, we would like to thank the workshop organizers for their workshop ideas and the engagement and energy they put into making the workshops a reality. Namely, we thank:

- Meike Ullrich, Peter Fettke, Peter Pfeiffer, Selina Schüler and Michael Striewe for organizing the workshop on Modellierung in der Hochschullehre (MoHoL’22)
- Dominik Bork, Peter Fettke and Ulrich Reimer for organizing the workshop on Modelle und KI (MoKI’22)
- Judith Michael and István Koren for organizing the workshop on Modeling in (and for) Production (MoPro’22)
- Wolfgang Böhm, David Schmalzing and Nico Jansen for organizing the workshop on State of the Art Methods and Tools in Model-based Systems Engineering (SpesML’22)
- Michael Goedicke and Ulrike Lucke for organizing the workshop on Research Data Management in Modeling (RDM4MOD’22)
- Ronald Schmidt-Vollus for organizing the workshop on Modellierung und Simulation im Engineering und zur virtuellen Inbetriebnahme im Maschinen- und Anlagenbau (VDI/VDE-GMA FA 6.11 Virtuelle Inbetriebnahme)

Furthermore, we are grateful to the members of the workshop program committees, who reviewed the workshop submissions and ensured the quality of the presented research. Additional thanks go to the authors of all workshop submissions and the attendees of the workshops for making Modellierung’22 an interesting venue.

A special thanks goes to the General Chairs of the Modellierung’22 Matthias Riebisch and Marina Tropmann, as well as the tools and demo chairs Dominik Borg and Simon Hacks, as well as the program committee co-chairs Stephanie Schulte, Sabine Schumann and Sabrina Göllner.

Finally, we would like to acknowledge the team of the GI Digital Library who made publishing this volume possible, as well as the EasyChair team, whose software was instrumental during the review processes.

Aachen, Stuttgart, June 2022
Judith Michael, Jérôme Pfeiffer, and Andreas Wortmann

Workshop "Modelle und KI"

Dominik Bork[1], Peter Fettke[2], Ulrich Reimer[3]

Abstract: The workshop focuses on topics at the intersection of the fields of conceptual modeling and AI and explores the value conceptual modeling brings to AI, and, vice versa, the value that AI can bring to conceptual modeling. This covers a wide range of issues such as how to combine learned and manually engineered models, data-driven modelling support, automatic incremental model adaptation, and how to achieve the explainability of learned models e.g. by utilizing conceptual models as background knowledge.

Keywords: Conceptual Modelling; AI; Model Learning; Explainability

Vorwort

With the increasing availability of large amounts of data in practically all application areas, the field of artificial intelligence (AI) has been attracting increasing attention for some time now. The new paradigm of data-driven AI, i.e. learning (domain) models and keeping them up-to-date by using data mining techniques, is highly attractive because it reduces the effort of creating application systems. However, it also has many disadvantages. For example, models generated from data usually cannot be inspected and understood by a human being, and it is difficult to integrate already existing domain knowledge into learned models ? prior or after learning.

The approaches to conceptual modelling as well as earlier approaches to AI have mainly been focusing on the manual engineering of models, which requires a great deal of time and money. Thus, depending on the application domain, these approaches scale up poorly.

In this workshop, we are interested to discuss all kinds of topics at the intersection of the fields of conceptual modeling and AI. More specifically, we would like to explore the value conceptual modeling brings to AI, and, vice versa, the value that AI can bring to conceptual modeling.

[1] Business Informatics Group (BIG), Institute of Information Systems Engineering, TU Wien, Favoritenstr. 9-11 / 194-3, 1040 Wien, Austria, dominik.bork@tuwien.ac.at
[2] German Research Center for Artificial Intelligence, Saarland University, 66123 Saarbrücken, Germany, peter.fettke@dfki.de
[3] Eastern Switzerland University of Applied Sciences, Institute for Information and Process Management, Rosenbergstrasse 59, 9001 St. Gallen, Switzerland, ulrich.reimer@ost.ch

The workshop has been deliberately designed to be strongly discussion-oriented. Apart from an initial introduction into the topic the workshop comprises two papers, which together with additional provocative theses provided by the organizers form the starting point for the discussion part of the workshop.

The paper by Marvin Hubl presents a manually engineered mathematical model for a process coordination problem in agricultural crop production. To estimate appropriate values for the variables in the model, considerable domain knowledge would be needed since the values depend on a multitude of factors from the specific application setting. As a more viable alternative approach, the authors suggest to determine proper variable values using machine learning.

Rittelmeyer and Sandkuhl present an approach to support the usage of AI techniques for enterprise modelling. In several industrial use cases the authors have observed that due to a lack of understanding of AI concepts many companies encounter problems when using AI. To cope with this, the authors have developed a so-called “morphological box” for AI solutions for the purpose of enterprise modelling. A morphological box divides a given problem into different aspects that can be seen as parameters or features and it identifies potential values for those features. The paper concludes with an account of the experiences made when using the morphological box.

Both papers illustrate the fact that there is a huge potential in combining manual model engineering with data-driven model generation: the advantages of both approaches can be combined and their respective disadvantages be mitigated.

J. Michael, J. Pfeiffer, A. Wortmann (Hrsg.): Modellierung 2022 Satellite Events,
Digital Library, Gesellschaft für Informatik e.V. 6

Modeling an Agricultural Process Coordination Problem to Enhance Efficiency and Resilience with Methods of Artificial Intelligence

Marvin Hubl[1]

Abstract: Modeling of relations in a domain is a fundamental basis for solving domain problems. However, even well-formulated mathematical models do not always allow for satisfactory solutions. Here, methods from Artificial Intelligence bring value for solutions based on the formal models, e. g. by meta-heuristics. Furthermore, variables in a mathematical model may require manifestations although exact values are not known or measured. Machine-learning-based methods can enhance the appropriateness for the variable manifestation. We study upon these issues at the example of a process coordination problem in agricultural crop production. We analyze how methods of Artificial Intelligence can enhance processual efficiency and resilience. Therefore, two domain objectives are formalized: (i) maximization of machine utilization; (ii) maximization of aggregated area output. We identify and discuss the contribution of Artificial Intelligence for solving the mathematically formalized problem appropriately.

Keywords: Mathematical Modeling; Optimization Problem; Process Coordination; Resource Allocation; Agricultural Engineering

1 Introduction

Methods of Artificial Intelligence (AI) provide promising solutions "in two basic situations: 1. A problem may not have an exact solution because of inherent ambiguities in the problem statement or available data. [...] 2. A problem may have an exact solution, but the computational cost of finding it may be prohibitive." [Lu09, pp. 123 et seq.][2] We study the merits of AI methods at the example of a process coordination problem in the agricultural engineering domain. Therefore, we develop a formal model with the endeavor for analytical solutions in the mathematical sense, i. e. ideally by non-numerical solving of equations. However, it turns out, first, that this ideal does not appear to be feasible due to computational complexity. Second, the formal model contains variables that are crucial for the appropriateness of the overall result but cannot be exactly determined. Against this backdrop, we discuss the incorporation of AI methods to attain reasonably expected satisfying solutions. The AI-based solutions most probably will not be first-best solutions as opposed to the mathematical endeavor. The main value that AI methods bring here is to make appropriate solutions of complex and vague problems feasible.

[1] University of Hohenheim, Dept. of Information Systems II, Schwerzstraße 35, 70599 Stuttgart, Germany marvin.hubl@uni-hohenheim.de

[2] Line breaks and indentations edited out.

Novel ideas for the application of AI provide value in the agricultural engineering domain. Declining harvest yields [BM20], presumably caused by climate change, more frequent extreme weather events [BM20] and scarce soil require efficient and resilient crop production. Scarce soil is to some extent caused by climate change, too, but also by high population and other soil utilization than for plant-based food production for humans, e. g. for crop cultivation for biofuel, for animal feed or for urbanization and traffic purposes. With efficiency we refer to the demand that for the given available utilizable soil the harvest yields shall be as high possible. With resilience we refer to maintaining high harvest yields at exogenous "shock" events. An exogenous shock event is any event that significantly decreases output, if not undertaken measures for adaptation, and occurs beyond the control of the affected agents.

In general efficiency is defined as a relation between output and input.

$$\text{Efficiency} := \frac{\text{Output}}{\text{Input}} \,. \tag{1}$$

Domain-related, *agronomic* metrics focusing machine efficiency are

$$\text{Field efficiency} := \frac{\text{Productive machine hours } [h] \textit{ (Output)}}{\text{Total time of machine usage } [h] \textit{ (Input)}} \,, \tag{2}$$

$$\text{Area output} := \frac{\text{Farmed hectars } [ha] \textit{ (Output)}}{\text{Hours } [h] \textit{ (Input)}} \,. \tag{3}$$

As a metric for resilience we simply define a relationship of the output with an exogenous shock event to the planned output without the exogenous shock event.

$$\text{Resilience} := \frac{\text{Farmed area with exogenous shock } \hat{A}}{\text{Total area to be farmed } A} \tag{4}$$

In this metric a shock event is a dichotomous variable, i. e. it is either present or not, but is not further quantified. Possibilities to further quantify a shock event as a metric variable could be the output difference when not undertaking any measures or the time needed to recover from the shock event with given measures. However, a typical shock event in the agricultural domain are unplanned weather conditions, such as a thunderstorm. An upcoming thunderstorm often sets an unplanned time limit for farming an area A. This time limit, respectively the remaining time as of the point in time of knowing about the thunderstorm can serve as a quantification for the shock amplitude, too.

Time limiting shock events require farmers in the short term for temporal extensions of their agricultural machine capacity. With the initially planned and available machine capacity it is only possible to effectively farm the area $\hat{A}$ until the unplanned time limit. An extension of the agricultural machine capacity enhances economic resilience of crop production, if it leads to an $\hat{A}' > \hat{A}$. In the agricultural domain temporal extension of machinery can be realized with rental resp. shared machines, e. g. via a cooperative machinery ring [Br91; PA16]. However, machines are usually a scarce resource. The “Maschinenring Baden-Württemberg”, for instance, has 25 rental tractors for 29 regional rings, i. e. on average less than one tractor per region. Note that a shock event, like a thunderstorm, usually affects several or even all farmers in a region. Hence, then, many farmers need a temporal machinery extension at the same time.

Intelligent coordination of cooperatively shared machines contributes to solving the problem of maintaining high output. The coordination problem is to allocate shared machines being scarce resources among the demanding farmers. The allocation of the shared machines, in turn, basically, constitutes a logistical problem, namely in providing the right machine, at the right time, at the right place [Pf10, p. 12]. Objective functions for the allocation are to (1) maximize the field efficiency of the shared machines or to (2) maximize the aggregated area output of a region. While the first objective is relevant for the machine provider to minimize unproductive idle time, the second objective is of macro-economical and of societal importance w. r. t. food security. We will analyze how both objective functions relate to each other.

The remainder of the paper is structured as follows. Section 2 analyzes the state of the art in applying AI-related methods for selected logistical coordination problems in the agricultural domain. Section 3 develops a formal, mathematical model and deduces first analytical findings from it. Section 4 discusses the integration of AI methods with the model to acquire solutions for the underlying problem. Section 5 concludes the paper.

2 State of the art

2.1 Infield logistical problems in agriculture

Logistical coordination problems as well as AI-related solutions for them have long been acknowledged in agricultural research pertaining to infield operations. An important problem while farming an area is the preservation of the farmed soil as growth medium [BSG12] Soil compaction impairs the capacity of the soil to be a growth medium. Hence, harvest yields are reduced. High soil moisture and high tire pressure are fraught with high risk of compaction. Bochtis et al. [BSG12], therefore, refer to controlling for the wheel and track loads. Tracks with high risk of compaction, e. g. moist tracks, should be driven on with little tank content, i. e. with little fuel tank content or little seed tank content. A preventive measure is also to use low tire inflation pressure for tracks with high risk of compaction. However, low tire inflation pressure increases the consumption of diesel and should only be used purposefully deliberated. While deflating is fast, inflating can last long for the tire volumes.

From these characteristics an allocation problem for infield logistics is derived [BSG12]. Therefore, a route be a sequence of tracks on a field that are farmed without refilling the seed tank. The number of routes, hence, is given by

$$\text{Number of routes} = \frac{\text{Seed application rate} \times \text{Field area}}{\text{Seed tank capacity}}\,. \tag{5}$$

To be determined is an allocation from tracks to routes, such that the tank content is the lower and the tire inflation pressure is the lower the higher the risk of compaction is for a track. Note that refilling the tank and inflating the tires are time consuming and hence expensive. For this reason the number of tank refills and of inflating the tires should be as small as possible. The problem can be modeled and solved as a vehicle routing problem [AVO09; BS09]

2.2 Strategic logistical problems in agriculture

A strategic logistical problem in agriculture concerns the long-term planning of sowing. Relevant parameters are [SLM20]: (i) the variety of a seed, (ii) the number of fields, (iii) the expected profits from selling the sown crop variety, (iv) the necessary amount of seeds and (v) water for establishing a variety as well as (vi) the labor costs. Two strategic decision have to be made: (1.) Which crop variety shall be established on which field? This pertains to the logistical goal of having the right good at the right place. (2.) What amount of seeds shall be purchased for which variety? This pertains to the logistical goal to have the right amount of a good.

Solano et al. [SLM20] formulate an optimization program for the decision problem. The objective function seeks to maximize the sum of the expected profits over each field and each variety, incorporating a binary decision variable $X_{i,k}$, specifying whether variety i is established on field k. The objective function is constrained by the secondary conditions that (i) on each field at most one variety can be established, (ii) the yields of the sown varieties shall satisfy the demand for the varieties, (iii) the purchased amount of seeds must cover the required amount of seeds and that (iv) the required amount of water must not exceed the available amount of water.

3 Model development

3.1 Initial model formulation

Be $L_1, \ldots, L_n$ farmers and R a machinery ring with a machine m^* that is needed as soon as possible by all farmers in the discourse world. All farmers use the shared machine m^* productively for the duration d. Be $V = \{v_0, v_1, \ldots, v_n\}$ the set of nodes of an undirected, connected graph where $v_0 := R$ and for $i > 0 : v_i = L_i$. Be $D(h,i)$ the duration of the shortest path from v_h to v_i. Be the n-tuple $\mathbf{b} \in \{1,\ldots,n\}^n$ with $b_j \neq b_k, \forall j,k \in \{1,...,n\}, j \neq k$ a permutation of the nodes $\{v_1,\ldots,v_n\} \subset V$. (There are $n!$ many $\mathbf{b}$.) The total path duration be

$$c := D(0,b_1) + \sum_{j=1}^{n-1} D(b_j, b_{j+1}) \ . \tag{6}$$

Be the field efficiency of the shared machine m^*

$$\varphi_{m^*} := \frac{n \cdot d}{n \cdot d + c} \ . \tag{7}$$

The numerator reflects the productive time of the shared machine which is, by provisional assumption, the number of farmers multiplied with the duration for farming a field. The assumption is that the farmers have all the same area to farm, do not possess own machinery and that there is only one shared machine. The denominator reflects the productive time plus the travel time of the shared machine, given by the total path costs c. The objective function for the field efficiency is to maximize φ_{m^*} over the variable parameters. Here, only c is variable. Maximizing φ_{m^*} over c is equivalent to minimizing c over $\mathbf{b}$, because in expression 7 c occurs in the denominator and c depends on $\mathbf{b}$.

$$\max_{\textit{var. param.}} \varphi_{m^*} \Rightarrow \max_{c} \varphi_{m^*} \Leftrightarrow \min_{\mathbf{b}} c \ . \tag{8}$$

Be $f_i(m^*)$ the area output of the shared machine m^* at farmer L_i. The area output of the same machine can diverge on different farmers, e. g. due to soil characteristics. Be F_i^* the total area output of L_i and $F_i^* = f_i(m^*)$. This reflects the provisional assumption that the farmers do not possess own machinery and hence their total area output is equals to the area output of the shared machine. Be F the aggregated area output over all farmers:

$$F = \sum_{i=1}^{n} F_i^* \ . \tag{9}$$

Be A_i the area that is to be farmed by L_i and

$$A = \sum_{i=1}^{n} A_i \ . \tag{10}$$

From the provisional assumption that all farmers use only the shared machine m^* for the same duration d follows with $d = \frac{A_i}{f_i(m^*)}$ provisionally that $A_h = A_i, \forall h,i \in \{1,\ldots,n\}$. The aggregated area output

$$F = \frac{A}{n \cdot d + c} \ . \tag{11}$$

Analogously as for φ_{m^*} it holds for F.

$$\max_{var.\ param.} F \Leftrightarrow \min_{\mathbf{b}} c \ . \tag{12}$$

This means that with the initial model formulation, minimizing c over $\mathbf{b}$ simultaneously maximizes F as well as φ_{m^*}.

Minimizing c over $\mathbf{b}$ constitutes a Hamiltonian path problem: Does a path that visits each node exactly one time exist with path costs $c \leq k, k \rightarrow \min$? The requirement of visiting each node exactly once can be guaranteed by inserting the transitive hull into the connected graph. Figure 1 illustrates an example graph for the problem. The dotted edges are the transitive hull representing the shortest paths between two nodes. The durations for distances shall be assumed to be proportional to the distances by the factor $\hat{v}^{-1}$, where $\hat{v}$ is a constant resp. mean velocity (in km/h). There are $3! = 6$ permutations, i. e. orders in which the nodes can be visited resp. in which the shared machine can be allocated to the farmers $\{L_1, L_2, L_3\}$. Table 1 provides the resulting path costs for each node permuation in the example graph. A Hamiltonian path is a Hamiltonian cycle without return to the start node, which has been shown to be $\mathcal{NP}$-complete [Ka72]. AI methods provide feasible solutions [Lu09, pp. 79 et seqq.], [RN03, pp. 59 et seqq.].

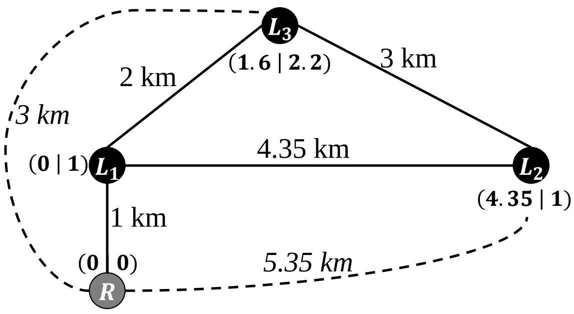


Fig. 1: Example graph for a coordination problem of allocating a shared machine m^* to farmers.

Permutation $\mathbf{b}$	Path costs c
(1, 2, 3)	8.35 km
(1, 3, 2)	6 km
(2, 1, 3)	11.7 km
(2, 3, 1)	10.35 km
(3, 1, 2)	9.35 km
(3, 2, 1)	10.35 km

Tab. 1: Resulting path costs for all node permutations in the example graph depicted by figure 1.

If we drop the assumption that the different areas of the farmers have different sizes, nothing changes essentially. Minimizing c over $\mathbf{b}$ still maximizes F and φ_{m^*} simultaneously:

$$d_i = \frac{A_i}{f_i(m^*)} \,, \tag{13}$$

$$D := \sum_{i=1}^{n} d_i \,, \tag{14}$$

$$\varphi_{m^*} := \frac{D}{D+c} \,, \tag{15}$$

$$F = \frac{A}{D+c} \,. \tag{16}$$

$$\left(\max_{\textit{var. param.}} \varphi_{m^*} \Leftrightarrow \max_{\textit{var. param.}} F \right) \Leftrightarrow \min_{\mathbf{b}} c \,. \tag{17}$$

3.2 Model extension

Now, we want drop the assumption that the farmers do not possess own machinery. Be M_i the set of utilizable machines, possessed by L_i. Be F_i the total area output of L_i *without the shared machine* m^* and be $f_i(m)$ the area output of a machine $m \in M_i$:

$$F_i = \sum_{\forall m \in M_i} f_i(m) \,. \tag{18}$$

Be t the elapsed time since a defined starting time. Be $A_i(t)$ the area that is still to be farmed by L_i after L_i has already farmed her area A_i with her own machines M_i for the duration of t.

$$A_i(t) = \begin{cases} A_i - t \cdot F_i & \text{, if } t < \frac{A_i}{F_i} \\ 0 & \text{, else} \end{cases} \,. \tag{19}$$

Be t_j the point in time from which m^* is utilizable for L_{b_j}. As from t_j the shared machine m^* is utilized additionally to M_{b_j} until A_{b_j} is completely farmed. Then, m^* has at L_{b_j} a productive time $d_{b_j}(t_j)$ of:

$$d_{b_j}(t_j) = \frac{A_{b_j}(t_j)}{F_{b_j} + f_{b_j}(m^*)} \tag{20}$$

with

$$t_j = \begin{cases} t_{j-1} + d_{b_{j-1}} + D(b_{j-1}, b_j) & \text{, if } 1 < j \leq n \\ D(0, b_1) & \text{, if } j = 1 \end{cases} \,. \tag{21}$$

For the case $1 < j \leq n$ in expression 21, the first summand is (a) the point in time, when the previous farmer, according to **b**, gets the shared machine, the second summand is (b) the remaining field working time at the previous farmer according to **b** and the third summand are (c) the path costs from the previous farmer, according to **b**, to farmer L_{b_j}. Be $\mathbf{t} = (t_1, \ldots, t_n)$:

$$D(\mathbf{t}) = \sum_{j=1}^{n} d_{b_j}(t_j) \ , \tag{22}$$

$$\varphi_{m^*} = \frac{D(\mathbf{t})}{D(\mathbf{t}) + c} \ . \tag{23}$$

$D(\mathbf{t})$ depends upon $\mathbf{t}$ which relates to c over $\mathbf{b}$ as follows: Be

$$c_j := D(0, b_1) + \sum_{h=1}^{j-1} D(b_h, b_{h+1}) \ . \tag{24}$$

Then, it holds for t_j:

$$t_j = t_{j-1} + d_{b_{j-1}}(t_j - 1) + c_j - c_{j-1} \ ; \ c_n = c \ . \tag{25}$$

Note that $c_j - c_{j-1}$ are the path costs from the previous farmer to L_{b_j}. Since $D(\mathbf{t})$ occurs in both the numerator and the denominator, φ_{m^*} is not maximized only by minimizing c. Hence, the objective function for the field efficiency reads:

$$\max_{\mathbf{b}} \varphi_{m^*} \tag{26}$$

The aggregated area output is

$$F = \frac{A}{D(\mathbf{t}) + c} \ . \tag{27}$$

Here, the objective function can be stated by

$$\max_{\mathbf{b}} F \Leftrightarrow \min_{\mathbf{b}} \left(D(\mathbf{t}) + c \right) \ . \tag{28}$$

With the model extension that farmers possess own machinery which they already use before the shared machine arrives, maximization of φ_{m^*} does not imply maximization of F and vice versa maximization of F does not imply maximization of φ_{m^*}. For example with $n = 2; A_1 = A_2 = 100\,ha; F_1 = 50\,\frac{ha}{h}; F_2 = 25\,\frac{ha}{h}; f_1(m^*) = f_2(m^*) = 50\,\frac{ha}{h}; D(0,1) = \frac{1}{3}\,h; D(1,2) = \frac{2}{3}\,h$ it results for $\mathbf{b} = (1,2)$ that $\varphi_{m^*} \approx \frac{1.555\,h}{2.555\,h} \approx 0.609$ and $F \approx \frac{200\,h}{2.555\,h} \approx 78.278\,\frac{ha}{h}$ and for $\mathbf{b} = (2,1)$ that $\varphi_{m^*} \approx \frac{1.3395\,h}{2.3395\,h} \approx 0.573$ and $F \approx \frac{200\,h}{2.3395\,h} \approx 85.488\,\frac{ha}{h}$.

4 Discussion: Integration of AI methods

4.1 Integration of machine-learning-based methods

In the preceding section, we developed a mathematical model that seems to be complete, at least for the assumptions on which it is based. However, the model contains variables on which the appropriateness of the overall solution depends crucially but are probably not known in the suggested exactness: These variables are (1) the exact durations $D(h,i)$ and (2) the actually realized area outputs F_i and $f_i(m^*)$.

One possibility to work with not exactly known or measurable variables is to use substantiated assumptions. For the durations, for instance, assumptions can be substantiated by dividing the exactly known distances of a section by the expected mean driving velocity for this section. For the area output the technically possible values can be calculated using machine characteristics, such as the width of the tractor attachments and technically optimal working resp. driving velocity on a field. However, both values, durations for traveling from one node to another and area output, a prone to environmental vagueness. The technically optimal area output, for instance, is almost never achieved [St18] and depends on fuzzy influences like soil conditions. An agricultural field is usually less homogenous than it seems. For example, the moisture and the soil composition varies in the field. If feasible, this variance would even suggest to apply seeds with a variable rate. This affects, in turn, again the technically optimal working velocity. Also the best depth in which seed is placed may depend on varying nutrient content, moisture and soil compaction. And if the seed placement depth is adjusted to field conditions, the technically optimal working resp. driving velocity usually varies, too.

The estimation of appropriate values for the two variable sets – there is not only one duration and not only one area output to determine – requires specific domain knowledge, e. g. from soil science and agricultural machine engineering for the area outputs and from traffic science for the travel durations between the nodes. Machine-learning-based methods promise to be particularly valuable when specific relations are not known. However, specific domain knowledge on causal and logical relationships is beneficial when using machine-learning-based methods to estimate values for the two variable sets, too, e. g. for feature engineering. The mathematical model, developed in section 3, precisely indicates where machine-learning-based methods can particularly contribute to more appropriate problem solving. The usage of machine-learning-based methods to specify values for the durations and area outputs is by no means an end in itself, but embedded in an overall, precisely described problem context.

4.2 Integration of heuristic methods

Assuming that appropriate values for the durations and area outputs are given, the mathematical model seems to suggest to provide exact solutions only by calculating some equations.

However, this is not the case due to the $\mathcal{NP}$-completeness of the underlying Hamiltonian path problem. $\mathcal{NP}$-complete problems are combinatorial problems that would be solved by a *non*-deterministic Turing-machine in a number of calculation steps that is bounded by a polynomial depending on the input length. Non-deterministic Turing-machines would have the property that they can multiply their computation capacity with each computation step. Naturally, such a machine cannot exist. A deterministic Turing-machine, which is a commonly agreed model to describe the set of algorithmically decidable problems, cannot solve $\mathcal{NP}$-complete problems efficiently, i. e. not with a number of calculation steps which is bounded by a polynomial depending on the input length. The number of necessary calculation steps for $\mathcal{NP}$-complete problems usually is exponential or even factorial w. r. t. the input length.

It is a core characteristic of $\mathcal{NP}$-complete problems that for finding an optimum no procedure is known for this problem class than to try out all possibilities. Since the number of possibilities is exponential or factorial w. r. t. the elements that are subject to the combinatorial problem, a solution in an acceptable time becomes infeasible already for relatively small input lengths. AI research has long been studying the integration of heuristic methods for problem solving. A heuristic method shall deliver an appropriate result with acceptable computational effort. However, in general heuristic methods cannot always guarantee to find an appropriate solution. Heuristic methods try to guide the search for a solution early into a promising direction. The search logic is given by so-called meta-heuristics. Several meta-heuristics haven been studied, i. e. best-first search, hill climbing, simulated annealing but also genetic algorithms [Lu09, pp. 133 et seqq.], [RN03, pp. 110 et seqq.].

Heuristic methods integrate *heuristics* in a meta-heuristic. Heuristics are estimation functions that shall exploit specific information to find an appropriate solution with acceptable computational effort. For this reason, heuristic search methods are also referred to as informed search [RN03, pp. 94 et seq.]. A good heuristic needs to incorporate specific domain knowledge. Therefore accurate conceptual models in the sense of formal theories with high prognosis power can bring crucial value to heuristic search. It is a fundamental insight from the so-called no-free-lunch-theorems that there cannot be a heuristic that outperforms other heuristics for all problem domains [WM96; WM97]. If a heuristic is more appropriate for one problem domain than another heuristic, then there exists another heuristic which is more appropriate for another problem domain.

5 Conclusions

We discussed how methods of AI can provide value in enhancing domain-specific process efficiency. Therefore, a coordination problem in the agricultural domain has been modeled mathematically. The problem is to allocate a shared agricultural machine, which is demanded by many farmers at the same time due to unplanned exogenous conditions, like an upcoming thunderstorm. The unplanned exogenous conditions require temporal extension of agricultural machine capacity in the near term to complete production within a shortened time limit. This targeted ability to efficiently adapt to an exogenous "shock" event is considered to be a form of economic resilience.

The objective functions for the allocations are to maximize field efficiency of the shared machine or to maximize the aggregated area output in the respective region. The mathematical model revealed that, in general, it is not possible to maximize both target variables at the same time. The model also revealed in its basic formulation, that the underlying problem is a Hamiltonian path problem and hence a complex problem in the computational theoretical sense.

For finding appropriate solutions, the integration of meta-heuristics and the role of specific heuristics that exploit domain knowledge were discussed. To set values for mathematical variables for which the characteristics are not known or measured exactly, the integration of machine-learning-based methods were discussed. Here, domain knowledge can contribute to feature engineering. Hence for both, heuristics and machine-learning-based methods, conceptual knowledge on domain-specific relations brings pivotal value to the generic AI techniques.

The formal model in this study contained only one shared machine. In this respect, an open and methodologically interesting issue is an extension to $r > 1$ shared machines $m_1^*, \ldots m_r^*$. Then, the productive times for an m_k^* depend on the allocation of all other shared machines $m_{-k}^*, k \in \{1, \ldots, r\}, -k \in \{1, \ldots, k-1, k+1, \ldots r\}$ and there are up to $(n!)^r$ allocations. Here, benefits of methods from distributed AI should be considered, too.

Furthermore, resilience had been defined as the relation between output at a shock event to the planned output without the shock event. The time limit forcing the farmers for short-termed temporal machinery extension still needs to be explicitly incorporated into the model.

Acknowledgements

This study has been developed in the research project KINERA and has been supported by the Federal Ministry of Food and Agriculture, Germany, under grant 28DK109A20.

References

[AVO09] Ali, O.; Verlinden, B.; van Oudheusden, D.: Infield logistics planning for crop-harvesting operations. Engineering Optimization 41/2, pp. 183–197, 2009.

[BM20] BMEL: Erntebericht 2020: Mengen und Preise, tech. rep., Berlin: Bundesministerium für Ernährung und Landwirtschaft, 2020.

[Br91] Brandes, W.: ‚Maschinenringe‘ – Their Role for Farmers‘ Income and Impacts on Structural Change in West Germany. Japanese Journal of Farm Management 29/3, pp. 78–85, 1991.

[BS09] Bochtis, D. D.; Sørensen, C. G.: The vehicle routing problem in field logistics part I. Biosystems Engineering 104/, pp. 447–457, 2009.

[BSG12] Bochtis, D. D.; Sørensen, C. G.; Green, O.: A DSS for planning of soil-sensitive field operations. Decision Support Systems 53/, pp. 66–75, 2012.

[Ka72] Karp, R. M.: Reducibility among combinatorial problems. In (Miller, R. E.; Thatcher, J. W.; Bohlinger, J. D., eds.): Complexity of Computer Computations. The IBM Research Symposia Series. Springer, Boston, pp. 85–103, 1972.

[Lu09] Luger, G. F.: Artificial Intelligence: Structures and Strategies for Complex Problem Solving. Pearson Education, Boston, 2009.

[PA16] Puschmann, T.; Alt, R.: Sharing Economy. Business & Information Systems Engineering 58/1, pp. 93–99, 2016.

[Pf10] Pfohl, H.-C.: Logistiksysteme: Betriebswirtschaftliche Grundlagen. Springer, Heidelberg, 2010.

[RN03] Russel, S.; Norvig, P.: Artificial intelligence: A modern approach. Prentice Hall, New Jersey, 2003.

[SLM20] Solano, N. E. C.; Llínás, G. A. G.; Montoya-Torres, J. R.: Mathematical model for planning of sowing in agricultural production systems. Manuscript for Big DSS Agro, Lleida, 2020.

[St18] Steckel, T.: Entwicklung einer kontextbasierten Systemarchitektur zur Verbesserung des kooperativen Einsatzes mobiler Arbeitsmaschinen. Dissertation, Univ. Hohenheim, URN: urn:nbn:de:bsz:100-opus-14767, URL: http://opus.uni-hohenheim.de/volltexte/2018/1476/, Stuttgart, 2018.

[WM96] Wolpert, D. H.; Macready, W. G.: No Free Lunch Theorems for Search, tech. rep. SFI-TR-95-02-010, Santa Fe: The Santa Fe Institute, 1996.

[WM97] Wolpert, D. H.; Macready, W. G.: No Free Lunch Theorems for Optimizations. IEEE Transactions on Evolutionary Computation 1/1, pp. 67–82, 1997.

J. Michael, J. Pfeiffer, A. Wortmann (Hrsg.): Modellierung 2022 Satellite Events,
Digital Library, Gesellschaft für Informatik e.V. 18

Features of AI Solutions and their Use in AI Context Modeling

Jack Rittelmeyer[1], Kurt Sandkuhl[2]

Abstract: Despite the implementation of many new artificial intelligence (AI)-based solutions in research and practice every year, companies still encounter problems while introducing AI solutions. One reason for that, from our own experience, are significant problems with understanding the concepts of AI. To cope with this problem, we aimed for developing a morphological box for AI solutions. The developed morphological box, its features and their values are based on four own industrial cases of AI solutions covering different application domains. We previously presented an enterprise architecture-based AI context model to help to better understand the context of an AI solution in a company and thereby minimize the risks of an implementation. We also analyzed that the morphological box supports the AI introduction process by improving and enhancing the three steps of the AI context model, which lead to more complete requirements for AI solutions.

Keywords: Enterprise architecture; AI context; organizational AI solutions; artificial intelligence; morphological box

1 Introduction

Development and introduction of AI solutions into enterprises or public administration is considered as a complex process [Ho16]. Recommendations by AI competence centers point at the importance of different perspectives in this process [Br19]:

- how business model, organizational structures and business processes are affected and have to be adjusted
- what competences and competence development of employees and management staff are required
- how the overall process of change management is performed when introducing AI-based innovations
- technology, data and composition of the actual AI solution

Studies about organizational readiness for AI solutions confirm those aspects [JWW21]. In the context of this complexity, conceptual modeling can help to understand the dependencies,

[1] University of Rostock, Institute of Computer Science, Albert-Einstein-Str. 22, 18057 Rostock, Germany jack.rittelmeyer@uni-rostock.de
[2] University of Rostock, Institute of Computer Science, Albert-Einstein-Str. 22, 18057 Rostock, Germany kurt.sandkuhl@uni-rostock.de

specifics and effects of changes in the perspectives described above. In particular, enterprise modeling (EM) and enterprise architecture (EA) modeling have proven to be supportive for innovation scenarios [NP20], for example because capturing multiple perspectives in one model is common practice in EM and EA modeling. However, specifics of AI solutions have not been sufficiently addressed in modeling and require more research work [Sj21].

We intend to contribute to the field of research by investigating the use of features of AI solutions when modeling organizational requirements for them. During our work with enterprises interested in AI solutions, particularly with small and medium-sized enterprises (SME), we observed that the concept of artificial intelligence, the different types of AI solutions and the varying preconditions for introducing AI, resulted in substantial problems to understand the concept of AI. In this context, we started to compare AI solutions with other software solutions and we used analogies to existing products. As this seemed to ease the understanding of what AI is, we developed the conjecture that a morphological box could be helpful. We started to develop this box based on our own industrial cases of AI solutions and to apply the box in meetings with SME. The development and current version of the morphological box is the first contribution of the paper.

In a second step, we investigate the potential of using the morphological box when preparing the introduction of AI solutions in enterprises. In previous work, we developed a method proposal for elaborating organizational requirements to AI solutions [Sa19], [SR22]. The core idea here is to identify the "context" of planned AI solutions in enterprise architecture models and extract requirements and frame conditions from this context. The leading question is, can the morphological box be used as an instrument to enrich the context model and, thereby, make the organizational requirements more detailed and complete?

The paper is structured as follows: in Section 2 the background for our work from EA management, AI context modeling and development of morphological boxes is introduced. Section 3 addresses development of the morphological box, including the industrial cases of AI solution development that form the empirical basis. Section 4 discusses the use of the morphological box in AI context modeling. Section 5 summarizes the findings and gives an outlook on future work.

2 Theoretical Background

In this chapter, we will describe the theoretical knowledge required to understand our research approach. Because a morphological box is developed and used in our AI context model, both of them are explained in more detail (sections 2.1 and 2.3). EA is the basis for the AI context model and will therefore also be introduced shortly (section 2.2).

2.1 Morphological box

The concept of a “morphological box” is based on work by Zwicky as a creativity technique for product development [Zw69]. A morphological box is often used to solve a problem by combining (partial) solutions already available in other areas. The general approach is to logically divide the problem into different aspects that can be seen as parameters or features, and to identify potential values for these features.

The process of building a morphological box starts by exactly defining the problem or phenomenon under consideration (scoping). This is followed by the selection of features that characterize the chosen phenomenon, which in our case are the AI solutions currently available. All features have to be made explicit and should not have any interference or affect one another. The features are noted down vertically and they form the first column of the box. At the same time, for each feature one individual row is included in the box that captures the values that the feature can have. These values for each feature have to be identified and must be specific and characteristic for the particular feature. All values are represented as columns for the features in their individual row. The result is a table with a specific number of rows (features) in the first column and varying number of columns (values).

Many applications of the morphological box are known and were the subject of publications. Some examples are the use as a tool for planning market innovations [SN12], identifying new electricity business cases [PL21], support of strategic decision-making [dWR07], business model prototyping [SSL14], knowledge transfer in organizations [KG09] or business model creation [Le11].

2.2 Enterprise Architecture Management

Enterprise Architecture Management (EAM) offers a systematic approach that includes various methods, tools and functions to plan, develop and coordinate the EA of a company in alignment with the company’s goals, visions and strategies [Ah12]. Thereby, it offers the possibility to improve the overall performance of a company as well as its effectiveness and efficiency by for example reducing redundancies or using existing synergies in the companies processes or infrastructure. An EA itself depicts all relevant components of a company in a model grouped by different architecture layers, such as business processes and roles on the business layer or application components and data objects on the application layer [Ah12].
A wide variety of different research areas exist in EAM, but most relevant to this research approach is work about the benefits of using EM and EAM for innovation management [NP20].

2.3 AI Context Modeling

In previous work, we offered a proposed method addressing feasibility studies and requirements elicitation of AI applications [Sa19]. The motivation is that many organizations investigate the introduction or development of AI solutions, but do not have experience with the technical complexity of AI applications. The method proposal assumes that an EA model for the enterprise under consideration exists. If this is not the case, the organizational context (see step 1 below) for the EA has to be developed. The method proposal consists of five steps, summarized below:

- Step 1. Model organizational AI context: aims at extracting all information of organizational structures, processes and resources required for or affected by the planned AI solution from the EA model of the organization under consideration. The extracted information is captured in a conceptual model, i.e., represented by using a modelling language. This conceptual model ideally is a subset or view of the analyzed EA model of the organization.
- Step 2. Elicit AI requirements: aims at documenting the requirements from candidate AI technologies that have to be fulfilled if a certain candidate technology is to be used in an organization (we recommend that an AI expert is involved in this step).
- Step 3. Analyze AI context: systematically analyzes the AI context model (from step 1) using the AI requirements (from step 2) for each EA layer of the context model separately.
- Step 3a. Enrich the EA model: this is as a complement to Step 3 “analyze AI context” to be performed optionally if step 3 discovers incompleteness of the model attributes or descriptions. The step is particularly important, e.g., for (a) images, videos and audio recordings where repository or storage systems only can be identified by company-insiders due to the naming in the EA model, (b) AI applications depending on the evaluation of historic data, where the extent of stored historic data and its quality are not described in the attributes of the model elements, and (c) data sources to be integrated for joint evaluation, which cannot be done with good performance and efficiency “on the fly”, but need detailed information about acceptable processing times or possibilities to extract data from operational systems
- Step 4. Decide on feasibility: gives support for deciding on feasibility based on the results from step 3.
- Step 5. Designing the future EA: a serious investigation on feasibility has to include the initial design of new or changed data, application and business architectures prepared for the AI application, and the migration planning. Only if migration is not only possible but also economically acceptable and resource-wise doable, the implementation should be started.

In addition to these procedural steps, the method also provides important concepts to observe and document, and a notation on how to do this.

3 Features of AI Solutions

First, the different use cases will be described and analyzed in this chapter (section 3.1). One of the use cases will be described in more depth. Secondly, we explain the morphological box we developed based on the findings and experience from the analyzed use cases (section 3.2).

3.1 Industrial Cases

From 2019 to 2022, the authors actively participated in several industrial projects that aimed at introducing new kinds of AI solutions in enterprises of different application domains. Four of these projects were selected for analysis in this paper as they each showed different features of AI solutions (Table 1). In all case studies, we collected documents, minutes of meetings and interviews with company representatives, field notes taken when working with the companies as well as models of processes, information structures and business models, and other relevant information. This material deals with the situation before making changes to products and services, the intermediate steps are taken and therefore before the situation at the end of the project. For reasons of brevity, we only describe one of the case studies (case study A) in more detail.

Case	Domain	AI solution	Literature
A	Clean room and air conditioning	Energy optimization on fleet level; anomaly detection on facility level	This section
B	Financial industries	Fraud detection for instant payments	[DSM19]
C	Marketing	AI for Object Recognition and Marketing Support	[Re21]
D	Garden products	Forecast of transaction development; anomaly detection	[Sa22]

Tab. 1: Analyzed industrial case studies

Case study A is an industrial case from air conditioning and clean room technologies (ACT). The case study company is a medium-sized enterprise from Germany designing, developing, installing and operating large ACT facilities. For energy optimization purposes and as a basis for predictive maintenance, additional sensors and control systems have to be integrated into the ACT facilities and connected to a network. This results in an Internet-of-Things (IoT) solution that forms the basis for new business services. The complexity and scope of such systems are increasing more and more in the field of industry and public buildings. Inspections of air handling units in operation often reveal significant deviations from the assessed energy efficiency. In the course of automation as well as I&C, the amount of data is increasing strongly. The direct and indirect processes involving air handling units can no longer be operated in an energy-efficient or -optimal way without intelligent data processing. The need for system solutions for self-recognition and self-organized learning and control of the systems is therefore high. AI solutions are capable of supporting this.

The envisioned IoT solution is supposed to implement diagnosis support of possible optimizations in air handling units as well as for the operational processes of the case study company. In such a solution, it must be possible to process a large amount of data from different sources in the systems. The IoT solution has to be integrated into the case company's operational processes to support new types of services. In an ACT facility, different technical devices are installed that are integrated into a control flow and together provide the desired AC functionality. Examples for such devices are ventilators, recuperators, humidifiers, heating or cooling units, and air filters. Some of these devices are already equipped with sensors for capturing energy consumption, temperature, revolution speed or other relevant parameters; other devices will be equipped with additional sensors during the development of the IoT solution. These sensors provide information that is evaluated on facility-level to determine energy consumption, anomalies or error situations.

ACT facilities can be grouped according to their functionality, which often corresponds to the type of buildings they were designed for and are installed in. The ACT groups relevant for the case study company are hospitals, manufacturing buildings, shopping centers and educational organizations. For all ACT facilities within a group, evaluation of the collected facility-level information is relevant during planning and operations, for example for dimension planning, i.e., deciding on the required performance class.

The use cases supported by AI solutions are summarized in table 2.

No.	**Use Case**	**Required AI solution**
1	Maintenance Support	Detection of unusual sensor values and anomalies based on rule set and knowledge base
2	Energetic Inspection	Detection of errors in control system, errors caused by the operators, energy consumption exceeding usual patterns
3	Contracting	Economic evaluation if repair of facilities and required spare parts are recommendable
4	Dimensioning of new facilities	Evaluation of historic data for categories of ACT facilities to recommend dimensioning of new facilities

Tab. 2: Use Cases and AI Solutions in case A

3.2 Development of Morphological Box

The analysis of the industrial cases for characteristics and features included the analysis of decision criteria for introducing AI, configurations of solutions, general functionalities and features of the application cases (see section Industrial Cases). As a result of the analysis, we identified the following features:

Feature	Values					
Computing source	cloud	local	device	hybrid		
Maturity	COTS	Commercial components	Open source components	Individual		
Special hardware required	Computing	Data capture	Data visualization	Data output		
Data source	Own data	Open data	Commercial collection	Synthetic data		
Data and model update frequency	Continuously	None	In case of changes in regulation	In case of quality problems		
Time to decision	Real-time	Near real-time	Several hours	Not time critical		
Primary data type	Audio	Video	Raster image	Vector image	Transaction records	Time series data
Primary purpose	assistance	decision making	forecasting	classification	anomaly detection	
AI use in solution development	Design time	Runtime	Accompanying runtime	Hybrid		
AI focus	Processing input	Generating output	Computing task			
Extent of effect on enterprise	Isolated solution	Effect on single process	Complete process	Work system	Business model	
Reliability and precision of results	100% required	Defined by enterprise	Defined by domain			

Fig. 1: Morphological box for introducing AI solution

- Computing source describes, where the data processing and execution of AI algorithms physically takes place. For use case B (fraud detection) it was important that this was the local computing center of the enterprise; for case D it was mandatory to do the computing in the cloud as some customers required this. Some applications in case A require computing on the actual device used in the process.
- Maturity denotes, to what extent the solution is a standard product or has to be developed individually for the requirements of the case. All cases preferred COTS products, which in most cases were not available. This is why collections of configurable components were accepted.
- Special hardware required describes the need for computing equipment or data capturing devices that deviate from established IT infrastructures in enterprises.
- Data source defines the origin of the data used for developing the model or training the algorithm underlying the solution. This could be publicly available data, such as in case C, private data from the enterprise (case A), synthetic data or mixed forms.
- Data and model update frequency describes the expected time between developing

the model or training the AI solution and having to update it due to changes in the domain.

- Time to decision describes, what the required time frame is until the result of the AI solution use (e.g., the decision it proposes) has to be available, i.e., the time between the business event triggers the solution use until the result is computed. In case A, the decision is needed in close to real-time, in case D there are scenarios where it would be sufficient several hours later.
- Primary data type describes on what data the AI solution's functionality is based on: audio, video, images, text, documents, time series, structures records, or any other kind of data.
- Primary purpose aims at characterizing the core functionality. From the cases, we identified assistance to human actors, decision-making, forecasting, classification and anomaly detection.
- Position of the AI functionality in the development process denotes whether the AI is used during design time of the overall solution, during run-time, accompanying run-time or hybrid.
- AI focus helps the prospective users to understand the type of AI. Here, the values would be functionalities for processing information input (speech processing, image processing, text processing, . . .), generating output (speech generation, text generation, image generation) or the actual computing task.
- Extent of effect on enterprise describes if the AI solution replaces only isolated tasks or machinery without affecting the neighboring processes, is changing complete processes, work systems or business functions, or is changing large parts of / the complete business model.
- Reliability and precision of the results of the AI solution: has it to be always 100% correct or could a lower level of precision and correctness also be acceptable?

4 Application of Features in AI Context Modeling

Based on the features of AI solutions identified in the morphological box, this section investigates if these features can help to improve the AI-context modeling method. As a first step, we walked through all method steps and their descriptions in search for potential applications for the features:

- in „Model organizational AI context" (step 1), the features were not applicable, as the focus here is on working with the current situation in the enterprise represented by the existing EA model. Features of the envisioned AI solution should be determined after the requirements are defined (step 2), not before.

- in „Elicit AI requirements“ (step 2), the list of AI solution features and the different values of the features seem to be useful. Some of the features can help to clarify frame conditions of the future AI solutions, which then again can lead to new requirements. Examples are „computing source“ and „data source“. If the enterprise wants the computing related to the AI solution to happen locally, the usage of existing cloud services has to be ruled out, which will lead to a higher complexity of the solution and in-house maintenance. If primarily open data are required for training purposes, the existence and suitability of such data sets needs to be investigated.
- in „Analyze AI context“ (step 3) and „Enrich the EA model“ (step 3a), the features are not very useful. The main reason is that detailed instructions exist on how to do the analysis or enrichment steps. An additional checklist or hints are not needed.
- in „Decide on feasibility“ (step 4), on the other hand, the features and their different values seem to be useful. Some features are relevant for the overall feasibility of the approach, such as time to decision and the required reliability and precision of the results. If real-time decisions are required but not possible due to the amount of data or complexity of computing, another solution than the AI-based approach might be required. If 100% correctness is mandatory but not possible due to missing information in the specification, more refinement work is required before deciding on feasibility.
- in „Designing the future EA“, the features can help to validate the initial design of new or changed data, application and business architectures. If 100% correctness is mandatory, the design of the solution has to be adapted accordingly.

The initial analysis of the method steps shows that two of five steps seem to have clear benefits from using the features (requirements analysis; feasibility analysis) and one more step (designing the future EA) probably also has advantages.

To further investigate the initial results, we examined for every feature and every case, in what method step (if at all) this feature was discovered or elaborated. Table 3 shows the result of this step.

Case / *Feature*	**A – ACT energy optimization**	**B – fraud detection**	**C – object recognition**	**D – forecast for operations**	**Recom-mended**
Computing source	Requirements Analysis	Feasibility Analysis	Design Future EA	Requirements Analysis	Requirements Analysis
Maturity	Requirements Analysis	Design Future EA	Design Future EA	Design Future EA	(depends on case)
Special hardware required	Requirements Analysis	Design Future EA	Design Future EA	Requirements Analysis	Requirements Analysis
Data source	Requirements Analysis	Feasibility Analysis	Feasibility Analysis	Requirements Analysis	Requirements Analysis
Data and model update frequency	Feasibility Analysis	Requirements Analysis	Feasibility Analysis	Requirements Analysis	Requirements Analysis
Time to decision	Requirements Analysis	Requirements Analysis	Requirements Analysis	Requirements Analysis	Requirements Analysis
Primary data type	Requirements Analysis	Before project start	Before project start	Requirements Analysis	Latest at Requirements Analysis
AI use in solution development	Requirements Analysis	Requirements Analysis	Before project start	Before project start	Latest at Requirements Analysis
AI focus	Before project start	Before project start	Before project start	Before project start	Before project start
Extent of effect on enterprise	Design Future EA	Feasibility Analysis	Design Future EA	Feasibility Analysis	(depends on case)
Reliability and precision of result	Requirements Analysis	Requirements Analysis	Feasibility Analysis	Feasibility Analysis	Requirements Analysis

Tab. 3: Analysis of Features in Use Case

The analysis shows that some features should have been elaborated in requirements analysis, but were only discovered in later method steps (e.g., computing source in cases B and C). Other features have to be known before the project starts and thus are not helpful for the method.

5 Summary and Conclusions

Starting from the observation of recurring features of established AI solutions, we began to develop a morphological box. The morphological box developed has a number of limitations and requires more work. We primarily based the development on only four industrial cases and the features visible in these cases or reflected in discussions with stakeholders related to these cases. More features probably will emerge when enriching the empirical base by additional cases, expert interviews or analysis of literature. The same is true for the values identified for the different features. We expect that the list of values is complete only for a few of the features. Furthermore, we need to analyze the features for interference between each other and between the values for completeness and potential overlaps.

The analysis of feature use in the AI context method steps showed that requirements analysis (step 2) and feasibility analysis (step 4) gain benefits from using the features. The benefits are a more complete set of requirements and, as a result, more efficient method steps. Furthermore, in the step of designing the future EA, inspiration can be gained from the features for what potential solutions could be envisioned. All this improvement potential has to lead to a minor revision of the method that includes the feature use in the instructions for the mentioned method steps. This will be part of future work.

Bibliography

[Ah12] Ahlemann, Frederik; Stettiner, Eric; Messerschmidt, Marcus; Legner, Christine: Strategic enterprise architecture management: Challenges, best practices, and future developments. Management for Professionals. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012 edition, 2012.

[Br19] Braun, Martin; Kutzias, Damian; Dukino, Claudia; Leuteritz, Jan-Paul: , Künstliche Intelligenz für die betriebliche Praxis: Leitfaden zur Umsetzung im Unternehmenskontext: Report im Projekt KI-Ultra, 2019.

[DSM19] Diadiushkin, Alexander; Sandkuhl, Kurt; Maiatin, Alexander: Fraud detection in payments transactions: Overview of existing approaches and usage for instant payments. Complex Systems Informatics and Modeling Quarterly, (20):72–88, 2019.

[dWR07] de Waal, A.; Ritchey, T.: Combining morphological analysis and Bayesian networks for strategic decision support. ORiON, 23(2):105–121, 2007.

[Ho16] Holtel, Stefan: Artificial intelligence creates a wicked problem for the enterprise. Procedia Computer Science, 99:171–180, 2016.

[JWW21] Jöhnk, Jan; Weißert, Malte; Wyrtki, Katrin: Ready or not, AI comes—an interview study of organizational AI readiness factors. Business & information systems engineering, 63(1):5–20, 2021.

[KG09] Kumar, J. Ajith; Ganesh, L. S.: Research on knowledge transfer in organizations: a morphology. Journal of knowledge management, 2009.

[Le11] Lee, Ji Hwan; Hong, Yoo S. et al.: A morphological approach to business model creation using case-based reasoning. In: DS 68-9: Proceedings of the 18th International Conference on Engineering Design (ICED 11), Impacting Society through Engineering Design, Vol. 9: Design Methods and Tools pt. 1, Lyngby/Copenhagen, Denmark, 15.-19.08. 2011. pp. 165–175, 2011.

[NP20] Niemi, Eetu; Pekkola, Samuli: The benefits of enterprise architecture in organizational transformation. Business & information systems engineering, 62(6):585–597, 2020.

[PL21] Park, Chankook; Lee, Daeyoun: Analysis on new types of electric power businesses using a morphological box. Energy & Environment, 32(1):113–133, 2021.

[Re21] Reiz, Achim; Albadawi, Mohamad; Sandkuhl, Kurt; Vahl, Matthias; Sidin, Dennis: Towards More Robust Fashion Recognition by Combining of Deep-Learning-Based Detection with Semantic Reasoning. In: AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering. 2021.

[Sa19] Sandkuhl, Kurt: Putting AI into context-method support for the introduction of artificial intelligence into organizations. In: 2019 IEEE 21st Conference on Business Informatics (CBI). volume 1, pp. 157–164, 2019.

[Sa22] Sandkuhl, Kurt; Shilov, Nikolay; Seigerroth, Ulf; Smirnov, Alexander: Towards the Quantified Product - Product Lifecycle Support by Multi-Aspect Ontologies. In (Yuval, Cohen, ed.): Proceedings 14th IFAC Workshop on Intelligent Manufacturing Systems. IFAC, 2022.

[Sj21] Sjödin, David; Parida, Vinit; Palmié, Maximilian; Wincent, Joakim: How AI capabilities enable business model innovation: Scaling AI through co-evolutionary processes and feedback loops. Journal of Business Research, 134:574–587, 2021.

[SN12] Storbacka, Kaj; Nenonen, Suvi: Competitive arena mapping: market innovation using morphological analysis in business markets. Journal of business-to-business marketing, 19(3):183–215, 2012.

[SR22] Sandkuhl, Kurt; Rittelmeyer, Jack: Use of EA Models in Organizational AI Solution Development. In (Aveiro, David; Proper, Henderik A.; de Vries, Marne; Guerreiro, Sergio, eds): Advances in Enterprise Engineering XV. LNBIP. Springer, 2022.

[SSL14] Seidenstricker, Sven; Scheuerle, Stefan; Linder, Christian: Business model prototyping–Using the morphological analysis to develop new business models. Procedia-Social and Behavioral Sciences, 148:102–109, 2014.

[Zw69] Zwicky, Fritz: Discovery, invention, research through the morphological approach. 1969.

J. Michael, J. Pfeiffer, A. Wortmann (Hrsg.): Modellierung 2022 Satellite Events,
Digital Library, Gesellschaft für Informatik e.V. 30

Workshop on the State of the Art Methods and Tools in Model-Based Systems Engineering

Wolfgang Böhm [1] Nico Jansen, David Schmalzing [2]

Preface

The notion of model-based systems engineering (MBSE) is an increasingly applied area in both research and industry. It envisions a transition from document-based development of cyber-physical systems (CPSs) to a model-driven approach using models as the primary engineering artifacts. Modern systems become more and more complex and involve an increasing number of different domains. Engineering in this interdisciplinary context is a crucial competitive factor and demands efficient and sustainable CPS development. This cross-domain development requires the communication and agile coordination of experts from different disciplines, who are responsible for individual sub-areas but also must organize their work in the context of the entire system. MBSE tackles these challenges with the vision of seamlessly integrated system modeling, in which experts of various domains contribute to a single source of truth using domain-specific modeling techniques.

Overcoming the challenges in interdisciplinary CPS development is essential for successfully applying MBSE. Collaboration often suffers from insufficient tool support or a lack of semantics for the employed modeling languages. While abstraction, composition, and refinement techniques have already been intensively researched, they still find little application in practice. Domain experts should be methodically guided by the modeling tool to enable consistent and concise development. This goal requires unambiguous semantics of a language to clarify the meaning of a model, even in an interdisciplinary context.

This workshop aims to bring together people involved in model-based systems. In particular, through this workshop, we intended to promote the exchange between industry and research and, by linking theory and practice, exchange knowledge and experience and discuss the application of MBSE methods. Therefore, relevant topics for this workshop were experiences and challenges of applying MBSE tools, methods, and analyses, and efforts to conceive tools and methods to support MBSE.

[1] Technische Universität München, Institut für Informatik - Software & Systems Engineering, Boltzmannstraße 3, 85748 Garching, Germany, boehmw@in.tum.de

[2] RWTH Aachen, Lehrstuhl für Software Engineering, Ahornstraße 55, 52074 Aachen, Germany, {jansen/schmalzing}@se-rwth.de

For presentation during the workshop, we accepted six papers highlighting different aspects of system modeling. Contributions include the development of tools and methods to support MBSE, as well as experience and evaluation reports on applying these. In addition, we invited Prof. Dr. Andreas Vogelsang to a keynote presentation and two talks from the industry to present their efforts and achievements in applying model-based system engineering methods.

We thank Prof. Dr. Andreas Vogelsang for offering to give a keynote presentation and all authors for the careful elaborations and reprocessing of their results. In addition, we thank the program committee members for reviewing the contributions and providing feedback to the authors. Finally, we thank the organizers of the conference and the workshops for their effort in planning the event.

Program Committee: Andreas Bayha (fortiss), Michael Jastram (Formal Mind GmbH), Maximilian Junker (Qualicen GmbH), Walter Koch (Schaeffler AG), Jan Philipps (foqee GmbH), Andreas Vogelsang (University of Cologne), Sebastian Voss (fortiss)

Embedding Textual Languages in MagicDraw[1]

Florian Drux, Nico Jansen, Bernhard Rumpe, David Schmalzing [2]

Abstract: Stakeholders in model-based systems engineering projects often rely on heterogeneous modeling languages and tools. Efficient and seamless model-based engineering requires analyzing consistency, maintaining tracing information, and propagating changes of models of these languages even in different technological spaces. However, research on software language integration and inter-model consistency often only considers modeling languages and tools within the same technological space. We present a method for language composition across the technological spaces of the graphical modeling framework MagicDraw and the language workbench MontiCore. We realized language integration between these technological spaces by applying concepts of language aggregation to exchange essential model information and performing analysis on this information in an automated toolchain. The presented concepts can guide software language engineers and modeling tool developers on how to combine concepts of language composition to bridge technological spaces.

Keywords: language engineering; language composition; tool integration; model-based engineering

This work was supported by the German Ministry for Education and Research (BMBF) in the SpesML project (https://spesml.github.io/index.html/; grant number 01IS20092B).

1 Introduction

The development of large, complex, model-based systems [Ni15] necessitates the collaboration of experts from different fields, which may use a variety of modeling languages and tools. Managing intermodel consistency and supporting interpreting models together across the boundaries of technological spaces is crucial for seamless model-based engineering but costs a lot of effort and is prone to errors if not done automatically. Software language engineering [Kl08] has given rise to various language composition mechanisms [VV10, HRW18], of which language aggregation enables models of different languages to be interpreted together. However, these mechanisms are mostly constipated for modeling languages originating from the same technological space [GVM12], whereas domain experts may operate in different technological spaces. Furthermore, software languages and workbenches come in different shapes as they may be textual, graphical, or projectional.

Aggregating languages of different shapes and across technological spaces can facilitate automated and seamless model analyzes in model-based systems engineering projects.

[1] This work was supported by the German Ministry for Education and Research (BMBF) in the SpesML project (https://spesml.github.io/index.html/; grant number 01IS20092B).

[2] RWTH Aachen, Lehrstuhl für Software-Engineering, Ahornstraße 55, 52074 Aachen, Germany, {drux/jansen/rumpe/schmalzing}@se-rwth.de

We, therefore, investigate the challenges of integrating textual and graphical languages between different technological spaces at the example of the graphical modeling framework MagicDraw [3] and the language workbench MontiCore [4]. We do this by making essential model information explicit that can then be exchanged between languages. Using the example of embedding textual expressions into graphical state machines, we then show how this model information can be utilized in modeling tools to analyze inter-model conformity automatically. The presented results can guide language engineers on how to combine language embedding with language aggregation to systematically develop a shared understanding of models from different technological spaces.

In the remainder, we illustrate the challenges of language integration across technological spaces in section 2. We then present the background of MontiCore and MagicDraw in section 3 before we then present a systematic concept for language embedding in MagicDraw through essential model information in section 4. Next, we explore language embedding using the example of embedding textual expression languages into graphical state machines in MagicDraw in section 5. Finally, we present related work on language composition and tool integration in section 6 and discuss our solution in section 7.

2 Challenges

Integrating textual with graphical modeling languages across heterogeneous technological spaces poses several challenges. The different language constituents must be embedded appropriately, capable of interacting with each other and provided to the user in a convenient form. With respect to ISO 25010 [IS10], the norm for software product quality, the occurred challenges are related to functional suitability, compatibility, and usability.

Functional suitability refers to providing the required functionalities in the product. For embedding textual languages in graphical MagicDraw diagrams, this involves enabling the modeler to seamlessly define graphical and textual portions in the MagicDraw editor, which are automatically checked for validity. An integrated editing environment is essential to prevent the modeler from switching back and forth between multiple tools, which could disturb the overall modeling effort. Compatibility basically means the ability of a software system to interoperate with others. This is particularly important in our case, as the main task is to connect two different technological spaces. Thus, model elements of both spaces must be able to interact with each other, and well-formedness must be checked on the integrated (i.e., the entire) model. Usability refers to the applicability of a product to the end-user, considering usability, intuitiveness of functionalities, and general user interface. Accordingly, textual languages must be seamlessly integrable into graphical model elements. Furthermore, the modeler requires direct feedback on the well-formedness of the modeled elements and support for the modeling process itself.

[3] https://www.3ds.com/products-services/catia/products/no-magic/magicdraw/
[4] https://monticore.de/

This leads to the following overall challenges:

C1 Textual model elements must be editable and storable in MagicDraw's graphical editor.

C2 Textual model parts must be checkable for correctness with respect to the context of the graphical model components.

C3 Graphical model parts must be referencable via textual qualifiers.

C4 Textual model parts and corresponding functionalities must be embedded seamlessly into MagicDraw.

C5 The user requires direct feedback on modeled elements and modeling support.

Since this paper focuses on embedding textual languages into MagicDraw, functional suitability (C1) and compatibility (C2, C3) are mandatory goals. While usability (C4, C5) often cannot be determined objectively and requires user studies, we can at least qualitatively evaluate whether these challenges have been addressed appropriately.

3 Preliminaries

MontiCore [HKR21] is a workbench and a framework for engineering modular textual modeling languages from context-free grammars (CFGs), which define a language's concrete and abstract syntax. From a CFG, MontiCore generates the basic infrastructure for the engineering of modeling languages, which includes a parser, a symbol table, model consistency and model transformation infrastructure, and a common workflow. Language engineers can implement language specifics by filling provided extension points and extending the generated infrastructure. The common workflow of modeling languages engineered with MontiCore starts with parsing a textual model to construct its abstract syntax tree, deriving a model's symbol table, performing handcrafted context-condition checks, and serializing essential model information in exchangeable artifacts. To facilitate modular development, MontiCore provides syntax-focused mechanisms for composing language constituents. A provided library of literals, expressions, and types further facilitates language composition [Bu20].

MagicDraw is a graphical collaborative and extendable modeling tool that implements various standardized languages, including the UML, SysML, and BPMN. In MagicDraw, profiling enables language engineers to adapt predefined language profiles and create custom and specialized languages on top of the provided profiles. MagicDraw furthermore enables customization of modeling language-specific tools by supporting the development of custom plugins. MagicDraw provides an open development API whose code can be reused, extended, and modified, to facilitate profiling and plugin development.

4 Embedding Textual Languages

Besides the graphical modeling capabilities, MagicDraw offers a set of predefined textual languages. These include scripting languages such as Groovy or JavaScript and a specific OCL dialect. Additionally, MagicDraw supports structured expressions, which are assembled in a tree-like fashion via the editor and are encoded in XML. Nevertheless, the possibilities for textual modeling are severely limited. The existing textual languages have only rudimentary access to the elements of the containment tree in MagicDraw, and thus elements can often only be referenced very unintuitively via detours. Otherwise, there is only the possibility of inserting plain text. However, the resulting string is neither checked in itself nor concerning the context of other model elements. An integrated language requires a seamless connection between graphical and textual modeling. Furthermore, MagicDraw does not provide any direct hook points for mounting additional textual languages besides the existing ones.

Therefore, to embed an external, textual language into an otherwise graphical DSL in MagicDraw, it must be integrated in such a way that it: (1) accepts only the allowed set of syntactic sentences, and (2) these are validated for well-formedness concerning their context. To meet the first requirement, we have to integrate a parser for the textual language into MagicDraw. The basic idea is to use MagicDraw's plain text panels to enable textual modeling. For this purpose, we can either reuse existing plain text fields or create new fields for the MagicDraw language under development by creating them in the corresponding metamodel. In both cases, we have to bind a parser to the plain text field for validation. This parser can be provided externally by using appropriate technology to create textual domain-specific languages, such as MontiCore [HKR21]. Here we provide a context-free grammar that defines the concrete and abstract syntax of the textual language and automatically generates a parser for its models. MagicDraw allows adding customized validation rules for model elements. These rules are checked at modeling time and return an error message to the modeler in case of a violation. We exploit this extension point by specifying a validation rule for the designated plain text field for embedding the textual language. The input string is forwarded to the parser, which converts it to an AST, or in case of erroneous input, returns an error message, which in turn is displayed to the modeler.

The second requirement must be validated on the created AST (thus, after parsing the textual input), checking if the syntactically correct model is also well-formed. Therefore, the nodes of the textual model's AST must be checked against the other elements, both textual and graphical. The particular characteristic of the well-formedness rules is highly dependent on the language. Thus, in the following, we discuss the different types of validation rules, as well as a general approach for addressing these. Generally, these rules arise from known context conditions for modeling languages [HKR21] but pose a special challenge in the case of language embedding and aggregation across different technological spaces. First, basic direct checks exist on an AST node, such as whether the name of an element begins with an uppercase letter. Furthermore, it can be checked whether certain elements can be referenced in a textual model, such as variables or method calls. This also includes checking whether these elements are accessible and, e.g., in the case of methods, the signature of the definition fits the signature applied in the usage. Finally, the types of the referenced elements must be compatible with each other.

While basic validation rules on a few AST nodes of the textual model can usually be implemented easily, references between model elements pose a major challenge, especially for references between the different technological spaces. To resolve these references, programming and textual modeling languages introduce the concept of the symbol table [HMSNR15, Ha15]. A symbol table is an infrastructure that stores uniquely named model elements (so-called symbols) and enriches them with additional information. It elevates the tree structure of the AST to a graph structure allowing for quick navigation and cross-referencing. To achieve referencing of graphical model elements in embedded textual language models, we extend this symbol table concept to the entire containment tree in MagicDraw. We traverse the entire containment tree and create a symbol for each model element uniquely identifiable by its qualified name. Here, we opt for the application of the Visitor Pattern, which is supported by in MagicDraw. For each relevant element, a corresponding symbol is created.

Depending on the element's type, we instantiate a symbol of a respective kind. Thus, the symbol for, e.g., an attribute is distinct from a method's as they need to provide different information and should be referenceable separately. The precise assignment of symbols is language-specific, although there are common candidates, such as (object-oriented) type definitions, fields, functions, or methods. After creating the symbols, they are bound to their model elements (e.g., via a map). Sophisticated language workbenches such as MontiCore follow a more integrated approach between AST nodes and symbols. In the case of MagicDraw, we have opted for a more straightforward binding mechanism for simplicity. Ultimately, the referenced symbols can be resolved, thus accessing the heterogeneous model elements throughout the entire containment tree. Well-formedness rules for referencing other model elements can be added via a validation rule in MagicDraw, similar to the parser.

Finally, the implemented validation rules can be packed into a MagicDraw plugin together with the customized language and the embedded parser. Since MagicDraw is based on the Java programming language, it is useful to rely on compatible technologies for parsing and symbol table creation.

5 Case Study

To evaluate our concept for tool interoperability, we extend MagicDraw's graphical state machines with textual expressions. Our goal is to use state machines to model the behavior of distributed systems. To this end, we want to interpret state machines together with internal block diagrams and block definitions. Unfortunately, while MagicDraw offers behavior descriptions, these don't fulfill the specific semantics we require. Using strongly typed expressions consisting of boolean operations, variable assignments, and method calls, we want to enable modelers to define transition triggers and effects that can reason over port and variable values.

MagicDraw's API enables us to embed expressions into transition guards and effects on a syntactical level. Using an expression parser provided by MontiCore's library of literals, types, and expressions and attaching it to a plain text panel in MagicDraw's state machines, we can now parse plain text input and derive their abstract syntax tree. While this provides us with at least the most basic checks for ensuring correct concrete syntax, we can not yet check the well-formedness of expressions with respect to the availability of referenced symbols and type conformity; that is, we can not yet check inter-model well-formedness.

To enable well-formedness checks between models, we need to make the available system elements, ports, variables, and types known to the type system. However, here two problems arise. First, MagicDraw does not provide a straightforward way to export such information; secondly, and more importantly, type access differs significantly for graphical and textual languages. The graphical framework MagicDraw stores elements in a containment tree and enables access to elements through direct object inks, whereas textual languages manage access to symbols either through namespaces or scopes, with MontiCore supporting the latter. That is, symbols are available in certain scopes, and access to elements is provided through their fully qualified name. Not all symbols in scope are available from the outside, though, as scopes may only export certain symbol kinds. Furthermore, a symbol may shadow another symbol of the same name in another scope.

Using the above-presented concept for language aggregation, we make symbol information explicit to type check on expressions and thus bridge the gap between the technological spaces of MagicDraw and MontiCore. We do this by traversing the containment tree using a visitor that collects element information along the way. We then synthesize, aggregate, and serialize the symbol table of the stored model elements from this information. While the serialized symbol table functions as a pivot model between the two technological spaces, it is pruned down to the essential model information for conformity checks.

To integrate well-formedness and type checks provided by MontiCore, we rely on the MagicDraw's validation suite, which provides validation rules which can be applied to model elements to check conformity. Filtering model elements for textual elements of state machines, we can use these to check the conformity of the embedded expressions. First, the embedded parser receives the abstract syntax tree. Then, the type check is applied to the abstract syntax tree, consuming aggregated symbol information to check well-formedness. Usages of names must be available in the symbol table as names of ports and fields. Their symbols provide information about their type, visibility, and access rights. Symbol information is made visible to the scope of an expression through import statements, with the containment tree hierarchy of elements defining their fully qualified name. The type check then consumes this information to check the conformity of the left-hand and right-hand sides of expressions, or that expressions in transition guards evaluate to `Boolean`.

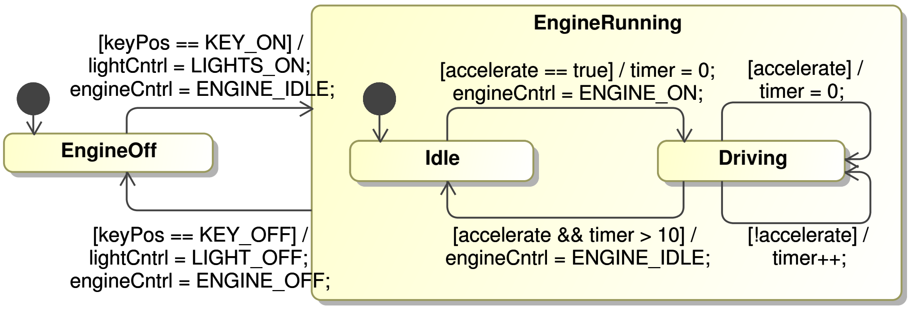


Fig. 1: A visual MagicDraw model of a car integrating textual model elements of our language both inside the rectangular brackets and after the slashes; The car may be turned on or off. When accelerating with the engine in its idle state the engine is turned on. After not accelerating for a while, the engine returns to its idle state.

Example: In Fig. 1 in the top left the guard `keyPos == KEY_ON` can be seen. When validating the model, the provided text is parsed to an AST with an equality expression containing two names. Afterward, referenced names are resolved in the symbol table. As both symbols have the same type, they are comparable, and thus the type of the expression is of type `Boolean`.

To report on ill-formed elements, we use the annotations-Application Programming Interface (API) provided by MagicDraw. Error messages are annotations on elements which are then displayed in the Graphical User Interface (GUI). By adding the annotations-API, MagicDraw will highlight annotated elements as erroneous. In addition, the errors are also listed in an additional window of GUI, which results from fetching the error output of the embedded expression language.

6 Related Work

Most related work on language composition only considers homogeneous technological spaces, e. g., languages developed in a common language workbench. MontiCore supports systematic, syntax-oriented reuse of individually developed language components through language aggregation, language embedding, and language inheritance [KRV10, Lo13, Ha15]. These mechanisms mostly enable language integration as defined in [EGR12]. Furthermore, language aggregation enables to combine independent modeling languages such that models of varying aspects defined in different artifacts can be interpreted together. Serialization and deserialization of model essentials provides a basic foundation for integration of languages of heterogeneous technological spaces. However, MontiCore only focuses on syntactic language composition and does not provide support for composing semantics or development environments.

The language workbench MPS [Vo13] supports language composition of concrete syntax with mechanisms similar to MontiCore, though they are called differently. In addition to syntax-oriented reuse of languages, MPS also addresses modularity of type systems, generators, and development environments. However, mechanisms that enable composition of modeling languages of heterogeneous technological spaces, such as language aggregation, are missing in MPS.

Another approach [Se17] describes integrating the textual modeling language Alf [Se14] into MagicDraw UML diagrams. As an action language, a modeler can use Alf code snippets to specify behavior. For instance, the operation body of a method could be written in Alf, or smaller snippets could be used in conjunction with graphical UML behavior diagrams (such as activity diagrams or state machines). In general, this concept is similar to our work. A textual language is also embedded in MagicDraw, which is evaluated context-sensitively with respect to the graphical model elements. The main difference lies in the actual application, as the described Alf plugin adds additional modeling capability and does not, as in our case, formalize existing model elements syntactically through a language. Furthermore, the article describes the features and benefits of the Alf plugin rather than providing a general concept for embedding textual languages in MagicDraw.

Generally related are various works in the area of tool integration. Highly heterogeneous models are linked together, and information is exchanged between them. For example, the FTG+PM framework [Mu12] defines model transformations as graph structures to automatically pass information between different representations. Furthermore, there are attempts to represent information of different models via knowledge base systems [Fe15], via which rules for the consistency of information can then be derived and inferred. Besides many other works describing the linking of information across heterogeneous models and tool landscapes [Br10, Ch15, ZLVH18, DDFV09, SWA10, Mi09, CMM09, Zi07, Ba10], these approaches are usually only loosely related to our solution. While generally, an aggregation of models and their information takes place, our approach describes a concrete embedding in the MagicDraw modeling platform. Consequently, elements do not have to be exchanged between different tools but are directly implemented together in an integrated environment.

7 Discussion

Our presented approach of embedding textual languages into graphical languages is based on the modeling framework MagicDraw. For designing and integrating the textual language, we rely on the MontiCore language workbench because it is based on the same technological space (i.e., Java) and highly fosters the use of a symbol management infrastructure, which is heavily used in our solution. The particular challenge in our work was not only to make a textual language accessible within a predominantly graphical modeling environment but also to integrate it seamlessly into the graphical model. This means that the model must be not only syntactically correct in itself but also well-formed under the context of the graphical model elements. Table 1 presents the evaluation results of our approach concerning the initially identified challenges (cf. section 2).

Tab. 1: Fulfillment of embedding challenges for textual languages in MagicDraw (• = fulfilled, ⊙ = partially fulfilled, ∘ = not fulfilled)

Challenge	Description	Fulfilled
C1	Textual model elements must be editable and storable in MagicDraw's graphical editor.	•
C2	Textual model parts must be checkable for correctness with respect to the context of the graphical model components.	•
C3	Graphical model parts must be referencable via textual qualifiers.	⊙
C4	Textual model parts and corresponding functionalities must be embedded seamlessly into MagicDraw.	•
C5	The user requires direct feedback on modeled elements and modeling support.	⊙

Generally, our approach extends language composition techniques known from textual languages and applies these to the graphical modeling space of MagicDraw (**C1**). Using the symbol table to resolve elements by their qualified name enables the modeler to cross-reference elements over the boundaries of the textual model. By integrating the resolve mechanism into a custom validation suite, we can check the well-formedness of the textual elements automatically (**C2**). However, the overall referencing could be seen as incomplete from a graphical modeling point of view, as not all parts are uniquely accessible by their names (**C3**). In MagicDraw, all model elements are theoretically accessible independent of their name. Referencing is accomplished via internal identifiers enabling arbitrary cross-referencing. Therefore, both ways of referencing are not fully compatible, resulting in potential conflicts and ambiguities. However, since internal identifiers are unintuitive in textual modeling and we can usually assume that relevant model elements have meaningful (and uniquely qualified) names in graphical modeling as well, we decided to follow this approach. A potential extension could combine these concepts by presenting the textual representation to the user while simultaneously managing MagicDraw's identifiers in the symbol table.

Finally, we primarily concentrated on the general feasibility and interoperability of the languages' constituents in an integrated fashion (**C4**). Accordingly, there is still a considerable lack of editor functionalities for the embedded textual language, such as syntax highlighting or autocompletion. However, by integrating validation rules, we can provide instant feedback, which satisfies at least the most rudimentary requirements of this challenge (**C5**). The integration of these functionalities, for example, via the language server protocol [BK19], still needs to be explored. In this context, extending the integration regarding refactorings of the graphical elements (such as renaming or deletion) and automatically tracing these changes into the textual model parts is important. Currently, the integrated validation rules would directly detect the results of such modifications but without automated adaptations.

Our solution demonstrates the integration of textual MontiCore languages into the predominantly graphical modeling environment of MagicDraw. While the approach is specific to both technological spaces, it also provides a foundation for generally integrating textual languages into graphical tools. First, although strongly established by MontiCore, the general concept for building a symbol table, which maps individual model elements to uniquely resolvable symbols, can be applied commonly. Moreover, the realization approach can also be considered generalizable as it is based on the prevalent visitor pattern. Here, we assume that most modeling tools manage the model internally in a tree-like construct, which immensely benefits the use of this pattern. Otherwise, the underlying technology stack and the supported extensibility of the tools used are crucial for the integrability of different languages. Our approach benefits heavily from the fact that MagicDraw supports Java extensions, and MontiCore can easily fill this hook point. However, for too diverse or hardly extensible tools, realizing such integration can be difficult.

8 Conclusion

We presented a concept for embedding textual modeling languages in graphical languages of MagicDraw, discussing different challenges for usability and interoperability. At its core, the required infrastructure to resolve textual references on graphical model elements and vice versa is elaborated. Additionally, we presented a case study, realizing our concept based on context-free grammars written in MontiCore. We integrated a textual language in consideration of mutual access to the different technological spaces by leveraging integrated symbol tables, thus bridging the gap not only between heterogeneous languages but also tools. Ultimately, our approach provides a contribution to fostering a seamless transition between graphical and textual modeling.

Bibliography

[Ba10] Baumgart, Andreas: A Common Meta-Model for the Interoperation of Tools With Heterogeneous Data Models. Citeseer, 2010.

[BK19] Bünder, Hendrik; Kuchen, Herbert: Towards Multi-editor Support for Domain-Specific Languages Utilizing the Language Server Protocol. In: International Conference on Model-Driven Engineering and Software Development. Springer, pp. 225–245, 2019.

[Br10] Brecher, C; Özdemir, D; Feng, J; Herfs, W; Fayzullin, K; Hamadou, M; Müller, AW: Integration of Software Tools with Heterogeneous Data Structures in Production Plant Lifecycles. IFAC Proceedings Volumes, 43(4):48–53, 2010.

[Bu20] Butting, Arvid; Eikermann, Robert; Hölldobler, Katrin; Jansen, Nico; Rumpe, Bernhard; Wortmann, Andreas: A Library of Literals, Expressions, Types, and Statements for

Compositional Language Design. Special Issue dedicated to Martin Gogolla on his 65th Birthday, Journal of Object Technology, 19(3):3:1–16, October 2020. Special Issue dedicated to Martin Gogolla on his 65th Birthday.

[Ch15] Chen, DeJiu; Maffei, Antonio; Ferreirar, João; Akillioglu, Hakan; Khabazzi, Mahmood R; Zhang, Xinhai: A Virtual Environment for the Management and Development of Cyber-Physical Manufacturing Systems. IFAC-PapersOnLine, 48(7):29–36, 2015.

[CMM09] Crnković, Ivica; Malavolta, Ivano; Muccini, Henry: A Model-Driven Engineering Framework for Component Models Interoperability. In: International Symposium on Component-Based Software Engineering. Springer, pp. 36–53, 2009.

[DDFV09] Didonet Del Fabro, Marcos; Valduriez, Patrick: Towards the Efficient Development of Model Transformations Using Model Weaving and Matching Transformations. Software & Systems Modeling, 8(3):305–324, 2009.

[EGR12] Erdweg, Sebastian; Giarrusso, Paolo G.; Rendel, Tillmann: Language Composition Untangled. In: Proceedings of the Twelfth Workshop on Language Descriptions, Tools, and Applications. LDTA '12, Association for Computing Machinery, New York, NY, USA, 2012.

[Fe15] Feldmann, Stefan; Herzig, Sebastian JI; Kernschmidt, Konstantin; Wolfenstetter, Thomas; Kammerl, Daniel; Qamar, Ahsan; Lindemann, Udo; Krcmar, Helmut; Paredis, Christiaan JJ; Vogel-Heuser, Birgit: Towards Effective Management of Inconsistencies in Model-Based Engineering of Automated Production Systems. IFAC-PapersOnLine, 48(3):916–923, 2015.

[GVM12] Giachetti, Giovanni; Valverde, Francisco; Marín, Beatriz: , Interoperability for Model-Driven Development: Current State and Future Challenges, 2012.

[Ha15] Haber, Arne; Look, Markus; Mir Seyed Nazari, Pedram; Navarro Perez, Antonio; Rumpe, Bernhard; Völkel, Steven; Wortmann, Andreas: Integration of Heterogeneous Modeling Languages via Extensible and Composable Language Components. In: Model-Driven Engineering and Software Development Conference (MODELSWARD'15). SciTePress, pp. 19–31, 2015.

[HKR21] Hölldobler, Katrin; Kautz, Oliver; Rumpe, Bernhard: MontiCore Language Workbench and Library Handbook: Edition 2021. Aachener Informatik-Berichte, Software Engineering, Band 48. Shaker Verlag, May 2021.

[HMSNR15] Hölldobler, Katrin; Mir Seyed Nazari, Pedram; Rumpe, Bernhard: Adaptable Symbol Table Management by Meta Modeling and Generation of Symbol Table Infrastructures. In: Domain-Specific Modeling Workshop (DSM'15). ACM, pp. 23–30, 2015.

[HRW18] Hölldobler, Katrin; Rumpe, Bernhard; Wortmann, Andreas: Software Language Engineering in the Large: Towards Composing and Deriving Languages. Computer Languages, Systems & Structures, 54:386–405, 2018.

[IS10] ISO/IEC: ISO/IEC 25010 System and Software Quality Models. Technical report, International Standardization Organization (ISO), Madrid, Spain, 2010.

[Kl08] Kleppe, Anneke: The Field of Software Language Engineering. In: International Conference on Software Language Engineering. Springer, pp. 1–7, 2008.

[KRV10] Krahn, Holger; Rumpe, Bernhard; Völkel, Stefen: MontiCore: a Framework for Compositional Development of Domain Specific Languages. International Journal on Software Tools for Technology Transfer (STTT), 12(5):353–372, September 2010.

[Lo13] Look, Markus; Navarro Pérez, Antonio; Ringert, Jan Oliver; Rumpe, Bernhard; Wortmann, Andreas: Black-box Integration of Heterogeneous Modeling Languages for Cyber-Physical Systems. In: Globalization of Modeling Languages Workshop (GEMOC'13). volume 1102 of CEUR Workshop Proceedings, 2013.

[Mi09] Milanovic, Nikola; Cartsburg, Mario; Kutsche, Ralf; Widiker, Jürgen; Kschonsak, Frank: Model-Based Interoperability of Heterogeneous Information Systems: An Industrial Case Study. In: European Conference on Model Driven Architecture-Foundations and Applications. Springer, pp. 325–336, 2009.

[Mu12] Mustafiz, Sadaf; Denil, Joachim; Lúcio, Levi; Vangheluwe, Hans: The FTG+PM Framework for Multi-Paradigm Modelling: An Automotive Case Study. In: Proceedings of the 6th International Workshop on Multi-Paradigm Modeling. pp. 13–18, 2012.

[Ni15] Nielsen, Claus Ballegaard; Larsen, Peter Gorm; Fitzgerald, John; Woodcock, Jim; Peleska, Jan: Systems of Systems Engineering: Basic Concepts, Model-Based Techniques, and Research Directions. ACM Computing Surveys, 48(2), September 2015.

[Se14] Seidewitz, Ed: UML With Meaning: Executable Modeling in Foundational UML and the Alf Action Language. In: Proceedings of the 2014 ACM SIGAda Annual Conference on High Integrity Language Technology. pp. 61–68, 2014.

[Se17] Seidewitz, Ed: A Development Environment for the Alf Language within the MagicDraw UML tool (Tool Demo). In: Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering. pp. 217–220, 2017.

[SWA10] Seifert, Mirko; Wende, Christian; Aßmann, Uwe: Anticipating Unanticipated Tool Interoperability using Role Models. In: Proceedings of the First International Workshop on Model-Driven Interoperability. pp. 52–60, 2010.

[Vo13] Voelter, Markus: Language and IDE Modularization and Composition with MPS. In (Lämmel, Ralfand Saraiva, João; Visser, Joost, eds): Generative and Transformational Techniques in Software Engineering IV: International Summer School, GTTSE 2011, Braga, Portugal, July 3-9, 2011. Revised Papers. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 383–430, 2013.

[VV10] Völter, Markus; Visser, Eelco: Language Extension and Composition with Language Workbenches. In: Proceedings of the ACM International Conference Companion on Object Oriented Programming Systems Languages and Applications Companion. OOPSLA '10, Association for Computing Machinery, New York, NY, USA, p. 301–304, 2010.

[Zi07] Ziemann, Jörg; Ohren, Oddrun; Jäkel, Frank-Walter; Kahl, Timo; Knothe, Thomas: Achieving Enterprise Model Interoperability Applying a Common Enterprise Metamodel. In: Enterprise Interoperability, pp. 199–208. Springer, 2007.

[ZLVH18] Zou, Minjie; Lu, Boyang; Vogel-Heuser, Birgit: Resolving Inconsistencies Optimally in the Model-Based Development of Production Systems. In: 2018 IEEE 14th International Conference on Automation Science and Engineering (CASE). IEEE, pp. 1064–1070, 2018.

J. Michael, J. Pfeiffer, A. Wortmann (Hrsg.): Modellierung 2022 Satellite Events,
Digital Library, Gesellschaft für Informatik e.V. 44

Development of a SysML Profile for Network Configurations in Safety-critical Systems

Andreas Hemmert[1] Andreas Schweiger[2]

Abstract: The contribution describes an approach for defining avionics network architectures in an existing system model. To this end, a SysML profile is developed containing stereotypes to specify such a network. The corresponding model captures the network's configuration. Such a configuration defines a network of a safety-critical complex system. As a result, systems engineers can use the profile in the systems development process with digital continuity. An example demonstrates the application of the profile in the model development.

Keywords: Avionics; Systems Engineering; Software Engineering; Model-based Development; Requirements; Qualification; Certification; Simulation; Process; Tool; Platform; Architecture; SysML

1 Introduction

1.1 Context

For the development of safety-critical complex systems highest quality standards are required. Applying these, improving productivity, and reducing development time are of uttermost importance in the limited avionics market with an increasing quantity of competitors. Therefore, creating digital twins of their products leads to the reduction of the time to market. Digital twins capture the digital representation of a real product and relating product information [Ku17]. These require models, that are exchangeable between different engineering disciplines. An essential key for digital twins is Model-based Systems Engineering (MBSE). MBSE means the central data management of all product-related information in models created during the systems development process [Ei14]. Consequently, a standardized modeling language is needed to support this approach and to create tool-independent models [Wa16].

OMG's SysML™ can be regarded as an enabler for MBSE, while taking into account the drivers mentioned above [Obb]. These language requirements are met by SysML: "The system model expressed in SysML provides a cross-disciplinary representation to enable integration with other engineering models and tools" [Obb]. Basically, SysML is an extension of a subset of elements and diagrams defined in the Unified Modeling Language (UML)

[1] Airbus Defence and Space GmbH, Manching, Germany, andreas.hemmert@airbus.com
[2] Airbus Defence and Space GmbH, Manching, Germany, andreas.schweiger@airbus.com

standard [Obj17b]. UML defines elements and diagrams used for software development. SysML targets the discipline of systems engineering, while reusing metaclasses of UML. The stereotypes, diagrams, and relations between metaclasses of the UML extension are defining the SysML metamodel [Obj19].

In the context of the presented systems development approach, a system model is created using SysML. The needs of the system are analyzed and necessary system components are designed. These components are communicating via networks. The network architecture is specified in parallel to the system model. Currently no data continuity exists between the system model and the network architecture model. Based on the previous statement of integrating models using SysML, the connection of the system model with the network architecture model seems to be achievable. Furthermore, the integration of these discipline models into a single one is an essential step towards developing a digital twin. In order to achieve the integration of the network architecture model into the system model, analyses have to be performed. They have to reveal, whether SysML (v1.6) is providing suitable means (i.e. SysML stereotypes) required for network configurations.

1.2 Research Question

As described in Sec. 1.1, the system model is developed for a safety-critical complex system. However, when it comes to network configurations, no data continuity to system models exists. In order to integrate the network architecture and the configuration of the communication into the system model, existing SysML capabilities are analyzed. As a result, the research question "What is the gap of elements and properties between SysML standard v1.6 and the required classes for network configuration?" will be answered [He20]. The taken approach and results are described in this contribution.

1.3 Related Work

Shames and Sarrel developed a pattern for defining network architectures [SS15]. The approach for defining communication ports for each network participant for each applicable Open Systems Interconnection (OSI) Layer is required for defining the network architecture. While Shames and Sarrel are providing means for defining network architectures, the ones for configuring a network are missing. Thus, our contribution finds suitable means for network configurations with SysML.

2 Selection of Network Architecture Classes

2.1 Method for Developing Network Architectures

Because SysML is offering many redundant system development means, a method has to be developed, which selects suitable stereotypes for developing the network architecture. Furthermore, such a method needs to define the model element organization of the system model. In addition, it has to specify the dedicated usage of SysML elements for the relevant steps of the development process. Two basic principles for network architecture development are found in literature and used in the approach of this contribution: **(1)** Separation into views: A network architecture can be defined by the requirements, functional, logical, and physical aspects (i.e. views or viewpoints) [St18, Bo21]. According to ISO 42010 [ISO11], a view is defined as a "work product expressing the architecture of a system from the perspective of specific system concerns" [ISO11, p. 2]. The view principle can also be seen as a pattern: Dividing a design into views offers the opportunity to focus on problem specific needs in a view with reduced model properties. The discussion in section 2.2 and the developed profile is organized according to the views Requirements, Functional, Logical, and Physical as suggested in the research project SPES 2020 [Po12]. **(2)** Abstraction of the communication into different layers: This principle is covered, e.g. in the OSI basic reference [Int94]. The OSI model is created based on the layer pattern for reducing the development complexity.

2.2 Selection of Classes to Express Network Architectures

Classes have been introduced in the object-oriented software engineering paradigm. A class in this domain is a template of methods and attributes for an object category [Jo17]. Nevertheless, the principle is useful for defining systems, as well. A system usually consists of functionality (i.e. class methods) and attributes denoting and storing the state of the system. This Sec. introduces the selected classes that are required to describe network architectures. The selection of classes is necessary, in order to avoid ambiguities of the same aspect. This step limits also the quantity of different stereotypes. The selection is done by analyzing each class for the context of network configuration of a safety-critical complex system. The classes that are discussed in this Sec. are indicated by using the format `Classname`.

The Functional View is consisting of `Tasks` [Eb15, p. 605] connected via `Dataflows`. A `Task` is selected instead of a `Function`, because `Function` is more suitable at system abstraction level and independent of its implementation [SAE10, p. 11]. `Dataflows`, which are the only identified means, are defining the functional interfaces [Eb15, p. 605]. The functional interface description is important in the context of a safety-critical system: The dataflow analysis is performed in a safety assessment and analyzes the functional dependencies. Dependent on the safety-criticality of a function one needs to ensure, that a failure of the function must not lead to failures of other functions.

Building the bridge to the Logical View, `Tasks` are allocated to an `<OSI Layer> Entity` [Int94]. The angle brackets are placeholders for the respective OSI Layer. The entity is named dependent on the contextual OSI layer. `<OSI Layer> Entity` is generic, whereas `Module` is related to modular architecture [FE17, p. 1942]. Therefore, `<OSI Layer> Entity` is selected. The entities are connected via `Virtual Links` to express the logical connection [Ri17, p. 5]. `Virtual Link` is selected, because of the highest occurrence in literature. Furthermore, because of the term virtual it becomes clear, that the link is not a real connection. Instead, it represents one or multiple connections. `Auxiliary Link` [Yu19, p. 2655] could be used instead, but is not found that often in literature.

A `Node` is a generic name for a network element of the physical view [STVH19, p. 6268]. `Node` can be used instead of `Module`. Modules shall be used, if the system is modular and flexible. Flexibility in a critical system is important, whereas all configurations need to be certified. `Nodes` can be further specialized, e.g. for denoting the functionality of a `Bus Controller`[3] [Pl18, p. 114]. The specializations can vary depending on the project context.

A `Network` is a class required for grouping connections and is implemented with `Switches` [Yu19, p. 2654] or `Buses` [Ch16, p. 206] together with their necessary `Communication Protocols`. While a `Bus` connects `Nodes` located all over the system, a `Switch` is in a `Zone` [Eb15, p. 605], which denotes a location in the system. A `Node` can be part of a `Zone` or `Domain` [St18, p. 73]. In order to connect the `Node` to the `Network`, a `Network Port` is necessary. The selection of `Switch`, `Bus`, `Domain`, `Zone`, and `Network Port` is straightforward.

3 Transition from Classes to SysML Stereotypes

The classes selected in Sec. 2.2 need to be mapped to SysML stereotypes. This is performed based on the class nature (e.g. structural or behavioral). As a preparation, SysML diagram and element types are allocated to the views of the view pattern according to Pohl et al. [Po12], which simplifies the mapping.

3.1 SysML Stereotypes for Functional View Classes

System functions relevant for the Functional View are expressed via `SysML::Activities` [Po12]. Activities define the transformation behavior from input to output parameters [FMS14]. A `SysML::Activity Diagram` is used to define the functional

[3] "The Terminal assigned the task of initiating information transfers on the data bus" [U.S18, p. 2].

interactions and interfaces via SysML::Object Flows [Obj19]. SysML::Control Flows could be used, as well. However, SysML::Object Flows define the data exchanged between functions, while the data (object) interfaces between functions are again essential for a safety assessment. Functions could be expressed as SysML::Blocks instead [Obj19]. SysML::Blocks are powerful, when it comes to specializations, generalizations, and system breakdown. However, the focus in the Functional View lies on functional interfaces and dependencies. Therefore, no blocks are used in this view. For allocating the classes of Sec. 2.2 to the SysML elements of this view, Tasks are implemented with SysML::Activities. The functional interfaces that are implemented with Dataflow classes can be expressed by SysML::Object Flows. A SysML::Object Flow is related to SysML::Activity Parameters. Whereas SysML::Parameters defines the interface of a SysML::Activity, the SysML::Object Flow defines the connection to another SysML::Activity. SysML::Parameters need to be typed by SysML::Value Types to express the kind of data exchanged via the SysML::Parameter [Obj19]. The resulting stereotype extensions are shown in Fig. 1

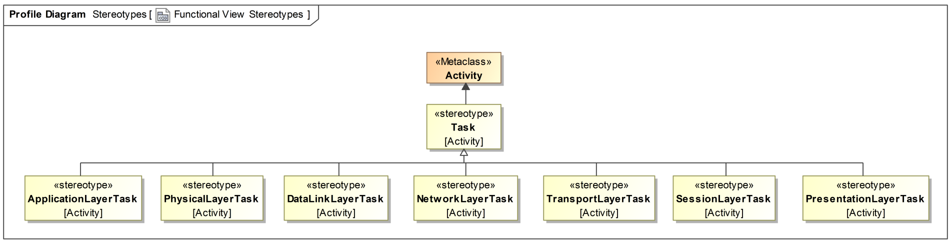


Fig. 1: Functional View Stereotypes

3.2 SysML Stereotypes for Logical View Classes

SysML::Blocks denote logical and physical elements of each abstraction level [Po12]. Hence, the physical and logical system decomposition is implemented using SysML::Blocks [Obj19]. The SysML::Block decomposition is expressed by UML::Composite Aggregations. The SysML::Block Definition Diagram is the most suitable diagram type to authorize and decompose blocks. Alternatively, tables can be used to define the part associations [Obj19] in a semantically equivalent way. A SysML::Block expresses the Logical View element <OSI Layer> entity. The SysML::Block Definition Diagram is used to relate the <OSI Layer> entity as parts of the logical parts, i.e. items by using UML::Composite Aggregations. Whereas specializations of SysML::Block are used for elements introduced by the network architecture model, specializations of UML::Named Element can be used for elements that are already defined in the system model. This is, because stereotype specializations of SysML::Block cannot be applied to existing block elements.

The `SysML::Internal Block Diagram` is introduced with SysML to define the internal relations of system elements [Obj19]. Particularly the interface definition of a system and connections to other systems is defined with the `SysML::Internal Block Diagram`. While the Logical View shows only logical connections, the Physical View visualizes real connections in a system in this diagram type. Logical interfaces are expressed via `SysML::Proxy Ports` [Obj19]. This stereotype is adequate, because it adds no additional part properties to the system definition. Instead, a `SysML::Proxy Port` represents the aggregated real interfaces via several hardware elements. In order to specify the data exchanged via the ports, a `SysML::Flow Property` and `SysML::Item Flow` can be used. `SysML::Flow Properties` are properties of a `SysML::Block` specifying data items flowing into or out of `SysML::Block` or both [Obj19]. Each `SysML::Flow Property` represents a data item that can flow into or out of a block. A `SysML::Flow Property` could be a property for each `SysML::Block` denoting a system element.

In the example of a computer, a `SysML::Flow Property` could be a video signal provided to a display. Connecting two `SysML::Blocks`, the respective `SysML::Flow Properties` need to match. This means, the display block needs to have a video signal flow property, as well. Other system elements connected to the computer are not requiring the video signal. Therefore, it is more suitable to attach the flow property to a port. This is implemented by defining a `SysML::Interface Block`.

Similar to the `SysML::Block`, the `SysML::Interface Block` can have `SysML::Flow Properties`. Because the `SysML::Interface Block` is specifying the flow behavior of the port instead of the whole block, it limits the potential receivers of the data items. Again, in the computer example one port is expressing the display interface. It is typed by an interface block containing the video signal flow property. Another computer port is typed by an interface block containing the power signal flow property.

`SysML::Item Flows` are like flow properties. `SysML::Item Flows` can be attached to `SysML::Connectors` [Obj19]. Due to this fact, a port can have multiple connections with different items flowing on each connection. In the computer example this could be a voltage meter attached to the display interface using the same port. Whereas the interface block contains a video signal, i.e. voltage and current flow properties, the voltage meter only receives the voltage information[4]. `SysML::Item Flows` can be linked to `SysML::Activities` [Obj19]. This is a useful feature in the view pattern, because the Functional View (`SysML::Activities`) can be linked to the Logical and Physical View (`SysML::Blocks` exchanging `SysML::Item Flows`). The element contained in the `SysML::Item Flow` is called conveyed item and can be a `SysML::Block`. A `SysML::Block` can have `SysML::Signals`. `SysML::Signals` can be used to

[4] In fact, a voltage meter receives a small amount of current, as well. Due to the high impedance of the voltage meter, this current can be assumed to be 0 A.

define Interface Control Data (ICD) messages. Attributes of a `SysML::Signal` can express the actual ICD signals as part of an ICD message [FMS14, p. 146].

A `Virtual Link` expresses the routed connection between two end points independent from the physical connections. Therefore, it should be expressed as a `SysML::Connector`. However, a `SysML::Connector` simply describes, that there is a connection, but it is not specifying, which objects are exchanged. In order to specify the content flowing via the connections, `SysML::Item Flows` are used. The `Virtual Link` can be expressed as a conveyed item of a `SysML::Item Flow`. Alternatively, `SysML::Interface Blocks` can be used to type the `SysML::Proxy Ports` and contain `SysML::Flow Properties`. The `SysML::Flow Properties` contain information that can be provided via the typed `SysML::Proxy Port`. Both the `SysML::Interface Block` and the `SysML::Block` can be attached as conveyed items to the `SysML::Item Flow`, in order to specify the connection. However, increased effort is expected by applying both. Therefore, the `SysML::Item Flow` is used only, because it describes the actually flowing items. The messages exchanged via the `Virtual Link` are contained in a `SysML::Block` as `SysML::Signals`. Each signal has attributes representing the variables exchanged via the `SysML::Item Flow`. The attributes are typed with `SysML::Value Types`. The resulting stereotype extensions are shown in Fig. 2. It illustrates the extension of `UML::Named Element` instead of `SysML::Block` for the model elements as explained above.

3.3 SysML Stereotypes for Physical View Classes

Real interfaces, which are necessary for the Physical View (see Fig. 3) are expressed via `SysML::Full Ports` [Obj19]. In contrast to the `SysML::Proxy Ports` full ports are expressing real physical parts of the system. E.g., the `SysML::Proxy Port` expresses, that there is any interface type from a computer to the keyboard. The `SysML::Full Port` specifies, that the interface is implemented via a Universal Serial Bus (USB) Port.

By mapping the Physical View classes to stereotypes, it can be seen, that `SysML::Blocks` are beneficial, if it comes to decomposition and interface definition. Therefore, a `Node` and a `Module` is expressed using a `SysML::Block`. `Nodes` are set in relation to `Domains` and `Zones` using `UML::Composite Aggregations`. Thus, `Network`, `Domain`, and `Zone` are `SysML::Blocks`. A `Node` can be a part of a `Network` and a `Zone` at the same time. The same applies to the `Domain` and its parts.

`Network Ports` are expressed using `SysML::Full Ports`, because they exist as real parts of a `Node`. `Network Ports` are connected using `SysML::Connectors` authorized in a `SysML::Internal Block Diagram`. `SysML::Interface Blocks` representing different connection types are used to type the `Network Ports`. Possible connection types are `Ethernet`, `Electrical`, `Fiber Channel` etc. `Network Ports` are sometimes not peer-to-peer connections, which is dependent on the network type.

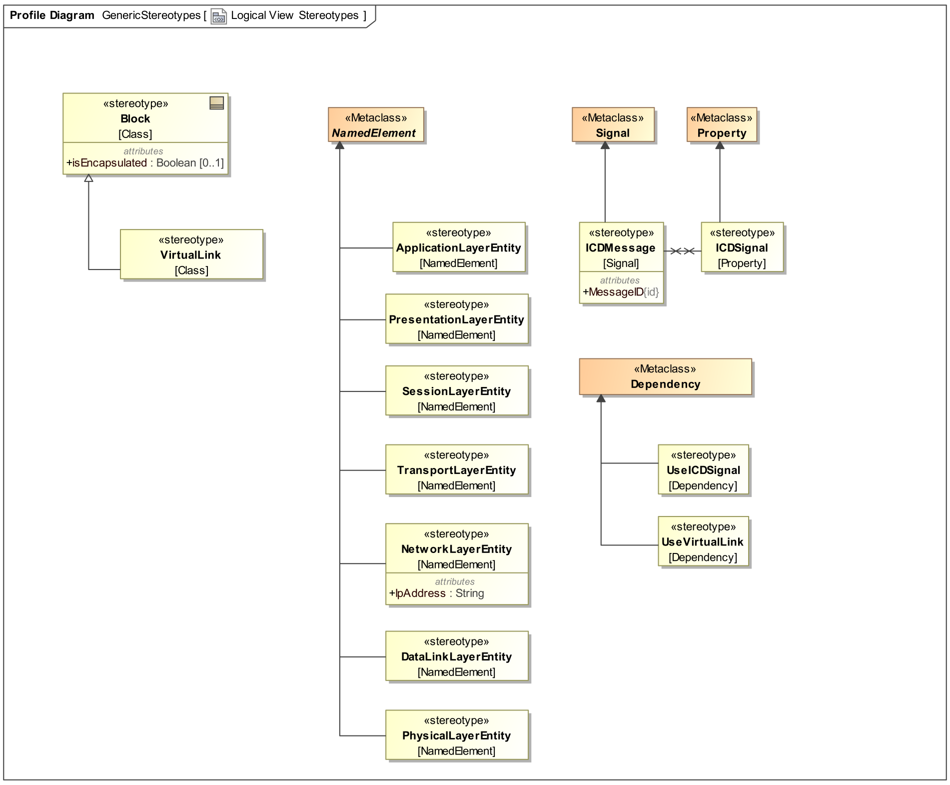


Fig. 2: Logical View Stereotypes

Therefore, `Switch` and `Bus` are introduced via `SysML::Blocks`, as well. For full detailed modelling, a communication protocol behavior can be added to the `SysML::Interface Block` by `SysML::State Machine Diagrams`. `SysML::State` machines are used to describe the different states of a network node in a network [SS15]. However, this approach is not analyzed yet as part of this contribution.

4 Application and Verification of Method and Stereotypes in a Demonstrator Model

A demonstrator model[5] is developed, in order to apply and to verify the method and stereotypes. Cameo Systems Modeler 19.0 is used to define the system and the network architecture. As the demonstrator requests further variables to be defined, further specializations of the

[5] For the verification purposes a demonstrator model is necessary, because it evaluates the applicability of the developed profile. This model can also be regarded as an example.

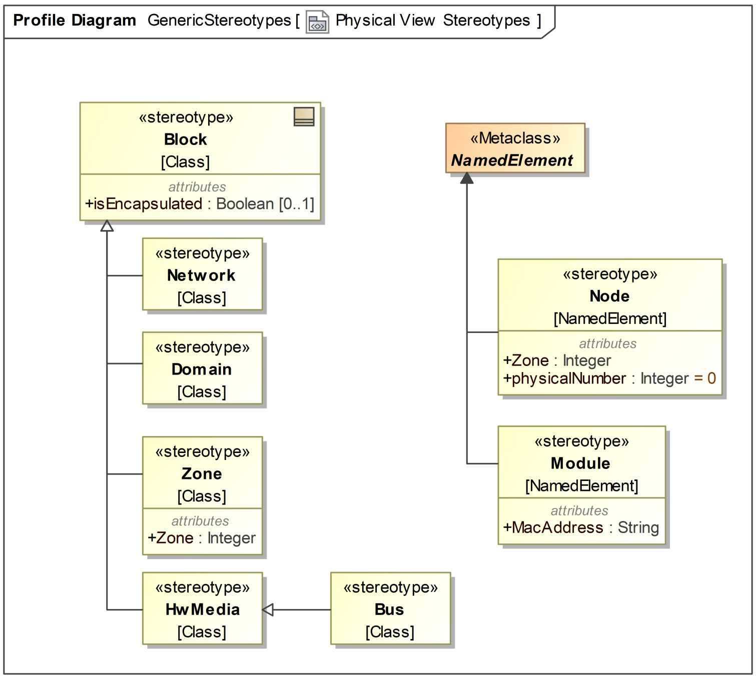


Fig. 3: Physical View Stereotypes

stereotypes are defined. They are «`NetworkCard`» (extending «`NamedElement`») and «`NetworkNode`» (specializing «`Node`»). The example model defines a network, which is consisting of network nodes. The network nodes consist of network cards to access the network. Information of several OSI layers are defined at the network cards. Fig. 4 shows the relations between the model elements of the Physical View. An excerpt of the connections between the network nodes are illustrated in the internal block diagram in Fig. 5. The diagram shows a connection implementing the MLD-STD-1553B bus communication and an adapter for adapting the Positioning Unit. The adapter is necessary, because just an Ethernet interface exists.

In the logical view, end-to-end communication is defined. This relates to OSI layer 3 (Network Layer). Therefore, Network Layer Entities are defined. The end-to-end-communication is defined by ports and connectors. Item flows are realized by the connectors to attach the traffic to the connectors. The traffic is defined by the `Virtual Link Block` with relations to ICD messages. Fig. 6 illustrates the example of the Logical View, where «`Network Layer Application`» is a specialization of «`Network Layer Entity`».

As a result, the network configuration approach using the developed profile has been verified by the demonstrator model. However, the final application of the approach and thus its full

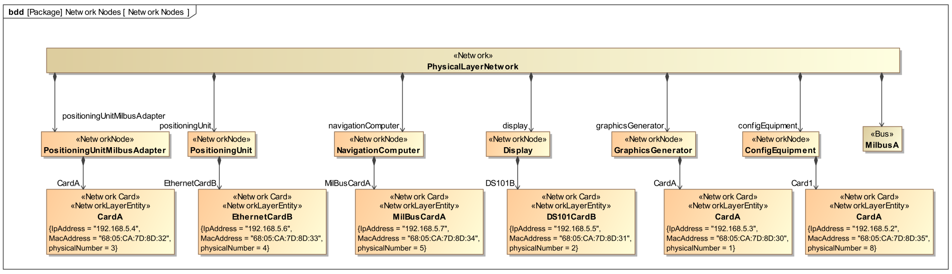


Fig. 4: Network Nodes

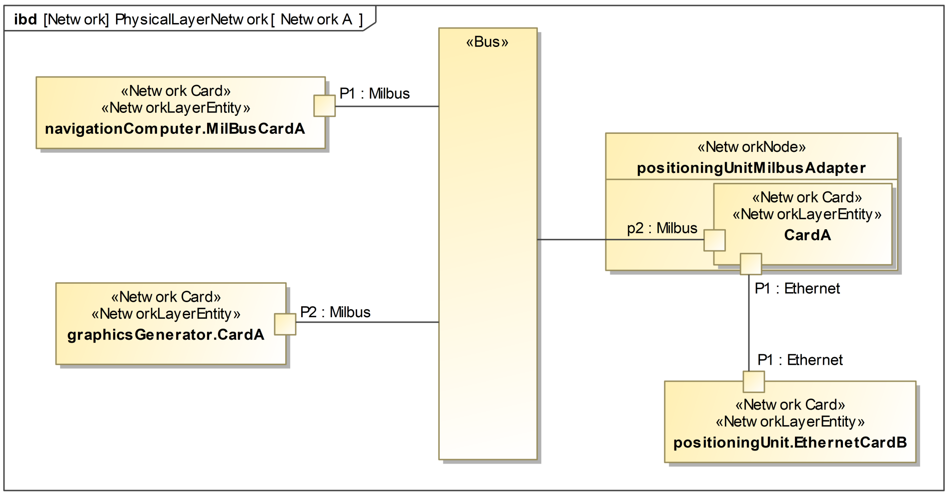


Fig. 5: Example Physical Layer Network Connections

verification including functional view is left to be performed, once the complete system model of the real product is available.

5 Fit-gap Analysis between Tool, Metamodel, and SysML v1.6

As the result of the analysis to select a suitable tool, Cameo Systems Modeler based on SysML v1.5 [Obj17a] is chosen. This Sec. analyzes the differences between v1.5 and v1.6 for SysML stereotypes applicable for the stereotypes of the developed profile. Furthermore, differences between the metamodel of Cameo Systems Modeler and SysML v1.5 are analyzed. The result is the traceability of differences between Cameo Systems Modeler to the current SysML standard v1.6. The reworked changes between SysML v1.5 and v1.6 are published by the authorizing standards body OMG. In this Sec. each issue is referenced by the key numbers of the changes given in [Oba] (e.g. SYSML 16-001).

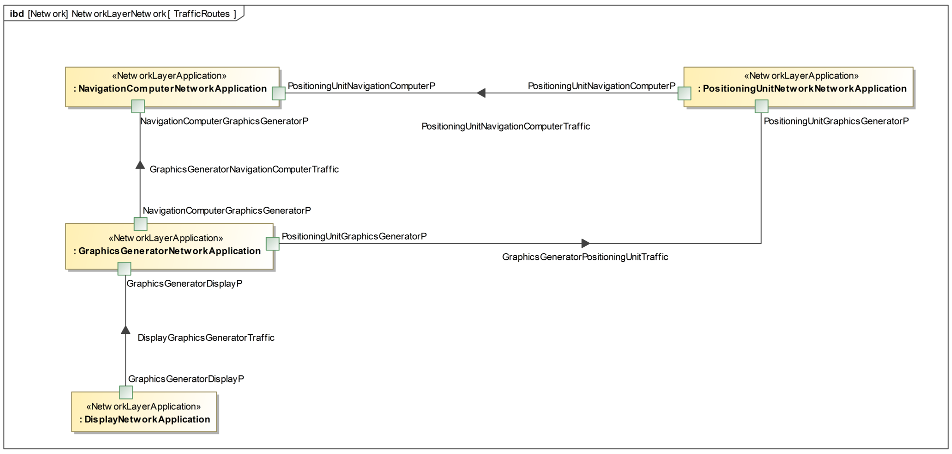


Fig. 6: Example Logical Layer Network Connections

In SysML v1.6 the constraint of locating a SysML::Instance Specification in the same SysML::Package, where it belongs to, is deleted (SYSML16-185). SysML::Instance Specifications are required for the usage of expressions, e.g. UML::Opaque Expressions. UML::Opaque Expressions are required to express hexadecimal values. In Cameo the constraint of SysML v1.5 is strictly implemented, so that a SysML::Instance Specification is only allowed to be owned by a SysML::Package. Subsequently, a SysML::Instance Specification cannot be nested in a SysML::Block it belongs to. This means, the value itself is decoupled from the usage. This may lead to a confusion of the engineer during the configuration of the system. The configured value shall be located under the SysML::Block to see the dependency already by the "own"-relationship. If the Cameo metamodel is switched to SysML 1.6, it should be possible to add a SysML::Instance Specification to a SysML::Block. Thus, the configuration of networks using SysML::Value Properties instead of stereotype tags simplifies the definition of hexadecimal values with UML::Opaque Expressions.

Since SysML v1.6 it is not required anymore to specialize each part association of a SysML::Block, if the block itself is specialized by another block (SYSML16-154). This impacts the network profile in the specialization of Physical Parts by Nodes. The workload of the tool user is dependent on the number of mouse clicks. Therefore, it saves time and avoids consistency conflicts, if the metamodel is adapted to SysML v1.6.

No further differences between the Cameo metamodel and SysML v1.5 are identified. The analysis needs to be repeated for future versions to ensure SysML standard conformance.

6 Answer of Research Question

The identified gaps between SysML v1.6 and required classes for network configurations are organized in the Functional, Logical, and Physical Views. In the Functional View, the `Task` is a specialization of the `SysML::Activity`. The main features of a `SysML::Activity` are suitable for defining the behavior of the logical elements (e.g. Data Link Entity) by connecting them with `SysML::Object Flows`. However, the Functional View is not reflected in the verification through the demonstrator model (see Sec. 4). The demonstrator model requires the configuration of the Logical and Physical View. The Functional View is ignored. Therefore, it could differ from the actually implemented behavior of the logical elements. Finally, the behavior located in the Functional View has to be verified by further quality assurance methods (e.g. tests). A possible future scenario would be to host software code in the Functional View and create software builds from the model. This enables the execution of the model or its parts.

The Logical View also addresses each OSI layer like the Functional View. The `<OSI Layer> Entity` is used in each OSI layer. The `SysML::Block` is used, because of its structural features using part associations. It is observed, that the specialization of the item is not the best way to add additional information of the same thing. By specializing a `SysML::Block` all parts are inherited. However, a new block has to be created as the specialized block. The specialized block inherits all ports of the base `SysML::Block`. During the network configuration, ports have to be redefined. A redefinition of a port decouples it from the original port at the base block. In fact, a new block with ports is created. The behavior is not supporting the idea of having different views and layers, because the information is displayed redundantly. The SysML standard needs to be enhanced to support such a requirement. Another central stereotype in the Logical View is the `SysML::Item Flow`. The `Item Flow` is used in this profile to attach the `Virtual Link` (`SysML::Block`) as a conveyed item. Whereas the `SysML::Item Flow` specifies the items that flow from block to block, the `SysML::Interface Block` typing a `SysML::Proxy Port` or `SysML::Full Port` specifies the items, that can potentially flow. Using the `SysML::Interface Block` instead would lead to multiple ports, because each port has to be used for each connection. The idea of defining both leads to an increased effort depending on the project. Regarding the context of a safety-critical complex system, the effort could lead to delays in system development due to formally redundant specifications. There should be a trade-off for defining one instead of both. In this contribution the approach of specifying only a `SysML::Item Flow` by the conveyed item is used in the Logical View.

In the Physical View the same issues apply to block specialization as discussed in the Logical View. In here, `Nodes` are specializing `Physical Parts`. Additional `Network Ports` are added to the `Node`, which redefine the ports of the `Physical Part`. As a result, the `Network Ports` are decoupled from the ports of the Physical Part. Consequently, a change of the port at the Physical Part does not directly trigger a change of the `Network`

Port. In the context of safety-critical complex system, this leads to a gap of traceability, which is unacceptable.

Independent from the views SysML lacks a way to define number formats different to the decimal format. SysML::Opaque Expressions can be used for expressing the value within a dedicated language (e.g. JavaScript). SysML v1.6 is on the right track by deleting the constraint of locating the SysML::Instance Specification in the SysML::Package, where the block is owned.

Because the contribution is limited to the Network, Data Link, and Physical Layer, further analysis is required for upper OSI layers. These layers could impact the network configuration. Therefore, constraints between layers could need to be taken into account. The analysis is not part of this contribution.

As a result, all network architecture classes required for network configuration can be implemented with SysML. Verification of the selected classes implemented in stereotypes was successful. However, further verification needs to be performed by proof of concept based on different verification methods and network configuration tool vendors.

7 Consequences

7.1 Consequences for System Model Definition in SysML

The network profile developed in the contribution can be used for the specification of network architectures in SysML. The single application of the profile does not ensure a consistent network architecture definition. Therefore, a method is developed. The method of separating the model into different views ensures the addressing of stakeholders' specific needs. Furthermore, the model separation into OSI layers enables the management of the complexity of the network architecture development. Consequently, by the application of the method and the profile, the network architecture is defined.

Regarding the identified gaps discussed in the research question (see Sec. 6), SysML has to evolve to fulfill specific needs. One of the major drawbacks is the lack of possibilities to define values in different number formats. Even if SysML v1.6 introduces means to improve this issue, the issue itself is not solved yet.

Although SysML provides features to create views (e.g. by using the specialization relationship), the application is not trivial. This leads to blocking points and misunderstandings by engineers. SysML should improve the communication between engineers and avoid misunderstandings. The standard should evolve to tackle this issue to be more comprehensible. Alternatively, the focus has to be on educating the engineers.

For systems development decisions have to be taken, if all information shall be covered in a single model. Different modelling languages are evolving and address domain specific needs. E.g., the authorization of ICD messages in the SysML tool has turned out to be not straightforward. Dedicated ICD management tools need to be used. The ICD messages could be imported to the SysML tool.

The developed profile is generic. However, it can only be verified, whether the model information can be reused. Therefore, the model is exported for further use (e.g. network configuration software). In the scope of the presented work a network configuration tool is developed by a tool vendor. Consequently, the verification is dependent on the information exchanged with the network configuration tool. The profile is adapted using stereotypes to enhance the generic stereotypes, in order to capture the information specific to the tool vendor.

7.2 Consequences for Model-based Systems Engineering

At the beginning of each system development process it needs to be decided, what the scope of the model is. In theory, all aspects of a system design shall be entered into the system model. Practically, this theory is limited by project constraints, e.g. costs or development time. However, it is expected that an increased effort in system modeling in the development phase (frontloading [Fa09]) will reduce costs in the production and service phases.

The network configuration based on the network architecture in the system model builds the bridge between the digital and the real dimension of the system. In the context of a safety-critical complex system, the required traceability from the specification of the system model to the network configuration is ensured now. Finally, the objective of creating a digital twin is getting closer.

8 Conclusion and Outlook

The presented approach is developed for SysML v1.6. Further adaptions could be required due to the release of SysML v2.

The scope of the verification is limited to Network, Data Link, and Physical Layer. However, the developed method intends to define upper OSI Layers, as well. It needs to be verified, whether these layers can be defined using the profile and method. Regarding upper OSI Layers the design is driven by the decision of message-based or content-centric communication design. For example, the Data Distribution Service (DDS) Standard is shifting the message-based communication to data-centric communication [DD]. The impacts on the selected classes (see Sec. 2.2) have to be analyzed to fulfill the needs of such technologies.

The network architecture is dependent on the given DDS Quality of Service (QoS) information defined in upper OSI Layers. Lower OSI Layers have to implement the achievement of a specific QoS. Therefore, further analysis has to be performed on the impact between the OSI Layers for such a technology.

Once the network architecture is defined mostly, further analysis is performed: As an example, the allocation of traffic routes to TDMA (Time Division Multiple Access) scheduled time slots is analyzed regarding the traffic load. The schedule is necessary, as it defines the network communication[6]. Therefore, a permutation of all possible configurations is generated and analysed. The approach is visualized in Fig. 7 and described with the following steps: **(1)** Logical and physical views of the network architecture are imported into a Permutation

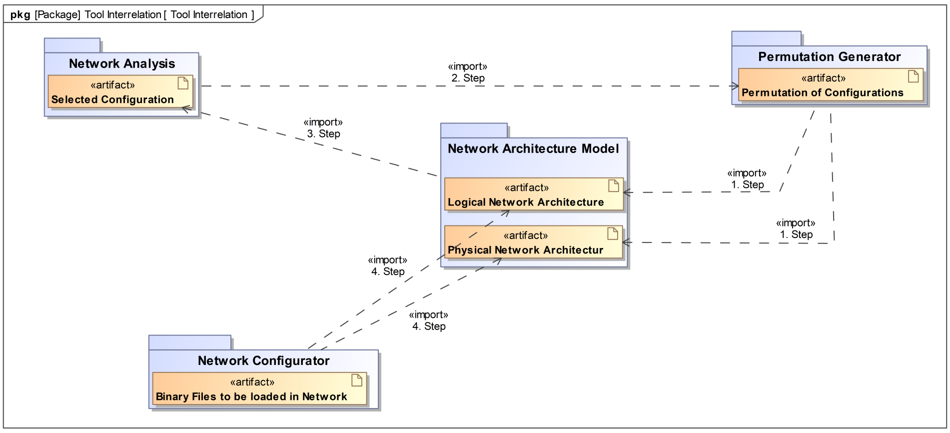


Fig. 7: Example Logical Layer Network Connections

Generator. The generator creates several possible configurations of virtual links to time slot allocations. A Traffic Route defines the End-to-End communication without mentioning all network nodes in the routing. **(2)** The generated configurations are imported into the Network Analysis Tool. The best configuration is proposed based on given criteria of the network calculus [WT]. **(3)** The selected configuration is imported into the Network Architecture Model. **(4)** Logical and Physical Architecture are imported into the Network Configurator to generate binary files, which are loaded into the network nodes: The network architecture consisting of the information above is exported into an XML file. The file is imported into the Network Configurator. It generates binary files, which are loaded into each network node of the real avionics application. This enables the respective nodes to communicate according to the specified patterns.

Most recent standardization documents require the coverage of cyber-security for aircraft certification purposes. Since this aspect is not yet covered by this contribution, relevant aspects along e.g. UMLsec [Ju10] have to be transferred to our approach as part of future work.

[6] The defined network communication is finally loaded into the network nodes of the real avionics application.

Bibliography

[Bo21] Boehm, W.; Broy, M.; Klein, C.; Pohl, K.; Rumpe, B.; Schroeck, S.: Model-Based Engineering of Collaborative Embedded Systems. Springer Cham, 2021.

[Ch16] Champeaux, P.B.; Faura, D.; Gatti, M.; Terroy, W.: A Distributed Avionics Communication Network. In: 2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSN-W). pp. 206–209, 2016.

[DD] DDS Foundation: How does DDS work? `https://www.dds-foundation.org/how-dds-works/`, last access: 4th May 2022.

[Eb15] Ebeid, E.; Medina, J.; Quaglia, D.; Fummi, F.: Extensions to the UML profile for MARTE for distributed embedded systems. In: 2015 Forum on Specification and Design Languages (FDL). pp. 1–8, 2015.

[Ei14] Eigner, M.; Apostolov, H.; Dickopf, T.; Schäfer, P.; Faißt, K.-G.: System Lifecycle Management: Am Beispiel einer nachhaltigen Produktentwicklung nach Methoden des Model-Based Systems Engineering. Zeitschr. f. wirtsch. Fabrikbetrieb, vol. 109, no. 11:853–860, 2014.

[Fa09] Faust, P.: Lean durch Frontloading. Zeitschrift für wirtschaftlichen Fabrikbetrieb, 104(5):366–370, 2009.

[FE17] Feng, H.E.; Ershuai, L.I.: Deterministic bound for avionics switched networks according to networking features using network calculus. Chinese Journal of Aeronautics, 30(6):1941–1957, 2017.

[FMS14] Friedenthal, S.; Moore, A.; Steiner, R.: A Practical Guide to SysML: The Systems Modeling Language. The MK/OMG Press. Elsevier Science, 2014.

[He20] Hemmert, A.: Development of a SysML profile for network configuration in safety critical complex systems. Master's thesis, FOM Hochschule für Ökonomie and Management – Hochschulzentrum München, 2020.

[Int94] International Telecommunication Union. ITU-T Recommendation X.200: Data Networks and Open System Communications – Open Systems Interconnection – Model and Notation, 1994.

[ISO11] ISO/IEC/IEEE. 42010:2011 Systems and software engineering – Architecture description, 2011.

[Jo17] Jones, C.: Software Methodologies: A Quantitative Guide. CRC Press, 2017.

[Ju10] Juerjens, J.: Secure Systems Development with UML. Springer, 2010.

[Ku17] Kuhn, T.: Digitaler Zwilling. Informatik Spektrum, 40, no.5:440–445, 2017. DOI: `https://doi.org/10.1007/s00287-017-1061-2`.

[Oba] Object Management Group: OMG System Modeling Language - Closed Issues. `https://issues.omg.org/issues/spec/SysML/1.6/fixed`, last access: 4th May 2022.

[Obb] Object Management Group: SysML SPECIFICATION AT OMG – Introduction. `https://www.omg.org/intro/SysML`, last access: 4th May 2022.

[Obj17a] Object Management Group. OMG Systems Modeling Language (OMG SysML™), v1.5 edition, 2017.

[Obj17b] Object Management Group. OMG Unified Modeling Language (OMG UML), v2.5.1 edition, 2017.

[Obj19] Object Management Group. OMG Systems Modeling Language (OMG SysML™), v1.6 edition, 2019.

[Pl18] Plankl, H.: 6.1 – MIL-STD-1553B and it's potential for the future. In: ettc2018 - European Test and Telemetry Conference 2018-06-26 - 2018-06-28 Nürnberg, Germany. pp. 114–122, 01 2018. DOI: 10.5162/ettc2018/6.1.

[Po12] Pohl, K.; Hönninger, H.; Achatz, R.; Broy, M.: Model-Based Engineering of Embedded Systems – The SPES 2020 Methodology. Springer Berlin, Heidelberg, 2012.

[Ri17] Ricker, T.: Avionics bus technology: Which bus should i get on? In: 2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC). pp. 1–12, 2017.

[SAE10] SAE International. ARP4754 (R) Guidelines for Development of Civil Aircraft and Systems, 2010.

[SS15] Shames, P.M.; Sarrel, M.A.: A modeling pattern for layered system interfaces. 25th Annual INCOSE International Symposium (IS2015), Seattle, WA, July 2015.

[St18] Steinbach, T.: Ethernet-basierte Fahrzeugnetzwerkarchitekturen für zukünftige Echtzeitsysteme im Automobil. Springer Vieweg, 2018.

[STVH19] Sollfrank, M.; Trunzer, E.; Vogel-Heuser, B.: Graphical Modeling of Communication Architectures in Network Control Systems with Traceability to Requirements. In: IECON 2019 – 45th Annual Conference of the IEEE Industrial Electronics Society. volume 1, pp. 6267–6273, 2019.

[U.S18] U.S. Department of Defense (DoD). Digital Time Division Command/Response Multiplex Data Bus, 2018.

[Wa16] Wasson, C.S.: System Engineering - Analysis, Design, and Development. Wiley, Hoboken, New Jersey, USA, 2016.

[WT] Wandeler, E.; Thiele, L.: Modular Performance Analysis with Real-Time Calculus. `https://www.mpa.ethz.ch/`, last access: 4th May 2022.

[Yu19] Yunus, N.A. Md; Othman, M.; Hanapi, Z.M.; Yeah Lun, K: Enhancement Replicated Network: A Reliable Multistage Interconnection Network Topology. IEEE Systems Journal, 13(3):2653–2663, 2019.

Implementation of the SpesML Workbench in MagicDraw[1]

Nikolaus Regnat[2] Rohit Gupta[3] Nico Jansen[4] Bernhard Rumpe[5]

Abstract:

Model-Based Systems Engineering (MBSE) is a formalized methodology that focuses on creating models at the centre of system design, rather than traditional document-based approaches. While general purpose modelling languages such as the Systems Modelling Language (SysML) and their corresponding methodological approaches such as the Software Platform Embedded Systems (SPES) framework are available, the combination of a modelling language, method, and tooling is still lacking. Typically, industrial language engineers simply provide guidelines for users on how to use a particular modelling language as there is no methodical support for using the same with a well-defined method and a well-suited graphical modelling tool. This puts all the burden on the user, often resulting in a failure of the whole approach. To solve this challenge, we introduce a coherent and systematic approach for the efficient development of a SysML workbench that combines SysML and the SPES methodology using a modelling tool, MagicDraw. In this paper, we showcase the construction of a comprehensive methodical workbench by integrating key aspects of the modelling language, the SPES methodology and MagicDraw. Ultimately, the resulting SysML workbench for SPES serves as a reference point for future MBSE implementations, relieves users from the many burdens of traditional approaches, and helps mastering the complexity of creating collaborative model-based systems with efficient methods and tools.

Keywords: Model-Based Systems Engineering; Domain-Specific Languages; Industrial Language Engineering

1 Introduction

With the advancement in digitization of various systems engineering domains, there is a notable shift in the way modelling is introduced using a model-based systems engineering (MBSE) [Se03] approach into an organization. However, there still exists a conceptual gap between the fundamentals of system engineering and the combination of a relevant methodology along with modelling tools [FR07, Re18] in effectively describing and using a modelling language. The modelling of systems, therefore, requires a methodologically sound approach to software and system development. To this end, in the German Federal Ministry of Education and Research (BMBF), funded projects such as **"Software Platform**

[1] This work was supported by the German Ministry for Education and Research (BMBF) in the SpesML project (https://spesml.github.io/index.html/; grant number 01IS20092D and 01IS20092B).
[2] Siemens AG, Munich, Germany nikolaus.regnat@siemens.com
[3] Siemens AG, Munich, Germany rg.gupta@siemens.com
[4] RWTH Aachen, Aachen, Germany jansen@se-rwth.de
[5] RWTH Aachen, Aachen, Germany rumpe@se-rwth.de

Embedded Systems" (SPES) [Po12] and its follow-up project **"Software Platform Embedded Systems Extended" (SPES_XT)** [Po16] have been developed to provide foundations for a comprehensive methodological toolkit in model-based development. The complexity in developing embedded systems is addressed using the methods described in these projects. However, there still exists the challenge of combining such a methodology with a modelling tool effectively which often leads to laborious efforts for the end user in adopting such methodologies.

SPES and SPES_XT have further advanced the development of automated embedded systems with a solid methodology based on MBSE. The SPES methodology has been developed to provide guidelines for the specification, design and models of complex systems. SPES is based on a number of specific scientific modelling theories, with a special focus on consistency and semantic coherence called *FOCUS* [BS12, Br10]. SPES is comprised of four different system *viewpoints*, which are collections of model elements addressing a set of concerns of various stakeholders. Each of these viewpoints contribute to the description and understanding of the system, while still supporting the separation of different aspects into different models. SPES is able to achieve system decomposition into finer, less complex granular levels using these separation of concerns. Further, the BMBF project "Collaborative Embedded Systems" (CrESt) extends the SPES framework to networks of collaborative embedded systems (CESs), thereby addressing challenges of complex networks of CESs.

The SPES methodology provides a relevant direction to modelling experts in the model-based development of software intensive cyber physical systems [PB20]. Modelling languages such as SysML [FMS14], were built *by* modelling experts *for* modelling experts, ignoring the fundamental fact that most end users are not and probably never will be modelling experts. Any attempt at building a comprehensive SPES workbench introduces challenges in extensibility, to what extent systems can be decomposed, and how independently viewpoints are developed. These challenges can be alleviated by providing users with a complete tooling framework that bundles the language and the method. Such a tooling framework provides language engineers with scope for customizations to realize a successful SPES methodology and reduces laborious manual efforts for users in their modelling. A successful adoption of the SPES methodology covers extensibility, both in implementation and architecturally, for future SPES projects. Bringing these aspects together is a challenge, as small and medium enterprises do not have the resources to build a comprehensive solution that combines the aspects of a modelling language, methods provided by SPES, and a modelling tool into a cohesive whole (Section 2). Modelling tools must be capable of providing editing capabilities at a greater level of flexibility for language engineers as well as for end users. These include support for a complete language definition [CGR09, Cl15, HRW18], templates for assisting users in quickly designing models, choosing tool features based on their modelling needs, and other extensions that add to the default functionalities of the tool. Only with the combined use of a modelling language, a method, and a modelling tool can efficient and semantically sound modelling be achieved.

Given the basis of SPES as a methodology foundation, the project SpesML provides a direction to these challenges. In this paper we put the choice on a firmer basis:

- We explore the creation of a SpesML workbench using a graphical modelling tool, MagicDraw, and integrating the SPES methodology, described in Section 3, with it.
- We show the editing capabilities that MagicDraw offers to define custom plugins consisting of the SPES language profile, a predefined template for creation of models, application of features based on the modelling experience of a user and additional extensions written in Java, that help realize the complete language definition.
- In Section 4, we take a detailed look into how the SPES methodology is configured in MagicDraw, by showcasing the internals of the composition of the language. These include the creation of different viewpoints as building blocks of a language, individual customization on language elements to enhance the aesthetics [Mo09, Ni00], validation rules and the interoperability between the different viewpoints.
- We detail the separation of concerns used in MagicDraw, and show how individual SPES language components are composed at different levels of granularity and are structured to the users modelling needs.
- Ultimately, in Section 5 we discuss how adopting SPES as a methodology can be beneficial to users in mastering the complexities of modelling in a practical environment such as MagicDraw and in Section 6 we conclude the paper.

2 Background

Models are an abstraction of the original system, which aims to reduce the gap between the domain problem and its implementation [Ad20]. Model-based approaches intend to revolutionize the tools for software engineering as well as the process of their definitions from classical documents to models [Ru16]. Model-based development is not only about drawing or setting up models, but also about the inclusion of a comprehensive modelling methodology. Typically, just introducing models is not sufficient. Modelling experts must also think about how systems can be broken down into smaller parts, that can be better designed and later synthesized back into a whole [Gu21]. Engineering of such systems require principles, concepts and methods, which form the basis of MBSE. MBSE is a formal methodology used to support requirements, design, analysis, and a number of other modelling concepts to capture system properties precisely [Bö21]. The three main aspects of MBSE that must be considered independently and also in good coordination with others are: (i) modelling language, (ii) methodology; and (iii) modelling tools. Only when these aspects are considered as a coherent whole, can the realization of an MBSE approach work cohesively. MBSE is applied on a number of complex heterogeneous systems such as Cyber-Physical Systems (CPSs) and Collaborative Embedded Systems (CESs) [Ru19].

CPSs are software controlled, interconnected physical machines that typically sense their environment and are able to interact with their contexts in some way. CESs, on the other hand, are a result of the transition from traditional embedded systems to a network of CESs working with other systems to achieving a common goal.

The drastic increase in the scope of variant diversity across domains [TK05] and the complexity of systems and system networks, presents engineering of embedded systems with new and challenging problems. In the funded projects, SPES and the follow-up project SPES_XT, the basis for a comprehensive methodical construction kit for the consistent model-based development of embedded systems has been developed. The SPES framework consists of methods and tools based on specific modelling theories, that help master the complexity of embedded systems in an efficient, controllable and verifiable manner. By using separation of concerns, SPES ensures the central principles of consistency, assessability and tool support are all taken into account while solving the different engineering challenges. The project CrESt, on the other hand, aims to create a comprehensive framework for the development of collaborative embedded systems that addresses the new challenges in the development of collaborative embedded systems in dynamic system networks, by leveraging the SPES methodology.

Despite the progress made in these projects to create a solid methodology for MBSE development, the realization of the methodology is left to the end users. This leads to challenges for modelling experts in adopting the methodology using relevant modelling tools. Few possible explanations for this are that organizations often struggle in incorporating modelling tools whose integration with a modelling language and a methodology is laborious, the modelling tool may be just too expensive or it does not support handling of different variants of their heterogeneous complex systems. In the new funded project, SpesML, a SysML workbench for the SPES method has been developed aiming to provide a custom tailored SysML profile that integrates the SPES method using the tool MagicDraw.

3 Methodology

3.1 SPES Methodology

The SPES methodology is based on a solid scientific foundation of consistency and semantic coherence called FOCUS [BS12]. It is based on three important principles [Bö21]: (1) The design process must consider interfaces consistently; (2) Decomposition of the interface behaviour and description of systems via subsystems and components at different levels of granularity; and (3) Definition of models for a variety of cross-sectional topics and analysis options. SPES defines a system model as a conceptual model for describing systems and their properties. System models define the components of systems, the structure, essential properties, and other aspects that have to be considered during development. SPES defines an MBSE artefact model based on the concepts in the standard ISO-42010 that assumes a System-under-Development (SuD) has an architecture and provides functions for determining the functional properties of the system. During the development process, different *viewpoints* separate the concerns of different stakeholders and allows for managing different artefacts while completely describing a system. SPES predefines four basic viewpoints: requirements viewpoint, functional viewpoint, logical viewpoint and technical viewpoint are described in Figure 1 along different layers of granularity with the topmost layer denoting the models of the SuD. The requirements viewpoint constitutes the system requirement engineering activities. The functional viewpoint describes a set of system functionalities. The logical viewpoint describes the decomposition of the system functions in terms of logical components. Finally, the technical viewpoint combines software and hardware components related to the SuD. The views for the different viewpoints are parts of the overall model that describes a system.

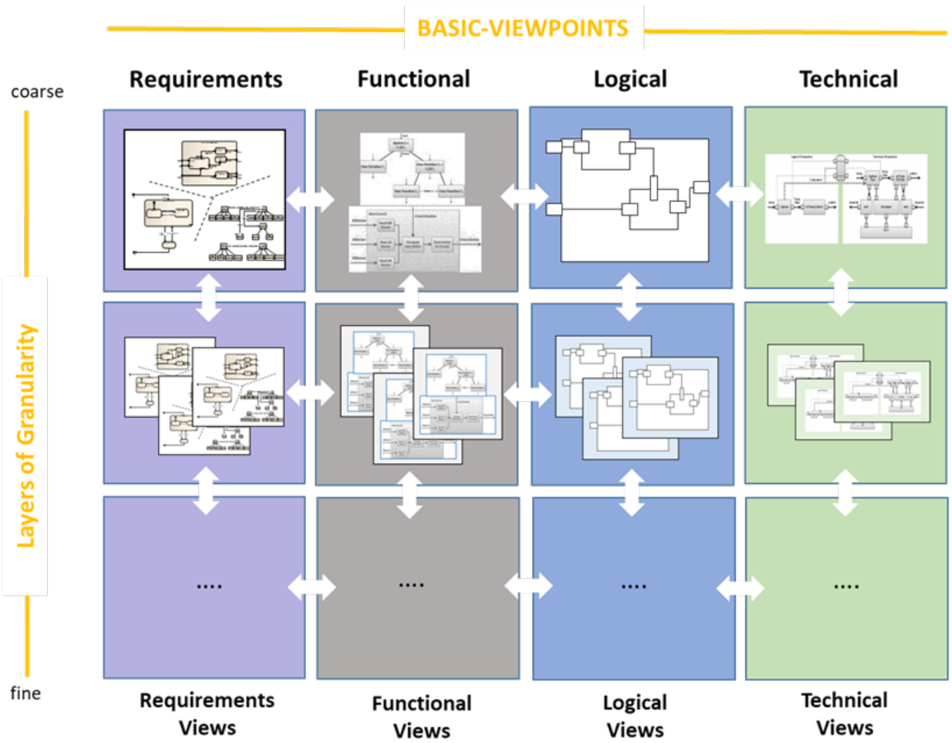


Fig. 1: An overview of the four basic SPES viewpoints: Requirements, Functional, Logical and Technical viewpoints described in [Bö21] across different layers of granularity.

3.2 MagicDraw: Modelling Tool

A good user experience is key to successfully introducing MBSE into any organization [Re18]. At the beginning of the SpesML project, all involved partners decided to use MagicDraw as the tool of choice for the prototypical implementation of SPES. MagicDraw is a modelling tool based on the Unified Modelling Language (UML). It comes with comprehensive extensions (such as a SysML plugin) and provides a wide range of customization possibilities that can be used to enhance the user experience, thus bringing the modelling language, method, and tool closer together. Gupta et al. [Gu21] describe a systematic engineering process of developing industrial domain-specific languages (DSLs) [Fo10] using modular reusable DSL Building Blocks in MagicDraw. Note that we consider SpesML a DSL consisting of the different viewpoints, or DSL Building Blocks, even if we only build upon the SysML and UML. Figure 2 describes the parts of this development approach conceptually by separating the concerns of industrial engineering and deployment of DSLs on the following levels: (1) Concept level: in this level, language engineers define the following parts: (i) the re-usable language components that, wholly or in part, defines the language [Ru16]; (ii) the method, describing a suitable methodology for the language to help users achieve their intended modelling goals; and (iii) the user experience design, where standards and usability heuristics related to user experience are described; (2) Tool-specific implementation level: in this level, language engineers realize the viewpoint aspects described in the concept level using MagicDraw; (3) Usage level: the level where end users model using MagicDraw. Ultimately, DSL Building Blocks, here SPES viewpoints, are composed together to create the SpesML DSL, consisting of heterogeneous domain constructs that the SpesML workbench leverages as different viewpoints.

MagicDraw provides the capability to define custom plugins that can be installed for the end user. Such a plugin typically consists of a MagicDraw project containing the profile, a template for new models, MagicDraw perspective definitions, and additional Java extensions to support dynamic customizations including defining context conditions. Standard modelling languages such as SysML are similarly bundled together as plugins and any MagicDraw user can install the plugin with few simple clicks. These functionalities of MagicDraw as a tooling environment makes it a good fit for realizing the SPES methodology.

4 SpesML Workbench

4.1 SpesML Profile

When implementing the SpesML Workbench in MagicDraw, we started with the definition of the SpesML *Language Components*. We created a dedicated MagicDraw profile to define all needed parts of the SpesML modelling language. A profile in MagicDraw does not only consist of stereotypes and tag definitions but also allows defining *customization* elements. These MagicDraw-specific elements allow to define additional rules or context conditions

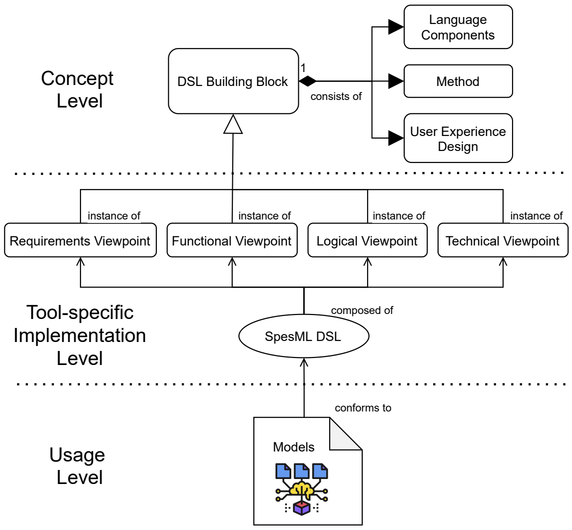


Fig. 2: A conceptual model for the development and usage of a graphical DSL describing the different levels in the engineering process that includes defining the DSL Building Blocks, or SPES viewpoints, each consisting of the method, language components and a user experience design part. The resulting SpesML DSL is composed of the different viewpoints.

for each stereotype. This also allows us to not only embed parts of the SpesML *Method* but also enables us to directly influence the *User Experience Design*.

The first step was to create dedicated stereotypes that represent the chosen SpesML model structure. Instead of using standard UML Package elements, we created individual stereotypes for each structural level. When defining the customizations for these stereotypes we allowed only certain other elements to be created. For example, under the *Logical Viewpoint* (package) we only allow specific elements or diagrams that are part of the *Logical Viewpoint* to be created. This step already provides instructive guidance to the user, as it combines the modelling language, method and tool in an easy to use way. This is less error prone compared to the typical approach of simply giving users access to every UML/SysML element without a defined scope and later providing guidelines on what to do [Re18]. It also ensures that all SpesML models are much more uniform, improving not only readability but overall model quality. Subsequently, it allows for easier integration of automation techniques, be it a document generation or any other form of accessing the model data using the API.

Fig. 3 shows an exemplary stereotype and its corresponding configuration from the SpesML profile. The stereotype «SpesML Logical Viewpoint» is defined with the metatype *Package* and comes with a distinct icon, setting it apart from normal packages. The corresponding *customization* element defines additional aspects for this stereotype. The *abbreviation* defines the default name of the element when the user creates a new element of that type. The *category* influences the user interface: when a user right-clicks on the viewpoint this element will show up in a category called *SpesML Packages*. The *disallowedRelationships* is setup so that all possible relationships on this element are forbidden. The *hiddenOwnedDiagrams* and *hiddenOwnedTypes* configurations hide all normal UML and SysML diagrams and elements from the user; these will not show up in the context menu of this element. The *suggestedOwnedTypes* setting references only those SpesML stereotypes that we want the user to be able to create under this element. Due to this configuration, users do not have to create generic SysML elements and manually apply stereotypes, but can directly create the SpesML elements.

The «SpesML Logical Component» stereotype is defined with the metatype *Class* as MagicDraw does not allow the usage of SysML stereotypes as metatypes. However, we can inherit from the *SysML Block* stereotype. This way our new stereotypes works like a SysML block from the users' perspective. Most model elements, including the different viewpoints, that are used in the SpesML profile come with their individual stereotypes and customizations, similar to these examples.

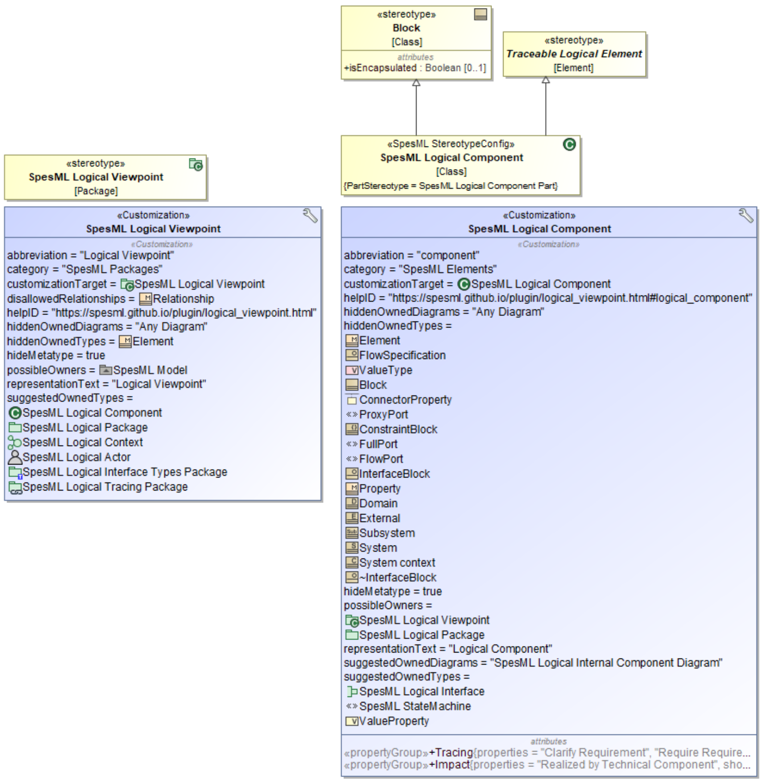


Fig. 3: An example of a SpesML profile stereotype and customization that allows the configuration of custom language elements in MagicDraw.

Fig. 4 shows the result of the customization configuration. During modelling, when the user clicks on *Create Element* on the «SpesML Logical Viewpoint» element, the upcoming dialog box will no longer show all UML/SysML elements, instead only the defined SpesML elements that are appropriately scoped are shown. Elements are also grouped into dedicated categories (such as *SpesML Packages*, *SpesML Elements*, and so on) making it easier for the user to work with the SpesML profile.

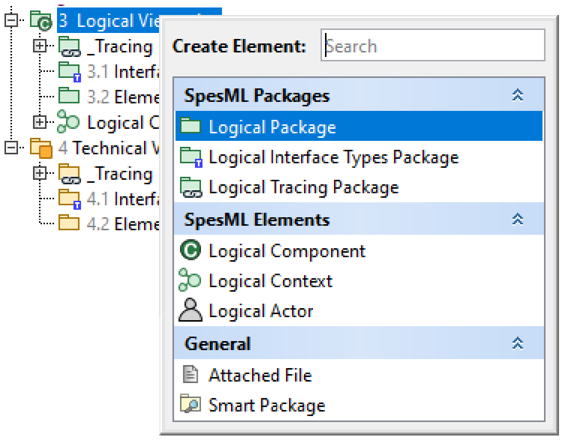


Fig. 4: An example of a dialog box to create a Logical Viewpoint element that shows the categorization and grouping of different elements.

4.2 Custom SpesML Diagrams

In the next step we created dedicated SpesML diagrams. MagicDraw allows the creation of custom diagram definitions based on UML diagrams. Note that we could not create customized SysML based diagrams due to limitations in MagicDraw. These custom SpesML diagrams allowed us to provide customized toolbars that only show those elements that are needed for SpesML. We can also directly reference our dedicated SpesML elements in these toolbars, enabling users again to create these elements without having to first create SysML elements and manually applying stereotypes. Apart from custom UML based diagrams MagicDraw also allows language engineers to define custom *Matrices*, *Tables* and *Relation Maps*.

These special diagrams are used for dedicated purposes and are pre-configured to show only those elements, their attributes or relationships that are required for a certain purpose. For example, in the SpesML Requirements Viewpoint, we have defined a *SpesML Requirements Table* that shows all requirements in a hierarchical table, also allowing users to directly create and modify requirements and their attributes directly on the table.

Fig. 5 shows an example diagram from the SpesML Logical Viewpoint. It is based on an UML Composite Structure Diagram and comes with a reduced set of diagram toolbar

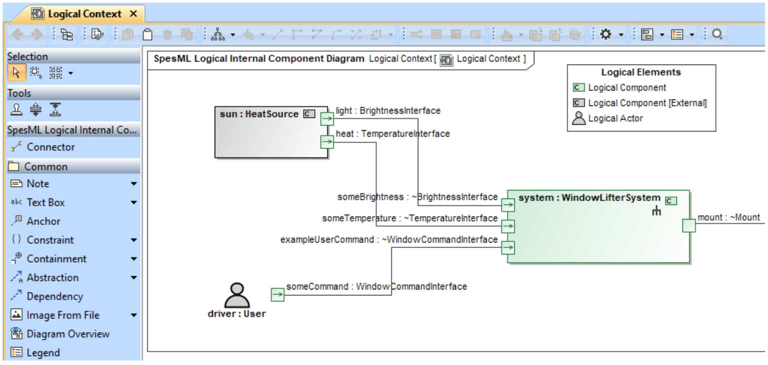


Fig. 5: An example model for a window lifter system using the Logical Viewpoint Diagram.

elements, allowing users only to create *connectors* between ports. This forces users to create all the *Logical Components* in the appropriate packages first, before using them on the diagrams. The diagram also comes with a pre-configured legend that describes all the relevant SpesML elements.

4.3 Embedding Textual Languages

MagicDraw predominantly fosters graphical modelling. However, some specific model elements have proven to be better written in textual form. Examples are guards and actions in state machine diagrams. While the state machine itself remains in its graphical representation, guards and actions on transitions are expressions in text form. MagicDraw comes with a predefined set of (semi-)textual languages such as structured expressions (coded in XML), groovy, or OCL. However, these languages are hardly extensible and did not integrate well into our SpesML workbench, where a modeller should be able to reference arbitrary model elements in the containment tree. Simple expressions for validating or assigning values are not supported.

Thus, we developed a method to integrate custom textual languages into MagicDraw and evaluate their models in our custom validation suite (cf. Section 4.6). While a modeller enters the expression as a String in the default text box for defining guards or actions, the expression is forwarded to an integrated parser. We developed this parser based on an existing library of textual language components [Bu20] using the MontiCore language workbench [HKR21]. If the input text cannot be parsed, our plugin reports the parsing errors back to the user. Otherwise, the plugin continues to construct a so-called symbol table of the containment tree. It is used for resolving references in the expressions to access various attributes of the existing model elements. Thus, it can also be checked whether the referenced elements are accessible from the current scope.

4.4 Model Template

Once we have created our dedicated SpesML elements and diagrams we can setup the SpesML model template. This template is used whenever a user creates a new SpesML project. It consists not only of a dedicated icon and description but a predefined package structure based on our dedicated SpesML stereotypes. While simple to setup, a model template not only makes things much easier for the user but also directly guides the user to follow the SpesML approach in a specific order. Fig. 6 shows the SpesML template as it is presented to the user. It does not only contain a predefined numbered package structure but can also be configured to contain certain other model elements (for example a *Logical Context* element), custom diagrams, matrices, tables and relation maps.

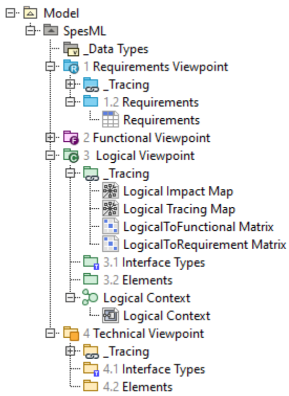


Fig. 6: A model template for the SpesML project that allows quicker and consistent creation of model elements in MagicDraw.

4.5 Perspectives and Help

In the next step, we defined the MagicDraw *perspectives*. These perspectives further customize the user interface of MagicDraw by removing toolbar menu entries, context menu actions, and a wide variety of MagicDraw functionalities. MagicDraw and similar modelling tools do typically offer many features, that are often overwhelming to new and inexperienced users. Reducing the user interface to a minimum will help these users to focus on the relevant functionalities of the tool, while still allowing more experienced users to make use of the full potential by choosing a perspective with more functionalities. In the context of SpesML, we have defined the *SpesML (Novice)* and *SpesML (Expert)* perspectives to support this approach.

Another aspect users often struggle with is finding the appropriate help or documentation when working with modelling tools. MagicDraw does provide the possibility to add dedicated hyperlinks to all stereotypes and custom diagram definitions. To this end, we created dedicated SpesML web pages [6] and linked the complete SpesML DSL with the implementation of all the elements and diagrams to these pages. This allows a SpesML user to click on the *Help* button of any element or diagram and the SpesML specific help is shown to the user. In addition, we have added an API plugin that adds a new entry in the MagicDraw Help menu, also allowing direct access of the dedicated SpesML web pages.

4.6 API Extensions

As a last step we have created and integrated additional API-based extensions to overcome some MagicDraw limitations and to enhance the SpesML Workbench implementation. MagicDraw provides extensive APIs and allows the enhancement of the tool by creating Java extensions for a variety of purposes. We started by implementing the *SpesML Stereotype Plugin* that allows us to automatically apply our SpesML stereotypes to model elements under certain conditions. For example, when a user drags and drops a *SpesML Logical Component* to a *SpesML Logical Internal Component Diagram* we want to apply a the «SpesML Logical Component Part» stereotype to the instance of the part element that is created by MagicDraw under the diagram. This has the benefit that we have full control on this element: we can show the part element with a defined icon and colour and can also define what properties is visible to the user. The result is a more consistent user experience as practically all model elements the users interacts with, have a common look and feel.

We have also added a plugin to enhance the visualization possibilities of MagicDraw. In SpesML, we defined that we want to model external elements (for example an external Logical Component) not by using dedicated stereotypes but instead by allowing the user to define on a part level if an element is external or not. This approach has the benefit that, depending on the development subject we can change whether a certain part is considered external or not. MagicDraw provides the capability to change an elements icon based on an enumeration and this is perfect for the *Containment Tree*. However, a change for an icon is not very distinct on diagrams. With our *SpesML Visualization Plugin* we are able to additionally change the colour of an element based on the enumeration.

MagicDraw also allows to create custom validation rules to check the accuracy, completeness, correctness and well-formedness of a model and in the process, marks invalid elements in the model. We have implemented the *SpesML Validation Plugin* that bundles all rules that have been defined in the SpesML methodology. For example, we want to ensure that certain SpesML elements have at least one port defined or that certain SpesML elements require the user to provide proper naming conventions. Simple rules can be expressed using Object Constraint Language (OCL) but more complex rules can be implemented using dedicated Java classes. In order for the user to easily access and execute these rules we have also created a dedicated *SpesML Validation Suite*.

[6] `https://spesml.github.io/plugin/overview.html/`

5 Discussion

In this paper, we present the SPES methodology and its implementation in MagicDraw in brief. The SPES framework provides a solid MBSE foundation in model-based development but does not necessarily specify the order in which different models should be created for the different viewpoints. This means the end users often struggle in realizing such a methodology effectively using modelling tools and eventually spend laborious manual efforts to achieve their modelling. Therefore, demonstrating the combination of a modelling language with a method using a modelling tool is imperative.

In the funded projects, SPES and SPES_XT, the basis for a comprehensive methodical construction kit for the consistent model-based development of embedded systems has been developed. The projects describe the concepts needed for a methodology-based on MBSE for CPSs, embedded systems and more recently CESs. Ultimately, it is the end user who should benefit from such solid methodologies that provide the relevant direction in modelling. SPES has been realized by different users in the past but so far a reference implementation with the modelling language and the method itself in a modelling tool is missing. The creation of a UML and SysML based workbench for SPES in MagicDraw, in the project SpesML, alleviates tooling shortcomings by providing focus on the end user. MagicDraw is a modelling tool having a range of custom functionalities, and is therefore a good fit for a reference implementation for covering broadly the aspects of SPES.

Industry grade modelling tools such as IBM Rational Rhapsody, Enterprise Architect, Arcadia Capella and MagicDraw offer integrated workbench capabilities and were investigated to create prototypes. In our experience, as it supports the Open Java API, MagicDraw enables many flexible extensions, as described in Section 4, directly into the tool. The MagicGrid reference implementation in MagicDraw also provides a suitable process for the methodology but lacks the composition of smaller SPES artefacts into a single whole. In contrast, our implementation allows decomposition of the SPES viewpoints that can be used independently. We used MagicDraw to create a profile that allows creation of stereotypes and tag definitions along with its customization properties that directly influence the user experience. Dedicated elements and diagrams were created to help realize the different viewpoints and their respective sub-elements, reducing the effort needed to create individual SPES elements. The availability of MagicDraw templates provide the end-users with a predefined model structure of SPES on the tool, eliminating the need to realize every single SPES aspect from scratch. MagicDraw also offers perspectives that influence how many functionalities of the tool a user sees, benefiting both novice practitioners and advanced modelling experts. Documentation can be added to individual elements in the form of hyperlinks to easily navigate to dedicated SpesML webpages. Limitations on the functionalities of MagicDraw are overcome using Java based API plugins bundled in the SpesML workbench, allowing additional functionalities to be introduced, such as dedicated context conditions or dynamic modification of the appearance of model elements. We believe using these customization functionalities of MagicDraw to implement the SPES methodology is more beneficial than creating a completely new method-specific tool because it helps reuse existing tooling functionalities and provides ample scope for generalizing specific aspects of the methodology independent of other modelling tools. Furthermore, the SPES methodology is intentionally designed to be language-agnostic to be applicable to different DSLs. Developing the SpesML workbench, we explored its application on SysML.

As of today, we are unaware of the existence of any other reference implementation of SPES using a modelling tool that is capable of providing such a wide range of customizations for a modelling language and a method. The realization of the SPES methodology in MagicDraw therefore presents a reference implementation for novice and advanced users as well as for small and medium organizations wanting to integrate the SPES methodology into their projects. Bringing heterogeneous domains together and building a library of reusable units is important in achieving modularity in systems engineering. Further, a solid methodology needs to be supported by appropriate tooling mechanisms to lessen the burden on end users. Finally, good user experience design aspects must be considered in order to achieve effective modelling for the end user. We believe the combination of a modelling language, method and tool needs further discussion between researchers and practitioners to foster efficient MBSE development in the future. To this end, we consider the SpesML workbench, realized in MagicDraw, a good reference point for future SPES implementations.

6 Conclusions

As various system domains become complex, heterogeneous and digitized, approaches to engineer such systems need solid methodological foundations. Software Platform Embedded Systems (SPES) and its follow up extended project (SPES_XT) have been developed to provide for a comprehensive methodological toolkit in model-based development for engineering cyber-physical systems and collaborative embedded systems. While such projects based on MBSE have further advanced the development of automated embedded systems, there still exists the challenge of combining such a methodology with a modelling tool to reduce laborious manual modelling efforts. To address this challenge, we developed a reference implementation of SPES in MagicDraw, a modelling tool, as the SpesML workbench. We developed the SpesML workbench by utilising the wide range of functionalities in MagicDraw: creation of custom language profiles consisting of stereotypes and diagrams, predefined model templates for consistent and ordered model creation, availability of tool functionalities based on the level of modelling expertise of a user, specific API plugins and documentation, all of which help realize the SPES methodology. The SpesML workbench, is therefore, designed to relieve users from the burden of laborious complex system engineering activities and improves the overall user experience in their modelling. Further, the SpesML workbench leverages techniques of separation of concerns by modularising language components fostering re-usability. Naturally, the SpesML workbench does not solve all problems related to systems engineering. As part of the SpesML workbench, we have identified the integration of the modelling tool with a solid MBSE methodology a key aspect in effectively designing complex, heterogeneous systems. We believe the implementation of SPES, as the SpesML workbench, in MagicDraw is beneficial for both novice and advanced modelling users and thus provides a good reference point for future SPES implementations.

Bibliography

[Ad20] Adam, Kai; Michael, Judith; Netz, Lukas; Rumpe, Bernhard; Varga, Simon: Enterprise Information Systems in Academia and Practice: Lessons learned from a MBSE Project. In: 40 Years EMISA: Digital Ecosystems of the Future: Methodology, Techniques and Applications (EMISA'19). volume P-304 of LNI. Gesellschaft für Informatik e.V., pp. 59–66, May 2020.

[Bö21] Böhm, Wolfgang; Broy, Manfred; Klein, Cornel; Pohl, Klaus; Rumpe, Bernhard; Schröck, Sebastian, eds. Model-Based Engineering of Collaborative Embedded Systems. Springer, January 2021.

[Br10] Broy, Manfred: A logical basis for component-oriented software and systems engineering. The Computer Journal, 53(10):1758–1782, 2010.

[BS12] Broy, Manfred; Stølen, Ketil: Specification and development of interactive systems: focus on streams, interfaces, and refinement. Springer Science & Business Media, 2012.

[Bu20] Butting, Arvid; Eikermann, Robert; Hölldobler, Katrin; Jansen, Nico; Rumpe, Bernhard; Wortmann, Andreas: A Library of Literals, Expressions, Types, and Statements for Compositional Language Design. Special Issue dedicated to Martin Gogolla on his 65th Birthday, Journal of Object Technology, 19(3):3:1–16, October 2020. Special Issue dedicated to Martin Gogolla on his 65th Birthday.

[CGR09] Cengarle, María Victoria; Grönniger, Hans; Rumpe, Bernhard: Variability within Modeling Language Definitions. In: Conference on Model Driven Engineering Languages and Systems (MODELS'09). LNCS 5795. Springer, pp. 670–684, 2009.

[Cl15] Clark, Tony; Brand, Mark van den; Combemale, Benoit; Rumpe, Bernhard: Conceptual Model of the Globalization for Domain-Specific Languages. In: Globalizing Domain-Specific Languages. LNCS 9400. Springer, pp. 7–20, 2015.

[FMS14] Friedenthal, Sanford; Moore, Alan; Steiner, Rick: A practical guide to SysML: the systems modeling language. Morgan Kaufmann, 2014.

[Fo10] Fowler, Martin: Domain-specific languages. Pearson Education, 2010.

[FR07] France, Robert; Rumpe, Bernhard: Model-driven Development of Complex Software: A Research Roadmap. Future of Software Engineering (FOSE '07), pp. 37–54, May 2007.

[Gu21] Gupta, Rohit; Kranz, Sieglinde; Regnat, Nikolaus; Rumpe, Bernhard; Wortmann, Andreas: Towards a Systematic Engineering of Industrial Domain-Specific Languages. In: 2021 IEEE/ACM 8th International Workshop on Software Engineering Research and Industrial Practice (SE&IP). IEEE, pp. 49–56, May 2021.

[HKR21] Hölldobler, Katrin; Kautz, Oliver; Rumpe, Bernhard: MontiCore Language Workbench and Library Handbook: Edition 2021. Aachener Informatik-Berichte, Software Engineering, Band 48. Shaker Verlag, May 2021.

[HRW18] Hölldobler, Katrin; Rumpe, Bernhard; Wortmann, Andreas: Software Language Engineering in the Large: Towards Composing and Deriving Languages. Computer Languages, Systems & Structures, 54:386–405, 2018.

[Mo09] Moody, Daniel: The "Physics" of Notations: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering. IEEE Trans. Softw. Eng., 35(6):756–779, November 2009.

[Ni00] Nielsen, Jakob: Designing web usability. 2000.

[PB20] Proper, Henderik A.; Bjeković, Marija: Fundamental challenges in systems modelling. In (Mayr, Heinrich C.; Rinderle-Ma, Stefanie; Strecker, Stefan, eds): 40 Years EMISA 2019. Gesellschaft für Informatik e.V., Bonn, pp. 13–28, 2020.

[Po12] Pohl, Klaus; Hönninger, Harald; Achatz, Reinhold; Broy, Manfred: Model-based engineering of embedded systems: The SPES 2020 methodology. Springer, 2012.

[Po16] Pohl, Klaus; Broy, Manfred; Daembkes, Heinrich; Hönninger, Harald: Advanced model-based engineering of embedded systems. In: Advanced Model-Based Engineering of Embedded Systems, pp. 3–9. Springer, 2016.

[Re18] Regnat, Nikolaus: Why SysML does often fail - and possible solutions. In (Schaefer, Ina; Karagiannis, Dimitris; Vogelsang, Andreas; Méndez, Daniel; Seidl, Christoph, eds): Modellierung 2018, 21.-23. Februar 2018, Braunschweig, Germany. volume P-280 of LNI. Gesellschaft für Informatik e.V., pp. 17–20, 2018.

[Ru16] Rumpe, Bernhard: Modeling with UML: Language, Concepts, Methods. Springer International, July 2016.

[Ru19] Rumpe, Bernhard; Schaefer, Ina; Schlingloff, Bernd-Holger; Vogelsang, Andreas: Special Issue on Engineering Collaborative Embedded Systems (Editorial). SICS Software-Intensive Cyber-Physical Systems, 34(4):173–175, December 2019.

[Se03] Selic, Bran: The pragmatics of model-driven development. IEEE software, 20(5):19–25, 2003.

[TK05] Tolvanen, Juha-Pekka; Kelly, Steven: Defining Domain-Specific Modeling Languages to Automate Product Derivation: Collected Experiences. In: Software Product Lines, 9th International Conference, SPLC 2005, Rennes, France, September 26-29, 2005, Proceedings. pp. 198–209, 2005.

Challenges in Multi-View Model Consistency Management for Systems Engineering

Sebastian Bergemann[1]

Abstract: A way to handle the complexity of cyber-physical systems is model-based systems engineering (MBSE) with multiple viewpoints. These viewpoints satisfy different concerns, but they likely have information dependencies and overlaps among each other. Inconsistencies can be introduced whenever there are changes in only some of the views without consistent synchronization in other dependent views. In this paper, we motivate why consistency management is important in multi-view MBSE and define requirements for it. By analyzing the State of the Art, we identify limitations in (multi-view) consistency management approaches, especially for inconsistency detection. Besides general performance issues, we notice primarily that most approaches are limited to or at least tested on only very specific views and tools with homogeneous models and few specific predefined consistency rules. Furthermore, in most approaches we cannot find solutions regarding subsequent updates of consistency rules by the user, allowance of tolerating inconsistency, and handling confidentiality. These literature gaps pose open research challenges for making multi-view consistency management more applicable in the industry.

Keywords: consistency management; multi-view modeling; model-based systems engineering

1 Introduction

The development of cyber-physical systems (CPSs) is much more difficult than of usual software systems since CPSs combine software with hardware, embedded in the physical world [KS08, TS18]. An approach to reduce the resulting higher complexity of CPSs is model-based systems engineering (MBSE) with multiple viewpoints. The key aspect of MBSE is the usage of models instead of or in combination with textural documents to design a system. A model is an abstraction of something real that holds enough information to represent this real subject regarding a specific purpose but nothing beyond to simplify the handling of it. Related to this abstraction concept are views and viewpoints, which play an important role in this paper. We are following their definition in ISO 42010 [In11]. A view is like a specific perspective on the system. It is a model representing the system regarding specific concerns. Therefore, it contains or represents only the information needed for these concerns while hiding everything else, resulting in a lower complexity of the system (for these concerns). A viewpoint is like the instructions for a view where it is specified how to construct it, meaning that a view is an instance of a viewpoint. Although views often represent different information due to different concerns, they might still share some

[1] fortiss GmbH – Research Institute of the Free State of Bavaria, Munich, Germany; bergemann@fortiss.org

information of the system or are in other ways related, which is why dependencies and overlaps between them exist. Therefore, it is likely that inconsistencies between views will occur if several engineers are working on the same system in different views (of different viewpoints). We want to clarify that we do not mean different views of a single model, i.e., a single source of truth, because in this case it is much less likely or even impossible to have inconsistencies between the views. We are referring to views without a single source of truth, meaning that we have different models or reused parts of models in other models. Unsolved inconsistencies between such views can cause problems at the latest when the system should be realized based on the combined information of the inconsistent model(s). Therefore, multi-view model inconsistencies need to be detected as early as possible so that it can then be decided if the inconsistency can be tolerated or if it needs to be resolved.

Multi-view consistency is much harder to identify than consistency within a view due to the many, often hidden, relations and dependencies between views. We provide in Sect. 2 our own requirements for this specialization. The importance of automatic (multi-view) consistency management in the industry is also mentioned by Jongeling et al. [Jo22], who provide a current State of Practice analysis regarding agile model-based development with consistency management in several industrial settings. Besides giving an overview about current research in this field, the authors point out the struggles the industrial companies have with consistency management resulting in often insufficient and/or manual solutions.

In this paper, we are continuously using the term consistency, but to be precise we are meaning model consistency, which is about consistency within and between models during MBSE. Therefore, the context is always MBSE including for example SpesML [Mod22], where modeling in multiple viewpoints plays a major role.

1.1 Contribution

We want to highlight challenges in the field of multi-view consistency, which can mostly be applied to general consistency management as well. The challenges are derived from limitations of current approaches for managing multi-view consistency, mainly detecting inconsistencies between multiple views and viewpoints. The limitations are based on our current literature analysis, which we will present in an overview, and on requirements we define for applicable multi-view consistency management in the industry.

1.2 Outline

According to our contribution, we first provide the basis in Sect. 2 with requirements and supporting examples for multi-view consistency management in the industrial setting, and then continue in Sect. 3 with the literature analysis overview including current limitations. This results in the challenges presented in Sect. 4. We conclude in Sect. 5 with a summary and an outlook of our next steps.

2 Requirements for Multi-View Consistency Management

We first provide the basis with some requirements for a good applicability of multi-view consistency management in the industry. We do not limit this to the software engineering domain but keep it open to systems engineering, where cross-domain consistency management is often needed. The upcoming requirements in Sect. 2.2 might not be complete, but based on our experience they are the most important ones while not being trivial. We have collected the requirements mainly by listening to feedback from industrial partners in previous research projects, where they told us with what problems they have to deal and what they still need to have an efficient workflow. Before we show the requirements in Sect. 2.2, we first want to give a short example of a setting in which the requirements are important. This should help to better understand the requirements, although we will stay abstract with the example since a detailed example would be out of scope for this paper.

2.1 Introducing Example

Several users, usually engineers, from different companies are working in different views on the same product, e. g., an autonomous vehicle. These views are heterogeneous: they separately model requirements, the logic, the technical implementation, the 3D appearance (via CAD), and the simulation of the system. Relations exist between elements (intra- and inter-view) like references to or from other elements (e. g., requirements tracing or allocation/deployment) or shared information (e. g., identical/related names). Some relations are important to exist/persist in a specific way, which is why consistency conditions are defined to specify which properties these relations need to fulfill.

Within this general setting, we create the example scenario that a user wants to reuse a part of a sensor from a library or another system. The part is reused inside the logic view, but it has also modeled information inside the technical and the CAD view. Due to the reuse, information in these views has changed, which means that probably some relations between the views have changed as well. Therefore, it needs to be checked if they are still according to the consistency conditions, because otherwise the change could have created mismatches between views, resulting in (future) failures. Defining and checking of consistency conditions are usually easy for simple relations, like related names, but it becomes challenging in our example due to the following four aspects:

A1) Not explicitly modeled information. The following consistency condition exists: *The weight of a product part must be identical over all views that are referring to this product part*. The sensor has an explicitly annotated weight inside the technical view, but inside the CAD view the weight is not explicitly stored. Nevertheless, the weight information (indirectly) exists also in the CAD view, because the geometry and material is modeled there and based on this data, a weight can be deduced in the CAD view. Therefore, the consistency check needs to consider this not explicitly modeled information as well, since not only the annotated weight in the technical view could have changed due to the reuse but also the geometry or material in the CAD view, resulting in a new deduced weight. This means that the consistency check needs to know about this not explicitly modeled information, and it needs a deduction mechanism to get the weight information from the CAD view based on the modeled geometry and the material.

A2) Semantic relevance. Another consistency condition exists: *The step size in the simulation view needs to fit to all simulated logical and technical components modeled in their views*. The updated sensor in the logic view has a worst-case execution time of 0.005 s, in the technical view it has a sampling rate of 100 Hz and the step size in the simulation view is 1 ms. A consistency check solely based on the syntactical value would find that the value of the worst-case execution time is less than the value of the step size in the simulation view and consider the consistency condition as fulfilled. However, if the unit is taken into account, it can be seen that the condition is actually violated, because a worst-case execution time of 0.005 s is 5 ms, which is larger than the step size of 1 ms. Therefore, the consistency check needs to be aware of the semantics, otherwise it will not work. It also needs to know that 100 Hz is 0.01 s or 10 ms to compare it with the step size. Hard coding this for every consistency check is not efficient and error-prone.

A3) Changing situation. Due to the reuse and the resulting new situation, the engineers gain new insights like new technical annotations they need to add and respect in the model. Therefore, the engineers want to add new consistency conditions or update existing ones for future consistency checks. This can be difficult or impossible, because their consistency tools have only predefined consistency conditions that cannot be changed or only by the developers of the tools and not the engineers as end-users. This means that consistency tools/concepts without the possibility of easy extension will be inefficient and, in some cases, unusable for modern industrial projects, especially in the agile context.

A4) Confidential information. The internal logic of the reused sensor part is confidential because it was developed by another vendor. Usually, requirements will make sure that the confidential parts are kind of consistent with the remaining parts, e. g., which ASIL level can be guaranteed, but especially in agile development the situation can often change, e. g., a higher ASIL level might be required after the reuse. An option would be to exchange again every time the requirements with the other partners, but especially for a first quick overview it would be more efficient if consistency conditions could be checked also in combination with confidential parts (without violating confidentiality).

These challenging aspects in our example will be converted in the next section into our requirements for multi-view consistency management, with which we will continue during the remaining paper.

2.2 Derived Requirements

The given examples of challenging aspects while applying multi-view consistency management in the previous section can be converted to these requirements (R1 is based on A1, R2 on A2, etc.):

R1) It should be possible that consistency can also be checked based on not explicitly modeled information. Information might not always be modeled and stored explicitly in the view but can exist hidden behind a combination of other information. This not explicitly modeled information is for consistency as important as explicitly modeled information and needs to be deduced for the consistency check. Therefore, the consistency checking mechanism not only needs to be aware of such hidden information, but also needs to know how to access it via a deduction mechanism.

R2) Inconsistency detection should also use semantic information. As shown in the example in Sect. 2.1, just comparing the syntax and pure values will not lead every time to a correct consistency statement. Consistency checks need to be aware of the semantics as well, which is usually more difficult than syntax checks.

R3) Consistency specifications should be extensible at any time. In the context of agile development but also beyond, it happens often that new insights occur, which needs to be applied as soon as possible to the system model. This can also result in the need to update the relations and consistency conditions. Therefore, the end user, and not the developer of the consistency tool, needs to have the possibility to easily extend the conditions.

R4) Inconsistency detection should be possible even with confidential model parts. In industrial projects it is common that model parts are confidential, especially if different manufacturer are involved. In this case, it will become difficult to get all information needed for the consistency checks. Outdated requirements after agile iterations and a continuous back and forth with the partners of the confidential parts about requirement changes will consume much time. A way to check for consistency with even confidential parts can be beneficial.

We will use these four requirements to analyze approaches in the literature (see Sect. 3) and to identify open challenges (see Sect. 4).

3 Literature Analysis

In this section we analyze the State of the Art in literature for how well it can fulfill the requirements of Sect. 2.2. Based on these findings we will identify open challenges later on in Sect. 4.

For the State of the Art analysis we have analyzed 124 papers. These papers are not directly the result of a systematic literature review but were continuously collected during our overall research and orientation in the field of consistency management. We used the literature collections of studies and reviews like [CCP19, Pe13, TvBS20, UTZ17, KM18, LMT09] as foundation and filtered them for multi-view consistency management approaches. During further research in this area, we found more interesting multi-view consistency management papers, which we added to build our current paper base. One of our next plans is to extend this literature analysis and provide a systematic literature review to have a more structured base.

3.1 Current Approaches

As a short overview, we will first briefly discuss the categorization of consistency approaches and afterwards present the three most interesting approaches regarding the requirements of Sect. 2.2.

Categorization. The literature does not provide a universal categorization of consistency approaches. Spanoudakis et al. [SZ01] differentiate between logic-based approaches, model checking approaches, specialized model analysis approaches, and human-centered collaborative exploration approaches. Torres et al. [TvBS20] provide not directly categories but list strategies to keep consistency between different domains, e. g., parameter or constraint management, or ontology. Other categorizations exist as well, but they are often focused on UML approaches [KM18, LMT09, Hu05]. A valuable contribution is the categorization by Feldmann et al. [Fe15]. They distinguish the approaches in proof-theory-based, rule-based, and synchronization-based (e. g., via model transformations). Similarities exist to Haesen et al. [HS05], where the approaches are labeled by how they generally manage consistency: by construction, by monitoring or by analysis. Consistency by construction matches to Feldmann's synchronization-based category, where an approach generates a new consistent model based on a changed one. Consistency by monitoring means simple rule checking that is executed after every new change and is similar to Feldmann's rule-based category. Consistency by analysis means one algorithm is run once in the end to check for all defined possible inconsistencies, but in our opinion, this is very similar to the monitoring category, because both are basically executed checks based on rules where only the time and frequency of the checks are different. What we did not find as explicit category is consistency by design, where the viewpoints and thus the resulting views are already designed from the beginning on in a way that inconsistencies cannot occur due to the precisely defined relations between the viewpoints. In summary, we think that all categories and their approaches can be fitted into the following three general categories: consistency be design (everything where the concept and the system of viewpoints are already designed to prevent inconsistency), consistency by rule checking (everything where models are checked explicitly for inconsistency based on rules), and consistency by construction/generation (everything where models are (newly) generated to preserve consistency).

1) Consistency by design. The *SPES* methodology [Po12] aims on consistency by design, because the underlying concept is defining the different viewpoints and their relations in a

consistent way and predefined well-formedness rules are preventing basic inconsistencies like in the currently ongoing SpesML project [Mod22]. However, the concept does not enforce consistency in all aspects but rather suggests specific designs and leaves it to the user to ensure final consistency. Therefore, inconsistency between views can still occur and they can only be detected if they are covered by the predefined well-formedness rules, otherwise the inconsistency remains hidden.
Regarding requirements of Sect. 2.2: Although it is theoretically possible to hard code the consideration of not explicitly modeled information (R1) and semantics (R2) inside the well-formedness rules, it is not really supported with this approach. The well-formedness rules are predefined and not intended to be modified and extended by the end user (R3). Confidentiality in connection with consistency (R4) is not yet a topic within *SPES*.

2) Consistency by rule checking. A rule-based approach is the *Model/Analyzer*, which was first proposed for solely UML models [Eg06] and was then generally extended [Eg11]. The approach is not directly about how to check consistency and define consistency rules. It rather sees the consistency rules as black boxes to check which model elements are used for each rule. This way the scope of a rule is determined, and it provides a fast and automatic way to identify which rules need to be evaluated again after a certain model change instead of just checking everything each time or a typed-based scope checking. Although this is an important improvement and provides flexibility, it mainly leaves all the difficulties to the developer of the consistency checker and rules, and the engineer as end user. The approach ignores the semantics of rules and model information and does not specify how to handle different heterogeneous views and models. However, this is also not directly the scope of the paper. We also assume that the consistency checker needs to have direct access to the whole model (set), which is critical regarding confidentiality.
Regarding requirements of Sect. 2.2: As already mentioned, the requirements regarding not explicitly modeled information (R1), semantics (R2) and confidentiality (R4) are not met. However, an advantage is definitely the extensibility (R3), because this is one of the few approaches where the end user can freely extend the consistency checks by any language.

3) Consistency by construction/generation. As a third major direction of research we want to present a consistency by construction or synchronization-based approach, which is *Vitruvius*. It was developed over several dissertations and papers, e. g., [Bu13, Kr17, La17, Kl18, Kl21], and proposed with a virtual single underlying meta-model (V-SUMM). Instead of explicitly collecting all information in one central model (like in Orthographic Software Modeling (OSM) [ASB09]), the idea is to combine all meta-models virtually to one V-SUMM, coupled by model transformations. These transformations are based on consistency conditions and will generate again consistent models of the other meta-models after a model change. This has many benefits, but we still see several limitations. 1) V-SUMM is currently only focused on software engineering and not generally on systems engineering. 2) Model transformations themselves have limitations and they are difficult to define for heterogeneous views. 3) They also hinder easy consistency modifications by the later engineers because for example adding a new consistency specification would mean

updating the whole V-SUMM and model transformations. 4) Only greenfield development is well feasible because of the assumption/requirement of V-SUMM that the whole model structure is in the beginning already consistent [An18]. 5) V-SUMM does not have the possibility to tolerate inconsistency. The generation is executed automatically and will always enforce consistency preservation.
Regarding requirements of Sect. 2.2: We could not find any source stating that this approach is dealing with the confidentiality issue (R4). As already mentioned, it is difficult to update the model transformations regarding new or modified consistency conditions and it is not intended for the end user to do this (R3). Due to the model transformations it might also be difficult to consider semantics (R2) and not explicitly modeled information (R1).

3.2 Current Limitations

Based on the previously mentioned literature research and to the best of our knowledge, we see following limitations that exist in the literature on consistency (management). None of the approaches in literature can fulfill all requirements from Sect. 2. We will divide them into 1) the group of missing features, 2) the group of too high restrictions and 3) the group of general performance problems.

Missing features. As many others [Ba91, We18, Eg11], we argue that inconsistency is tolerable in some cases, either temporarily for tests or even permanently depending on the inconsistency type. Therefore, a consistency approach should enable this option. However, many approaches force automatic consistency preservation either by direct (re-)construction without separate inconsistency detection or by forced inconsistency repair after detection. The next feature we miss in many approaches is the possibility to update the consistency specifications at any time after the actual consistency approach was implemented. Most approaches can only take the predefined consistency rules into account or provide a poor extension interface. However, consistency rules might change and evolve over time, especially during agile development, which is why the engineer should always be able to modify them. The last major feature is about confidentiality. In the industry this is a critical issue and approaches should at least consider this in some way during consistency checking but based on our research this is nearly never the case.

Not generalizable. We are sure that it is not possible to have a completely generic approach for consistency management across domains, but the level of restrictions and specialization in current approaches is often too high and thus not suitable for industrial applicability. This starts with the usage of very specific consistency rules that cannot consider, e. g., semantics or not explicitly modeled information. The next level is limiting oneself to only few specific views if different views are included at all. Finally, consistency can in many approaches only be checked for specific models that are usually solely in the software domain and/or very homogeneous, where defining consistency conditions is not complicated. As an example, many approaches, especially the older ones, are completely focused on only UML models.

General performance problems. Some approaches are not scalable for large models and projects, especially if they hold high complexity. In addition, consistency approaches should be fast and executable at any time, or at least after each model change, to have the possibility of immediate feedback, but this is not to case for several reviewed approaches. We do not want to focus on these performance limitations, but to ignore them is also not an option regarding industrial applicability.

In sum, all these limitations make it difficult to apply current consistency approaches in the industry, where project models are usually heterogeneous, based on multiple views and tools, often confidential, large in scale, and should be easy to handle by engineers who are usually not experts in programming. This leads to the open challenges presented next.

4 Challenges

The previously mentioned limitations in the literature show that the stated requirements in Sect. 2.2 are not yet covered. Therefore, we can derive from them open challenges that are beneficial to solve for a good applicability of multi-view consistency management in the industry. For all following challenges it should be considered to enable the possibility of inconsistency toleration, because always directly forcing automatic consistency preservation is a limitation (see Sect. 3.2).

Consistency between heterogeneous views. Inconsistency detection across multiple views is already difficult because of the many dependencies between them, which are not always easily identifiable. However, it becomes even more complex when the views are heterogeneous and not only limited to the software domain. In addition, they might even be distributed among different tools, requiring tool interoperability. Overall, this makes it more difficult to compare them and finding as well as checking dependencies between them.

Not explicitly modeled information. Checking consistency relations is becoming more challenging when the information on which the relation is based is not explicitly modeled. With explicitly modeled information we mean directly accessible/stored information in the model. If it needs to be checked because of a consistency condition, it can easily be received. However, if the information is not explicitly modeled, we need to deduce the needed information from other explicitly modeled information (with a deduction mechanism). Such deduction needs to be possible during consistency checks.

Semantic-related consistency. We consider consistency checking as challenging when it is semantic-related. This is already mentioned as missing feature for many approaches in the literature [CCP19, TvBS20]. It is even more important for multi-view consistency because a system information can be interpreted and represented differently by different views. If we only consider syntax and not semantics during the comparison, we might easily get false positives or false negatives during inconsistency checks.

Continuous extensibility. Another challenge we see, especially in the context of agile development but also beyond, is the extensibility possibility of the consistency management system by the engineer. It is easy for developers of consistency management approaches to predefine consistency rules and only consider them for the consistency checks. However, if the engineer as end-user gets new insights during agile development or has very specific project-related consistency rules, it should be possible to extend and update the consistency rules at any time, especially during the actual modeling. This extensibility gets even more challenging when the extension and modification of consistency conditions should not be very restricted but more user-friendly, i.e., more understandable for end-users like engineers.

Confidentiality. The confidentiality issue is broadly factored out in the field of consistency, but especially for large CPSs it is very common that different companies are working together, and they do not want to expose all of their own intellectual property. However, this information hiding makes it difficult to check consistency. A possibility is to only check consistency for the parts that are accessible and everything else is based on pure requirements that are given to the other companies for their black box models. For agile development this is getting more difficult and costly because new insights might result in changes of own models and to ensure that they are still consistent with the external black box models, requirements might need to be updated or the suppliers need to be asked directly. This could be saved when consistency checks would be possible while still respecting confidentiality.

5 Conclusion

As part of this challenge paper, we have shown that consistency management between views plays an important as well as challenging role for MBSE. For this purpose, we have provided requirements for a good industrial applicability of multi-view consistency management.

Based on our literature analysis, we have stated that the combination of these requirements is not yet fulfilled by state-of-the-art approaches. Therefore, we have derived several open challenges for multi-view consistency management in industrial systems engineering. The presented list of requirements does not aim to be exhaustive. Hence, there can very well be further challenges not mentioned in this paper.

The derived challenges are based on the major difficulties of not explicitly modeled information, the relevance of semantics, the heterogeneity of occurring consistency relations, the need for continuous extensibility of them, and handling confidentiality. All of this is even more complex when consistency is managed between multiple tools. However, overcoming these challenges will make consistency management more applicable and efficient for the industry, especially with respect to agile development.

As a next step, we are planning a systematic literature review of specific multi-view consistency management and comparing it with the State of Practice to confirm our current research as well as refine it.

Acknowledgement

This work was in parts supported by the German Ministry for Education and Research (BMBF) in the SpesML project (`https://spesml.github.io/index.html/`; grant number 01IS20092G).

Bibliography

[An18] Ananieva, Sofia; Klare, Heiko; Burger, Erik; Reussner, Ralf: Variants and Versions Management for Models with Integrated Consistency Preservation. In: Proceedings of the 12th International Workshop on Variability Modelling of Software-Intensive Systems. pp. 3–10, 2018.

[ASB09] Atkinson, Colin; Stoll, Dietmar; Bostan, Philipp: Orthographic Software Modeling: A Practical Approach to View-Based Development. In: Evaluation of Novel Approaches to Software Engineering. Springer, pp. 206–219, 2009.

[Ba91] Balzer, Robert: Tolerating inconsistency. In: Proceedings of the 13th International Conference on Software Engineering. pp. 158–165, 1991.

[Bu13] Burger, Erik: Flexible views for view-based model-driven development. In: WCOP 2013 - Proceedings of the International Doctoral Symposium on Components and Architecture. pp. 25–30, 2013.

[CCP19] Cicchetti, Antonio; Ciccozzi, Federico; Pierantonio, Alfonso: Multi-view approaches for software and system modelling: a systematic literature review. Software and Systems Modeling, 18(6):3207–3233, 2019.

[Eg06] Egyed, Alexander: Instant consistency checking for the UML. In: Proceedings of the 28th International Conference on Software Engineering. pp. 381–390, 2006.

[Eg11] Egyed, Alexander: Automatically detecting and tracking inconsistencies in software design models. IEEE Transactions on Software Engineering, 37(2):188–203, 2011.

[Fe15] Feldmann, Stefan; Herzig, Sebastian J.I.; Kernschmidt, Konstantin; Wolfenstetter, Thomas; Kammerl, Daniel; Qamar, Ahsan; Lindemann, Udo; Krcmar, Helmut; Paredis, Christiaan J.J.; Vogel-Heuser, Birgit: A Comparison of Inconsistency Management Approaches Using a Mechatronic Manufacturing System Design Case Study. In: IEEE International Conference on Automation Science and Engineering. IEEE Computer Society, pp. 158–165, 2015.

[HS05] Haesen, Raf; Snoeck, Monique: Implementing Consistency Management Techniques for Conceptual Modeling. In: Proceedings of the 3rd International Workshop on Consistency Problems in UML-based Software Development. Springer, 2005.

[Hu05] Huzar, Zbigniew; Kuzniarz, Ludwik; Reggio, Gianna; Sourrouille, Jean Louis: Consistency Problems in UML-Based Software Development. In: Lecture Notes in Computer Science. volume 3297. Springer Verlag, pp. 1–12, 2005.

[In11] International Organization Of Standardization: , ISO/IEC/IEEE 42010:2011 - Systems and Software Engineering – Architecture Description, 2011.

[Jo22] Jongeling, Robbert; Ciccozzi, Federico; Carlson, Jan; Cicchetti, Antonio: Consistency management in industrial continuous model-based development settings: a reality check. Software and Systems Modeling, 2022.

[Kl18] Klare, Heiko: Multi-model Consistency Preservation. In: Proceedings of the 21st ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings. pp. 156–161, 2018.

[Kl21] Klare, Heiko; Kramer, Max E.; Langhammer, Michael; Werle, Dominik; Burger, Erik; Reussner, Ralf: Enabling consistency in view-based system development — The VITRUVIUS approach. Journal of Systems and Software, 171(110815), 2021.

[KM18] Knapp, Alexander; Mossakowski, Till: Multi-view consistency in UML: A survey. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), volume 10800, pp. 37–60. Springer, 2018.

[Kr17] Kramer, Max E.: Specification Languages for Preserving Consistency between Models of Different Languages. Phd thesis, Karlsruhe Institute of Technology (KIT), 2017.

[KS08] Karsai, Gabor; Sztipanovits, Janos: Model-Integrated Development of Cyber-Physical Systems. In: Software Technologies for Embedded and Ubiquitous Systems. volume 5287. Springer Berlin Heidelberg, pp. 46–54, 2008.

[La17] Langhammer, Michael: Automated Coevolution of Source Code and Software Architecture Models. Phd thesis, Karlsruhe Institute of Technology (KIT), 2017.

[LMT09] Lucas, Francisco J.; Molina, Fernando; Toval, Ambrosio: A systematic review of UML model consistency management. Information and Software Technology, 51(12):1631–1645, 2009.

[Mod22] Model-based system engineering with SysML and SPES methodology. `https://spesml.github.io/index.html/`, 2022. Online; accessed 16 April 2022.

[Pe13] Persson, Magnus; Törngren, Martin; Qamar, Ahsan; Westman, Jonas; Biehl, Matthias; Tripakis, Stavros; Vangheluwe, Hans; Denil, Joachim: A Characterization of Integrated Multi-View Modeling in the Context of Embedded and Cyber-Physical Systems. In: Proceedings of the International Conference on Embedded Software. IEEE Computer Society, 2013.

[Po12] Pohl, Klaus; Hönninger, Harald; Achatz, Reinhold; Broy, Manfred: Model-Based Engineering of Embedded Systems: The SPES 2020 Methodology. Springer, 2012.

[SZ01] Spanoudakis, George; Zisman, Andrea: Inconsistency Management in Software Engineering: Survey and Open Research Issues. In: Handbook of Software Engineering and Knowledge Engineering: Volume I: Fundamentals, pp. 329–380. World Scientific, 2001.

[TS18] Törngren, Martin; Sellgren, Ulf: Complexity Challenges in Development of Cyber-Physical Systems. In: Principles of Modeling, pp. 478–503. Springer International Publishing, 2018.

[TvBS20] Torres, Weslley; van den Brand, Mark G.J.; Serebrenik, Alexander: A systematic literature review of cross-domain model consistency checking by model management tools. Software and Systems Modeling, 20(3):897–916, 2020.

[UTZ17] Ul Muram, Faiz; Tran, Huy; Zdun, Uwe: Systematic review of software behavioral model consistency checking. ACM Computing Surveys, 50(2), 2017.

[We18] Weidmann, Nils: Tolerant Consistency Management in Model-Driven Engineering. In: Proceedings of the 21st ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings. 2018.

Machine trainable software models towards a cognitive thinking AI with the natural language processing platform NLX

Felix Schaller[1]

Abstract: Since the last decade, machine learning, especially with artificial neural networks, has triggered a new quantum leap in computer science. Despite the considerable achievements, these applications still lack a general purpose approach for artificial intelligence (AI). The main reason is the absence of the ability for cognitive reflection or self-awareness. They are mainly highly specialized trained patterns that can solve intricate problems but cannot describe themselves. I would like to contrast this with a new method of trainable software models that shall be capable for self-awareness. Implemented in the project Natural Language Platform NLX it shall be demonstrated that self-aware AI is key for human-like cognitive tasks. The hypothesis claims that to reach this goal, machines require to describe its system context semantically by a formal model. Neuronal networks are good at specific tasks, but the trained patterns cannot derive a reasoning for the trained solution. Only that it satisfies its intended functionality - but not why. Creating formal models instead of patterns has turned out, that the formal nature of natural language is the best to reach that goal of a self-aware AI. Certainly there are other AI's that do natural language processing with neuronal networks. But most of the models try to resolve the content with too rigid constraints and with little attention to the context. For this project context plays a key role to resolve the meaning of natural language. If the context is resolved correctly, such AI can be used for general purpose tasks resolving anything imaginable.

Keywords: Machine Learning; Software Models; AI; Cognitive thinking; Cybernetics and Systems Theory; Computer linguistics

1 Machine learning today

The last decade can be seen as the great breakthrough in artificial intelligence, where finally by the leading role of neuronal networks, computation solutions could be achieved which where not possible before by using primarily formal methods and algorithms. Meanwhile almost all groundbreaking improvements by AI go on the account and the subdomain of neuronal networks. They perform superior especially in domains where there are no formal solutions possible, because no formal rules are available [GBP17]. Such domain is in particular the computer vision domain where patterns play a major role. But also for other domains like natural language processing, neuronal networks meanwhile conquered a lot of computational domains where they now rule over formal methods and algorithms that attempt to do the same task [LBH15]. In the last few years the project openAI [Op21] with

[1] Validas AG, IT research, Arnulfstr 27, 80335 München, Germany schaller@validas.de

their newest language model GPT-3 [FC20] has achieved astonishing results, where this AI can even create intricate software code from natural language [Ch21] that is even executable. Also in other domains of natural language processing GPT-3 in conjunction with neural theorem prover has achieved astonishing results by solving some Math Olympics problems [Po22]. But yet when summarizing all these success stories the impression prevails, that these solutions can be good at a specific domain, but yet still require a human guided setup or operation frame to master the task. In other words it needs to be specialized for the domain to perform. Meaning that it would neither be capable to develop the solution by it's own nor to reflect it's result. Something you would expect by a general purpose tool. So it can be confidently stated it is yet a stupid - indeed a very efficient stupid - machine.

2 The nature of neuronal networks

What makes us confident in that claim? The reason lies in the nature of neuronal networks itself. Though if stacked together in huge networks they can even easily surpass a human on that specific domain, but still they can only perform tasks for what they are designed for. It can nevertheless be considered a general-purpose tool from this point of view, so that the problem posed may in fact be of general-purpose nature. But yet it processes only tasks for which it is specifically tailored for and not any task by one single interface like a HAL 9000, best known from Stanley Kubricks "A Space Odyssey". The reason is those networks do the task, but they do not know why. They doesn't even know if the result produced is correct, except that it satisfies an intended cross reference. But here lies the rub, that to develop unique solutions, an AI requires a formal reason.

Neuronal networks are not capable by nature for that task because they currently supersede on problems nobody knows a formal solution for, so you unleash a neuronal network on it to intuitively find a solution with heuristic methods, like humans develop a gut feeling for a certain domain by experience, a network achieves positive success rate over time. After achieving a finite amount of training the network became confident in estimating the result. But no matter how precise the network performs, it's yet an estimation - indeed a highly sophisticated estimation - of the solution. This solution is anyway just a product of the training set posed to the network and in an unluckily constellation in the selection of training material during training, the network later may show unintended behavior leading to false positives [FS10]. One reason are unknown patterns on a subliminal level or patterns that seem not obviously relevant by human interpretation being mixed in the training data which influences the final result, best known since decades from the 'tank problem' [Ri19]. Another example are adversarial attacks by noise. By nature a neuronal network is not so resistant against such type of signal jamming [NKP19]. Here the major reason is, that the network is applied to the raw data finding patterns on the atomic level. Would the signal instead be transposed by Fourier Transformation, a noise jamming could be isolated easily [ECK19].

2.1 Functional safety

Based on the fact, that no matter how sophisticated the network is, it is at the end still a highly sophisticated guessing, achieved through training experience by no formal rule. That makes it to the ultimate tool for unresolved problems but on the other hand makes those networks actually insufficient for functional safety domains [GB19]. Currently there are several safety standards under development which discuss criteria and evaluation methods under which circumstances they are acceptable. [Ts20] [et20]

2.2 Natural language processing and neuronal networks

As natural language has yet been seen as too ambiguous for formal methods the solution lays near to use that available tool that can handle problems with unknown formal rules. And as the success story of the GPT-3 model reveals [He21], it does this job pretty well and precise although language is seen as something too ambiguous for deterministic tasks like creating software code. The reason is, that the hypothesis of the ambiguous nature of language is actually not true. It is only true if you look on literal terms as formal or imperative expressions. And that's why solving natural language was yet prevailed to the domain of neuronal networks. But language has indeed a very formal nature and can be very powerful if you look at natural language from a different point of view. This matter will be very essential for the topic of this paper and will be dedicated in particular in a later chapter.

3 The formal method and machine learning

As laid out in the chapters before, it is obvious that machine learning methods - in particular with neuronal networks - computation has penetrated into areas which were not available before with formal methods. But that alone would not make a system more "intelligent". Because yet it does not provide a reasoning, at least and most for itself to build formal conclusions from it. The simple reason is that to know what I am dealing with, I require to describe it semantically. Furthermore this semantics requires a context. The context becomes key in that matter, and in this context also lies the reason why formal methods fail where machine learning succeeds - because they do not consider a context.

3.1 Structural science and the Hilbert program

When looking today on the domains of math and information science it is all about reasoning. In particular universal validity of claims and their proof. This was believed till Alan Turing introduced its Turing Machine as answer to David Hilbert's Program for an axiomatic proof theory [Pl14]. Turing demonstrated with this machine theory, that there is no ultimate law, that any task can be solved by formal rules. This paradigm thus defines the foundation of all information theory - but actually it is only half true. It is only true when applied in the frame and to problems of the David Hilbert Program. Although the problems stated there gather all formal problems in information technology till today. Basically what all axiomatic theorems do, is to set up a claim that pretend to have an ultimate validity and the algorithm has to be just sophisticated enough to handle that problem. To break out from that paradigm it requires to look at it from a different perspective. If an algorithm does not resolve or the theorem does not prove it's not necessarily the algorithm, but the constraints that are set up that cannot be fulfilled. So if a program does not return from it sub-states it does not automatically mean the problem is unsolvable, but the constraints are either specified wrong (misconception of the author) or are underspecified (missing information). Practically this is the result of bug-fixing where developers getting feedback about the misconception or suitable preconditions for the algorithms. It's the essence of all systems, that they require the right context to work in. Even in nature species can only exist in their suitable biotope. Axioms and theorems do not come with such a system context, thus the context must be created first and ensured it suits the formal method properly. Something like an immune system for theorems. That's mostly the job of functional safety. The functional safety area therefore closely examines the product and reports under which circumstances safety is ensured.

The deepest insight I got in that problematic myself when designing a sophisticated DSL to parse natural language documents. It requires to have so many information on certain abstraction levels that it is almost impossible to cast this into a rigid abstract syntax tree of a DSL. But on the other hand it revealed such a deep insight how system contexts are made up and that certain information is required on a certain branch of abstraction to decide further in higher levels of abstractions. Alongside it provided also a rough impression how neuronal networks learn such proper tree structures by a heuristic approach. I suggest that such an approach could maybe one day help to find a solid theory how neuronal networks develop their decisions. Nevertheless due to the high amount of semantic information, a DSL cast into unchangeable formal code is an unsuitable approach to create a solid DOM structure, but sufficient to provide a proof of concept for the natural language processing. So far to say; that whole structure problematic in the DSL alone holds enough material to occupy with it in a separate Paper.

Carrying on with the problematic of formal methods, the actual reason why they fail can be summarized:

1. Because they claim to have ultimate validity. All constructed models that reflect a certain aspect of a real problem are true only in a certain area. No model can claim from itself being valid for all circumstances.
2. If a Turing Machine fails because it does not return from its pushdown state it means the problem is underspecified and lacks constraints it requires to solve the posed problem.

3. Because of the self-entitlement of ultimate validity of axiomatic theorems they claim to work context free.
4. Due the negligence of meta structures in the data, those axioms are applied on raw and unstructured data on an atomic level. Thus lacking a structured semantic of the data and therefore any formal method will fail, because ambiguity is not resolved properly.

3.2 The key role of context

As already introduced in the last chapter, the problem with all formal methods is, that a theorem cannot stand by its own. All formal constraints require a certain frame where they can be seen as valid and that frame represents a suitable context.
EXAMPLE 1: Try to qualify a software for a functional safety domain. Here it is mandatory to draw a line around finite tasks, named use case, of within they can be declared as safe after undergoing a certain process of validation. Any contemporary program is intended to work in any context so the degree of freedom in that program is reduced to that amount of states to work safe under any circumstance the user can bring it to and what the safety case allows.
EXAMPLE 2: A further example what's closely related is the already mentioned challenge in the last chapter. The challenge to develop a grammar for a DSL that can be unleashed on any non-formal text, which is capable to resolve the document's structure with it as DOM-tree. DSL's will work reliably on formal languages, that's what DSL's where designed for, but there will be no ultimate valid and error-free grammar to solve the content of natural language documents - including the interpretation of the document structure - without a certain knowledge of the context of the text to resolve.
Further examples in other domains are like automating tasks in continuous integration (CI). It normally requires always someone who sets it up manually. The cannot be feed into an automated process, because those automation processes are lacking knowledge about what the given data represents and what shall be processed. Yet the maintenance of such CI pipelines is restricted to a developer, due to there is no additional information for the system to source in order to fix itself in case of an unexpected state or condition. Such systems thus require additional design information. And such information can only be provided by a proper knowledge of the context they process.

3.3 Fractals and meta structures of reality

By the theoretical proof through the Turing Problem someone may formulate a law, that reality is so complex it cannot be described by formal methods and therefore forget that formal methods though work but just within a certain frame. Applying it on a given context it reveals that there are so many ambiguities to resolve properly. The very reason for that is, that the reality has a self similar nature. I make a bold claim that this circumstance is causing all the ambiguities when applying a certain theorem on a matter. Not that it is just self similar but also fractal, those self similar features build upon fractal meta structures which build the ground truth and are the backbone of any cybernetic system. As it can be shown that systems work within a certain frame, it also shows that parts of the system can be described formally. Therefore, when there exist rules for sub-systems based on the fractal axiom, the entire system can be formally described in systems of systems, where each super-system create conditions where subsystems can exist. Gregory Bateson a pioneer in cybernetics and systems theory shaped the term of “meta structures” in his book “Mind and Nature” [Ba79]. He sees those structures as immanent and as a implicit consequence of growing complexity in systems. This means for a practical application, that without a proper knowledge of the context you are unable to distinguish self similar features from each other on an atomic level. Many different problems appear identical on an atomic level and thus cannot be distinguished. This problem leads to an ambiguous situation how to distinguish and categorize the data features. Unfortunately most applications work solely on the atomic level without a structure semantics. Simple experiments with computer vision mimicking the human perception can be made to understand the need to create meta structures of the given data and relate them to the given context. Here may be given two distinct problems

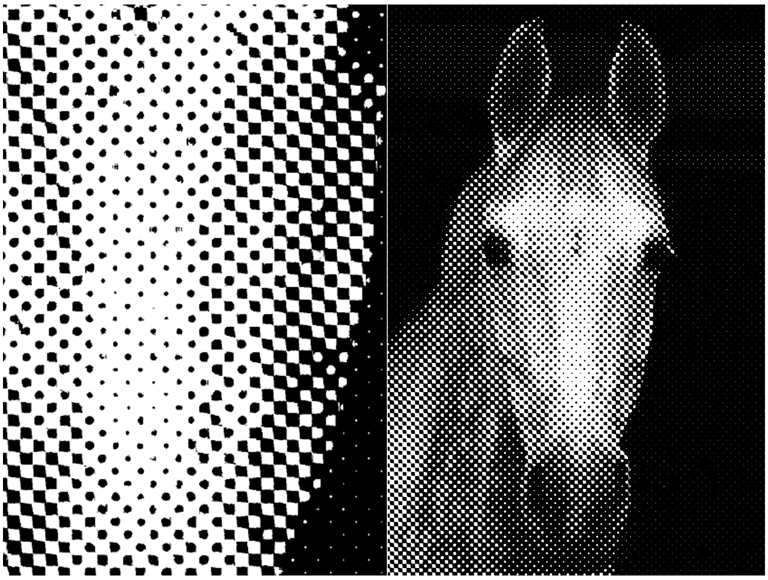


Fig. 1: (left) a closeup view on the raster does not reveal the motive, while on the right picture the motive can be clearly percepted

where this matters:

- Taking the halftone image as example in the Fig. 1, looking here in a closeup on the patterns would not reveal the content of the matter. Only when you zoom out,

Fig. 2: Problems in older computer vision algorithms like PTAM: Feature-trackers get confused by redundant patterns in the camera image which makes the result unusable

the human perception would resolve a pattern of a white horse. Which means the information is hidden when just looking on the atomic level of the patterns itself. They do not really distinguish from each other, because a single feature does not carry the information. The information is encoded deeper in the meta-structure.

- Same problem occurs when designing computer vision algorithms for autonomous driving or augmented reality. Resolving the observer position from a time sequence of a video-camera, it requires to find recurring patterns in the image sequence to track and locate them over time to render the motion of the observer in 3D-space. Such algorithms are know as SLAM algorithms (**S**imultaneous **L**ocalization **A**nd **M**apping). Earlier Algorithms have solved it by adding markers on very distinct features in the images like the PTAM algorithm in Fig. 2. But such an algorithm gets confused if it is opposed to redundant features in the image like on a checker board. In this case it is unable to distinguish to which of those features it shall assign, given by the last image frame. The result is, that trackers are jumping between the redundant features. This creates a jittering path, which makes a further resolution of the observer position impossible.

Concluding from these examples, it seems obvious that for a proper interpretation of posed problems it will be essential to develop a structured semantic for them that they can be formally resolved. Otherwise the problem gets stuck in ambiguity. So the next question arises here: how to build up meta structures on data to derive a context so it can be processed with formal methods.

4 Towards formal models in AI

Carrying on with the success story of neuronal networks from above I mentioned the circumstance that they penetrate in areas where there is no formal rule known by applying heuristic methods of intense training. By training the neuronal network derives hidden meta structures of the trained content and stores it in its patterns of several hidden layers. But those meta structures do not own a semantic. So they can not explain what they are doing and develop reasoning for that. Thus it requires to develop a semantic to interpret those meta structures.

4.1 Introducing software models to reduce complexity

Even when working with software models which promise by structuring the matter to reduce complexity, like models from the SysML and UML domain, they can help to sustain an overview for the user. So if created by experienced system engineers it can speed up the development process and in some cases it even can be generated software code from. But as the modeling languages UML or SysML often want to reserve the flexibility to describe systems, users can create huge mess. Many modeling automation projects in the industry, starting with great ambitions aiming to automate the development process and to leverage systems and software development to the next level not seldom died an inglorious death. Some of them are still riding a dead horse because the companies already poured millions of their budged into the project fearing all that investment to be lost if they redesign their concept from the bottom up, so they insisted to carry on and exacerbate. I have consulted some of them and was not reluctant to provide them with an honest analysis of the tricky situation. The main reason in those projects were that there was no proper modeling semantic given that maintained a certain methodology hygiene. Especially, because in many modeling tools available on the market neither type checking nor any other automatism for checking the methodology is implemented. With the final result that the created models were useless for the further processing in downstream system domains and can finally only served as a form of a documentation of the process. This circumstance arises the need for a proper modeling theory. For that purpose the FOCUS theory of distributed systems by Broy et.al. was developed [BS01] This modeling theory is currently implemented in the SPES modeling framework which is constantly being developed further in various research projects and industrial applications [Bö16]. Having a proper concept of various viewpoints and the concept of granularity levels, the framework does a good job in improving the quality of model based systems. Meanwhile it has been widely established in the industry - and added with a proper tooling, it has the potential becoming the new standard in systems engineering.

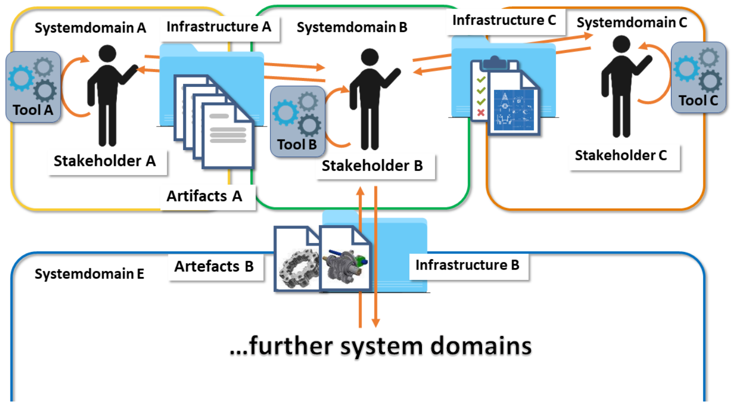


Fig. 3: Exchanging data artifacts across system domains require intense manual administration

4.2 Getting the developers knowledge into the model

As shown in the last chapter that a proper methodology accompanied with a powerful framework is key for a good success story in model based software engineering, and if such a proper methodology is properly implemented it can avoid nightmares in systems engineering. But although the model based approach improved the development in software and systems, those models still requires the interpretation of the developer because the models do not own its own semantics. To the computer all software models are simply a finite collection of nodes and edges - nothing more. Yet that is nothing that could live up with the fancy claim of a cognitive thinking AI as promised in the title, but we are not there yet. This is the case simply due to the lack of knowledge for the machine - and that remains with all data artifacts in information technology: They create meaning for the developer or expert, but not to the machine itself, simply because they hold no meta information for the machine about what they are and how they have to be treated. First and most, literals that are used by the developer to name identifiers give meaning to the developer and to understand what it is about, but the machine lacks an interpretation of the literals to create its own semantic.

4.3 Introduction of Meta Information

A first step to help the machine resolving the context would be to implement a semantic by introducing meta information as an overlaying layer that tells the machine how to handle the model elements. This meta information introduce a classification system with a proper taxonomy to describe a model element - lets say we have a model that describes artifact flow within a tool chain and contained within a model element that shall represent a specific tool e.g. a GCC compiler. Then we assign a class hierarchy which derives that specific compiler tool from a generic compiler up to higher super classes: tool, executable etc. Further it is defined that this tool has certain interfaces to other tools. Now when we then assign that class to a tool in a tool chain the modeling tool knows what kind of data that tool can interchange with other tools. If we now want to define an artifact flow from one tool to another, the modeling tool already can do some automatic assignment because of the semantic given in that meta-description. By that it can derive constraints of what kind of data can be interchanged between those tools. By building up such meta-information the system engineer no longer need to explicitly assign any data connections by himself manually but the software already knows what connections are possible and can assign all appropriate connection by themselves. Here at that stage it would be already a semi automatic and proactive tool. Such proactive actions can be a great help, but frequently proactive behavior in programs create a huge nightmare.

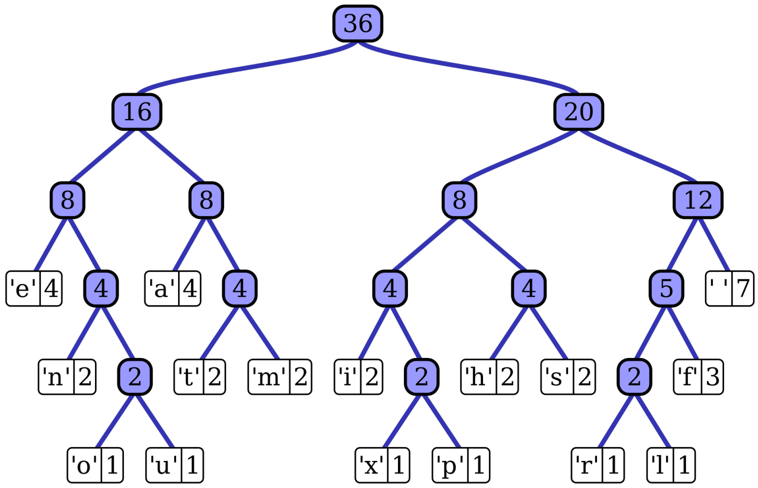


Fig. 4: A Huffman Tree describes a hierarchy of redundancies and is widely used for data compression and image-processing. It can be seen as a meta-structure describing redundant features in a context

4.4 Language as a modeling tool

Talking about natural language processing in the previous chapters, the current practiced approach is achieved by neuronal networks. Language expressions are yet widely understood as too ambiguous to formalize expressions and theorems. But this is only half true. Language is a tool that can be very precise in expressing. Looking at language as formal and imperative commands doesn't work. The nature of language is not imperative but rather cooperative. Language explicitly requires a context to work. At the beginning of a conversation the actual statement is often unclear. By its declarative nature, semantic in language is created by interaction with the other party. The interaction goes on until all uncertainties are clarified between the parties. This is important because:

- The background knowledge of both parties is mostly different, so common knowledge about the matter must be established.
- Terms in language are not of axiomatic nature and thus do not have a fixed meaning. So the real semantic of a term is either constrained by the context or requires to be negotiated with the other party, till both parties reached an consensus.
- It has a fractal nature and that makes language as a very efficient and information dense modeling tool, where it only requires detailed descriptions where it does not derive automatically by the given context
- A further fractal feature of language is a proper taxonomy of the terms where terms derive from a hierarchic taxonomy structure. This analogy is already successfully instrumented by object oriented programming (OOP) scheme and thus a big step towards natural language processing of instructions. It has turned out since its invention, when the OOP paradigm is understood properly it's a great help in designing applications.

This and more features of natural language makes it very efficient in the expression and communication of intentions because many things can be implicitly derived from the context and do not have to be explained in a redundant manner. Through its truly declarative nature, language has the ability to build formal models that can be used for cognitive human like problems and tasks.

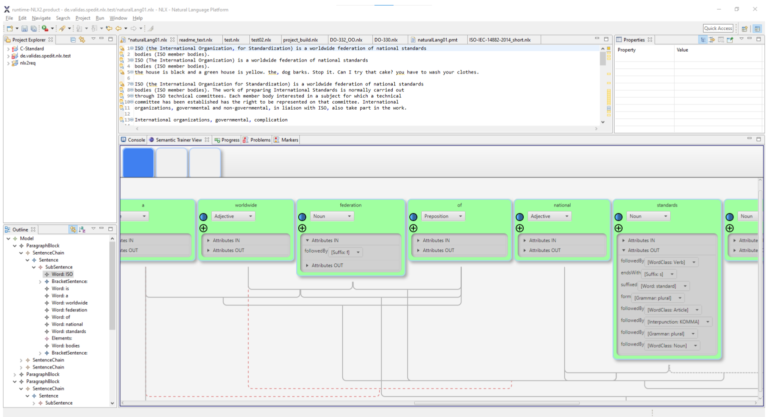


Fig. 5: Overview of the NLX tool. (top) Text document, (bottom) interactive grammar trainer, (bottom left) DOM-tree

5 Experiment: Proof of concept with the natural language project NLX

Based on the prior considerations about natural language processing in a truly formal manner with machine trainable software models, the NLX project was set up. Currently aimed on the goal to provide a proof of concept for the hypothesis of natural language based on machine trained models. Also to show that such models are capable to resolve cognitive tasks and provide an argued reasoning for its conclusions. Something that to our knowledge was not yet published or proven in the field of artificial intelligence. Going more in detail of the architecture, the prototype is built up in two parts. A front-end application, written in Java with the Eclipse Modeling Framework. As back-end I am using the Neo4j spatial graph database to formalize the semantic structure and to resolve patterns with the graphical querying language Cypher.

As textual pre-processor for language processing, the front-end currently uses a DSL, where the grammar is tailored in that way, that it can parse natural language documents containing various structure components like:

- Paragraphs
- Chapters with chapter number
- Sentences
- Tables
- Bullet point and numbered lists
- IT-Words like paths, emails, urls, camel case
- Source code
- Footnotes

Based on these structures it can not only resolve natural language content but also interprets the structure of the document itself. The DSL transfers the textual input into a document object model tree (DOM-tree). On top of this tree all kinds of other generators can be adapted to generate e.g. XMI-Models from that structure. So apart from the unfinished prototype of natural language processing the tool yet offers the working DSL platform that can be used to generate all kind of other products on top that can make use of the structure derived by the DOM-tree. This offers already a rich selection of ready to use use cases which can be set on top of that core platform. At Validas AG we e.g. generate requirements models from industrial standards that integrate the requirements from those standards into other software models.

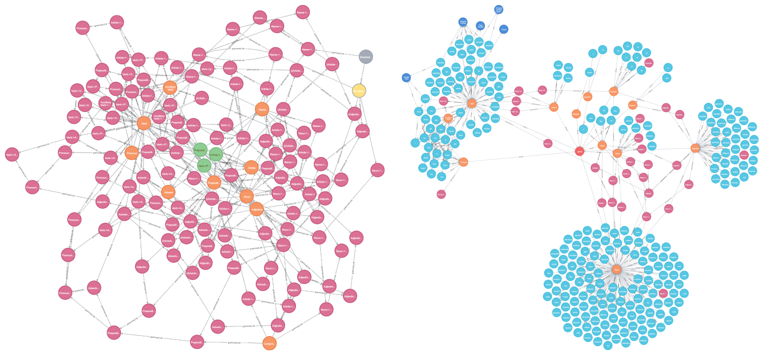


Fig. 6: (left) grammar rules network, (right) rules related with lexical dictionary and other attributes

5.1 Current state of project: Grammar tree and trainer

Currently the development of the natural language processing is in the state of developing a grammar trainer which resolves the sentences in a grammar model. At the moment it trains a grammar structure interactively on a given context of text samples. By working very close with a context it has implemented several common NLP modules like:

- Structure trainer
- Exception pattern trainer
- Lemmantizer
- Tense extraction
- Build-in implicit rules
- Implicit rules-builder

With these features it can be trained in which context certain words types are allowed and what words could be subtyped by providing extra patterns through training input of right and wrong detection. This helps to cluster words and their features in a model with high granularity. Yet not all intended features are working stable, but when this all works reliably then in the future an interlinked ontology model and constraint provers will be built on top. Those provers then would have the role to validate the created context of the statements and derive formal processes from it like analysis tasks, process automation, code generation and many more. The grammar trainer already can be seen as a first proof of concept of machine trainable models. It attempts to resolve the sentence structure like finding an exit through a maze. If the entire sentence has found one root and all branches cover the entire sentence the sentence is resolved. The resolved sentence will then be transferred into the overall ontology of the document. With the grammar trainer it is intended to train a model that is capable to resolve a sentence structure and separates the parts of subject predicate and objects. Further it determines time and modes like passive, active, conjunctive. What it basically does is that it intends to create a tree structure of a sentence. Starting with the detection of the right word type where it determines whether the word is a noun, verb, adjective and so on it accesses the database to derive the trained rules in which words can stand in. As a word can have different types, like the word “use” could be a noun or a verb and thus according to the given context it can detect in which type this word acts at that position of the context. The sentence structure resolves then in more and more higher hierarchies, until a final root for the entire sentence is found. When such a hierarchy of a sentence is resolved it can be parsed for entities to build an ontology via entity relationship models. The solution of a

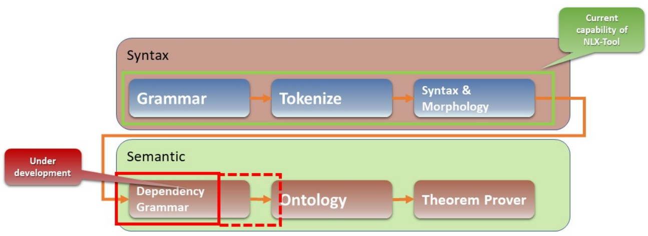


Fig. 7: Current development status of the project

trainable grammar is chosen by the reason, that constructed rules would be too unflexible for the almost infinite variances of sentence structures, thus the structures shall be trained with a training interface to improve the grammar model constantly. Currently the extension of the training capabilities are ongoing and are estimated to be complete soon.

5.2 Future tasks

Before the ontology model can be started the grammar trainer has yet to become more versatile. When this is finished the entities can be parsed into an ontology to derive its functional relationships and its attributes. This document-internal ontology shall then be linked to other related ontologies outside the document or with a database of ontologies acting as the background knowledge of the system. All this is then fed into a kind of “constraint prover” which does the validation of the statements on one hand and resolves the logic on the other hand. Using this prover can formulate all kinds of logical problems converted into software models that can be interpreted or turned into executable code.

6 Conclusion

Based on the general idea of this paper and the experiment findings it can be said: A machine is not automatically intelligent because it is trained excessively and it will finally reach human intelligence if the AI is just big enough. That humans have developed cognitive capabilities is rather exceptional and is able because language acts as a tool for the human consciousness to build formal models. With this mental models the human is capable understand and explain its environment and himself. The outcome of this ongoing experiment shall provide the final evidence that natural language has that essential capability to create conscious machines.

References

[Ba79] Bateson, G.: Mind and nature: A necessary unity. New York: EP Dutton. Bateson, MC (1994). Peripheral visions: Learning along the way, 1979.

[Bö16] Böhm, W.; Daun, M.; Koutsoumpas, V.; Vogelsang, A.; Weyer, T.: SPES XT Modeling Framework. In (Pohl, K.; Broy, M.; Daembkes, H.; Hönninger, H., eds.): Advanced Model-Based Engineering of Embedded Systems: Extensions of the SPES 2020 Methodology. Springer International Publishing, Cham, pp. 29–42, 2016, ISBN: 978-3-319-48003-9, URL: https://doi.org/10.1007/978-3-319-48003-9_3.

[BS01] Broy, M.; Stølen, K. In: Specification and Development of Interactive Systems: Focus on Streams, Interfaces, and Refinement. Springer New York, New York, NY, 2001, ISBN: 978-1-4613-0091-5, URL: https://doi.org/10.1007/978-1-4613-0091-5_2.

[Ch21] Chen, M.; Tworek, J.; Jun, H.; Yuan, Q.; Pinto, H. P. d. O.; Kaplan, J.; Edwards, H.; Burda, Y.; Joseph, N.; Brockman, G.; Ray, A.; Puri, R.; Krueger, G.; Petrov, M.; Khlaaf, H.; Sastry, G.; Mishkin, P.; Chan, B.; Gray, S.; Ryder, N.; Pavlov, M.; Power, A.; Kaiser, L.; Bavarian, M.; Winter, C.; Tillet, P.; Such, F. P.; Cummings, D.; Plappert, M.; Chantzis, F.; Barnes, E.; Herbert-Voss, A.; Guss, W. H.; Nichol, A.; Paino, A.; Tezak, N.; Tang, J.; Babuschkin, I.; Balaji, S.; Jain, S.; Saunders, W.; Hesse, C.; Carr, A. N.; Leike, J.; Achiam, J.; Misra, V.; Morikawa, E.; Radford, A.; Knight, M.; Brundage, M.; Murati, M.; Mayer, K.; Welinder, P.; McGrew, B.; Amodei, D.; McCandlish, S.; Sutskever, I.; Zaremba, W.: Evaluating Large Language Models Trained on Code, 2021, URL: https://arxiv.org/abs/2107.03374.

[ECK19] Esmaeilpour, M.; Cardinal, P.; Koerich, A. L.: A robust approach for securing audio classification against adversarial attacks. IEEE Transactions on Information Forensics and Security 15/, pp. 2147–2159, 2019.

[et20] et.al., J. M. C.: Concepts of Design Assurance for neural networks (CoDANN) - easa.europa.eu, Mar. 2020, URL: https://www.easa.europa.eu/downloads/112151/en.

[FC20] Floridi, L.; Chiriatti, M.: GPT-3: Its nature, scope, limits, and consequences. Minds and Machines 30/4, pp. 681–694, 2020.

[FS10] Forman, G.; Scholz, M.: Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement. Acm Sigkdd Explorations Newsletter 12/1, pp. 49–57, 2010.

[GB19] Gharib, M.; Bondavalli, A.: On the evaluation measures for machine learning algorithms for safety-critical systems. In: 2019 15th European Dependable Computing Conference (EDCC). IEEE, pp. 141–144, 2019.

[GBP17] Granter, S. R.; Beck, A. H.; Papke Jr, D. J.: AlphaGo, deep learning, and the future of the human microscopist. Archives of pathology & laboratory medicine 141/5, pp. 619–621, 2017.

[He21] Heaven, W. D.: OpenAI's New Language Generator GPT-3 is shockingly good-and completely mindless, Oct. 2021, URL: https://www.technologyreview.com/2020/07/20/1005454/openai-machine-learning-language-generator-gpt-3-nlp/.

[LBH15] LeCun, Y.; Bengio, Y.; Hinton, G.: Deep learning. nature 521/7553, pp. 436–444, 2015.

[NKP19] Naseer, M.; Khan, S.; Porikli, F.: Local gradients smoothing: Defense against localized adversarial attacks. In: 2019 IEEE Winter Conference on Applications of Computer Vision (WACV). IEEE, pp. 1300–1307, 2019.

[Op21] OpenAI, June 2021, URL: https://openai.com/.

[Pl14] von Plato, J.: The development of proof theory, Oct. 2014, URL: https://plato.stanford.edu/entries/proof-theory-development/.

[Po22] Polu, S.; Han, J. M.; Zheng, K.; Baksys, M.; Babuschkin, I.; Sutskever, I.: Formal mathematics statement curriculum learning. arXiv preprint arXiv:2202.01344/, 2022.

[Ri19] Riley, P.: Three pitfalls to avoid in machine learning, 2019.

[Ts20] Tsukiyama, A.: ISO/PAS 21448:2019, June 2020, URL: https://www.iso.org/standard/70939.html.

J. Michael, J. Pfeiffer, A. Wortmann (Hrsg.): Modellierung 2022 Satellite Events,
Digital Library, Gesellschaft für Informatik e.V. 106

Modeling an Anomaly Detection System with SpesML

An Experience Report

Maximilian Junker[1], Henning Femmer[2]

Abstract: SpesML is an instantiation of the SPES methodology for cyber physical systems using SysML. However, SpesML is still under development and urgently requires evaluation with practical examples. This experience report describes our study of using SpesML for an anomaly detection system. The goals of the case study are to evaluate feasibility, benefits, and shortcomings of both the tool and the methodology iteratively at early stages of the project. The results are already promising with respect to both methodology and tool; however, the work continuously identifies suggestions for adaptations and future work regarding both.

Keywords: Model-based Systems Engineering (MBSE), Iterative Development, SPES, SpesML

1 Introduction

Many companies are currently discussing introducing Model-based Systems Engineering (MBSE) into their processes. However, adopting MBSE in the industrial practice still comes with considerable effort to define, implement, and customize a suitable MBSE method and tooling.

The SPES series of projects are a joint effort by industry and academia to provide a methodology for MBSE to ease introduction [Br12]. Its primary aim is to base MBSE on a precise system model. Previous project succeeded in defining this methodology.

In the SpesML project the additional aim is to port SPES to SysML, a standardized notation that is picking up dissemination in industry, and to provide a tool that supports the SPES method. Part of such an endeavor is then, of course, to validate the developed methods and tools in exemplary cases.

In this work, we present the case of developing an anomaly detection system using the SpesML methodology. Due to the early stage of both methodology and research, this paper qualitatively evaluates the feasibility of the methodology, as well as the expressiveness and usability of the tooling.

[1] Qualicen GmbH, Rosa-Bavarese-Straße 15, 80639 München, Germany, maximilian.junker@qualicen.de
[2] FH Südwestfalen University of Applied Sciences, Haldener Straße 182, D-58095 Hagen, Germany, femmer.henning@fh-swf.de

In the following, in Sec. 2 first look at background and related work. Afterwards, in Sec. 3, we describe the case study and the different iterations, in which we created the model. We then describe the resulting model in Sec. 4. Finally, we discuss these results w.r.t. to the aforementioned research questions in Sec. 5, and summarize the presented work in Sec. 6.

2 Background and Related Work

This work is based on the MBSE frameworks SPES and SpesML. SpesML itself is based on SysML, for which few methodologies exist. In the following, we summarize these and provide pointers for further reading.

2.1 SysML and SysML Methodologies

SysML is a standardized modeling language for systems engineering. It is originally based on UML but extends UML to better match the needs for modeling systems instead of software. Please refer to [FMS14] for a detailed introduction into SysML [OM19]. SysML however, is a language, not a methodology. It therefore only provides the building blocks and endless possibilities how to apply the language.

Various approaches have tried to fill this gap. In no particular order, the most prominent approaches probably are Dassault's own MagicGrid [Mo20], the ARCADIA approach [Ro16; Ro17] strongly tied to the eclipse capella tool[3], and the SYSMOD methodology [We16].

2.2 SPES & SpesML

For simplicity, we do not want to explain SPES in all detail here. Please refer to any of the published material for fundamentals [Bö14a; Br12], extensions [Bö21; Po16], case studies [Bö14b] or introduction methodology [We21] for this. Instead, we just provide a very rough overview in this chapter.

SPES is a framework for MBSE. It defines a set of models to describe different aspects of a system under development with varying level of detail. To this end, SPES defines four core viewpoints:

- Requirements Viewpoint: Contains the requirements to the system.
- Functional Architecture Viewpoint: Contains the system function of the system and breaks those down into whitebox functions.

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

- Logical Architecture Viewpoint: Contains a component architecture of the system which is independent from the technical realization.
- Technical Architecture Viewpoint: Contains an architecture of technical, discipline-specific components (e.g., mechanical components and software components)

Additionally SPES defines the concept of layers of granularity which allows to identify subsystems and develop those independently. Finally, all models in SPES are based on a common universal interface model, a common system model, and an overarching architecture model providing the base for refinement and tracing across viewpoints and levels of granularity.

While SPES is independent of a specific tool and modeling language, SpesML is an instantiation of SPES using SysML and the commercial modeling tool Cameo Systems Modeler[4] by 3ds Dassault Systemes. The SpesML workbench is a plugin to Cameo providing SPES concepts as well as advanced analyses and simulation in Cameo.

2.3 Research Gap

Currently, there are no published experience reports for applying SPES to SysML with SpesML. This work addresses this gap.

3 Study Setup and Execution

In this chapter, we describe the goals of the study, the study object, and how the study was executed.

3.1 Goals of the Case Study

At this early stage, the goal of this case study is to identify potential for improvement in the method as well as in the tool. In consequence, the goals of the case study are two-fold:

- First, the case study shall evaluate the applicability of the SPES method and potentially serve as a blueprint for creating MBSE models based on SPES.
- Second, the case study shall evaluate the expressiveness and usability of the SpesML tooling.

[4] https://www.3ds.com/products-services/catia/products/no-magic/cameo-systems-modeler/

3.2 Study Object: Anomaly Detection

The System under Development is an anomaly detection system (ADS). The ADS example is taken from the following real world problem: When sensors fail, it is often not obvious that they are leaving a nominal operation mode, e.g., but not replying anymore at all. Instead, the sensors continue providing data, but just *incorrect* data. This is usually recognized by irregularities (i.e., anomalies) in the provided sensor data. The main task of an ADS system, therefore, is to use sensor data originating from a monitored system (e.g., a machine) to detect anomalies which could potentially lead to damage of that system. The ADS works in two phases: In the first phase it monitors the sensor data while the monitored system works nominal. From the data gathered in this way, the ADS creates a benchmark. In the second phase, the ADS uses this benchmark to asses the sensor data and determine if there is an anomaly.

3.3 Study Subjects

The model was created by the two authors of this work, as well as two further employees of Qualicen GmbH. Three modelers are SysML and SPES experts, whereas for one modeler it was his first expose to SPES, but not his first to SysML.

3.4 Case Study Execution

The case study was conducted during the course of the research project. The tooling used was developed in parallel with the study execution and the development of the model. Accordingly, the model was created in several iterations. In each iteration we changed the model according to changes in the method as well as integrated new aspects.

During the course of the modeling, we documented findings regarding the applicability of the method, possible improvements, and workarounds. However, we did not perform any specific analysis to uncover problems, but instead used an opportunistic approach. Hence, the list of issues cannot be considered complete.

Iteration 1: Scope and high-level requirements In the first iteration, we defined the scope of the system under development and developed an initial set of high-level stakeholder requirements. At this very early stage of the research project, there was only preliminary tool support. We nevertheless created this scope document and the requirements in the tool as Cameo supports these artifacts out of the box and we could later make adoptions as the SpesML tool made progress.

Iteration 2: Initial logical architecture In the second iteration, we would have ideally designed the functional viewpoint. However, since the logical viewpoint of SpesML was finished earlier, we decided to create a prototype of the logical architecture ahead, which we would later refine. Therefore, we created an initial logical architecture in the second iteration, where we defined coarse components, e.g., for data processing, storage, and interfacing.

Iteration 3: Requirements Refinement & Functional Architecture In a third iteration, we refined the initial coarse grained stakeholder requirements with a larger set of fine-grained system requirements. These included functional requirements as well as quality requirements and design constraints. From the functional requirements we developed a functional black-box model, containing the system functions located at the system interface. We then refined these black-box functions by giving a functional white-box model for each system function. We added trace-links connecting the different requirements levels as well as connecting the requirements to the functional architecture.

Iteration 4: Refined logical architecture and simulation In the fourth iteration, we refined the logical architecture to faithfully realize the functional architecture and the requirements. We further added behavior descriptions to logical components in order to be able to simulate the components.

4 Resulting Model

In the following, we show the models developed in the requirements-, logical-, and functional viewpoint. Although there is a preliminary technical viewpoint available, we will not go into details here, since at the time of writing, the tool support and the method for the technical viewpoint has not been completed yet.

4.1 Requirements Viewpoint

In the requirements viewpoint we have three main artifacts: (1) the scope document, (2) the stakeholder requirements, and (3) the refined system requirements. For the scope document we used the Free Forms Diagram, which is provided by Cameo and which supports creating informal documents containing text and images. The document describes the ADS and provides background information. For the stakeholder requirements and the system requirements, we created a SpesML Requirements Package as well as a SpesML Requirements Table. An excerpt can be found in Fig. 1. It shows the a subset of the functional requirements, focussing on reporting aspects (i.e. querying the sensor data by various variables). The only further requirements attributes are status, text, and type.

#	Name	Status	Text	Requirement Type
1	System Requirements			
2	Functional Requirements			
3	Reporting			
4	Query Sensor Data by Time	reviewed	When the system receives a sensor query specifying a time range, the system returns sensor data with timestamps in that range.	Functional
5	Time Data from Database	proposed	The system checks time range from database.	Functional
6	Data with Timestamps	proposed	The system returns sensor data with timestamps according to database time range.	Functional
7	Query Sensor Data by Source	reviewed	When the system receives a sensor query specifying data sources, the systems returns sensor data that originate from that source.	Functional
8	Specified Source Identification	proposed	The system identifies the specified source and retrieves the source's sensor data from database.	Functional
9	Sensor Data from Source	proposed	The system returns the sensor data originating from the specified source.	Functional
10	Query Sensor Data by Value	reviewed	When the system receives a sensor query specifying a value range , the system returns sensor data with values in that range.	Functional
11	Range of Values from Database	proposed	The system retrieves all sensor data values withing the specidifed value range from database.	Functional
12	Sensor Data by Value	proposed	The system returns all values withing the specified value range.	Functional
13	Query Sensor Data by Anomaly	reviewed	When the system receives a sensor query specifying an annomaly id, the systems returns all sensor data which contributed to the anomaly.	Functional

Fig. 1: An excerpt of the system requirements table.

4.2 Functional Viewpoint

For the functional viewpoint we first created a functional black-box model which contains the system functions (i.e., the functions located at the system boundary). For ADS, the black-box model consists of three system functions (see Fig. 2):

1. **Manage Benchmarks** to create and manage benchmarks of sensor data,
2. **Detect Anomalies** to monitor a system for anomalies during regular application, and
3. **Reporting**, to allow to retrieve historical sensor data and warnings based on a query.

Additionally, the method allows to have communication between black-box function, when the communication relates to a mode of the system. In our case, we used such mode channels to model the communication of created sensor benchmarks and warnings.

For each system function we created a whitebox model detailing how a system function is realized by a network of communicating whitebox functions. Fig. 3 shows an example of the whitebox model for the system function *Reporting*. In this case the whitebox model consists of eight whitebox function describing the internals of the black box system function. Note that the interface seen from the black-box perspective is the same as seen from the whitebox perspective.

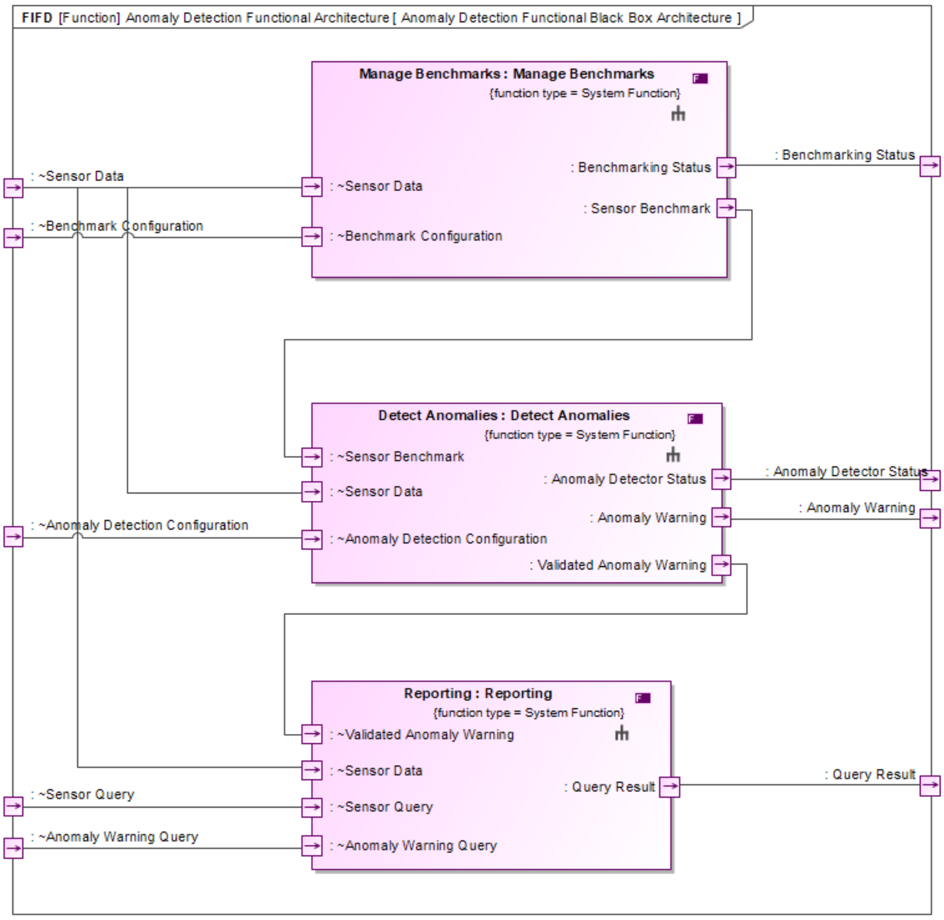


Fig. 2: Functional Black-Box Architecture of the Anomaly Detection

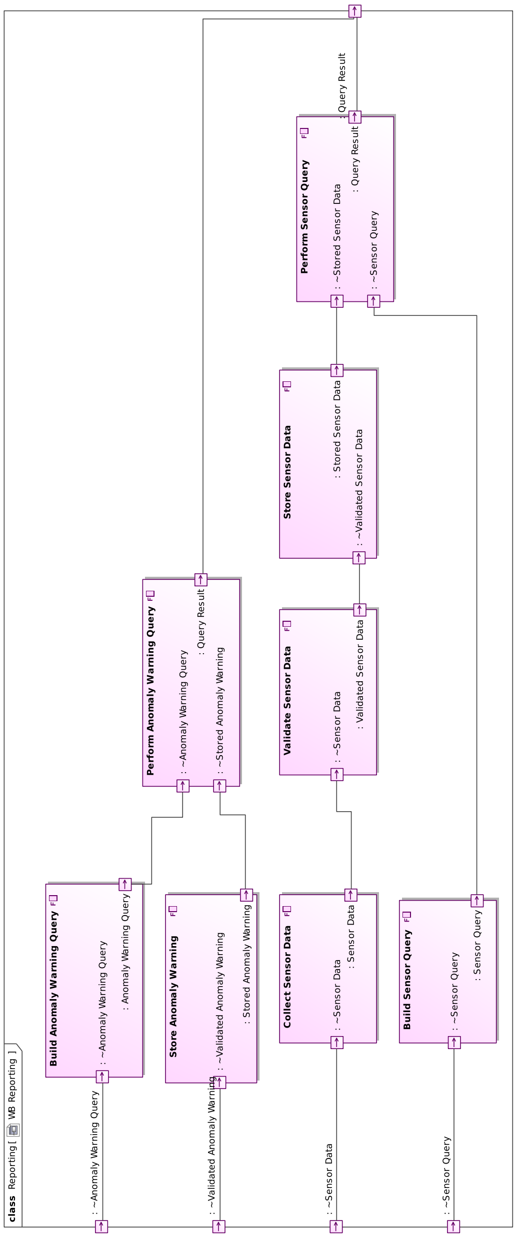


Fig. 3: Whitebox model of the system function Reporting

4.3 Logical Viewpoint

The logical viewpoint of the ADS describes how the functionality described by the functional viewpoint can be realized in a consistent architecture. Where in the functional architecture duplicate functionality exists, e.g., for preprocessing, this is resolved in the logical architecture (for example by a central data processing component). We created a logical architecture (see Fig. 4 with several decomposition levels (not to confuse with layers of granularity). Most components on the highest level were further broken down into further logical components.

Just when we could not break down a component any further, we modeled the behavior of the component using a state machine. Fig. 5 shows the state machine of the model controller. This component controls wether the system is recording a benchmark, performing monitoring or none of those.

4.4 Tracing

Apart from creating the artifacts outlined above, we established tracing relationship between different artifacts. Specifically, we created the following types of trace links

- From stakeholder requirements to system requirements
- From quality system requirements to functional system requirements, when a quality requirement (e.g., security) is realized through a system function or a functional requirement to a system function
- From functional requirements to black-box functions and whitebox functions
- From requirements to logical or technical components, in case a requirement is not realized by a functionality.
- From whitebox functions to logical components

Figure 6 shows an extract of the trace links between the system requirements and the functional architecture.

5 Discussion: Feasibility and Findings

The goal of the case study was, first, to evaluate the applicability of the SPES method, and, second, to evaluate the expressiveness and usability of the SpesML tooling.

Overall, we could so far successfully model the ADS. However, we found the following issues: During the modeling and project internal review rounds we gathered issues regarding the method and the current tooling. Below we report a selection of these issues.

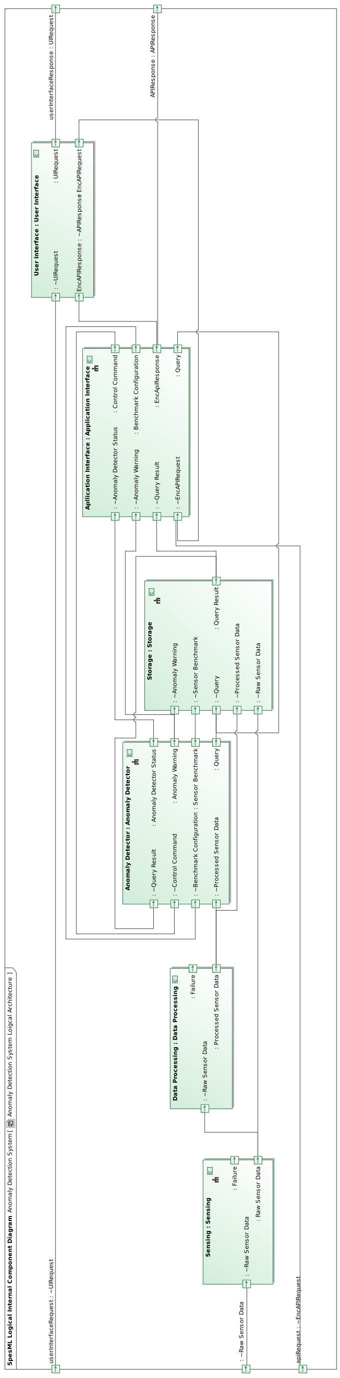


Fig. 4: Logical Architecture of the Anomaly Detection

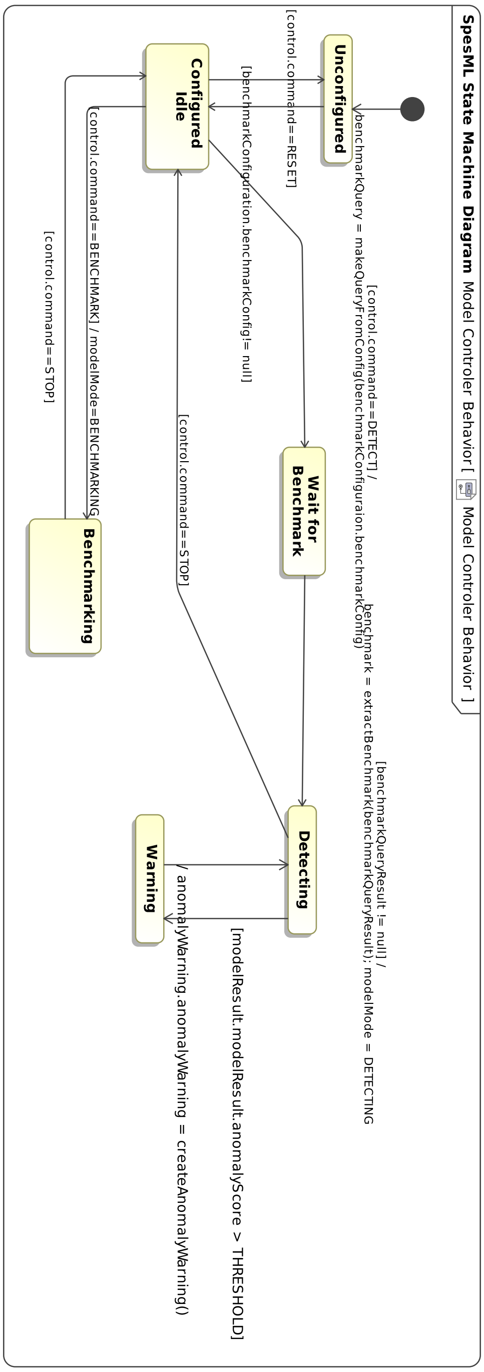


Fig. 5: State Machine of the Model Controller component

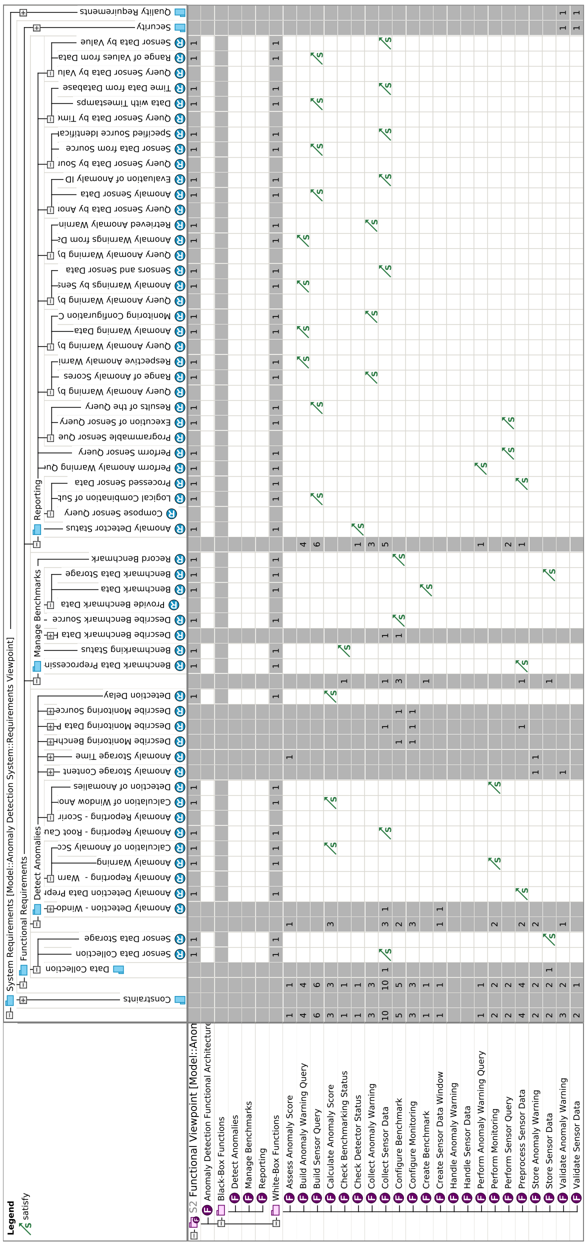


Fig. 6: Excerpt of the tracing links between system requirements and functional architecture

- Requirements: Currently there is the possibility to categorize requirements, link requirements to realizing model elements, and link requirements to each other. However, as pointed out in a review round, often requirements originate from the architecture work. Therefore, it would be helpful to link requirements to the design decision from which they originated.
- Tracing: Currently it is possible to trace requirements to whole functions. It is also possible to trace functions to logical components. However, especially in case of large interfaces, it would be helpful to create traces on a more fine granular level, e.g., between requirements and ports, for example to ease requirements verification and validation.
- Compatible ports: In order to be compliant with the underlying formal universal interface model, there are strict rules regarding the compatibility of ports. However, this leads to an inflexibility regarding the connection between ports and in general to a large number of ports.
- Behavior modeling: Currently, there are specific rules regarding the formulation of guards and effects in state machines, however advanced tool support (e.g., autocompletion in guards) is missing. This would ease the formulation of valid behavior models. Furthermore, currently only state machines can be used to model behavior. Certain types of behavior, e.g., data processing, are not naturally modeled with state machines. In this case, other types of behavior models could be beneficial.

W.r.t. the aforementioned goals, we could see the behavior modeling aspect as a future issue in expressiveness. However, since it did not impact us in our study, we suggest to analyze this issue in future work. For our own system under development, we did not identify gaps in expressiveness, neither in the tooling nor the methodology. We did however, identify a set of potential issues which could improve the usability in future work.

6 Summary

In this work, we gave an experience report on a case study that we conducted in the context of MBSE. With this report, we demonstrated the current state of the SPES modeling method and the SpesML modeling workbench based on Cameo Systems Modeler. We showed, at the example of an Anomaly Detection System, the models in the requirements-, functional-, and logical viewpoint. The focus of the execution of the study, however, was the analysis of the methodology and tooling against expressiveness and usability. The results show that we were able to model the ADS using the SpesML methodology and tooling. However, during the modeling, we identified four suggestions for adjustments.

The presented work was heavily influenced by the current status and the incremental improvement of the tooling during the project, as well as by the inside knowledge and experience of the modelers. Future work will therefore (1) extend the model in the technical viewpoint and execute simulations, (2) derive hypotheses on the quantitative and qualitative improvements through the method, (3) conduct further studies on different study objects and with other study subjects, in particular non-insiders as modelers, and finally (4) quantify the feedback and analysis provided based on the derived hypotheses.

7 Acknowledgements

We want to thank Alexander Knerr for his contributions during creation of the model. This work was funded by the German Federal Ministry of Education and Research (BMBF) under grant no. 01IS20092K.

Literatur

[Bö14a] Böhm, W.; Henkler, S.; Houdek, F.; Vogelsang, A.; Weyer, T.: Bridging the gap between systems and software engineering by using the SPES modeling framework as a general systems engineering philosophy. Procedia Computer Science 28/, S. 187–194, 2014.

[Bö14b] Böhm, W.; Junker, M.; Vogelsang, A.; Teufl, S.; Pinger, R.; Rahn, K.: A formal systems engineering approach in practice: An experience report. In: Proceedings of the 1st International Workshop on Software Engineering Research and Industrial Practices. S. 34–41, 2014.

[Bö21] Böhm, W.; Broy, M.; Klein, C.; Pohl, K.; Rumpe, B.; Schröck, S.: Model-Based Engineering of Collaborative Embedded Systems: Extensions of the SPES Methodology. Springer Nature, 2021.

[Br12] Broy, M.; Damm, W.; Henkler, S.; Pohl, K.; Vogelsang, A.; Weyer, T.: Introduction to the SPES modeling framework. In: Model-Based Engineering of Embedded Systems. Springer, S. 31–49, 2012.

[FMS14] Friedenthal, S.; Moore, A.; Steiner, R.: A practical guide to SysML: the systems modeling language. Morgan Kaufmann, 2014.

[Mo20] Morkevicius, A.; Aleksandraviciene, A.; Armonas, A.; Fanmuy, G.: Towards a common systems engineering methodology to cover a complete system development process. In: INCOSE International Symposium. Bd. 30. 1, Wiley Online Library, S. 138–152, 2020.

[OM19] OMG: OMG Systems Modeling Language (OMG SysML), Version 1.6, Object Management Group, 2019, URL: http://www.omg.org/spec/SysML/1.6/.

[Po16] Pohl, K.; Broy, M.; Daembkes, H.; Hönninger, H.: Advanced model-based engineering of embedded systems. In: Advanced Model-Based Engineering of Embedded Systems. Springer, S. 3–9, 2016.

[Ro16] Roques, P.: MBSE with the ARCADIA Method and the Capella Tool. In: 8th European Congress on Embedded Real Time Software and Systems (ERTS 2016). 2016.

[Ro17] Roques, P.: Systems architecture modeling with the Arcadia method: a practical guide to Capella. Elsevier, 2017.

[We16] Weilkiens, T.: SYSMOD-The systems modeling toolbox-pragmatic MBSE with SysML. http://model-based-systems-engineering.com, 2016.

[We21] Weyer, T.; Goger, M.; Koch, W.; Kremer, B.: Implementation Strategy for Seamless Model-based Systems Engineering. ATZ worldwide 123/7, S. 66–71, 2021.

Workshop Modeling in (and for) Production

Judith Michael[1], István Koren[2]

The production domain is permeated by heterogeneous data sources, a variety of IT systems, and complex industrial use cases — aspects that offer a very exciting field for research. Modern production systems involve a large number of sensors for monitoring industrial plants which help to gain an insight into a system's state. Sensors over time create datasets, that may be very detailed, need to be preprocessed such as time reduced, quantitatively, and qualitatively reduced, e.g., black and white instead of colored pictures, and enriched with metadata. To be able to define the datasets needed, we have to handle all these dimensions.

Our approach to handle these dimensions is the concept of digital shadows. A digital shadow is a set of contextual data traces and their aggregation and abstraction collected for a specific purpose with respect to an original system. These digital shadows are then used by digital twins. For us, a digital twin is a set of models of the system, a set of digital shadows and their aggregation and abstraction collected from a system, and a set of services that allow using the data and models purposefully with respect to the original system. Within [Be21], a conceptual model to describe digital shadows, data structures tailored to exploit models and data in smart manufacturing, was presented through a metamodel and its notion space. Using models helps handle this complexity in real-world scenarios.

The MoPro 2022 Workshop is a platform for researchers and practitioners within the production domain to exchange modeling techniques, interesting use cases and challenges. MoPro 2022 was searching for three types of contributions: *Research Papers*, *Novel Directions Talk* and *Digital Shadow Use Case*. We received eight submissions and accepted two Research Papers, two Digital Shadow Use Cases and three Novel Directions Talks.

- *Semantic Reasoning for Automated Factory Planning* (Niklas Schäfer) discusses potentials and challenges in capacity planning of factories with semantic information models.
- *Modelling Pig Rearing as Digital Shadow* (Tobias Zimpel) demonstrates how digital shadows can help to model data traces to enhance animal welfare.
- *Modeling Digital Shadows in Manufacturing by Using Process Mining* (Tobias Brockhoff, Merih Seran Uysal and Wil van der Aalst) realizes performance-aware digital shadows that provide holistic views on shopfloor-level processes.

[1] Software Engineering, RWTH Aachen University, Germany michael@se-rwth.de
[2] Process and Data Science, RWTH Aachen University, Germany koren@pads.rwth-aachen.de

- *Digital Shadows for Cross-Organizational Data Exchange* (István Koren) envisions a trade of production data to unlock new business models.
- *Enhancing Digital Shadows with Workflows* (Malte Heithoff, Judith Michael and Bernhard Rumpe) integrates human-machine-interactions to the modeling of digital shadows by utilizing workflows.
- *Modelling Human Factors in Cyber Physical Production Systems by the Integration of Human Digital Shadows* (Alexander Mertens, Philipp Brauner et al.) outlines an anthropocentric approach that considers human actors in production systems.
- *A Vision Towards Generated Assistive Systems for Supporting Human Interactions in Production* (Judith Michael) investigates how the software engineering processes of assistive systems in production can be improved.

Program Commitee

We thank our program committee members for their work, adherence to deadlines and contribution to MoPro 2022.

- Pascal Bibow, RWTH Aachen University (IKV)
- Philipp Niemietz, RWTH Aachen University (WZL)
- Felix Ocker, Technical University of Munich
- Rick Rabiser, JKU Linz
- Bianca Wiesmayr, JKU Linz
- Manuel Wimmer, JKU Linz
- Andreas Wortmann, Universität Stuttgart
- Alois Zoitl, JKU Linz

Furthermore, we are grateful to the members of the workshop program committees, who reviewed the workshop submissions and ensured the quality of the presented research. Additional thanks go to the authors of all workshop submissions and the attendees of the workshops for making MoPro 2022 an interesting venue.

MoPro 2022 takes place on June 28 in Hamburg as part of the Modellierung'22 conference. Thus, a special thanks goes to the whole organizing team of the Modellierung'22 for their continued support.

Aachen, June 2022
Judith Michael and István Koren

Bibliography

[Be21] Becker, Fabian; Bibow, Pascal; Dalibor, Manuela; Gannouni, Aymen; Hahn, Viviane; Hopmann, Christian; Jarke, Matthias; Koren, Istvan; Kröger, Moritz; Lipp, Johannes; Maibaum, Judith; Michael, Judith; Rumpe, Bernhard; Sapel, Patrick; Schäfer, Niklas; Schmitz, Georg J.; Schuh, Günther; Wortmann, Andreas: A Conceptual Model for Digital Shadows in Industry and its Application. In: Conceptual Modeling, ER 2021. Springer, pp. 271–281, 2021.

Semantic Reasoning for Automated Factory Planning

Niklas C. Schäfer [1]

Abstract: This article outlines the novel directions talk on semantic reasoning for decision-making support and automation in the domains of production engineering. With the use case of factory planning, the potentials and challenges are demonstrated on an application example in capacity planning. On the basis of a factory information model, a calculation model was defined to deduce planning results automatically. Recommendations for future research topics are given in the conclusions.

Keywords: Digital Factory · Production Design · Information Modeling · Ontologies.

1 Modeling Production Systems for Decision-Making Support

Resulting from today's dynamic market environment, the changing requirements to production systems justify continuous planning efforts to maintain cost-effective production. These planning efforts in designing and dimensioning the production system are associated to the task of factory planning. State-of-the-art planning approaches are characterized by modularity and project-specific configuration [SKW10]. Experts are provided with digital tools and applications to support in planning tasks. However, available integrated solutions limit adaption to project-specific requirements and raise the problem of interoperability by lacking interfaces for information exchange across applications and databases [DRA18].

Recent research investigates semantic information modeling for management of factory data to represent production systems virtually [Bü16]. Ontology-based information modeling allows connecting heterogeneous data sources in a data model that offers machine-interpretable data for processing in subsequent use cases. With such basis, implicit information can be logically deduced to support decision-making in factory planning tasks. Planning scenarios can be validated by checking for inconsistencies, e.g., as demonstrated in MEP design (mechanical, electrical and plumbing aspects of the production facility) [Bu21]. Another example of recalculating planning results will be given in the following section. Thereby, factory planning will serve as a use case to demonstrate the potentials of extending the modeling of production systems by semantic reasoning. Particular focus is set on how decision-making can be supported for automation in production engineering. As a result, currently used planning tools such as simulation software can be advanced in their functionality. Additionally, the approach can be applied

[1] Laboratory for Machine Tools and Production Engineering (WZL) of RWTH Aachen University, Campus-Boulevard 30, Aachen, 52074, n.schaefer@wzl.rwth-aachen.de, https://orcid.org/0000-0003-4329-9244

to further use cases in related domains of production engineering, e.g. production planning, that are also profiting from automation support in decision-making.

2 Application Example in Capacity Planning

Capacity planning is a typical task of factory planning by determining the required number of production resources. By comparing capacity demand of the products to be manufactured with the supply of capacity through the production resources, the capacity utilization is calculated and then considered in dimensioning the production system.

Following the ontology engineering methodology [SSS09], application-specific requirements were collected before conceptualizing and modeling an ontology for the described task of capacity planning. It was implemented as a conceptual factory information model in the ontology editor Protégé. In the following step, concrete factory information from an industrial factory planning project was imported. The source data in tabular format from an Excel file was mapped to the ontology with the plug-in Cellfie. Finally, the semantic web rule language (SWRL) was used to define a generic calculation model with rules for semantic reasoning. For example, capacity utilization for specific production resources is deduced from available factory information for different planning scenarios, as shown in Fig. 1.

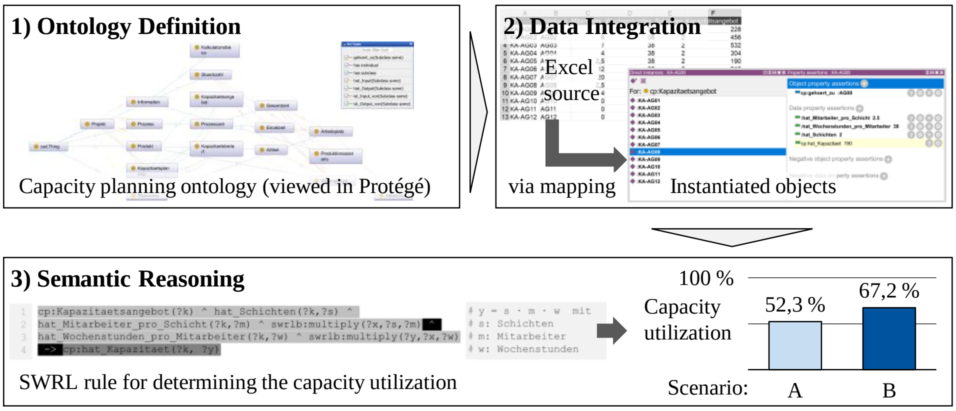


Fig. 1: Applied procedure for semantic reasoning of capacity utilization in capacity planning.

Eventually, the calculation results are automatically added to the information model and is available for further planning processes. Recalculations are automatically performed with updates of factory information. Similarly, factory information from different projects can also be processed, as the calculation model is defined generically.

A major challenge was found in the notably long processing times for reasoning (e.g., 7,7 s calculation time from 526 instantiated objects for a part of an industrial factory planning project), even rendering some proposed calculation rules unusable. Furthermore, the applied software tools are often not accessible to end users. The definition of the calculation rules also proves unrealistic for practical factory planning projects as their setup exceeds acceptable efforts for the average factory planner. However, this drawback can be countered by pre-defining the calculation rules for the conditions typically encountered in capacity planning tasks. Otherwise, integration of semantic reasoning into planning software could provide end users with improved calculation rule formulation in user interfaces.

3 Conclusions and Future Research

The implementation of semantic reasoning to support planning experts in capacity planning shows promising potentials for automation in the use case of factory planning. With semantic information models becoming increasingly available across production engineering, domain experts are encouraged to exploit them for support and automation of decision-making. To implement further identified use cases, their underlying decision-making processes need to be specified systematically. From the perspective of software development, end-user-accessible interfaces need to be designed. Equally important is the realization of methods for faster semantic reasoning, e.g., by using semantically-enriched but local data bases, as outlined in [BS16], or with ontology-oriented programming.

Acknowledgements

I would like to express my gratitude foremost to Marius Schmitt, who accomplished valuable thesis work on this topic. Further thanks for valuable guidance in my research go to Prof. Peter Burggräf and Tobias Adlon.

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC-2023 Internet of Production – 390621612.

Bibliography

[BS16] Biffl, S.; Sabou, M.: Semantic Web Technologies for Intelligent Engineering Applications. Springer International Publishing, Cham, 2016. https://doi.org/10.1007/978-3-319-41490-4.

[Bu21] Burggräf, P.; Dannapfel, M.; Ebade Esfahani, M.; Scheidler, F.: Creation of an expert system for design validation in BIM-based factory design through automatic checking of semantic information. Procedia CIRP 99, pp. 3–8, 2021. https://doi.org/10.1016/j.procir.2021.03.012

[Bü16] Büscher, C.; Voet, H.; Krunke, M.; Burggräf, P.; Meisen, T.; Jeschke, S.: Semantic Information Modelling for Factory Planning Projects. Procedia CIRP 41. pp. 478–483, 2016. https://doi.org/10.1016/j.procir.2015.12.022

[DRA18] Dombrowski, U.; Reiswich, A.; Alexander, K.: Designing digital tools for factory planning: Integrating requirements for usability on a meta-level. In: Proceedings of 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA). IEEE, Turin, pp. 99–106, 2018. https://doi.org/10.1109/ETFA.2018.8502669

[SKW10] Schuh, G.; Kampker, A.; Wesch-Potente, C.: Condition based factory planning. Prod. Eng. Res. Devel. 5, pp. 89–94, 2011. https://doi.org/10.1007/s11740-010-0281-y

[SSS09] Sure, Y.; Staab, S.; Studer, R.: Ontology Engineering Methodology. In (Staab, S.; Studer, R. ed.): Handbook on Ontologies. Springer, Berlin, pp. 135–152, 2009. https://doi.org/10.1007/978-3-540-92673-3_6

Modeling Pig Rearing as a Digital Shadow

Tobias Zimpel[1]

Abstract: Pig rearing and animal welfare are increasingly in the interest of society. To enhance animal welfare using data-driven analyses, modeling the pig rearing process is essential to create corresponding data sets. Pig rearing is a complex process for increasing pigs' weight (from approx. five to 25 kilograms) involving various actors (e. g. farmers, veterinarians) to provide goods (e.g., food, water) and services (e.g., medical care). Thereby, pigs live in pens equipped with condition-measuring sensors, like the pen's temperature or pigs' activity. Manual measurements (e.g., weights) are also conducted, resulting in various data sources. For analyzing these data, measured in different contexts, a digital shadow appears as an approach for modeling these data traces. Therefore, we report on a digital shadow for pig rearing, including the assets pen and pig, sensor sources, data traces (e.g., pens' temperature), and the purpose of analyzing causes of necrosis (dead tissue) with association rules.

Keywords: digital shadow, pig farming, process modelling.

1 Motivation

Pig rearing pursues the complex task of raising pigs (increasing pig's weight from approx. five to 25 kilograms) over several weeks while maintaining animal welfare. Animal welfare can be defined by using the five freedoms, including the freedom from hunger, stress, pain, and injury [Fa09]. To achieve this goal, various actors work together to provide goods (e.g., food, water) and take care of the pigs. Farmers, veterinarians, craftspeople, and food suppliers can represent such actors. Each actor can collect data, resulting in potentially different data sources (e.g., different sensors in a pen) [Ri20]. Therefore, data may differ in their structure, meaning or aggregation level. Data is combined into a dataset using extract, transform, and load processes. Then further data analysis is based on this entire data [Ne20]. In pig rearing, we analyze pig and pen data to suggest causes for necrosis (dead tissue) using association rule mining. Therefore, contextual information is required.

The concept of digital shadows (DS) may be able to support this analysis of necrosis in pig rearing. A DS is a set of data traces with corresponding aggregation or abstraction functions collected for a specific purpose [Bi20]. Therefore, a DS provides a view of assets (like pig and pen), including contextual information [Be21]. Elements in DS are data traces, data sources, data points, metadata, model, and purpose [Be21]. Data traces corresponds to a set of data points (e.g., temperature) recorded by a source (e.g.,

[1] University of Hohenheim, Department of Information Systems 2, Schwerzstraße 35, Stuttgart, 70599, tobias.zimpel@uni-hohenheim.de

temperature sensor) and linked to a DS and metadata [Be21]. Metadata describes relevant contextual data for the DS, like the pen's target temperature or sensor precision. [Be21]. The model adds relevant information about the asset, like data structure, to the DS [Be21]. Purpose describes the objective of the DS, such as analyzing pig and pen data for causes of necrosis [Be21]. Against this backdrop, we address the following question: *How to design a digital shadow for the analysis of necrosis in pig rearing?*

This paper is structured as follows: In Section 2 describes the use case of pig rearing. In Section 3, we present digital shadows for the assets pen and pig. Finally, Section 4 discusses the potential implications and challenges of the digital shadow and concludes the paper.

2 Use case description

The use case describes the pig rearing process based on processes at the Boxberg Teaching and Research Centre (LSZ) in Germany. The LSZ is the central educational, experimental, and testing facility of the state of Baden-Wurttemberg in the field of pig farming. Pig rearing is part of a supply chain for the production of pork. Such a supply chain is shown in Figure 1, based on the supply chain reference model (SCOR) version 12 (see [AP17]) and according to [LS13]. The supply chain in Figure 1 includes the SCOR level-1 processes source (sS), make (sM), and deliver (sD) and is limited to one actor per stage. The lowercase "s" in sS, sM, and sD stand for SCOR [AP17]. End customer describes various types of customers, like butchers or restaurants.

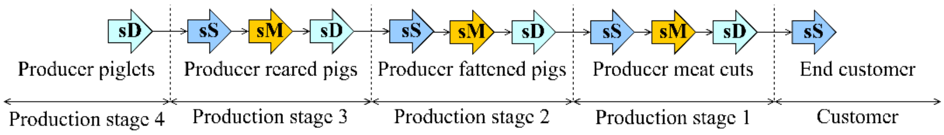


Fig. 1: Supply chain production of pork (SCOR diagram, level-1 processes)

The pig rearing process is located in production stage three, resulting in corresponding effects for the subsequent stages. Such an effect can originate from pigs' necrosis (e.g., medical care or loss), which can occur in the rearing process (as well as in production stages four and two) [Re20].

Figure 2 shows a pig farming supply chain, including one upstream production stage and a selection of producers for production stage two. Missing producers include, for example, producers of manipulable material and lightning. We are aware of potential inconsistencies in the distinction between internal and external actors. However, Figure 2 presents the complexity of the rearing process and potential data traces along with each good flow between the production stages.

We also assume that potential necrosis causes are in pig's environment (the pen) and thus in the production stage two. [FS81, FU79, KFS82]. For example, such causes could be failures in the ventilation system (producer air), or water system (producer water), resulting in a factor supporting necrosis (e.g., stress due to hot air). To analyze such potential causes of necrosis using association rules, we generated a dataset containing data from production stages one and two.

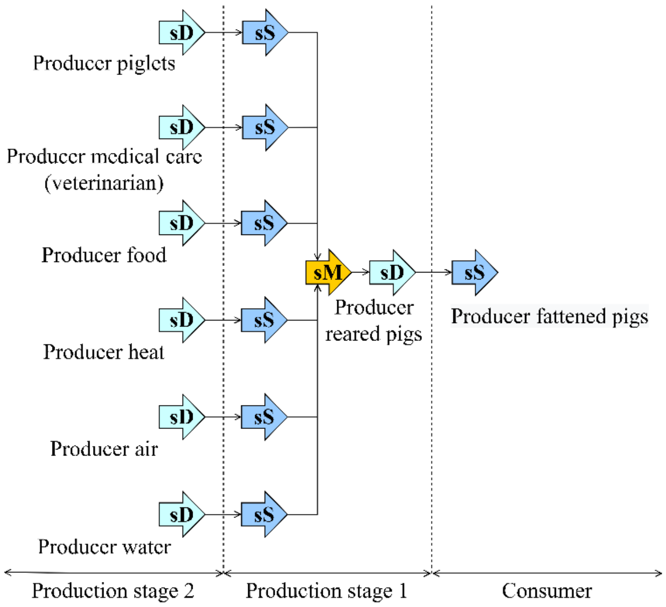


Fig. 2: Supply chain pig rearing (SCOR diagram, level-1 processes)

3 The Digital Shadow

We identified two assets for the purpose of data-driven proposals for necrosis causes in pig rearing: pen and pig. While Figure 3 presents the pen's digital shadow, the pig's digital shadow is shown in Figure 4.

The asset pen in the corresponding digital shadow contains a name and an identifier. The pen is also connected to the asset pig to access the pig's data. The digital shadow should contain data traces for each property or state that may be relevant to necrosis analysis. Therefore, the pen’s digital shadow should include information about each actor in production stage two (see Figure 2) as a data trace (e.g., temperature history provided by the heat producer). Due to the number of actors in Figure 2 and the corresponding amount of data traces, we only included data traces for temperature and food. However, the temperature data is similar to water and humidity data (provided by air and heat producers). Therefore, analogous to temperature, a data trace for water and humidity can be added into the digital shadow. The data trace for medical data is ignored due to the underlying complexity. Metadata for temperature data traces should consist of information about data quality and data collection to understand and analyze underlying data (e.g., handling missing data). Another relevant information in the metadata is the target temperature to enable the comparison with the actual temperature. Each temperature data point reports the actual temperature in Celsius at a given point in time for the pen. The food trace is measured and integrated similarly to temperature, with changes in labeling and unit of measurement. Different sensors record temperature and food data. The pen's model is a Unified Modeling Language (UML) class diagram.

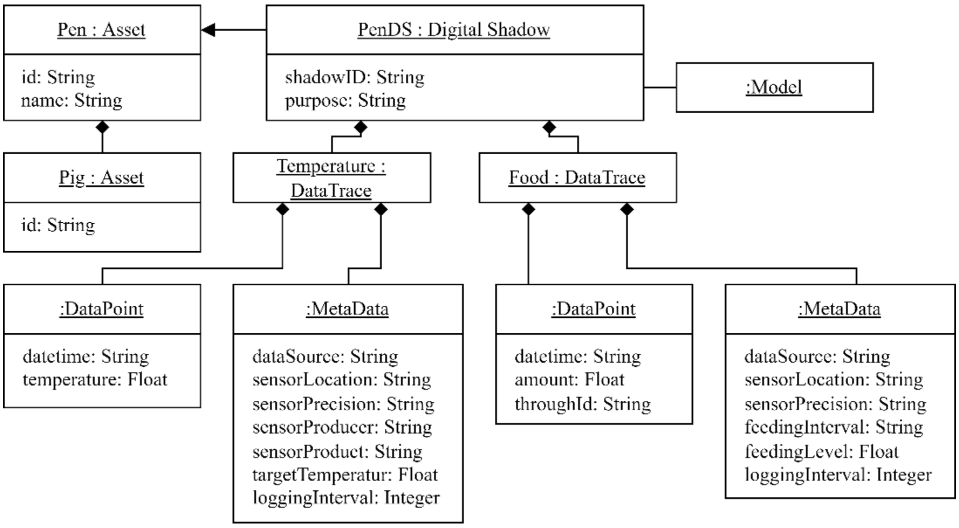


Fig. 3: Pen's digital shadow (UML class diagram)

The asset pig in the digital shadow of the pig has the attributes identifier, birthday, gender, and breed. Furthermore, the asset pig is linked to the asset pen to access pen data. The data trace consists of pigs' assessments regarding the presence of necrosis at different dates during the rearing. An employee decides at each assessment on the presence or absence of necrosis according to a uniform scheme. One assessment corresponds to one data point. Therefore, the employee who performed the rating and the rating scheme constitute necessary information in the metadata to consider these factors in the analysis (e.g., to compare assessments). Another data trace could be the pigs' weight, which can be integrated similarly to the pen's temperature in the corresponding digital shadow. Similar to the pen's model, a UML class diagram is a model for the digital shadows of pigs.

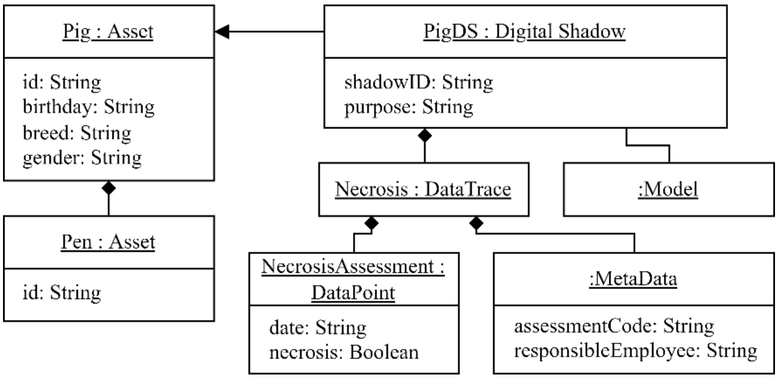


Fig. 4: Pig's digital shadow (UML class diagram)

4 Discussion and Conclusion

In this work, we presented pig rearing and corresponding data-driven analysis as a use case for digital shadows. We proposed two interconnected digital shadows for pens and pigs. Both digital shadows consist of selected data traces, like pens' temperature or pig's presence of necrosis. Further data traces can be integrated similarly in the future.

The digital shadow supports the data-driven analysis of necrosis in two ways. Firstly, the digital shadow includes context information resulting in an increased understanding and another way of presenting this information. Increased understanding of context information is justified by the otherwise necessary manual search and analysis of context information. We assume reduced time for data understanding by providing contextual information in the digital shadow metadata. There is often separate documentation for each data-trace. In addition, sometimes, no or only partial documentation is directly available. Secondly, we assume that the digital shadow can simplify the combination of relevant data into an asset due to the prior defined data structures. Thus, we assume that an increased data understanding of data and prior defined data structures accelerate the analysis process concerning the required time. Therefore, improvements in data analysis can benefit the underlying process – in this use case, pig rearing. However, analyzing the actual implications and challenges in practice is future work.

Acknowledgments

The authors thank the staff at the Boxberg Teaching and Research Centre for their advice especially Hansjörg Schrade and Andrea Wild. This work was supported by the project “Landwirtschaft 4.0: Info-System (Phase 2)”, funded by the Ministry of Rural Development and Consumer Protection of Baden-Wurttemberg, Germany.

Bibliography

[AP17] APICS: Supply chain operations reference model SCOR. Version 12.0, 2017.

[Be21] Becker, F. et al.: A conceptual model for digital shadows in industry and its application. In (Ghose, A. et al. Eds.): Conceptual modeling. Springer International Publishing, Cham, pp. 271–281, 2021.

[Bi20] Bibow, P. et al.: Model-driven development of a digital twin for injection molding. In (Dustdar, S. et al. Eds.): Advanced information systems engineering. Springer International Publishing, Cham, pp. 85–100, 2020.

[Fa09] Farm Animal Welfare Council: Farm animal welfare in Great Britain: Past, present and future, 2009.

[FS81] Fiesjå, K. I.; Solberg, I.: Pathological lesions in swine at slaughter. IV. Pathological lesions in relation to rearing system and herd size. Acta Veterinaria Scandinavica 2/22, pp. 272–282, 1981.

[FU79] Fiesjå, K. I.; Ulvesæter, H. O.: Pathological sesions in swine at slaughter. I. Baconers. Acta Veterinaria Scandinavica 498-514/20, 1979.

[KFS82] Kjell I., F.; Forus, I. B.; Solberg, I.: Pathological lesions in swine at slaughter. V. Pathological lesions in relation to some environmental factors in the herds. Acta Veterinaria Scandinavica 23, pp. 169–183, 1982.

[LS13] Leukel, J.; Sugumaran, V.: Formal correctness of supply chain design. Decision Support Systems 56, pp. 288–299, 2013.

[Ne20] Neethirajan, S.: The role of sensors, big data and machine learning in modern animal farming. Sensing and Bio-Sensing Research 29, p. 100367, 2020.

[Re20] Reiner, G. et al.: Swine inflammation and necrosis syndrome is influenced by husbandry and quality of sow in suckling piglets, weaners and fattening pigs. Porcine health management 1/6, p. 32, 2020.

[Ri20] Riekert, M. et al.: Towards animal welfare monitoring in pig farming using sensors and machine learning. In (Gandorfer, M. et al. Eds.): 40. GIL-Jahrestagung, Digitalisierung für Mensch, Umwelt und Tier. Gesellschaft für Informatik e.V, Bonn, pp. 271–276, 2020.

Modeling Digital Shadows in Manufacturing Using Process Mining

Tobias Brockhoff[1], Merih Seran Uysal[2], Wil M.P. van der Aalst[3]

Abstract: Friction in shopfloor-level manufacturing processes often occurs at the intersection of different subprocesses (e. g., joining sub-parts). Therefore, considering the Digital Shadows (DSs) of individual materials/sub-parts is not sufficient when analyzing the processes. To this end, holistic views on shopfloor-level processes that integrate multiple DSs are needed. In this work, we discuss how material-centric DSs supported by discrete assembly events can be integrated using techniques from process mining. In particular, we propose to utilize DSs that contain additional structural information to overcome the main challenges of concurrency and the presence of many different objects.

Keywords: Digital Shadow; Process Mining; Bill of Materials; Manufacturing Process Discovery

1 Introduction

With the advent of Industry 4.0, Digital Shadows (DSs) become increasingly important for decision-making in production. At the same time, companies collect increasing amounts of data on their operational processes. Despite the importance and feasibility in terms of data availability, realizing compatible DSs, which can be integrated and linked to create new insights, often remains difficult. To facilitate the use and create a common foundation, a DS meta model has been proposed in [Be21] (see Fig. 1). Still, implementations can become very and specific, and, therefore, realizing DSs remains difficult. For example, predicting waste for a machine can require to consider and fuse many different sensors and models.

In our research, we focus on general-purpose DSs that exist on the shopfloor-level of manufacturing processes—namely, *DSs of assembly executions* and *DSs of sub-components/materials composition as well as production line plans*. In particular, we consider assembly execution DSs built on discrete event data (e. g., assembly activity events). To gain insights into the production process, the DS on the assembly execution and the structural material composition for multiple products need to be combined. To this end, we use techniques from Process Mining (PM) which is an emerging discipline that leverages event data to improve processes.

In PM, there are three major concepts: *events*, *cases*, and *process models*. *Events* are recordings of discrete business operations and their time of occurrence. Multiple events

[1] RWTH Aachen University brockhoff@pads.rwth-aachen.de
[2] RWTH Aachen University uysal@pads.rwth-aachen.de
[3] RWTH Aachen University wvdaalst@pads.rwth-aachen.de

(e. g., events related to a product) constitute a *case*. Finally, *process models* describe the behavior of a process. As depicted in Fig. 1, these concepts perfectly align with the meta model proposed in [Be21]. However, from the PM perspective, there are two main challenges

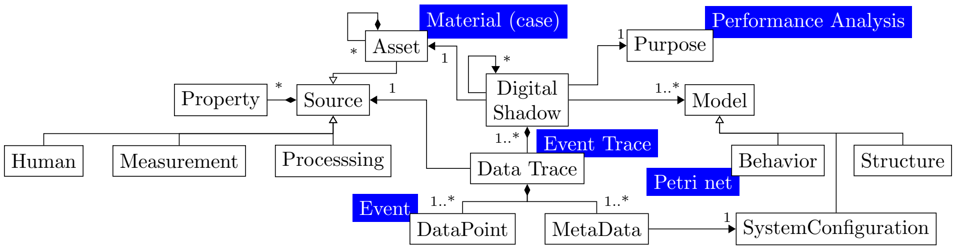


Fig. 1: Alignment of process mining concepts (blue) with the entities of the Digital Shadow (DS) concept model proposed in [Be21] (black).

for integrating the DSs: *concurrency* and *object-centricity* [Aa21]. Sub-materials can be assembled concurrently, and, when analyzing the performance of shopfloor-level processes, we can consider either the sub-materials or the product as case notions. When selecting the product as the case, the events related to its sub-materials may arbitrarily interleave; while, when selecting the materials as the case notion, friction at the intersection of materials remains unnoticed. In our research, we investigate bridging the gap towards a comprehensive production model by leveraging additional sources of information—namely, structural DSs on material composition (e. g., bill of materials) or the assembly line (e. g., assembly line plans). We thereby address the following research questions:

RQ1 How can dynamic DSs of process executions be combined with structural DSs of material composition and production lines?

RQ2 How can we create a performance-aware DS of a production line that reveals friction particularly at the intersection of subassembly boundaries?

While the first question focuses on modeling production lines, the second question targets the enhancement of this model such that it realizes the *purpose* of performance monitoring.

2 Related Work

Digital Shadows and Twins in manufacturing are increasingly gaining attention. Key applications are production planning (e. g., by simulation), control, and optimization [Kr18]. In this regard, one approach to describe shopfloor-level activities is by means of discrete events (e. g., events for starting or completing assembly activities). Discrete event simulation can then be applied to plan and optimize the process [KA16; YAL16]. While the required accurate process models would be valuable for a data-driven performance analysis, such models often do not exist. Moreover, tuning the parameters (e. g., service times) to obtain

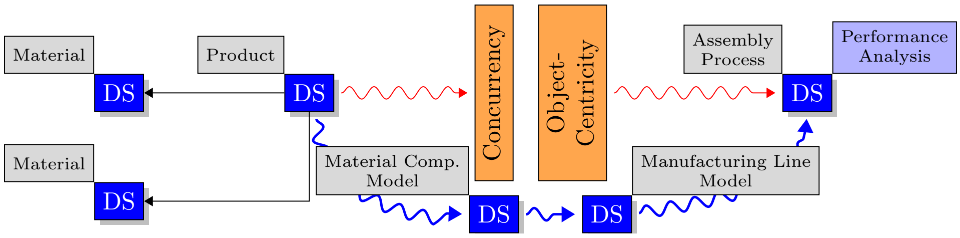


Fig. 2: Exploiting additional structural information to overcome the challenges of concurrency and object-centricity. We propose (blue snake line) to structure and integrate Digital Shadows (blue) for different assets (gray) by exploiting DSs on structural production information.

precise models is often tedious. In our research, we focus on the reverse direction, that is, given event and additional structural data, a model of the manufacturing process is built. Using this model, we generate unbiased performance assessments based on *real data*.

Within the last decade, there have been numerous works on the application of PM for manufacturing processes. Overviews over the potential of PM to improve manufacturing processes and how PM has already been applied can be found in [Aa20; DRG21]. Its focus on the actual dynamics distinguishes PM from classical process mapping that shows statically aggregated data [Lo21]. For example, it can show differences between the designed work flow and the as-is production [Lo21]. However, purely event data-driven model discovery becomes infeasible for large processes. Consequently, these works resort to process-global statistics such as the number of activities in progress or cumulative delay [Pa15]. Process models are only used within limited process scopes. To analyze processes based on comprehensive models, we exploit additional manufacturing-specific information.

3 Methods

In our research, we strive to create comprehensive views on shopfloor-level processes. To this end, we leverage event data generated on the shopfloor as well as additional structural information. Conceptually, we create a new DS that combines event data-based DSs of the process executions with DSs that contain structural material models (e. g., Multi-level Manufacturing Bills of Materials (M^2BOMs)). Techniques from PM thereby help to *integrate highly dynamic event data* and rather *rigid structural models*. In particular, using concepts from process mining and information from the structural DSs, we first discover a behavioral model—i. e., a process model—of the manufacturing process. Afterwards, we use PM to enhance the discovered model by performance information creating a performance-aware DS of the production. An illustration of our approach is depicted in Fig. 2. We start from DSs of material and subpart assembly execution that are built on and instantiated by discrete event data (i. e., events of assembly activities). Gaining insights into the process then requires to integrate the obtained DSs. However, the integration faces two major challenges:

concurrency and *object-centricity* [Aa21]. In our research, we exploit DSs that contain additional structural information to disentangle shopfloor-level manufacturing processes. Such models can either be manufacturing line models that disentangle the concurrency of assembly lines or material composition models. Due to physical constraints, it can usually be assumed that the data conforms to the model (e. g., products cannot skip stations at the conveyor belt). For example, in [Uy20], we modeled a car manufacturing process that consists of a general assembly line where some stations depend on concurrent sub-assembly lines. We used the model to replay the event data to compute KPIs (e. g., waiting or idle times) and visualized the evolution over time. While direct modeling is feasible for highly structured processes, it quickly becomes infeasible if the product flexibility increases. In this case, information on the material composition can help to disentangle and, eventually, model the process.

The resulting DS can be visualized, thereby, enabling a backward-looking analysis of the process that can reveal systematic production problems. Moreover, it can be used to query specific production KPIs.

Exploiting Structural Domain Information A common type of material composition information are M^2BOMs. M^2BOMs organize the materials built into a product in trees. Vertices correspond to materials whose assembly depend on the assembly of their child materials. The root vertex is the final product. Depending on the product, M^2BOMs contain hundreds of materials. For such large processes, in particular when subparts are assembled concurrently, automatic model discovery usually fails to find understandable models. Compared to the underlying highly-structured M^2BOM, the models are either unstructured and 'spaghetti' or overly general. This problem is worsened by products having similar but not necessarily equal M^2BOMs (e. g., certain materials are optional, or there might be a choice between different configurations). In [Br21], we investigated how M^2BOMs can be exploited to comprehensively model manufacturing processes for the purpose of performance analysis. We start with a collection of M^2BOMs and, targeting RQ1, output a M^2BOM-like process model. The latter is a tree that contains all materials from the input collection as well as optional materials, material choices, and additional material groupings (e. g., a choice between two material groups). Moreover, each material is endowed with an assembly task vertex that subsumes all activities related to its direct assembly. An example output for the offset printer manufacturing process introduced in [Br21] is depicted in Fig. 3 which shows all occurring (anonymized) materials as well as optional materials and choices between materials. The performance-aware coloring shows process-global bottlenecks as well as differences between similar materials.

Conceptually, we obtain the model by merging M^2BOMs. Based on counting arguments, we automatically identify shared materials and potential choices. The latter are then resolved manually as the resolution can be ambiguous. For example, consider two infrequent features that never occur together. It does not per se clear that these features are mutually exclusive. Since the so-obtained model has a direct correspondence to a process model (i. e., a process

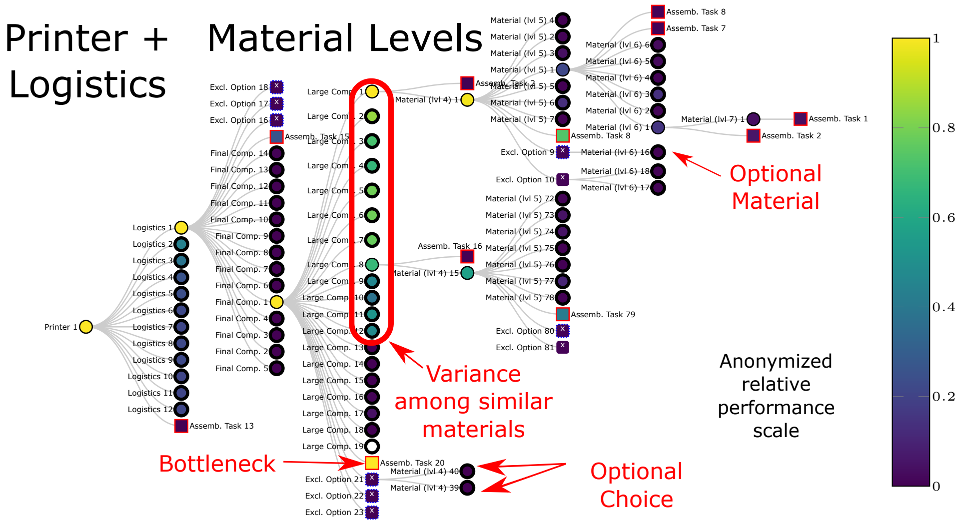


Fig. 3: Overall offset printer manufacturing process model with collapsible vertices. Each vertex corresponds to a material. Special vertices are dedicated to optional materials (blue dashed), choices (×), and assembly tasks (red vertices). The color depicts the cumulative material assembly time.

tree), it can be endowed with performance metrics derived from replaying the event data. Considering RQ2, this enables to use the model to detect performance problems. Since the model comprehensively integrates all materials, it can also be used to compare similar materials across the model as well as to analyze relations between parent and child materials.

4 Challenges and Conclusion

In this work, we presented our research on realizing performance-aware Digital Shadows (DSs) of shopfloor-level manufacturing processes. To this end, we propose to complement techniques from Process Mining (PM) by additional structural data to alleviate the challenges of concurrency and object-centricity. In doing so, we can visualize processes even if standard automatic model discovery fails. In future work, we aim to generalize our work to other sources of structural information. Considering the performance analysis, a major challenge lies in integrating additional process context into the model. While, in process model notations commonly used in PM, different orders are independent, this does not hold in real life. We therefore require models that capture the process context. Moreover, our current work only enables a backward-looking analysis. Even though this is sufficient to yield insights into systematic problems, it does not allow to react to and recover from real-time problems. The latter requires to continuously update the model turning it into a Digital Twin of the assembly line. While techniques from PM facilitate integrating dynamic performance updates with respect to the event data, structural updates can become more challenging.

Acknowledgment We thank the Alexander von Humboldt (AvH) Stiftung for supporting our research. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC- 2023 Internet of Production – 390621612.

References

[Aa20] van der Aalst, W. M. P. et al.: Removing operational friction using process mining: challenges provided by the internet of production (IoP). In: DATA. Springer, pp. 1–31, 2020.

[Aa21] van der Aalst, W. M. P.: Concurrency and Objects Matter! Disentangling the Fabric of Real Operational Processes to Create Digital Twins. In: ICTAC 2021. Springer International Publishing, pp. 3–17, 2021.

[Be21] Becker, F. et al.: A Conceptual Model for Digital Shadows in Industry and Its Application. In: ER. Springer International Publishing, pp. 271–281, 2021.

[Br21] Brockhoff, T. et al.: Analyzing Multi-level BOM-structured Event Data. In: Process Mining Workshops. Vol. 433, 2021.

[DRG21] Dreher, S.; Reimann, P.; Gröger, C.: Application Fields and Research Gaps of Process Mining in Manufacturing Companies. In: LNI. Gesellschaft für Informatik, Bonn, pp. 621–634, 2021.

[KA16] Kaylani, H.; Atieh, A. M.: Simulation Approach to Enhance Production Scheduling Procedures at a Pharmaceutical Company with Large Product Mix. Procedia CIRP 41/, pp. 411–416, 2016, ISSN: 2212-8271.

[Kr18] Kritzinger, W. et al.: Digital Twin in manufacturing: A categorical literature review and classification. IFAC-PapersOnLine 51/, pp. 1016–1022, 2018.

[Lo21] Lorenz, R. et al.: Using Process Mining to Improve Productivity in Make-to-Stock Manufacturing. IJPR/, pp. 1–12, 2021, ISSN: 0020-7543.

[Pa15] Park, M. et al.: Workload and Delay Analysis in Manufacturing Process Using Process Mining. In: Asia Pacific BPM. Springer International Publishing, pp. 138–151, 2015.

[Uy20] Uysal, M. S. et al.: Process Mining for Production Processes in the Automotive Industry. In: Industry Forum at BPM, Sevilla, Spain. 2020.

[YAL16] Yang, S.; Arndt, T.; Lanza, G.: A Flexible Simulation Support for Production Planning and Control in Small and Medium Enterprises. Procedia CIRP 56/, pp. 389–394, 2016, ISSN: 2212-8271.

Digital Shadows for Cross-Organizational Data Exchange

István Koren[1]

Abstract: Production settings typically involve heterogeneous systems that create a challenging environment for collecting data in light of digital transformation. Once overcoming these difficulties, data-driven opportunities for manufacturing companies include increasing efficiency and productivity, reducing costs, and improving quality control. On the shop floor, digital shadows and digital twins are elements of these modernization strategies, e.g., to leverage machine learning methods for decision support. Recently, some approaches have transferred these concepts to the organizational level, like digital twins of organizations. In this paper, we envision how we can use data collections from the shop floor, captured as digital shadows, to share data across organizational boundaries to create new business models and ultimately enter new markets. We discuss the necessary enhancements of our conceptual model for digital shadows presented in previous work. We are convinced that digital shadows can help companies embrace innovative, data-driven business models to face challenges like sustainability.

Keywords: Industry 4.0; Digital Shadows; Cross-Organizational Data Exchange

1 Introduction

Production settings are challenging digitalization targets as various engineering disciplines, heterogeneous devices, architectures, and protocols prevail on the shop floor. Data-driven approaches such as machine learning are necessary to tackle new requirements like mass customization. Digital twins are an important concept in this realm as virtual representations of physical objects or systems. Digital shadows can feed them as on-demand aggregations of data. In addition to the interactions in a machine shop itself, however, material and data flows beyond a single organization should not be neglected. There has been an increased focus on these cross-organizational data exchanges in manufacturing in recent years. They play a role in simple supplier-producer relationships and in new, data-driven businesses like subscription-based pricing models. Finally, regulatory measures such as reporting sustainability indicators require that companies open up internal data repositories. Overall, the complexity of internal and external data flows presents companies with new challenges.

In this paper, we argue that we can use the data infrastructure from the shop floor for new forms of data-driven collaboration beyond a single organization. Specifically, we adopt and reuse the concept of digital shadows. Initial approaches such as the *digital twin of an organization* are currently emerging in research (e.g., [PvdA21]). However, these

[1] RWTH Aachen University, Chair of Process and Data Science, Ahornstr. 55, 52074 Aachen, Germany
koren@pads.rwth-aachen.de

are based on business processes and extracting data from enterprise resource planning system databases, for example. The complexity of production environments is much more challenging. In the next section, we first identify elements of cross-organizational data exchange and show how they require an extension of our conceptual model developed in previous work [Be21]. Finally, we give an outlook on ample research opportunities of federated digital shadows.

2 Federating Digital Shadows

A digital shadow is a task-specific collection of data to overcome the heterogeneous system landscape and deal with the massive volume of data in production environments. It needs to tackle several dimensions from material to production line by aggregating and filtering data, for instance. Based on a model-based approach as an abstraction from reality, it forms a *view* of the data that can be used to feed digital twins, e.g., for analytics purposes or to manage the control flow of machines. In previous work, we presented a conceptual model for digital shadows and exemplified its utility by an injection molding use case [Be21] (the reader is kindly referred to Figure 1 of the referenced article for the metamodel).

Recent European debates have recognized the growing importance of cross-organizational data exchange with *data spaces* as managed collections of data. The International Data Spaces (IDS) consortium developed a reference architecture that mentions necessary prerequisites [Ot19], but it does not impose a technical realization. To this end, we envision the further use case of data exchange beyond company boundaries for our digital shadow metamodel. However, we recognize that a fully detailed digital shadow of a complex organizational relationship is impossible because of the volume of data and the varying degrees of data security. Instead, we propose the *federation* of digital shadow instances. This linking of at least two digital shadows requires a slight extension of the original metamodel. Specifically, a `Digital Shadow` has to inherit from `Source`. The `originates from` relation from `DataTrace` to `Source` then allows for chaining one or more digital shadows. Figure 1 shows the federated digital shadow instance of a supply chain risk assessment system from the car manufacturing industry, connecting a tire supplier with a car producer and a dealer. The central digital shadow captures data from a car production (right orange box). Its left-most data trace is connected to a supplier's digital shadow, that in turn gets its data from the digital shadow of a supplier. The `originates from` relations thereby represent interfaces between organizations. Note the usage policy `Metadata` on the bottom. It requires the data from the tire supplier to be only used by the risk management system and to be deleted within 14 days. These data usage concepts that we derived from the IDS, can be attached as metadata. For additional attributes, e.g., regarding pricing, please refer to the IDS reference architecture [Ot19]. In the next section, we conclude the paper with an outlook on possible future research in this area.

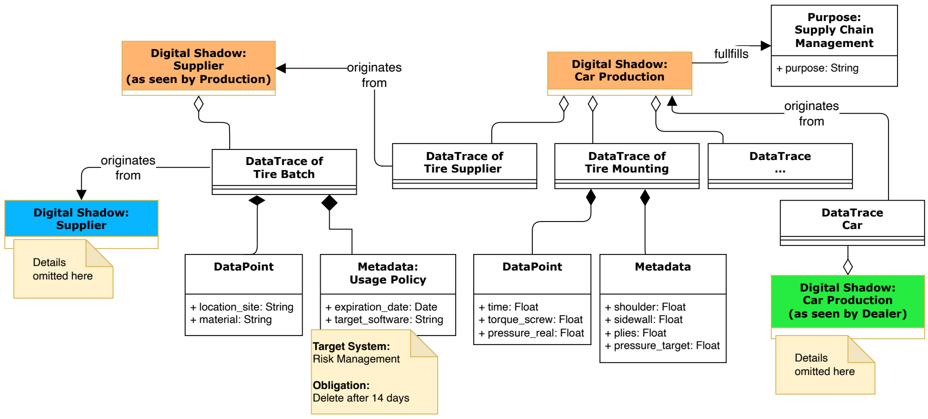


Abb. 1: Example of a supply chain as federated digital shadow instance; colors depicting ownership.

3 Conclusions

In this paper, we discussed the growing importance of cross-organizational data exchange for production companies in light of new business models and regulatory requirements. The extended use case of digital shadows opens some exciting paths for future research. First, we need to understand better how to materialize the conceptualized interfaces, e.g., via code generation of service interface description languages like OpenAPI and GraphQL. To combine machine and human intelligence, referred to as *hybrid intelligence*, this could also include automatically provided human interfaces. Second, processing and utilizing data across the product lifecycle could facilitate context-adaptive and circular production ecosystems for sustainability and resilience in production and usage. Finally, the secure data exchange could help manufacturing companies develop resilience to sudden threats such as supply shortages induced by natural disasters or pandemics.

Acknowledgment. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC-2023 Internet of Production – 390621612.

Literaturverzeichnis

[Be21] Becker, Fabian et al.: A Conceptual Model for Digital Shadows in Industry and Its Application. In (Ghose, Aditya et al., Hrsg.): Conceptual Modeling. Lecture Notes in Computer Science, Springer International Publishing, Cham, S. 271–281, Januar 2021.

[Ot19] Otto, Boris; Steinbuß, Sebastian; Teuscher, Andreas; Lohmann, Steffen; Association, International Data Spaces: Reference Architecture Model: Version 3.0. Bericht, Dortmund, Germany, Januar 2019.

[PvdA21] Park, Gyunam; van der Aalst, Wil M.P.: Realizing A Digital Twin of An Organization Using Action-oriented Process Mining. In: 2021 3rd International Conference on Process Mining (ICPM). IEEE, Eindhoven, Netherlands, S. 104–111, Oktober 2021.

J. Michael, J. Pfeiffer, A. Wortmann (Hrsg.): Modellierung 2022 Satellite Events,
Digital Library, Gesellschaft für Informatik e.V. 142

Enhancing Digital Shadows with Workflows

Malte Heithoff[1], Judith Michael[1], Bernhard Rumpe[1]

Abstract: The vast amount of data in modern manufacturing demands acquisition of contextualized data to enable fast decision making where domain expertise must be provided at run-time. Within this paper, we investigate the research question how to handle human-machine-interactions for engineering of digital shadows and still ensure the traceability of computation and simulation results. Current research for digital shadows concentrates on modeling key elements such as data sharing or metadata, but does not incorporate human-machine-interaction or the traceability of data aggregation. In this paper, we present a conceptual model which covers the base concepts for digital shadows integrating human-machine-interaction by utilizing workflows. We extend the conceptual digital shadow model defined within the "Internet of Production" excellence cluster and showcase our approach on an example. This contribution presents an applicable modeling approach for designing digital shadows which provide contextual information of the underlying human integrated process.

Keywords: Digital Shadow; Digital Twin; BPMN; Production; Human Computer Interaction

1 Introduction

Modern manufacturing processes produce huge amounts of data which demand for a contextualized data acquisition for fast decision making. Digital Shadows (DS) [Be21] promise to provide a data entity which can be used to reason about reality [vdA21] in a timely manner. In the following, "*A Digital Shadow is a set of contextual data traces and their aggregation and abstraction collected for a specific purpose with respect to an original system*" [Be21]. When aggregating data in digital shadows, we often assume a fully automatic process, executed either by an additional software component, e.g., a digital twin, or by some software service. Semi-automated processes arise when a DS operator might need to first set some parameters or must transfer intermediate results between software or hardware systems for which yet no digital communication exist.

Current research on digital shadows does not consider human-machine-interaction by domain experts which might be needed to execute all computations, and lacks the traceable modeling of the complex aggregation process. Within this paper, we investigate how to handle human-machine-interactions within the engineering of digital shadows and still ensure the traceability of computation and simulation results.

In an interdisciplinary work within the Cluster of Excellence "Internet of Production" (IoP)[2], Becker et al. [Be21] developed a conceptual model for digital shadows. In this paper,

[1] Software Engineering, RWTH Aachen University, {heithoff,michael,rumpe}@se-rwth.de

[2] Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2023 Internet of Production - 390621612. Website: https://www.iop.rwth-aachen.de

we aim to provide an extended version of this conceptual model which is able to model complex aggregation processes and allows human-machine-interaction using workflows.

2 Related Work

According to Bibow et al. [Bi20] "*a Digital Shadow is a set of contextual data traces and their aggregation and abstraction collected concerning a system for a specific purpose with respect to the original system*". Existing approaches include only some of these concepts, such as data analysis and data access [LKC19] or a data and knowledge holding [La21]. Van der Aalst et al. [vdA21] propose process mining to create digital shadows as an entity used to reason about the system. Nonetheless, their modeling lacks human interaction and the complex processes of lifting data to a useful aggregation.
Becker et al. [Be21] present a conceptual model for the engineering of digital shadows fulfilling the definition given above. Their model includes concepts such as assets describing the machine and its parts, data within data traces originating from a source, a purpose the DS must be tailored to, and models that the DS uses to fulfill its purpose. This approach provides the conceptual foundation for designing digital shadows. In contrast to this paper, the overall process is underspecified and remains domain knowledge which can not be modeled in the DS. Moreover, human interaction is limited to providing initial data.

To cover human interaction, the Business Process Management Notation (BPMN) [OM13] is commonly appreciated by business users [CT12] for its intuitive graphical notation which comprises the difficult details of business processes. Elements are, e.g., *data*, *flow objects* or *connection objects* to connect *activities* and *events* in a control logic. The BPMN provides further concepts which cope with the complexity of business process, e.g., hierarchical structuring or causal branching.

3 Implementing Workflows for Digital Shadow Creation

We make use of workflows to enable a digital shadow designer to capture the details of complex aggregation processes, having the data flow traceable, and integrating the human. Workflow modeling languages like the BPMN are designed to integrate human interaction into a semi-automated process. The conceptual model described in [Be21] already captures the conceptual basics for modeling DSs. However, it was clear that it has to be extended when we were exploring further use cases.

Fig. 1 shows the conceptual model using the UML class diagram [Ru16] notation with the workflow extensions. Please note that Fig. 1 shows only the related elements relevant for this extension. The digital shadow stands for an *Asset* and is tailored to a *Purpose*, e.g., a DS of a cyber-physical production system with the purpose of finding optimized parameters to

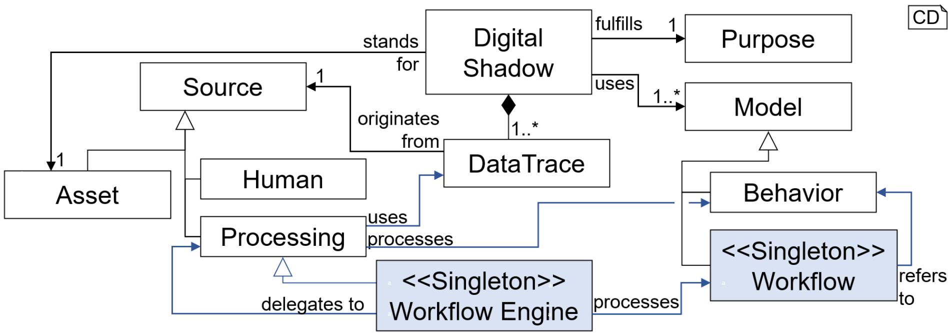


Fig. 1: Adapted conceptual model of digital shadows with workflows. The blue colored elements mark the added classes and associations.

minimize scrap. *Data Traces* build the core of the digital shadow and hold all data provided to and from the DS. Each data trace has its origin in a *Source* which could be the asset itself or, e.g., a *Human* via manual input or a *Processing* component that, for instance, converts data coming from the machine. *Models* are used to describe the asset or to express system or digital shadow behavior. Here, we have added a *Workflow* model which is processed by a *Workflow Engine*. The *Processing* component is super class to the workflow engine and has a connection to the *Behavior* models as well as to the data traces in use. The data traces can be input, intermediate result, or end result of the workflow execution; behavior models are utilized to specify the computation of the resulting data trace. This way, we have a clear traceability of how and with which input new data traces are computed. Our variant of the digital shadow conceptual model uses one main workflow model which controls the process of aggregating and abstracting. This way, there is no misconception on how data traces are produced and on how the digital shadow acts. Since the workflow model only presents a control structure, the actual computations are modeled in behavior models. Therefore, the workflow model needs to refer to those in the occurring tasks. The workflow engine can then delegate the execution of this behavior model to the responsible processing component.

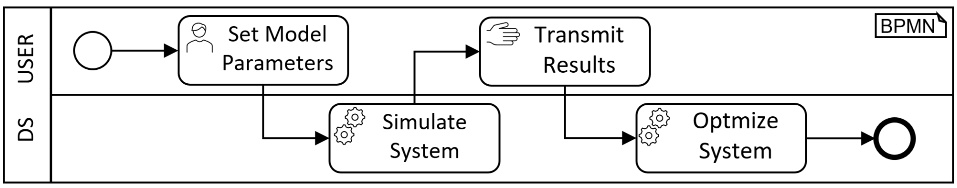


Fig. 2: A human integrated workflow to aggregate data modeled as BPMN.

We demonstrate our concept on a small example. Given a system, the DS first runs an analytic simulation on the initial data set and its result is then used to optimize the asset's parameters regarding some purpose. Before processing the initial data coming from the asset, a domain expert wants to set configuration parameters of both behavior models. The simulation is executed in one software component and the optimization is executed in another. In the current state, there exists no connection which results in a gap in communication. Therefore, a DS operator needs to interact with the system and manually transmit the simulation results to the optimization execution as input. This process is modeled in BPMN as shown in Fig. 2 with one lane for the DS operator and one for automatic tasks, as well as the four tasks of setting the parameters, simulate the system's behavior, manual transmitting between the software components and computing the optimized new machine parameters. For simplicity, we omitted modeling data transmissions in the BPMN model. Fig. 3 outlines how a digital

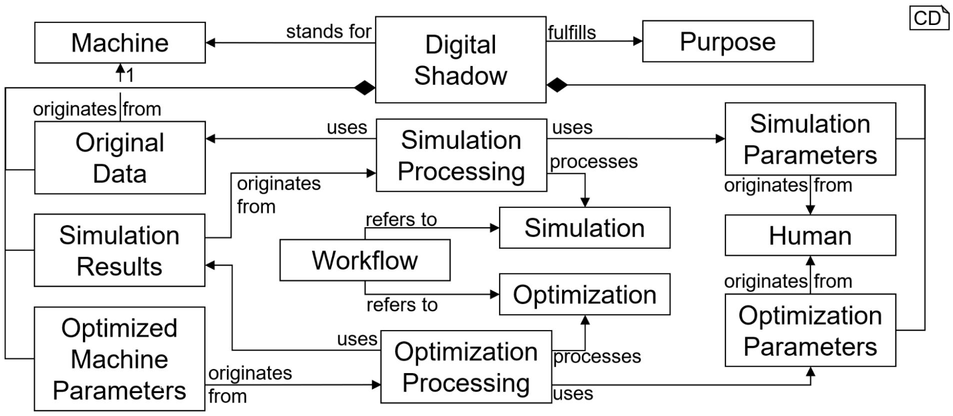


Fig. 3: Exemplary modeling of a digital shadow incorporating a workflow with human interaction.

shadow design benefits from our modeling technique. The DS is composed of the five data traces: *Original Data* as initial input which originates from the machine and is used by the *Simulation Processing*, the *Simulation Results* as intermediate result which now originates from the simulation processing and also acts as input for the *Optimization Processing*, the final result *Optimized Machine Parameters* which only originates from the optimization processing, and finally the two user entered *Simulation/Optimization Parameters*. Both of the processing components process the behavior models referenced by our *Workflow* model. Other elements are left out for the purpose of readability, such as the connection from the DS to the models in use or the workflow engine. In the BPMN model, these data traces would act as in- and output for the different tasks. Data stored in our digital shadow can now exactly be traced back to its origin and to the calculation specifications used for its computation. The human-machine-interaction is modeled in the BPMN model and all set parameters can be stored as well.

4 Discussion/Conclusion

In this paper, we proposed a methodology to incorporate complex and human interactive processes to the modeling of a digital shadow. We extended the conceptual model by a workflow model which captures the individual steps needed for computing the data the digital shadow shall provide. Integrating the human interaction modeled as part of the digital shadow aggregation process changes the way of perceiving the digital shadow as a fully autonomous software component. This lets a digital shadow designer model systems for which the digital automation is not yet as advanced. Workflows can model complex processes with many single activities which might split into parallel processes or loop back to previous tasks. Control logic such as conditional gates allows for actual logic affecting the aggregation process inside.

In the long run, such workflow models could be used to further increase automation or to train new users. If we analyze the manual tasks in workflow models we could further automate, e.g., data transferring between different systems as relevant input and output parameters are already defined in the workflow models. The workflow models can be used within training material and assistive systems [MRV20] which guides the human worker through the processes. Clearly, this approach has to be validated by further use cases.

Bibliography

[Be21] Becker, Fabian; Bibow, Pascal; Dalibor, Manuela; Gannouni, Aymen; Hahn, Viviane; Hopmann, Christian; Jarke, Matthias; Kröger, Moritz; Lipp, Johannes; Maibaum, Judith; Michael, Judith; Rumpe, Bernhard; Sapel, Patrick; Schäfer, Niklas; Schmitz, Georg J.; Schuh, Günther; Wortmann, Andreas: A Conceptual Model for Digital Shadows in Industry and its Application. In: 40th Int. Conf. on Conceptual Modeling (ER'21). Springer, 2021.

[Bi20] Bibow, Pascal; Dalibor, Manuela; Hopmann, Christian; Mainz, Ben; Rumpe, Bernhard; Schmalzing, David; Schmitz, Mauritius; Wortmann, Andreas: Model-Driven Development of a Digital Twin for Injection Molding. In: Int. Conf. on Advanced Information Systems Engineering (CAiSE'20). volume 12127 of LNCS. Springer, pp. 85–100, 2020.

[CT12] Chinosi, Michele; Trombetta, Alberto: BPMN: An Introduction to the Standard. Comput. Stand. Interfaces, 34(1), 2012.

[La21] Ladj, Asma; Wang, Zhiqiang; Meski, Oussama; Belkadi, Farouk; Ritou, Mathieu; Da Cunha, Catherine: A knowledge-based Digital Shadow for machining industry in a Digital Twin perspective. Journal of Manufacturing Systems, 58:168–179, 2021.

[LKC19] Loucopoulos, Pericles; Kavakli, Evangelia; Chechina, Natalia: Requirements Engineering for Cyber Physical Production Systems. In: Advanced Information Systems Engineering, volume 11483 of LNCS, pp. 276–291. Springer, 2019.

[MRV20] Michael, Judith; Rumpe, Bernhard; Varga, Simon: Human Behavior, Goals and Model-Driven Software Engineering for Assistive Systems. In: EMSIA 2020. CEUR, 2020.

[OM13] OMG: Business Process Model and Notation (BPMN), Version 2.0.2. Technical report, Object Management Group, 2013.

[Ru16] Rumpe, Bernhard: Modeling with UML: Language, Concepts, Methods. Springer International, July 2016.

[vdA21] van der Aalst, Wil MP: Concurrency and objects matter! Disentangling the fabric of real operational processes to create digital twins. In: International Colloquium on Theoretical Aspects of Computing. Springer, pp. 3–17, 2021.

Modelling Human Factors in Cyber Physical Production Systems by the Integration of Human Digital Shadows

Alexander Mertens[1], Philipp Brauner[1], Ralph Baier[1], Florian Brillowski[1], Hannah Dammers[1], Marc Van Dyck[1], Iris Kong[1], Peter Königs[1], Frauke Kordtomeikel[1], Gian Luca Liehner[1], Sebastian Pütz[1], Niklas Rodemann[1], Anne Kathrin Schaar[1], Linda Steuer-Dankert[1], Luisa Vervier[1], Shari Wlecke[1], Thomas Gries[1], Carmen Leicht-Scholten[1], Saskia K. Nagel[1], Frank T. Piller[1], Günther Schuh[1], Martina Ziefle[1] and Verena Nitsch[1]

Abstract: The future of industrial manufacturing and production will increasingly manifest in the form of cyber physical production systems. Here, Digital Shadows will act as mediators between the physical and digital world to model and operationalize the interactions and relationships between different entities in production systems. Until now, the associated concepts have been primarily pursued and implemented from a technocentric perspective, in which human actors play a subordinate role, if they are considered at all. This paper outlines an anthropocentric approach that explicitly considers the characteristics, behavior, and traits and states of human actors in socio-technical production systems[2]. For this purpose, we discuss the potentials and the expected challenges and threats of creating and using Human Digital Shadows in production.

Keywords: Human Digital Shadow, Cyber Physical Production Systems, Human Factors

1 Motivation for Integration of Human Digital Shadows

Across the whole life cycle of products from design, production, and use the usage of (digital) data gains in importance. These data are often referred to as Digital Twins or Digital Shadows and represent either all information available or task- and context-dependent aggregated views and enable, for example, retrospective analysis of defects or damages, the real-time control of production, or prospective estimation of products' properties, such as durability [Br22]. This enables companies to quickly react to changes despite the increasing complexity of the products and processes and will therefore gain in importance, especially due to the increasing interdependencies and volatility of global production chains. Research into corresponding approaches as part of the Internet of Production (IoP)[3] showed that, to the best of the authors' knowledge, Digital Shadows have been primarily characterized by a technocentric view, in which human actors are only considered as an abstract resource alongside other process parameters and physical objects. However, due to their unique skills and abilities and despite much progress in artificial intelligence and automation, humans will continue to be an immanent part of

[1] RWTH Aachen University, Aachen, Germany - Contact: a.mertens@iaw.rwth-aachen.de

[2] A detailed analysis and discussion using SWOT analysis can be found in [Me21]. This article contains a résumé of the core results regarding the use in model-based engineering.

[3] Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under German's Excellence Strategy - EXC-2023 Internet of Production - 390621612

production systems. Humans show a much higher inter- and intraindividual variance compared to technical components. Hence, smart adaptation of processes and interfaces offer large potential for optimization.

We therefore suggest Human Digital Shadows that integrate empirically determined human-related data to improve the adaptability and performance of cyber-physical production systems. These Human Digital Shadows integrate and aggregate data from human actors, their actions, and behaviors within the socio-technical system under consideration as a source or sink. The Human Digital Shadows are not distinct from the already established concept of Digital Shadows but represents an extension facilitating the analysis of existing interactions between people, technology, and organizations. They include, for example, behavior and movement patterns, individual working methods, anthropometric data such as measurements or forces, physiological and cognitive parameter progressions, personality states and traits, abilities, skills, and experiences, as well as socio-demographic information [Me21].

2 Discussion of Challenges and Threats

The significance of Human Digital Shadows for the sustainable design of cyber physical production systems is that corresponding information is an immanent part of process monitoring, and thus problems for and with human stakeholders are not only identified retrospectively but can be proactively avoided. Changes to equipment and tools can be made immediately if existing workstations are not compatible with a persons' body size and abilities. Interfaces can be customized to meet the needs of a changing workforce, or organizational measures can be automatically initiated to reduce workload spikes. Further, effective collaboration between humans and robots can be supported using Human Digital Shadows if, for example, a technical system knows the individual strengths and weaknesses and can adjust the task or level of support to the situation.

However, for all the expected benefits of integrating Human Digital Shadows in cyber-physical production systems, the ethical, legal, and social implications and potential risks must be considered and sensibly balanced with the benefits. This is particularly crucial, as different stakeholders in socio-technical production systems often have diverging interests and perceptions of capturing personal data (e.g., employees [Pü22] and mangers [To22]). Honoring ethical standards and respecting self-determination and perceived autonomy of the employees will improve the social acceptance of using Human Digital Shadows. The evaluation of this approach must carefully weigh the potential benefits through smarter adaptation against the disadvantages in form of possible misuse. Data protection must be a priority to prevent internal or external misuse. This is especially true as the Human Digital Shadows may contain sensitive personal data, such as the employee's health or performance, which carry the risk of misuse, extortion, and embarrassment.

Bibliography

[Me21] A. Mertens et al.: Human Digital Shadow: Data-based Modeling of Users and Usage in the Internet of Production. In: Proceedings of 14th International Conference on Human System Interaction (HSI), 2021, pp. 1-8, 2021.

[Br22] P. Brauner et al.: A Computer Science Perspective on Digital Transformation in Production. ACM Trans. Internet Things (3) 2, article 15. doi.org/10.1145/3502265

[Pü22] Pütz, S. et al. (2022) 'Using IoT devices for sensor-based monitoring of employees' mental workload: Investigating managers' expectations and concerns', Applied Ergonomics, 102, p. 103739. doi: 10.1016/j.apergo.2022.103739.

[To22] Tolsdorf, J., Reinhardt, D. and Iacono, L. Lo (2022) 'Employees' privacy perceptions: exploring the dimensionality and antecedents of personal data sensitivity and willingness to disclose', Proceedings on Privacy Enhancing Technologies, 2022(2), pp. 68–94. doi: 10.2478/popets-2022-0036.

J. Michael, J. Pfeiffer, A. Wortmann (Hrsg.): Modellierung 2022 Satellite Events,
Digital Library, Gesellschaft für Informatik e.V. 150

A Vision Towards Generated Assistive Systems for Supporting Human Interactions in Production

Judith Michael[1]

Abstract: Human workers need to cope with complex production settings when handling and monitoring cyber-physical production systems. Assistive systems can provide situational step-by-step support for human behavior, e.g., when interacting with a machine or for manual assembly. These systems need to take personal knowledge, workers skills or personal restrictions into account and are therefore subject to privacy concerns. However, the engineering of such interactive assistive systems within the production domain is a complex task as they might support critical functionality in dangerous environments and have a high need for safety and privacy considerations due to processing personal data. We want to investigate how the software engineering process of assistive systems in production can be improved to achieve higher reusability. Current research focuses on specific use cases and implements systems specifically for those needs without reusability in mind. We suggest using behavior and context models in a generative approach, to create a reusable method to engineer assistive systems for production environments, either as own applications or as services integrated within digital twins. We have already applied model-driven methods for assistive systems in the smart home domain and discuss the opportunities and challenges of an application of these methods for the production domain. These methods can facilitate the engineering of assistive functionalities within applications in production while meeting privacy, adaptability, and context-sensitivity requirements.

Keywords: Assistive Systems; Production; Human Support; Digital Twin; Process Models; Model-Driven Software Engineering

1 Novel Directions Talk

Motivation. The need for assistive systems in manufacturing grows due to a high complexity of tasks [LL12] and a large variety of products, production systems and tools [Fa22] resulting from mass customization. The engineering of interactive assistive systems within the production domain is challenging as they might support critical functionality in dangerous environments and have a high need for safety and privacy considerations due to the processing of personal data about human behavior. In research, current approaches assist employees on the shop floor [Ul16], use AR to assist manufacturing tasks such as assembly guidance or the inspection and evaluation of the machining processes before performing real machining [NO13], or use process models to visualize instructions for service operations [UFH19]. Current development approaches for assistive systems supporting human interactions (1) suggest mainly hand-written systems or (2) are not generalizable. The systems and graphical user interfaces must be newly developed for every use case.

[1] Software Engineering, RWTH Aachen University, michael@se-rwth.de

Other model-based development approaches support only certain use cases and are not generalizable.

Research question. Thus, we want to investigate *how the software engineering process of assistive systems in production can be improved to achieve a higher reusability*. We believe that the use of behavior and context models in a generative approach is a purposeful method to realize the engineering of reusable assistive systems for production environments, either as standalone applications or as services integrated into digital twins.

Assistive Systems. There are three main aspects that constitute an assistive system [Hö19]: (1) They provide situational support for human behavior, which requires them to be knowledgeable about situations and adaptable. (2) The support is based on information from previously stored and real-time monitored structural context and behavior data, which requires comprehensive models about the context as well as human behavior models. Moreover, they need monitoring capabilities to update the current status in processes and environment models. (3) They provide support at the moment a person needs it or asks for it, which requires intelligent support detection and interactive components.

Generated Assistive Systems for Smart Homes. We are successfully using the generator framework MontiGem [Ge20] to create assistive systems for smart homes supporting, e.g., cooking processes. Process and context models allow us to generate web-based multi-modal user interfaces that compose supporting texts from model information and provide pictures and audio support. We cover concepts for leading users to find or place objects using spatial relationships and object nesting.

The Need for Generated Assistive Systems in Production. These solutions can be transferred to other domains. Through our insights into production within the Cluster of Excellence "Internet of Production"(IoP)[2], we have noticed several areas where a generative approach for the creation of assistive systems is needed and would be well applicable. A generated assistive system could replace instruction manuals [SM18] for machines, which are often outdated or missing important aspects when tools are changed for a production site. Another area is manual assembly [HRU16], where tools have their fixed place and finding them in the right order or learning changed processes can be supported by training. Moreover, assistive systems can provide ergonomic support, e.g., when using data from working posture analysis [HMB18] or when monitoring the employees' mental workload [Pü22].

Challenges. When transferring model-based and generative concepts for the engineering of assistive systems to the production domain, we face several challenges.

- *High amount of relevant contextual information*: The meta-model for context information can be defined on an abstraction level which allows to reuse it for different use cases and domains [MS17]. However, the set-up of context-sensitive assistive systems is time consuming when we need to cover a high number of context objects. Methods

[2] Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2023 Internet of Production - 390621612. Website: https://www.iop.rwth-aachen.de

such as semantic annotation of user manuals [SM18] may fasten this process but further research, e.g., on digital shadows [Be21], or using controlled natural language in domain-specific modeling languages is needed to improve the set-up.

- *High heterogeneity of user interfaces*: Assistive systems in production could exist in loud environments, users could not be able to read screens from their working position or due to their movement. Thus, generated assistive systems have to cover various multi-modal user interfaces on different devices to be reusable.
- *Adaptability to personal needs*: Assistive systems need to cover a high diversity of personal needs, e.g., human abilities, their work and learning history [UFH19], personal restrictions [Lu21], or the stress level, and to be adaptable to these needs. Specific needs [Gr21] within the production domain have to be further analyzed and we have to develop methods to incorporate these needs into models usable in the generation process of assistive systems.
- *Privacy needs*: Every assistive system needs a certain amount of personal data to be accepted by users, e.g., to be adaptable and adjustable to user preferences. Assistive systems have to consider privacy-by-design principles. Model-based approaches for privacy preservation such as [Mi19] could be incorporated into assistive systems to make sure that the data is used purposefully.

Conclusion. Model-driven software engineering can facilitate the engineering of assistive functionalities in manufacturing applications while meeting the requirements for privacy, adaptability and context sensitivity. However, there are still several challenges to overcome that raise new research questions for the model-based and model-driven software engineering of assistance systems. Moreover, we need to further explore the generalizability of a model-driven software engineering approach for the engineering of assistive systems in different use cases.

Bibliography

[Be21] Becker, Fabian; Bibow, Pascal; Dalibor, Manuela; Gannouni, Aymen; Hahn, Viviane; Hopmann, Christian; Jarke, Matthias; Koren, Istvan; Kröger, Moritz; Lipp, Johannes; Maibaum, Judith; Michael, Judith; Rumpe, Bernhard; Sapel, Patrick; Schäfer, Niklas; Schmitz, Georg J.; Schuh, Günther; Wortmann, Andreas: A Conceptual Model for Digital Shadows in Industry and its Application. In: Conceptual Modeling ER '21. Springer, 2021.

[Fa22] Fadhlillah, Hafiyyan Sayyid; Feichtinger, Kevin; Meixner, Kristof; Sonnleithner, Lisa; Rabiser, Rick; Zoitl, Alois: Towards Multidisciplinary Delta-Oriented Variability Management in Cyber-Physical Production Systems. In: 16th International Working Conference on Variability Modelling of Software-Intensive Systems. VaMoS '22. ACM, 2022.

[Ge20] Gerasimov, Arkadii; Michael, Judith; Netz, Lukas; Rumpe, Bernhard; Varga, Simon: Continuous Transition from Model-Driven Prototype to Full-Size Real-World Enterprise Information Systems. In: 25th Am. Conf. on Information Sys. (AMCIS'20). AIS, 2020.

[Gr21] Grundy, John; Khalajzadeh, Hourieh; McIntosh, Jennifer; Kanij, Tanjila; Mueller, Ingo: HumaniSE: Approaches to Achieve More Human-Centric Software Engineering. In (Ali, Raian; Kaindl, Hermann; Maciaszek, Leszek A., eds): Evaluation of Novel Approaches to Software Engineering. Springer, pp. 444–468, 2021.

[HMB18] Hellig, Tobias; Mertens, Alexander; Brandl, Christopher: The interaction effect of working postures on muscle activity and subjective discomfort during static working postures and its correlation with OWAS. International Journal of Industrial Ergonomics, 68:25–33, 2018.

[Hö19] Hölldobler, Katrin; Michael, Judith; Ringert, Jan Oliver; Rumpe, Bernhard; Wortmann, Andreas: Innovations in Model-based Software and Systems Engineering. The Journal of Object Technology, 18(1):1–60, July 2019.

[HRU16] Hinrichsen, Sven; Riediger, Daniel; Unrau, Alexander: Assistance Systems in Manual Assembly. In (Villmer, Franz-Josef; Padoano, Elio, eds): Production Engineering and Management. p. 3–14, 2016.

[LL12] Liu, Peng; Li, Zhizhong: Task complexity: A review and conceptualization framework. International Journal of Industrial Ergonomics, 42(6):553–568, 2012.

[Lu21] Luy, Calvin; Law, Jeremy; Ho, Lily; Matheson, Richard; Cai, Tracey; Madugalla, Anuradha; Grundy, John: A Toolkit for Building More Adaptable User Interfaces for Vision-Impaired Users. In: 2021 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC). 2021.

[Mi19] Michael, Judith; Netz, Lukas; Rumpe, Bernhard; Varga, Simon: Towards Privacy-Preserving IoT Systems Using Model Driven Engineering. In: MODELS 2019. Workshop MDE4IoT. CEUR Workshop Proceedings, pp. 595–614, 2019.

[MS17] Michael, Judith; Steinberger, Claudia: Context Modeling for Active Assistance. In (Cabanillas, Cristina; España, Sergio; Farshidi, Siamak, eds): Proc. of the ER Forum 2017 and the ER 2017 Demo Track co-located with the 36th Int. Conference on Conceptual Modelling (ER 2017). pp. 221–234, 2017.

[NO13] Nee, A.Y.C.; Ong, S.K.: Virtual and Augmented Reality Applications in Manufacturing. IFAC Proceedings Volumes, 46(9):15–26, 2013. 7th IFAC Conference on Manufacturing Modelling, Management, and Control.

[Pü22] Pütz, Sebastian; Rick, Vera; Mertens, Alexander; Nitsch, Verena: Using IoT devices for sensor-based monitoring of employees' mental workload: Investigating managers' expectations and concerns. Applied Ergonomics, 102:103739, 2022.

[SM18] Steinberger, Claudia; Michael, Judith: Towards Cognitive Assisted Living 3.0. In: International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops 2018). IEEE, pp. 687–692, march 2018.

[UFH19] Uhlmann, Eckart; Franke, David; Hohwieler, Eckhard: Smart Maintenance – dynamic model-based instructions for service operations. Procedia CIRP, 81:1417–1422, 2019. 52nd CIRP Conference on Manufacturing Systems (CMS).

[Ul16] Ullrich, Carsten: Rules for Adaptive Learning and Assistance on the Shop Floor. In: Int. Conf. on Cognition and Exploratory Learning in the Digital Age (CELDA). 2016.

J. Michael, J. Pfeiffer, A. Wortmann (Hrsg.): Modellierung 2022 Satellite Events,
Digital Library, Gesellschaft für Informatik e.V. 154

1. Modellierung und Simulation im Engineering und zur virtuellen Inbetriebnahme im Maschinen- und Anlagenbau

Ronald Schmidt-Vollus[1], Eric Handschuh[1], Axel Gödrich[1], Christian Hölzer[1]

Abstract: Der Workshop stellt das Thema Virtuelle Inbetriebnahme (VIBN) sowie Modellbindung und Simulation im Engineering von automatisierten Maschinen und Anlagen vor. Im Mittelpunkt stehen die Meinungen und Thesen des VDI/VDE-GMA FA 6.11 Virtuelle Inbetriebnahme. Der Workshop stellt die VIBN mit ihren Methoden, Anwendungsszenarien und ihrem Nutzen vor. Der Workshop gibt, in Form eines Tutorials, einen Einblick in die Modellbindung und bietet einen Ausblick auf die weitere Nutzung von VIBN-Modellen im Lebenszyklus von Anlagen und Maschinen.

Keywords: Virtuelle Inbetriebnahme; VIBN

Vorwort

Die virtuelle Inbetriebnahme (VIBN) hat sich im Laufe der Jahre in vielen Bereichen des Maschinen- und Anlagenbaus zu einem etablierten Instrument im Engineering von Maschinen und Anlagen entwickelt. Dennoch gibt es noch immer zahlreiche Herausforderungen, und bei vielen Unternehmen ist die VIBN noch nicht integraler Bestandteil ihrer Unternehmensprozesse.
Die Grundidee einer virtuellen Inbetriebnahme ist das Testen eines Automatisierungssystems für eine Produktions- oder Fertigungsanlage bzw. Maschine an Hand eines Modells dieser, noch bevor die eigentliche physische Anlage bzw. Maschine vollständig gebaut ist. Das heißt, das reale Steuerungssystem wird, gegebenenfalls auf emulierter Hardware, mit einem virtuellen Abbild der Anlage, welches in einem Simulationssystem ausgeführt wird, verbunden. Dies erfolgt idealerweise in Echtzeit. Für das der Simulation zugrundeliegende Modell hat sich auch der Begriff des ausführbaren digitalen Zwillings als geeignet erwiesen. Die Richtlinie VDI/VDE 3693-1 liefert einen guten Überblick.
Neben der eigentlichen Idee der VIBN, dem Test eines Steuerungssystems, welche als VIBN im engeren Sinne bezeichnet werden kann, bietet der ausführbare digitale Zwilling zahlreiche weitere gewinnbringende Anwendungsmöglichkeiten im Engineering (VIBN im erweiterten Sinne), wie z.B. Operator Training oder produktionsbegleitende Simulation zur Optimierung von Anlagen.

[1] Technische Hochschule Nürnberg Georg Simon Ohm (THN), Nuremberg Campus of Technology, Keßlerplatz 12, 90489 Nürnberg, ronald.schmidt-vollus@th-nuernberg.de

Ziel des Workshops ist es, die VIBN einer breiteren Öffentlichkeit vorzustellen und deren Nutzen im Engineering von Maschinen und Anlagen zu vermitteln. Im ersten Teil des Workshops wird die VIBN vorgestellt, es werden praktische Beispiele gezeigt und die Modellbildung Form eines Tutorials vermittelt. Im 2. Teil des Workshops wird die Einführung der VIBN im Unternehmen sowie erweiterte Nutzungsmöglichkeiten für VIBN-Simulationsmodelle gezeigt. Hierzu wird ein wissenschaftlicher Beitrag präsentiert, welcher ergänzt wird um einen Work-In-Progress-Bericht ergänzt.
Insbesondere bei den Tutorials und den Diskussionen sind die Workshop-Teilnehmerinnen und Teilnehmer herzlich eingeladen, eigene Fragestellungen mitzubringen.

Insgesamt sollen mit dem Workshop Austausch und Diskussion über das Thema VIBN angeregt werden, um zukünftigen Forschungs- und Entwicklungsbedarf zu identifizieren.

J. Michael, J. Pfeiffer, A. Wortmann (Hrsg.): Modellierung 2022 Satellite Events,
Digital Library, Gesellschaft für Informatik e.V. 156

Digitale Technologien für die modellgestützte Überwachung, Wartung und Modernisierung von Verdichter- und Turbinenanlagen

Robert Thiel[1] Jens Jäkel[2] Kevin Schleifer[3] Rico Schulze[4]

Abstract: In dieser Arbeit werden Methoden untersucht und entwickelt, die auf Basis des digitalen Zwillings die automatische Erkennung und Überwachung des Anlagenzustandes (**Smart Monitoring**) verfolgen, eine ereignisgesteuerte Instandhaltung (**Smart Maintenance**) mit Hilfe von Fehleranalysen und Vorhersagen ermöglichen und Kennzahlen im Anlagenbetrieb ermitteln und durch Anpassung der Anlagen- und Reglerstruktur teilautomatisiert optimieren (**Smart Revamp**). Für den Einsatz im produktiven Betrieb wird eine geeignete IT-Plattform aufgebaut, die die entwickelten Methoden und prototypisch implementierten Analysealgorithmen als Service erstellt. Schließlich wurden die Entwicklungen anhand von Anwendungsbeispielen getestet und optimiert.

Keywords: digitaler Zwilling; Turboverdichter; Smart Monitoring; Smart Maintenance

1 Motivation

Mit Hilfe des digitalen Zwillings ist es möglich, immer mehr Informationen über eine Anlage zu sammeln. Die Herausforderung liegt vor allem in der intelligenten Nutzung dieser enormen Datenmenge, um die richtigen Schlüsse für den laufenden Anlagenbetrieb, die nächste notwendige Wartung oder eine mögliche Anlagenmodernisierung zu ziehen. In diesem Zusammenhang rückt die vorausschauende Instandhaltung zunehmend in den Vordergrund. Insbesondere geht es darum, ungeplante Anlagenstillstände aufgrund von Alterung und Verschleiß, defekten Anlagenteilen oder kritischen Betriebszuständen zu vermeiden. Ungeplante Anlagenstillstände können durch regelmäßige Instandhaltungsmaßnahmen reduziert werden. Dies erfordert jedoch regelmäßige Eingriffe in die Anlage, die in der Regel mit einer Abschaltung der Anlage verbunden sind. Zwar lassen sich so Produktionsunterbrechungen planen und die Anlagenverfügbarkeit wird erhöht. Dennoch bleibt der Gedanke, dass eine zeitgesteuerte Instandhaltungsmaßnahme nicht unbedingt notwendig ist. Hier bietet der Digitale Zwilling innovative Lösungsmöglichkeiten, um fehlerhaftes Anlagenverhalten frühzeitig zu erkennen und vorherzusagen, wodurch Instandhaltungsmaßnahmen vorausschauend gesteuert werden können.

[1] HTWK Leipzig, Fakultät für Ingenieurwissenschaften, Leipzig, Deutschland, robert.thiel@htwk-leipzig.de
[2] HTWK Leipzig, Fakultät für Ingenieurwissenschaften, Leipzig, Deutschland, jens.jaekel@htwk-leipzig.de
[3] Avicomp Controls GmbH, Leipzig, Deutschland, kevin.schleifer@avicomp.com
[4] Avicomp Controls GmbH, Leipzig, Deutschland, rico.schulze@avicomp.com

Ebenso unterstützt der Digitale Zwilling den Entscheidungsprozess bei der Modernisierung von Verdichter- und Turbinenanlagen. Mit Hilfe des Modells kann eine bestehende Anlage auf Engpässe oder ineffizientes Betriebsverhalten untersucht werden, was wiederum als Grundlage für die Bewertung von spezifischen Optimierungsansätzen dient. Darüber hinaus können geplante Anlagenänderungen vor dem Eingriff in die reale Anlage umfassend getestet, virtuell in Betrieb genommen und das zuständige Personal geschult werden.

1.1 Stand der Technik

Mit dem Digitalen Zwilling wurde die Grundlage geschaffen eine verbesserte Integration, Wartung und Analyse des realen Systems umzusetzen. Dabei ist eine kontinuierliche Anpassung der Modelle eines Digitalen Zwillings auf Grundlage der realen Messdaten notwendig und bedarf eines hohen Automatisierungsgrads bezüglich der Datenfilterung, -validierung und der Bestimmung von Betriebszuständen. Die komplexe numerische Herausforderung der Parameteroptimierung auf Modellebene im Digitalen Zwilling kann mithilfe von nichtlinearen Optimierungsverfahren angegangen werden [Sc14; UU12]. Anhand der Ergebnisse bei der Parameterbestimmung von elektrischen Maschinen wird die Effizienz solcher Verfahren aufgezeigt [Fl03]. Ein weiterer möglicher Ansatz besteht in der Verwendung von metaheuristischen Verfahren. Dazu gehören bspw. genetische Algorithmen, die sich dank ihrer hohen Flexibilität besonders für komplexe Parameteroptimierungsaufgaben eignen. Speziell bei stationären Betriebszuständen haben sich derartige Verfahren bewährt [BS93; MS03]. Für dynamische Betriebszustände werden zudem immer öfter Bayes-Optimierer verwendet, die mithilfe von historischen Messdaten eben die Dynamik des Systems einbeziehen [MPW11; Sh15].
Neben verschleißbedingten Veränderungen, die über den Lebenszyklus einer Anlage auftreten, können sich verändernde Anforderungen hinsichtlich Kapazität oder Effizienz Einfluss auf notwendige Wartungs-, Umbau oder Modernisierungsmaßnahmen haben. Reicht der einfache Austausch einer Komponente nicht aus, um diesen Herausforderungen zu begegnen, müssen strukturelle Änderungen an der Anlage vorgenommen werden. Für komplexe Verdichter- und Turbinenanlagen soll daher eine globale Design- und Strukturoptimierung bezüglich verschiedener Leistungskennzahlen (KPIs) vorgenommen werden. Diese Leistungskennzahlen beschreiben betriebswirtschaftliche Kennzahlen. Insbesondere soll der Fokus auf Energieeffizienz, Rohstoffproduktivität und möglichen Umbaukosten einer Anlage liegen.
Die Designoptimierung wird insbesondere zur Suche geeigneter Struktur und Parametrierung von einzelnen Komponenten verwendet. Besonders im Automobilbau oder im Flugzeugbau werden nach bestimmen Gütekriterien, wie Energieeffizienz, Wirkungsgrad, Gewicht und Robustheit die idealen Formen gesucht, die das Gesamtsystem verbessern sollen. Häufig kommen hierzu metaheuristische Methoden zur Anwendung, aber mithilfe der Finite-Elemente-Methode wird die optimale Form gesucht [Di86; EO01; PA06; WS07]. Eine zum Standard gewordene Methode ist die Topologieoptimierung. Bei diesem Verfahren werdend die mechanischen Grundelemente die Gesamttopologie einer mechanischen

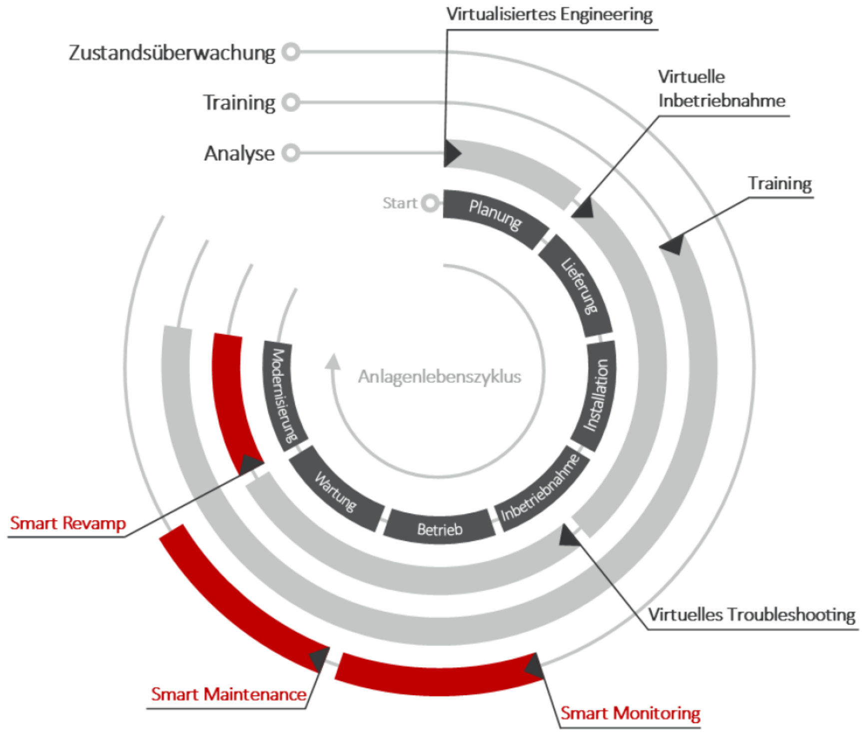


Abb. 1: Einordnung der Artefakte in den Anlagenlebenszyklus.

Maschine verändern. Dies führt zu einem diskreten Optimierungsproblem welches in der Wissenschaft gut untersucht wurde [Be99; MS13; RK92]. In der Literatur sind im Bereich der Turbomaschinen vor allem Optimierung hinsichtlich aerodynamischer Aspekte zu finden [LZ17; Mo14].

Die genannten Ziele erfordern ein digitales Abbild der Anlage. Dieser Digitale Zwilling wurde im Vorgängerprojekt TwinTurbo entwickelt und dient nun als Ausgangspunkt für die Entwicklung. In Abbildung 1 ist zu erkennen, wie sich Smart Monitoring, Smart Maintenance und Smart Revamp nach Abschluss der Entwicklung in den Anlagenlebenszyklus einbetten.

2 Methoden

Um die steigenden Komplexitäten und Qualitätsanforderungen sowie laufenden Betriebskosten verfahrenstechnischer Anlagen beherrschen und verbessern zu können, wird folgende Fragestellung untersucht, die bei Anlagenbetreibern allgegenwärtig ist:

Wann und wo ist welcher Eingriff in die Anlage erforderlich bzw. lohnenswert?

Die nötige Plattform liefert der Digitale Zwilling einer Anlage, auf dessen Basis spezifische Analysen zum zielgerichteten Eingriff in die Anlage und zu einem erheblichen Mehrwert des Anlagenbetreibers führen. In diesen Kontext betten sich die nachfolgend konkretisierten Ziele ein.

2.1 Smart Monitoring

Um eine valide Aussage treffen zu können, wann und wo ein Anlageneingriff erforderlich ist, muss zunächst eine Erkennung und Überwachung des Anlagenzustands erfolgen. Dieses wurde im Rahmen des ersten Projektziels Smart Monitoring umgesetzt. Dazu wurde ein automatisiertes Verfahren zum Anlernen des Digitalen Zwillings entwickelt, sodass ein ausreichend genaues, virtuelles Abbild der Anlage geschaffen wird. Dieses Abbild dient der Ermittlung des entsprechenden Referenzzustands bei gegebenen Eingangsgrößen der überwachten Anlage in Eingangsdruck, -temperatur, Stellsignale etc., sodass in der Folge der Vergleich mit dem aktuellen messtechnisch erfassten Anlagenzustand stattfindet. Dies führt wiederum zu aussagekräftigen Informationen, ob und wo ein abweichendes Anlagenverhalten vorliegt.

2.2 Smart Maintenance

Das zweite Projektziel wird unter dem Titel Smart Maintenance geführt und unterstützt den digitalen Wandel von zeitgesteuerter zu ereignisgesteuerter Instandhaltung bei Verdichter- und Turbinenanlagen. Unter Anwendung volldigitalisierter Analysewerkzeuge wird ein abweichendes Anlagenverhalten erkannt. Darüber hinaus werden die technologischen Komponenten bzw. die Kombination technologischer Komponenten identifiziert, welche das abweichende Verhalten verursachen. Durch zusätzliche Extrapolationsmodelle zur Vorhersage des Fehlerzustands erfolgt ein zielgerichteter Eingriff, um die Abweichung zu beseitigen und trägt damit wesentlich zur Kostenreduzierung des Anlagenbetreibers bei.

2.3 Smart Revamp

Als drittes Projektziel wird die Beseitigung ineffizienter oder sicherheitskritischer Betriebsweisen durch die Optimierung der Anlagen- und Regelungsstruktur mithilfe volldigitalisierter Analysewerkzeuge unter dem Namen Smart Revamp verfolgt. Durch Anwendung des Digitalen Zwillings wird hierbei die Anlagenperformanz durch anlagenspezifische Merkmale, sogenannte Key-Performance-Indikatoren (KPI) ermittelt und auf Basis dessen ineffizientes Betriebsverhalten identifiziert. Durch teilautomatisierte Optimierungsverfahren werden anschließend Lösungen zur Verbesserung der Anlagenperformanz gefunden. Dies kann die Anpassung der Regelungsstruktur, die Modifikation einer einzelnen Anlagenkomponente, aber die gezielte Strukturänderung eines Anlagenteils betreffen.

3 Umsetzung

Um auf der Grundlage des Digitalen Zwillings eine verbesserte Integration, Wartung und Analyse des realen Systems zu erreichen, ist eine kontinuierliche Optimierung der Modelle eines Digitalen Zwillings auf Basis der realen Messdaten notwendig und bedarf eines hohen Automatisierungsgrads bezüglich der Datenfilterung, -validierung und der Bestimmung von Betriebszuständen.

3.1 Optimierungsverfahren

Um die Komplexität und den möglichen Lösungsraum der Optimierung zu reduzieren, wird der gesamte Optimierungsprozess in drei aufeinanderfolgende Teilaufgaben unterteilt:

- der (stationären) Komponentenoptimierung,
- der stationären Modelloptimierung sowie
- der dynamischen Modelloptimierung.

Die drei Optimierungsaufgaben erfordern jeweils ausreichend Messinformationen der realen Anlage, sodass genügend Messpunkte für die Berechnung und für den Vergleich zwischen berechneter Modellgröße und realem Messwert mithilfe einer Kreuzvalidierung vorhanden sind.
Die Komponentenoptimierung passt Parameter einer einzelnen Komponente unabhängig vom Gesamtmodell an. Demnach muss für die Evaluierung der Zielfunktion nur die Komponente und nicht das gesamte Modell berechnet werden. Die durch die Komponentenoptimierung angepassten Parameter werden in der anschließenden stationären Modelloptimierung nicht weiter optimiert. Dies schränkt den möglichen Lösungsraum ein und reduziert damit den wesentlich komplexeren Aufwand zur Berechnung eines Modells. Die Optimierung des dynamischen Modellverhaltens verlangt Berechnungen über einen bestimmten Zeitraum und ist daher nochmals rechenaufwendiger. Der Startzeitpunkt einer solchen Simulation muss zudem in einem eingeschwungenen Zustand, z.B. einer stationären Ruhelage, gewählt werden. Zudem wird gewährleistet, dass das Ergebnis der dynamischen Modelloptimierung keinen rückwirkenden Einfluss auf das Ergebnis der stationären Modelloptimierung besitzt, da die zu optimierenden Parameter entweder nur einen stationären oder nur einen dynamischen Einfluss auf das Modellverhalten besitzen.

3.2 Zustandserkennung

In der Zustandserkennung werden die verfügbaren Messwerte und Stellsignale aus der realen Anlage mit den entsprechenden Zustands- und Prozessgrößen des Digitalen Zwilling verglichen. Neben Prozess- und Stellfehlern wirken sich die Messunsicherheiten der Sensorik und Modellfehler auf die Bildung der Residuen aus und beeinträchtigen so die Funktion der Zustandserkennung. Folglich sind für eine robuste Auslegung der Zustandserkennung die vorliegenden Modellfehler und Messunsicherheiten zu berücksichtigen. Die Abbildung

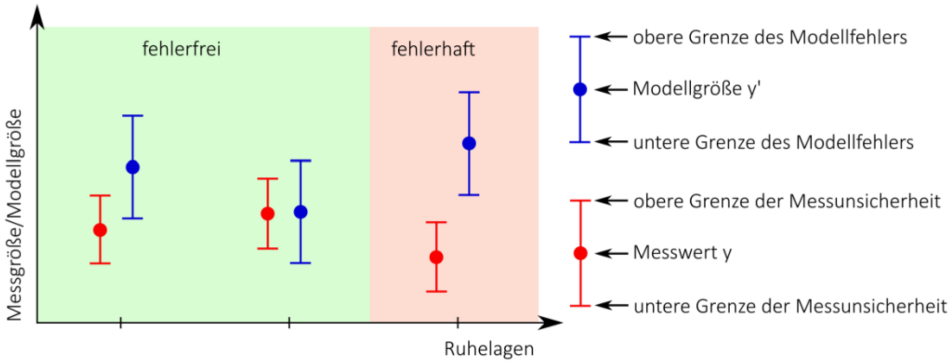


Abb. 2: Prinzipielle Funktionsweise der Zustandserkennung unter Berücksichtigung von Modellfehlern und Messunsicherheiten.

2 zeigt schematisch die Funktionsweise der Zustandserkennung unter Berücksichtigung des abgeschätzten Modellfehlers und der bestimmten Messunsicherheit für eine Mess- bzw. Modellgröße. Auf der x-Achse werden die diskreten Zeitpunkte der erkannten stationären Betriebspunkte aufgetragen. Jeder Ruhelage kann ein Messwert y und der Wert der entsprechenden Modellgröße y‘ des Digitalen Zwillings zugeordnet werden. Der rote Fehlerbalken stellt dabei das Konfidenzintervall des Messwertes dar. Der zulässige Modellfehler der je weiligen Modellgröße wird ebenfalls durch eine obere und untere Grenze beschränkt. Durch dieses Vorgehen ist es unter Berücksichtigung der Messunsicherheiten und des Modellfehlers möglich, den Zustand einer Messgröße eindeutig in fehlerfrei und fehlerhaft zu klassifizieren. Dabei gilt es zu beachten, dass die stochastischen Eigenschaften der Messunsicherheit vernachlässigt werden. Überlagern sich die Intervalle der Messunsicherheit und des Modellfehlers vollständig oder partiell, wird der Zustand der Messgröße als fehlerfrei angenommen. Liegt keine Überlagerung der Intervalle vor, wird der Zustand der Größe als fehlerhaft angenommen. Die Auswertung der Fehlerbalken erfolgt für jede Messgröße, die nicht als Berechnungsgröße des Modells definiert ist.

3.3 Fehlermodelle

Die Grundlage des entwickelten Verfahrens bilden Fehlermodelle, die in den Komponenten des Digitalen Zwillings implementiert sind. Durch sukzessives Zuschalten dieser Fehlermodelle und einem fortlaufenden Vergleich mit den Messdaten wird so das fehlerbehaftete Verhalten der realen Anlage im Digitalen Zwilling nachgebildet. Das Fehlermodell, welches die detektierten Abweichungen optimal über alle Messgrößen kompensiert, gibt eine direkte

Auskunft über die Ursache des Fehlers. Der Betrag des gefundenen Fehlermodellparameters liefert zudem Rückschlüsse auf das Ausmaß des Fehlers. Da lediglich die implementierten Fehlermodelle durch die Fehlerdiagnose untersucht werden, können folglich nur Fehler identifiziert werden, für die ein Fehlermodell im Digitalen Zwilling hinterlegt ist. Die Bestimmung der Fehlermodellparameter erfolgt mit Hilfe einer Optimierung, wobei jedem einzustellenden Parameter vorab feste Grenzen zuzuweisen sind. Diese Beschränkung der Parameter soll sicherstellen, dass das Modell den definierten Betriebsbereich der realen Anlage abbildet. Durch unbeschränkte Parameter könnten bspw. nicht physikalische Zustände innerhalb des Modells auftreten und so die Ergebnisqualität beeinflussen.
Die implementierten Fehlermodelle bilden verschiedene unerwünschte Anlagenzustände ab, die durch die unterschiedlichsten Schädigungsprozesse verursacht werden können. Potentielle Schädigungsprozessen in den zu untersuchenden Verdichter- und Turbinenanlagen sind vor allem Verschmutzungen, die durch verschiedene Ablagerungsprozesse entstehen, sowie Materialabtrag und Verformungen an den unterschiedlichsten Anlagenkomponenten. Eine Übersicht einiger im Digitalen Zwilling implementierter Fehlermodelle kann Tabelle 1 entnommen werden. Jedes der aufgelisteten Fehlermodelle wirkt direkt auf einen oder zwei physikalische Parameter der Anlagenmodells. Die entsprechenden Parameter sind der rechten Spalte der Tabelle zu entnehmen. Ein Fehlermodell wirkt entweder additiv oder multiplikativ auf die zugehörigen Modellparameter. Durch diesen generischen Modellierungsansatz ist es möglich die Komponenten des Digitalen Zwilling um beliebige Fehlermodelle zu erweitern. Zusätzlich wird durch die gewählte physikalische White-Box-Modellierung die einfache Interpretierbarkeit der Fehlerursache und des Fehlerausmaßes gewährleistet. Um den Rechenaufwand der Optimierung pro Fehlermodell möglichst gering zu halten, wurde zudem die Anzahl der einstellbaren Parameter pro Fehlermodell auf ein Minimum reduziert. So wird sich z.B. durch Ablagerungen innerhalb einer Rohrleitung der Leitungsquerschnitt verringern und die Rohrreibung erhöhen. Die Auswirkungen der beiden Effekte auf den Prozess sind aufgrund des verwendeten Modellierungsansatzes jedoch identisch. Somit ist es möglich den Effekt der Querschnittsverengung zu vernachlässigen.

Tab. 1: Auflistung einiger implementierter Fehlermodelle

Fehlermodell	**Parameter**
Verschmutzte Rohrleitung	Rohrreibungskoeffizient λ_{sta}
Kennlinienverschiebung Ventil	Durchflusskoeffizient K_v
Verschmutzter Wärmetauscher	Wärmeübertragungskoeffizienten α_M, α_R
Massenstrom Wärmetauscher Rohrseite	Rohrseitiger Massenstrom $\dot{m}_R$
Verschmutzte Saugseite Verdichter	Rohrreibungskoeffizient $\lambda_{1,sta}$
Verschmutzte Druckseite Verdichter	Rohrreibungskoeffizient $\lambda_{2,sta}$
Fehlerhaftes Vorleitgitter	Vorleitgitterstellung H_{ELA}
Kennfeldverschiebung Verdichter	Kennfeldverschiebung x_{off}, y_{off}

3.4 Key-Performance-Indikatoren-System

Die Key-Performance-Indikatoren (KPIs) dienen zur Bewertung der Performanz einer Anlage und können aus den verschiedensten anlagenspezifischen Kennzahlen ermittelt werden. Diese Kennzahlen umfassen technische und ökonomische sowie ökologische Randbedingungen und dienen z.B. der Bemessung des Gesamtwirkungsgrads, der Auslastung, der Energie- oder Ressourceneffizienz einer Anlage. Demnach werden für die Berechnung von KPIs neben messtechnisch erfassten Größen externe Kennzahlen wie Auslegungsdaten, Energie- und Materialkosten, Steuern, etc. benötigt.
Für die Einbindung von Messgrößen sowie konstanten Kennzahlen stellt der Digitale Zwilling durch die DataPoint- sowie ParameterAsset-Klasse (Definition konstanter Kenngrößen) eine geeignete Implementierung zur Verfügung. Nun gilt es ein Werkzeug zu entwickeln, die einerseits die vorhandenen Klassen mathematisch kombinierbar macht und andererseits die Einbindung externer Kennzahlen erlaubt, die nicht als konstant anzunehmen sind. Hierzu wird die DataOperation-Klasse eingeführt. Diese bietet die flexible Definition symbolischer, mathematischer Gleichungen. Eine automatisierte Symbolerkennung unterscheidet anschließend alle Variablen (Aliase) von mathematischen Operatoren. Den Aliasen können dann verschiedene Objekte wie Datenpunkte, Konstanten sowie Funktionen selbst zugewiesen werden. Dieser rekursive Aufruf erlaubt die Verrechnung mehrerer KPIs. Durch die Verknüpfung der Datenpunkte mit den Importdatensätzen sowie zu einer Komponente des Modells wird darüber hinaus die Möglichkeit geschaffen, KPI-Berechnungen auf Basis von Mess- und Simulationsdaten durchzuführen.
Um für die KPI Ermittlung komplexe, thermodynamische Realgasberechnungen zu ermöglichen, wird ferner die ThermodynamicOperation-Klasse implementiert. Dieser wird lediglich jeweils ein Datenpunkt für Druck und Temperatur sowie das entsprechende Fluid-Objekt zugewiesen. Mit der Auswahl einer gewünschten thermodynamischen Zustandsgröße, z.B. Dichte, spezifische Enthalpie und Entropie, ist die Konfiguration dieser Operation vollständig.
Ein einfaches Anwendungsbeispiel ist die Ermittlung der Energiekosten eines Verdichters anhand dessen innerer Leistung

$$P_i = (h_2 - h_1) \cdot \dot{m} \cdot K \tag{1}$$

wobei h_1 und h_2 der spezifischen Enthalpie von Eingangs- und Ausgangsseite und $\dot{m}$ dem Verdichtermassenstrom entsprechen, multipliziert mit den Energiekosten K pro MWh. Die spezifischen Enthalpien werden mittels thermodynamischer Berechnungen ermittelt.

4 Ergebnisse

4.1 Prototypische Umsetzung und Erprobung der Entwicklung Smart Monitoring

Für die Parameteroptimierung zum Anlernen eines Digitalen Zwillings wird ein neues Werkzeug in Form eines eigenständigen Moduls – das Model optimization module (MOM)

– integriert. Dieses beinhaltet die drei Teilaufgaben des definierten Optimierungsprozesses und besteht aus Funktionen und Methoden, die gezielt in das Klassenmodell und in bestehende GUIs eingebettet sowie an geeigneten Stellen mit weiteren Konfigurationsdialogen und Diagrammen zur Ergebnisauswertung erweitert wurden. Durch die Integration sogenannter Wrapper-Funktionen wird ferner sichergestellt, dass die drei Teilprozesse jeweils die ausgewählten Optimierungsalgorithmen verwenden können. Dies ermöglicht eine besonders effiziente Umsetzung, ohne die Algorithmen auf die entsprechenden Anforderungen der jeweiligen Optimierungsaufgabe anpassen zu müssen.
Durch die Verknüpfung von Datenpunkten in der Parametrieroberfläche der Komponenten erfolgt die Zuweisung realer Messdaten im Modell, auf deren Grundlage stationäre Berechnungen durchgeführt werden, siehe Abbildung 3 . Weitere Datenpunkte, die nicht für die Berechnung benötigt werden, werden als Referenzdatenpunkte verknüpft und dienen einerseits der Validierung durch den Vergleich zwischen Messwert und berechnetem Wert und andererseits zur Bildung der Zielfunktion. Die Zielfunktion setzt sich aus allen Referenzdatenpunkten zusammen (z.B. die Summe der quadratischen Abweichungen zwischen Messwert und berechnetem Wert) und wird durch das gewählte Optimierungsverfahren minimiert.
Mittels der vorhandenen Initialisierungsroutine können stationäre Ruhelagen für das Gesamtmodell berechnet werden, die wiederum einen validen Startzustand für die Simulation dynamischer Vorgänge darstellen. Diese Vorgehensweise erspart die Integration weiterer GUIs, sodass alle relevanten Informationen für den gesamten Optimierungsprozess über die Parametrieroberflächen der einzelnen Komponenten bzw. der zentralen Objekt-GUI verknüpft und gesteuert werden können.
Die Auswahl der zu optimierenden Parameter findet ebenso über die Parametrieroberfläche

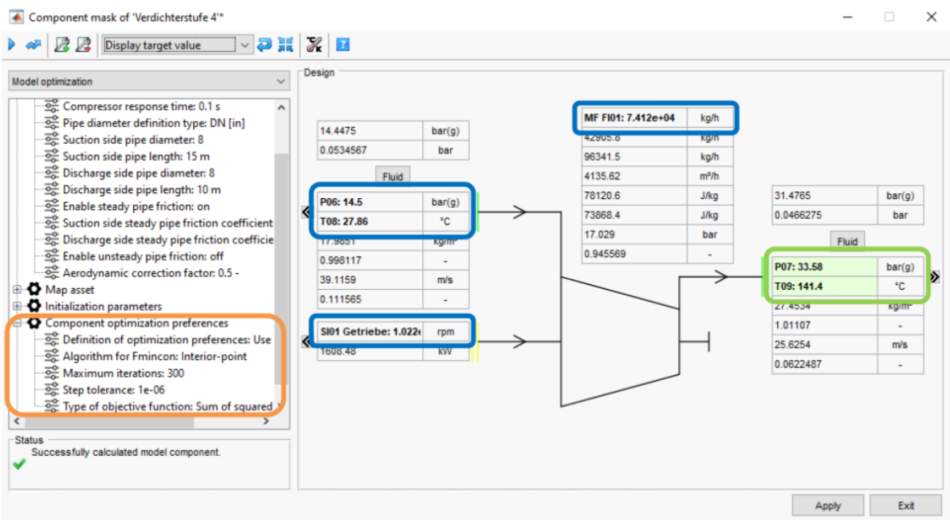


Abb. 3: Parametrieroberfläche eines Verdichters mit verknüpften Datenpunkten zur Berechnung (blau), zur Bildung der Zielfunktion (grün) und konfigurierter Komponentenoptimierung (orange).

der Komponenten statt. Dabei wird zwischen Parametern unterschieden, die ausschließlich Einfluss auf das stationäre oder dynamische Modellverhalten besitzen. Weiterhin können hier die untere und obere Grenze sowie die Schrittweite des zu optimierenden Parameters definiert werden. Die Erprobung der erzielten Entwicklungen fand anhand eines Testprojekts mit einem bestehenden Digitalen Zwilling statt. Hierbei wurden der Datenimport, die Modelloptimierung und Zustandserkennung eines einstufigen Butylverdichters mit zwei parallelen Laufrädern untersucht. Die erprobten Arbeitsschritte und deren Ergebnisse sind im Folgenden aufgelistet.

Erprobte Arbeitsschritte

1. Konfiguration des Datenimports und Erstellung der Zuweisungsliste
2. Import mehrerer Datensätze (CSV-Dateien) und Durchführung der Gleichgewichtserkennung
3. Verknüpfung der Datenpunkte in den Parametrieroberflächen der einzelnen Komponenten
4. Durchführung der Optimierung
   a) Konfiguration, Durchführung und Auswertung der Komponentenoptimierung
   b) Konfiguration, Durchführung und Auswertung der stationären Modelloptimierung
   c) Konfiguration, Durchführung und Auswertung der dynamischen Modelloptimierung
5. Durchführung der Zustandserkennung mit weiteren importierten Datensätzen

Ergebnisse

- Nach einmaliger Konfiguration des Imports sowie der Zuweisungsliste können Daten effizient und konsistent importiert werden. Der Import neuer Datensätze erfolgt bei gleichbleibendem Format automatisiert und ohne weiteren Aufwand.
- Mehrere Testdurchläufe konnten zeigen, dass die entwickelte Vorgehensweise der schrittweisen Modelloptimierung zeitlich schneller und wesentlich handlicher ist als eine Optimierung des Gesamtmodells ohne vorherige Komponentenoptimierung.
- Die Validierung der Zustandserkennung anhand fehlerfreier und fehlerhafter Datensätze hat gezeigt, dass das implementierte Verfahren unter dem Einfluss von Messunsicherheiten und Modellfehlern zuverlässig funktioniert.

- Eine identifizierte Problematik betrifft die Optimierung mit unsicherheitsbehafteten Messdaten, die zum einen für die Modellberechnung und zum anderen als Referenzgröße dienen. Aufgrund der Genauigkeitsanforderungen sind Lösungsansätze zu verfolgen, die insbesondere die Unsicherheit der zur Berechnung genutzten Größen auf die Unsicherheit der Berechnungsergebnisse bzw. der Parameter übertragen.

4.2 Prototypische Umsetzung und Erprobung der Smart Maintenance

Für die Fehlerdiagnose wird ein neues Werkzeug in Form eines eigenständigen Moduls – das Fault Diagnosis Module (FDM) – integriert. Das Modul beinhaltet neben der beschriebenen Fehlerdiagnose eine Methode zur Untersuchung der Isolierbarkeit potenzieller Anlagenfehler. In dieser Methode werden die Auswirkungen potenzieller Fehler auf die verfügbaren Vergleichsmessstellen untersucht. Diese Analyse wird einmalig vor der Fehlerdiagnose ausgeführt. Ermittelt werden die potenziellen Auswirkungen durch eine Berechnung des Modells bei aktiviertem Fehlermodell. Haben mehrere Fehler die gleichen qualitativen Auswirkungen auf die erfassbaren Prozessgrößen, können diese unter Umständen in der folgenden Fehleridentifikation nicht voneinander unterschieden werden. Weiterhin werden so Fehler identifiziert, die keinerlei Auswirkungen auf die verfügbaren Messgrößen besitzen. Diese Fehler können vollständig von der folgenden Fehleridentifikation ausgeschlossen werden, da diese von System nicht detektiert und nicht identifiziert werden können. Folglich erlauben die Ergebnisse dieser Untersuchung eine Aussage über die potenzielle Leistungsfähigkeit der Fehlerdiagnose.
Welche Fehlerszenarien in Betracht zu ziehen sind, ist zuvor vom Anwender festzulegen. Werden beispielsweise nur Einzelfehler betrachtet, wird iterativ jedes Fehlermodell einzeln zugeschaltet und parametriert. Bei der Betrachtung von Doppelfehlern werden immer zwei Fehlermodelle zugleich parametriert. Ist dieser Vorgang abgeschlossen, werden die Ergebnisse in einer Rangfolge aufgelistet. Der oder die Fehlermodi, welche die beste Kompensation der Abweichungen erzielen, sind als wahrscheinlichste Fehlerursachen in Betracht zu ziehen. Da die Fehler durch einstellbare Parameter modelliert sind, kann zudem auf das Ausmaß des identifizierten Fehlers geschlossen werden. Je größer die modellierten Parameter, desto größer das Ausmaß des Fehlers.
Die Erprobung der entwickelten Fehlerdiagnose fand anhand eines Testprojekts mit einem bestehenden Digitalen Zwilling statt. Hierbei wurde die Fehlerdiagnose auf den fehlerhaften Datensatz, an dem zuvor die Zustandserkennung getestet wurde, angewendet. Die erprobten Arbeitsschritte und deren Ergebnisse sind im Folgenden aufgelistet.

Erprobte Arbeitsschritte

1. Konfiguration und Durchführung der Isolierbarkeitsuntersuchung
2. Konfiguration und Durchführung der Fehlerdiagnose

3. Auswertung der durchgeführten Fehlerdiagnose

Ergebnisse

- Durch die Untersuchung der Isolierbarkeit konnten nicht detektierbare Fehler von der Fehlerdiagnose ausgeschlossen und die Berechnungsdauer der Fehlerdiagnose reduziert werden.
- Die Anwendung der Fehlerdiagnose hat gezeigt, dass mithilfe des implementierten Verfahrens detektierte Abweichungen auf eine Fehlerursache zurückgeführt werden können.
- Da die manuelle Auswertung aller Optimierungsergebnisse nach abgeschlossener Fehlerdiagnose sehr aufwendig ist, wird jedes untersuchte Fehlermodell mit einer Bewertungszahl versehen. Diese Bewertungszahl fasst das Fehlerausmaß sowie die maximale Abweichung und die gemittelte Abweichung vom Referenzzustand in einem Bewertungsmaß zwischen 0 und 100% zusammen. Wird für ein Fehlermodell eine verhältnismäßig große Bewertungszahl bestimmt, ist das Fehlermodell als potenzielle Ursache für die detektierten Abweichungen in Betracht zu ziehen.
- Aufgrund der physikalischen Modellierung ist eine einfache Interpretierbarkeit des Fehlerausmaßes gewährleistet.

4.3 Prototypische Umsetzung und Erprobung der Entwicklung Smart Revamp

Anhand eines vorhandenen Digitalen Zwillings wurden die getätigten Entwicklungen zusammengeführt und an einem konkreten Anwendungsbeispiel erprobt. Dieser Digitale Zwilling umfasst ein turbinengetriebenes, mehrstufiges Verdichtersystem ,siehe Abbildung 4, bei welchem nach einer Notabschaltung eine unerlaubte Drehrichtungsumkehr im gesamten Maschinensatz auftrat. Derartige Betriebszustände können zu Beschädigungen an der gesamten mechanischen Konstruktion in Getriebe, Kupplungen, Lager, Laufräder, etc. führen und sind daher zu vermeiden.
Um geeignete Lösungsansätze definieren zu können, muss zunächst die Ursache der Drehrichtungsumkehr identifiziert werden. Diese konnte mittels einer simulationstechnischen Untersuchung auf ein zu klein ausgelegtes Ausblaseventil zurückgeführt werden, wodurch die im System befindliche Menge nicht schnell genug abgeführt werden kann und es zu einer Umkehr der Strömungsrichtung kommt, die zum Rückwärtsdrehen der Laufräder führt. Eine geeignete Gegenmaßnahme ist das Platzieren eines oder mehrerer zusätzlicher Ausblaseventile, was eine schnellere Systementlastung erlaubt. Die genaue Position und Auslegung (Parametrierung) sollen nun durch das Optimierungsverfahren bestimmt werden. Weiterhin müssen entsprechende KPIs definiert werden, die die Zielfunktion des Optimie-

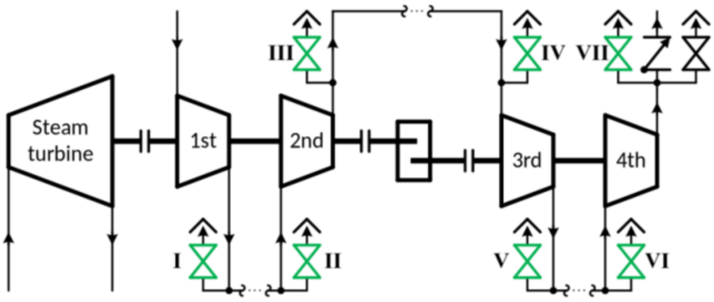


Abb. 4: Schematischer Aufbau des Verdichtersystems mit möglichen Positionen zusätzlicher Ausblaseventile.

rungsverfahrens bilden. Hierzu bietet sich zunächst die Definition einer Soll-Drehzahlkurve $f_{Ziel}(t)$ an, die als Zielfunktion zum Vergleich mit dem Simulationsergebnis $f_{Sim}(t)$ dient:

$$KPI_1 = \int_{t_{Start}}^{t_{Ende}} (f_{Sim}(t) - f_{Ziel}(t))^2 dt. \tag{2}$$

Da während einer Notabschaltung weitere unerwünschte Effekte bei der Verdichterpumpe wie starkes Oszillieren des Ausgangsdrucks mit verbundener Strömungsumkehr auftreten können, werden ferner positive Gradienten der Ausgangsdrücke $p_{aus,i}$ detektiert und in folgender Form aufsummiert:

$$KPI_2 = \frac{1}{n} \sum_{i=1}^{n} \int_{t_{Start}}^{t_{Ende}} max\left(0, \frac{p_{Aus,i}(t)}{dt}\right) dt \tag{3}$$

Nachdem mögliche Optimierungsansätze festgelegt und die relevanten KPIs definiert wurden, ist der gewählte Optimierungsalgorithmus Estimation of Distribution Algorithms zu konfigurieren. Dies betrifft einerseits allgemeine Einstellungen, wie die maximale Anzahl von Iterationen, und andererseits spezifische Einstellungen, wie die Wahl der Lernrate oder Mutationsstärke. Anschließend kann der Algorithmus gestartet und die Ergebnisse ausgewertet werden. Die oben beschriebenen Arbeitsschritte und deren Ergebnisse sind im Folgenden nochmals zusammengefasst.

Erprobte Arbeitsschritte

1. Definition und Zusammenfassung relevanter KPIs
2. Festlegung möglicher Optimierungsansätze und Vorgabe der notwendigen Modellierung
3. Konfiguration des Algorithmus

4. Durchführung der Designoptimierung
5. Auswertung der Ergebnisse

Ergebnisse

- Die KPI-Definition ist hochgradig problemspezifisch und erfordert ein hohes Maß an Kenntnissen aus dem Anwendungsbereich.
- Die Konfiguration des Algorithmus erfordert spezifische Kenntnisse zur Theorie des Algorithmus. Hier sind weitere Maßnahmen zu ergreifen, die die Notwendigkeit derartiger Kenntnisse minimieren.
- Die Arbeitsschritte zur Durchführung der Optimierung sowie der Ergebnisauswertung sind als iterativer Prozess anzusehen. Die Optimierungen sollten mehrfach bei gleichbleibender Konfiguration durchgeführt und anschließend ausgewertet werden. Entsprechend der Auswertung können gezielte Änderungen am Modellierungsansatz bzw. an der Konfiguration des Optimierungsalgorithmus vorgenommen werden.
- Für eine weitgehend automatisierte und wiederholte Durchführung der Designoptimierung bietet sich wiederum die Implementierung einer entsprechenden Job-Klasse an, die zur Auswahl und Konfiguration des Optimierungsverfahrens sowie zur Auswahl und Wichtung geeigneter KPIs dient.

5 Fazit

Der Bereich **Smart Monitoring** mit der Einbettung der Verfahren zur Parameteroptimierung sowie der automatisierte Datenimport durch die entwickelte Pipeline ermöglichen das Erreichen einer höheren Engineering-Qualität bei gleichzeitiger Reduzierung der Engineering-Dauer zur Erstellung und Anpassung eines Digitalen Zwillings.
Das implementierte Verfahren zur automatisierten Fehlerdiagnose im **Smart Maintenance**, welche ohne tiefere A-priori-Kenntnisse über den spezifischen Prozess erfolgreich Fehlerquellen lokalisieren und quantifizieren kann und reduziert den Engineering-Aufwand gegenüber einer manuellen Fehlersuche erheblich.
Die entwickelte Methodik **Smart Revamp** wurde prototypisch erfolgreich angewendet, es müssen aber noch weitere metaheuristische Optimierungsverfahren untersucht werden, um die erzielte Lösungsqualität sowie die benötigte Laufzeit bewerten zu können.

Förderung:

Dieses Kooperationsprojekt wurde vom BMWi mit der Nummer ZF4331002GR9 gefördert.

Literatur

[Be99] Beckers, M.: Topology optimization using a dual method with discrete variables. Structural optimization 17/1, S. 14–24, 1999.

[BS93] Bäck, T.; Schwefel, H.-P.: An overview of evolutionary algorithms for parameter optimization. Evolutionary computation 1/1, S. 1–23, 1993.

[Di86] Ding, Y.: Shape optimization of structures: a literature survey. Computers & Structures 24/6, S. 985–1004, 1986.

[EO01] Eschenauer, H. A.; Olhoff, N.: Topology optimization of continuum structures: a review. Appl. Mech. Rev. 54/4, S. 331–390, 2001.

[Fl03] Flockermann, D.: Einsatz von Optimierungsverfahren zur Bestimmung der Parameter nichtlinearer Ersatzschaltbilder von elektrischen Maschinen, Diss., Dortmund, Univ., Diss., 2003, 2003.

[LZ17] Li, Z.; Zheng, X.: Review of design optimization methods for turbomachinery aerodynamics. Progress in Aerospace Sciences 93/, S. 1–23, 2017.

[Mo14] Moroz, L.; Frolov, B.; Burlaka, M.; Guriev, O.: Turbomachinery flowpath design and performance analysis for supercritical CO2. In: Turbo Expo: Power for Land, Sea, and Air. Bd. 45615, American Society of Mechanical Engineers, V02BT45A004, 2014.

[MPW11] Marzat, J.; Piet-Lahanier, H.; Walter, E.: Min-max hyperparameter tuning, with application to fault detection. IFAC Proceedings Volumes 44/1, S. 12904–12909, 2011.

[MS03] Michalewicz, Z.; Schmidt, M.: Evolutionary algorithms and constrained optimization. In: Evolutionary Optimization. Springer, S. 57–86, 2003.

[MS13] Maute, K.; Sigmund, O.: Topology optimization approaches: A comparative review. Structural and Multidisciplinary Optimization 6/, 2013.

[PA06] Pedersen, C. B.; Allinger, P.: Industrial implementation and applications of topology optimization and future needs. In: IUTAM Symposium on Topological Design Optimization of Structures, Machines and Materials. Springer, S. 229–238, 2006.

[RK92] Rajeev, S.; Krishnamoorthy, C.: Discrete optimization of structures using genetic algorithms. Journal of structural engineering 118/5, S. 1233–1250, 1992.

[Sc14] Schaeffler, S.: Globale Optimierung: Ein informationstheoretischer Zugang. Springer-Verlag, 2014.

[Sh15] Shahriari, B.; Swersky, K.; Wang, Z.; Adams, R. P.; De Freitas, N.: Taking the human out of the loop: A review of Bayesian optimization. Proceedings of the IEEE 104/1, S. 148–175, 2015.

[UU12] Ulbrich, M.; Ulbrich, S.: Restringierte Optimierung. In: Nichtlineare Optimierung. Springer, S. 89–142, 2012.

[WS07] Wang, G. G.; Shan, S.: Review of metamodeling techniques in support of engineering design optimization./, 2007.

J. Michael, J. Pfeiffer, A. Wortmann (Hrsg.): Modellierung 2022 Satellite Events,
172 Digital Library, Gesellschaft für Informatik e.V.

RDM4MOD: Working Workshop on Research Data Management in Modelling in Computer Science - Summary

Michael Goedicke[1], Ulrike Lucke[2]

Abstract: This workshop reflects on science and scientific results in different ways than usual: it looks at the meta level what are the items of research in computer science in the context of models and modelling. Since this needs to be a community effort to agree on the classification and related properties this is a "working" workshop to discuss the challenges in this area to support the research data management in modelling in computer science and related infrastructure for implementing the so-called FAIR[3] principles. The main challenge is, to cover the research data management requirements of the modelling activities in computer science in their broad variety. The aim is to sketch a way how to agree on a comprehensible metadata standard and how to evolve it to include also future research directions. There are certain pieces of existing work which we discuss as a starting point in the area of UML-based modelling and artifact evaluation / assessment. The goal is to draft a manifesto for RDM for modelling activities in computer science.

Keywords: Research Data Management, Modelling in Computer Science, FAIR Principles, Nationale Forschungsdaten Infrastruktur

1 Preface

Modelling is an important approach to tackle research problems in computer science. It helps to lower the bar in terms of complexity and to support validation – in some areas also formal verification – to substantiate scientific claims and (published) results. However, to support the reproducibility of such results the basis in form of the used model(s) needs to be available as well. This type of research has also become much more empirical in nature: e.g. for assessing the usefulness of representation schemes and modelling concepts large repositories of models and experiments involving human developers play an important role.

In order to support such research based on empirical data and related publications the so-called FAIR-principles (Findable, Accessible, Interoperable, Reusable) need to be supported. Especially the processes to ***find*** and ***access*** the research data needs a community-wide consensus how to define and use metadata. This is a challenge for the still fast evolving discipline of computer science. The workshop builds upon existing experiences from related efforts. An invited contribution on the challenges and

[1] paluno/University of Duisburg-Essen, Faculty of Economics and Business Administration, Gerlingstraße 16, Essen, 45117, michael.goedicke@paluno.uni-due.de,
[2] Complex Multimedia Architectures/University of Potsdam, Department of Computer Science, An der Bahn 2, Potsdam, 14476, ulrike.lucke@uni-potsdam.de
[3] https://force11.org/info/the-fair-data-principles/

architectures for empirical research in software architecture brings in the experience of storing a substantial number of model descriptions. A further contribution reflects on assessing research data artifacts. Based on these impulses a discussion will be held to sketch a comprehensive outline of research data management for modelling in computer science in form of a manifesto.

This will initiate the community process to collaboratively define related meta data standards and additional aspects to support the aforementioned FAIR principles. This support will help to coordinate the many efforts not only at the national level.

We are grateful for the support of the team of this workshop to kick off this process which starts an important improvement of research processes in research based on modelling in computer science. The related effort and developments will be documented at the website / portal of NFDIxCS (www.nfdixcs.org).

June 2022, Essen / Potsdam

Michael Goedicke and Ulrike Lucke

CoChairs of RDM4MOD

J. Michael, J. Pfeiffer, A. Wortmann (Hrsg.): Modellierung 2022 Satellite Events,
Digital Library, Gesellschaft für Informatik e.V. 174

3. Workshop zur Modellierung in der Hochschullehre

Meike Ullrich[1], Peter Fettke[2], Peter Pfeiffer[2], Selina Schüler[1], Michael Striewe[3]

Abstract: Der Workshop befasst sich mit dem Thema Modellierung – nicht wie üblich aus der Perspektive der Modellierung zum Einsatz in Industrie und Unternehmen, sondern aus dem Blickwinkel der Hochschullehre. Somit soll die Frage nach passenden Lernzielen, Lerninhalten und innovativen Unterrichtsmethoden für die Modellierung im Vordergrund stehen, ebenso wie die Frage nach geeigneten Prüfungsformaten und Bewertungsverfahren für die von Studierenden erstellten Modelle.

Keywords: Hochschullehre; Modellierung

Vorwort

In der Praxis findet die Modellierung ihren Einsatz beispielsweise in der Softwareentwicklung, dem Datenbankentwurf oder bei der Geschäftsprozessmodellierung. Daher ist sie curricularer Bestandteil zahlreicher informatiknaher Studiengänge. Bislang wird die Modellierung in der Hochschullehre überwiegend in klassischen Frontalveranstaltungen, wie z.B. Vorlesungen unterrichtet. Jedoch wird aus der Wissenschaftsforschung und Hochschuldidaktik in den letzten Jahrzehnten verstärkt ein Perspektivenwechsel von einer dozenten- zu einer studierendenzentrierten sowie einer kompetenzorientierten Lehre hin gefordert, die eine aktive Rolle der Studierenden und praktische Anwendung theoretischer Inhalte vorsieht. Ebenso gewinnt nicht zuletzt durch den Bologna-Prozess die Frage nach der Qualität von Hochschulabschlüssen und damit verbunden die faire, objektive und lernzielgerichtete Überprüfung studentischer Leistungen zunehmend an Bedeutung. Mit diesen Herausforderungen muss sich auch die Hochschullehre rund um das Thema Modellierung auseinandersetzen.

Ziel des Workshops ist es, an der Modellierung in der Hochschullehre beteiligte und interessierte Personen zusammenzubringen. Insbesondere werden unterschiedliche Perspektiven von Studierenden, Dozenten und Tutoren gesucht. Darüber hinaus ist auch die Meinung von Personen aus der Praxis gefragt, die wertvolle Hinweise zu wünschenswerten Lernzielen und zur Praxisrelevanz des üblicherweise zur Modellierung gelehrten Stoffes

[1] Karlsruher Institut für Technologie (KIT), Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB), Kaiserstr. 89, 76133 Karlsruhe, vorname.nachname@kit.edu
[2] Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI) und Universität des Saarlandes, Institut für Wirtschaftsinformatik (IWi), Campus D3 2, 66123 Saarbrücken, vorname.nachname@dfki.de
[3] Universität Duisburg-Essen, paluno - The Ruhr Institute for Software Technology, Gerlingstr. 16, 45127 Essen vorname.nachname@uni-due.de

liefern können. Daher sind vor allen Dingen auch Beiträge ausgewählt worden, die Erfahrungswerte aus der (Hochschul-)Praxis liefern oder Ideen und Anregungen für zukünftige Entwicklungen vorstellen. Insgesamt sollen mit dem Workshop Austausch und Diskussion über das Thema Modellierung in der Hochschullehre angeregt werden, um zukünftigen Weiterentwicklungsbedarf und relevante Forschungsschwerpunkte zu identifizieren.

Zur Präsentation und Diskussion wurden fünf Langbeiträge angenommen, die verschiedene Aspekte der Hochschullehre abdecken. Es geht um Werkzeuge zur automatisierten Generierung von Klausur- oder Übungsaufgaben für die Modellierung sowie um Konzepte für die Gestaltung von Kursen zur Modellierung. Zudem konnte Prof. Dr. Wolfgang Reisig (Humboldt-Universität zu Berlin) für einen Keynote-Vortrag gewonnen werden.

Wir danken Herrn Prof. Dr. Wolfgang Reisig für die Bereitschaft, den Keynote-Vortrag zu übernehmen, sowie allen Einreichenden für die sorgfältige Aufbereitung ihrer Arbeitsergebnisse. Ebenso danken wir den Mitgliedern des Programmkomitees für die Mitwirkung bei der Begutachtung und Auswahl der Beiträge. Dem Organisationsteam der Tagung und der Workshops danken wir für die Unterstützung bei der Ausrichtung der Veranstaltung.

Progammkomitee: Michael Fellmann (Universität Rostock) Constantin Houy (Universität des Saarlandes), Judith Michael (RWTH Aachen), Andreas Oberweis (Karlsruher Institut für Technologie), Jana-Rebecca Rehse (Universität Mannheim), Kristina Rosenthal (FernUniversität Hagen), Andreas Schoknecht (avono AG), Janis Voigtländer (Universität Duisburg-Essen)

J. Michael, J. Pfeiffer, A. Wortmann (Hrsg.): Modellierung 2022 Satellite Events,
Digital Library, Gesellschaft für Informatik e.V. 176

Modellierung in der Informatik-Hochschullehre

ein Trauerspiel

Wolfgang Reisig[1]

Der Modellierer:

Habe nun, ach!
Philosophie,
Juristerei und Medizin,
Und leider auch Informatik
Durchaus studiert, mit heißem Bemühn.
Da steh' ich nun, ich armer Tor,
Und bin so klug als wie zuvor!
Heiße Magister, heiße Doktor gar,
Und ziehe schon an die zehn Jahr'
Herauf, herab und quer und krumm
Meine Schüler an der Nase herum –

Der Programmierer:

Verzeiht! es ist ein groß Ergetzen,
Sich in den Geist der Zeiten zu versetzen;
Zu schauen, wie vor uns ein weiser Mann gedacht,
Und wie wir's dann zuletzt so herrlich weit gebracht.

frei nach J.W. v. Goethe

Vorbemerkung

Es ist das Risiko des Veranstalters und das Privileg des Sprechers eines eingeladenen Vortrags, dass kein Gutachter „helfend" eingreifen kann. Ich kann also beliebig pointiert formulieren, und das will ich nutzen.

1 Die Lage

Jeder Schreiner macht einen Plan, ein Modell, bevor er einen Stuhl baut.

[1] Humboldt-Universität zu Berlin, Berlin, Germany, reisig@informatik.hu-berlin.de

Jeder Elektriker macht einen Schaltplan.

Jeder Architekt plant ein Haus, bevor er es baut. Modelliert es, visualisiert es, Studenten bauen 3D-Modelle, manche Architekten kümmern sich gar nicht ums Bauen, sondern machen nur Pläne.

Jeder Ingenieur-Student lernt erst mal Differential-/Integralrechnung und Differentialgleichungen als Modellierungs-Handwerkszeug.

Kurz: Jeder Handwerker und jeder Ingenieur-Student lernt erst mal modellieren.

Ein Informatik-Student lernt erst mal programmieren.

Informatik-Kurrikulum HU Berlin, aber auch sonst wo:

1. Semester: Hacken mit Java (andernorts, edlere Variante: Hacken mit Python);
2. Semester: Algorithmen und Datenstrukturen (man beachte die Reihenfolge: 1. Semester: Hacken. Dann im 2. Semester ein wenig Systematik);
3. Semester: Software Engineering. Noch mehr Hacken und Fummeln.

2 Ein Beispiel

Meine Erfahrung als Beisitzer bei „endgültig durchgefallen“-Prüfungen: Prüfling soll *Sortieren* erklären. Um es einfach zu halten, braucht er kein Java-Code hinschreiben, sondern kann Sortieren „gern auch intuitiv“ schildern.

Wenn er es gut macht, malt er eine Sequenz aus Kästchen, schreibt was rein, skizziert mit Pfeilen, welcher Eintrag wann wohin wandert. Was ist das, was er da macht? Eine umgangssprachliche Skizze? Ein Modell des Algorithmus? Der Algorithmus selbst? Kann man das nicht systematischer machen?

Fragen wir das HPI: Was ist *Bubblesort*? Die Website zu Bubblesort enthält [HP22]:

- ein Beispiel: Eine Sequenz von acht Spielkarten wird in sechs Schritten sortiert.
- ein Pseudocode-Programm:

```
function BubbleSort(A) {
    newRound := TRUE;
    for i := n-1 downto 1 do {
      if newRound then {
         newRound := FALSE;
         for j := 0 to i-1 do {
             if A[j] > A[j+1] then {
```

```
                    temp := A[j];
                    A[j] := A[j+1];
                    A[j+1] := temp;
                    newRound := TRUE;
                 }
             }
          }
          else break
        }
    }
```

„Bubblesort ist die Verschachtelung von zwei for-Schleifen“

- Die Idee des Nachweises der korrekten Sortierung durch „Bubblesort“ („Korrektheitsbeweis“) wird an der Tafel erläutert:
  „Die äußere for-Schleife macht . . . “ ; „Elemente sind falsch rum“; „Vergleich stimmt / stimmt nicht“

Warum stellt man so was ins Netz? Was ist die Rolle des Pseudocode-Programms? Warum nicht gleich Java? Ist das Pseudocode-Programm eine Art Modell des Algorithmus? Kann man nicht ohne eine Programmier-Syntax den Bubblesort-Algorithmus hinschreiben? Warum redet man nicht präzise? Was wäre denn hier „präzises Reden“?

Vorschlag: Vernünftig über Bubblesort reden

Gegeben seien

1. einige Mengen:
   Eine Menge A mit einer Ordnung, $\leq$.
   Daraus abgeleitet:
   Tupel, geordnetes Tupel, Liste über A, geordnete Liste.

2. einige Funktionen:
   - für Tupel (a, b):

     „Tupel (a, b) *ordnen”*:
     ord : *Tupel* $\rightarrow$ *Tupel*
     ord$(a, b) := (a, b)$, falls (a, b) geordnet, sonst (b, a).

   - für Listen $(a_0, \ldots, a_n)$:

     „(a_{i-1}, a_i) *ordnen“*:

i-Schritt: *Listen* $\longrightarrow$ *Listen*
i-Schritt$(a_0, \ldots, a_{i-1}, a_i, \ldots, a_n) := (a_0, \ldots, ord(a_{i-1}, a_i), \ldots a_n)$

Wir schreiben „*i-Schritt*“ statt „*i-Schritt*$(a_0, \ldots, a_n)$“, wenn $(a_0, \ldots, a_n)$ im Kontext klar ist.

„Das größte Element ganz nach rechts spülen“:
max : *Listen* $\longrightarrow$ *Listen*
$max(a_0, \ldots, a_n) :=$ *1-Schritt; 2-Schritt; ... ; n-Schritt.*

„a_i an die richtigen Stelle setzen, wenn $a_{i+1}, \ldots, a_n$ schon an der richtigen Stelle stehen“:
i-ordnen : *Listen* $\longrightarrow$ *Listen*
i-ordnen$(a_0, \ldots, a_{i-1}, a_i, \ldots, a_n) := (max(a_0, \ldots, a_i), a_{i+1}, \ldots, a_n)$

Wir schreiben „*i-ordnen*“ statt „*i-ordnen*$(a_0, \ldots, a_n)$“, wenn $(a_0, \ldots, a_n)$ im Kontext klar ist.

„$(a_0, \ldots, a_n)$ ordnen“:
geordnet: *Listen* $\longrightarrow$ *Listen*
geordnet$(a_0, \ldots, a_n) :=$ *n-ordnen*; *n*-1-*ordnen*; ... ; 1-*ordnen*.

Das ist ein operationelles, formales Modell des Bubblesort-Algorithmus. Die intuitive Idee des Algorithmus wird präzise, mit den passenden formalen Ausdrucksmitteln, formuliert. Die Funktion *max* schildert formal und zugleich anschaulich, wie ein „*bubble* aufsteigt“. Die Funktion *i-ordnen* schildert, wie alle „*bubbles* aufsteigen“. Das versteht auch ein dummer Student. Ein Korrektheitsbeweis für den Algorithmus gehört auf diese Ebene. Von hier aus, wenn's denn sein muss, kann man Programmcode generieren.

Es ist hier nicht der Ort, allgemeine Prinzipien der Formulierung von Algorithmen zu diskutieren. Allerdings zeigt dieses Beispiel: An den vielen Studienabbrechern haben wir Schuld! Wie [Pf94] zur Informatik-Ausbildung sagt: *Fertigkeiten ersetzen Wissen, Machen erdrückt das Verstehen. Typischerweise lernen die meisten, Programme zu schreiben, ohne jemals welche gelesen, geschweige verstanden zu haben.* Das ist kein Wunder: ein Programm soll ja von einem Rechner verstanden werden, nicht unbedingt von Menschen. Für Menschen braucht man stattdessen Modelle.

3 Historische Entwicklung

In den 1950/60er Jahren waren zentrale Themen der Informatik das Suchen und Sortieren von Daten auf Magnetbändern, die numerische Mathematik und Rechnerarchitekturen (Betriebssysteme, Compiler). Dafür wurden immer neue, vermeintlich bequemere Programmiersprachen vorgeschlagen. Zwei wichtige Publikationen: *Assigning Meanings to*

Programs [Fl67]. Immerhin, nach 10 Jahren Programmieren: ein Programm soll etwas Eindeutiges bedeuten! Ein Jahr später: *The Art of Computer Programming* [Kn68]. Ein Programmierer ist ein Künstler, eine Art Genie. Knuth verwendet eine „synthetische Programmiersprache". Was soll das? Fehlt ihm bei den damals gängigen Programmiersprachen die Präzision? Glaubt er, seine Sprache eigne sich besonders gut zur Formulierung der Algorithmen? Programmieren war damals schon eine fehleranfällige Angelegenheit, und es wurde eine „Software-Krise" ausgerufen. Zur Lösung wurden weitere Programmiersprachen vorgeschlagen, darunter monströse wie ALGOL 68 und ADA (Wie ein Zyniker bemerkt: „Manche der früheren Sprachen waren ein deutlicher Fortschritt gegenüber einigen ihrer Nachfolger . . . "). Der Grundfehler all dieser Bemühungen war und ist der Versuch, mit Programmiersprachen zu modellieren. Die ganz unbegründete Vorstellung, ein Algorithmus soll von seiner Implementierung her verstanden werden, und nicht zunächst einmal von seinem beabsichtigten Sinn und Zweck her. Dijkstra hat diese Vorstellung befördert mit seinem wiederholt vorgetragenen Vorschlag, zwischen der formalen und der Anwender-Sicht auf ein System eine gedankliche Mauer zu errichten. Seine Begründung: Das „correctness problem" des Informatikers verlange ganz andere Herangehensweisen als das „pleasantness problem" des Anwenders [Di89].

Heute sind typische zentrale Themen der Informatik beispielsweise: Ware bestellen und Rechnung bezahlen, Werbefläche im Internet versteigern, autonome Fahrzeuge steuern. Allgemeiner formuliert: Ein gegebenes realweltliches Problem so präsentieren, dass Rechner Teile der Lösung übernehmen, zusammen mit Menschen oder mechanischen Maschinen. Beispiel: Ausschreibung und Besetzung einer Stelle in einer Behörde. Im konkreten Ablauf der Besetzung einer Stelle greift ein Algorithmus auf rechnergestützte Datenbanken zu, kommuniziert mit andern Verwaltungen, kontrolliert die Einhaltung vorgegebener Regeln, überwacht Fristen, etc. Zu dem Algorithmus gehören auch Menschen, die unter den Bewerbern den geeignetsten Kandidaten wählen und Urkunden unterschreiben. Solche Algorithmen müssen modelliert werden mit Modellen, die Dijkstra's Mauer geradezu einebnen.

4 Was hat die Modellierungs-Community zu bieten?

Vorschläge der Softwaretechnik: *Abstract State Machines (ASMs) / Actor model / Alloy / ANSI/ISO C Specification Language (ACSL) / Autonomic System Specification Language (ASSL) / B-Method / CADP / Common Algebraic Specification Language (CASL) / Esterel / FOCUS / Java Modeling Language (JML) / Knowledge Based Software Assistant (KBSA) / Lustre / mCRL2 / MSC-LSC / Perfect Developer / Petri nets / Predicative programming / Process calculi: CSP, LOTOS, π-calculus / RAISE / Rebeca Modeling Language / SPARK Ada / Specification and Description Language (SDL) / Statecharts / TLA+ / USL / VDM: VDM-SL, VDM++ / Z notation*.

Vorschläge der Wirtschaftsinformatik: *ADONIS / ARIS / BPMN / EPK / MEMO / St. Gallen Approach / UML-Varianten*.

Einige dieser Modelle wurden und werden in großen Softwareprojekten verwendet, in der Wirtschaftsinformatik mehr als in der allgemeinen Softwaretechnik, weil in der Wirtschaftsinformatik der Bedarf offensichtlicher ist: Wirtschaftsinformatik gestaltet nicht nur Software, sondern auch Prozesse, Organisationen und Geschäftsmodelle.

Ich selbst habe viele Jahre lang eine Vorlesung für Bachelors mit einem Strauß verschiedener Methoden bestückt: ASM, BPMN, CASL, FOCUS, MSC&LSC, Petrinetze, Prozessalgebren, Statecharts, TLA, Z. Die Resonanz war mittelmäßig. Immerhin sind jedes mal ein paar Studenten für Bachelor-/Masterarbeiten und Promotionen dabei geblieben.

Jeder bisher genannte Vorschlag hat Schwächen. Es gibt keine einheitliche konzeptionelle, theorie-basierte und allgemein akzeptierte Grundlage. Indem die sog. „Theoretische Informatik“ sich auf die Manipulation von Zeichenketten beschränkt, trägt sie hier nichts bei. Damit gibt es auch keine akzeptierte Systematik für Modellierungs-Frameworks und Modellierungs-Infrastrukturen, die man in der Lehre verwenden oder darstellen könnte.

Vielleicht könne man sich auf einige Anforderungen für eine solche Systematik einigen, beispielsweise:

- Ein System soll zunächst aus Sicht der Anwender modelliert werden, nicht aus Sicht der Implementierbarkeit.
- Ein Modellierer überführt informelle, intuitive Ideen zu einem gegebenen oder intendierten System in ein formales Modell.
- Ein Modell beschreibt lebensweltliche, organisatorische, von Menschen ausgeübte, von Maschinen durchgeführte, sowie digitale Prozesse in einer einheitlichen Weise.
- Die Modellierung von Systemen skaliert, ist also auch zur Beschreibung umfangreicher Systeme geeignet.

5 Schluss: Thesen zur Modellierung in der Informatik-Lehre

- Um Informatik an einer wissenschaftlichen Hochschule vernünftig zu lehren, müsste man Informatik erst mal vernünftig systematisch als Wissenschaft formulieren. Dann ergibt sich die zentrale Rolle der Modellierung von selbst [Re20]. Modellierung in der derzeitigen Informatik-Lehre kann man nicht verbessern. Man muss sie neu aufsetzen, zusammen mit der Gestaltung einer Wissenschaft der Informatik.
- Wer nur Zeichenketten manipulieren will, meint oft, er brauche keine Modelle. Allerdings meint [Pf94]: *Es kann keine Rede davon sein, dass sich aus der Technik des Formalen die Grenzen der informatischen Modellierung verstehen lassen.* Die Vulgär-Version der Church'schen These: „Was man mit einem Computer machen kann, das kann man auch mit einer Turingmaschine machen“ ist offensichtlich blanker Unsinn, wird aber von den Studenten so verstanden.

- Wir haben kein Erkenntnis-Problem, sondern ein Akzeptanz-Problem: an sich ahnen viele Kollegen, dass man Konzepte, Ideen, Algorithmen besser mit passenden Notationen unterrichten sollte, also mit passenden Modellen. Aber üblich sind nun mal Programm-Notationen.
- Der Informatik geht es zu gut: Mit Hacken kann man herrlich Geld verdienen. Warum soll man es da ordentlich machen (also modellieren)? „Do kannsch au em Ochs ins Horn pfetze!“

Danksagung

Peter Fettke hat mich zu einigen Änderungen in einer früheren Fassungen dieses Textes angeregt. Ich danke ihm.

Literaturverzeichnis

[Di89] Dijkstra, E. W.: Reply to comments. Commun. ACM, 32(12):1414, 1989.

[Fl67] Floyd, Robert W.: Assigning Meanings to Programs. Proceedings of Symposium on Applied Mathematics, 19:19–32, 1967.

[HP22] HPI: `https://hpi.de/friedrich/teaching/units/einfache-sortierverfahren.html`, 2022.

[Kn68] Knuth, Donald E.: The Art of Computer Programming. Addison-Wesley, 1968.

[Pf94] Pflüger, J.: Informatik auf der Mauer. Informatik-Spektrum, 17:251–257, 1994.

[Re20] Reisig, Wolfgang: Informatics as a Science. Enterprise Modelling and Information Systems Architecture – International Journal of Conceptual Modeling, 15(6):1–13, 2020.

Modellierung von Geschäftsprozessen mit BPMN im Studiengang Bachelor Wirtschaftsinformatik

Ein Inverted-Classroom-Konzept mit offenen, digitalen Medien

Vera G. Meister [1]

Abstract: Der Beitrag präsentiert einen Kompaktkurs zur Geschäftsprozessmodellierung mit BPMN (Business Process Model and Notation), der primär für den Einsatz in der akademischen Lehre im Studiengang Bachelor Wirtschaftsinformatik entwickelt wurde. Der Kompaktkurs folgt einem Inverted-Classroom-Konzept. Gestaltungselemente dafür sind eine Reihe offener, digitaler Medien: Videovorlesungen, Seminaraufgaben, Selbsttests, Modellierungsübungen und ein Modellierungsprojekt.

Keywords: Geschäftsprozessmodellierung, BPMN 2.0, Camunda Modeler, Fehlermuster, gute Modellierungspraxis, BPMN-Informationsparadigma.

1 Einleitung

Zum klassischen Kompetenzprofil der Wirtschaftsinformatik gehört die Modellierung von Geschäftsprozessen. Dabei ist der weltweite Standard BPMN [IS13] die am weitesten verbreitete Notation für die Geschäftsprozessmodellierung. Die zunehmende Digitalisierung in Wirtschaft, Gesellschaft und Verwaltung verstärkt diesen Trend.

Zahlreiche wissenschaftliche Studien (vgl. z. B. [Ro13]) zeigen, dass BPMN-Modelle für Personen mit Fachexpertise jedoch ohne Modellierungsexpertise gut verständlich sind. Damit sind sie eine hervorragende Kommunikationsbasis in Digitalisierungsprojekten, sofern sie bestehende oder angestrebte Geschäftsprozesse unmissverständlich und klar abbilden. Die dafür notwendige Kompetenz zu erwerben, ist jedoch eine nicht zu unterschätzende Herausforderung. Davon zeugt eine Vielzahl syntaktischer und ablauflogischer Fehler, die sich in Modellsammlungen in der Wirtschaft, im öffentlichen Sektor und auch in der Wissenschaft finden lassen (vgl. z. B. [Ha20]).

Es muss daher Ziel einer akademischen Lehrveranstaltung sein, ein tiefes Verständnis nicht nur für die reine Syntax, sondern auch für das zugrundeliegende informatorische Konzept, für gute Modellierungspraxis und trügerische Fehlermuster, für methodische

[1] Technische Hochschule Brandenburg (THB), Fachbereich Wirtschaft, Magdeburger Straße 50, 14770 Brandenburg an der Havel, vera.meister@th-brandenburg.de, https://orcid.org/0000-0002-2780-0222

Herangehensweisen und effiziente Werkzeugnutzung zu entwickeln. Der hier vorgestellte Kompaktkurs widmet sich genau diesen Zielen.

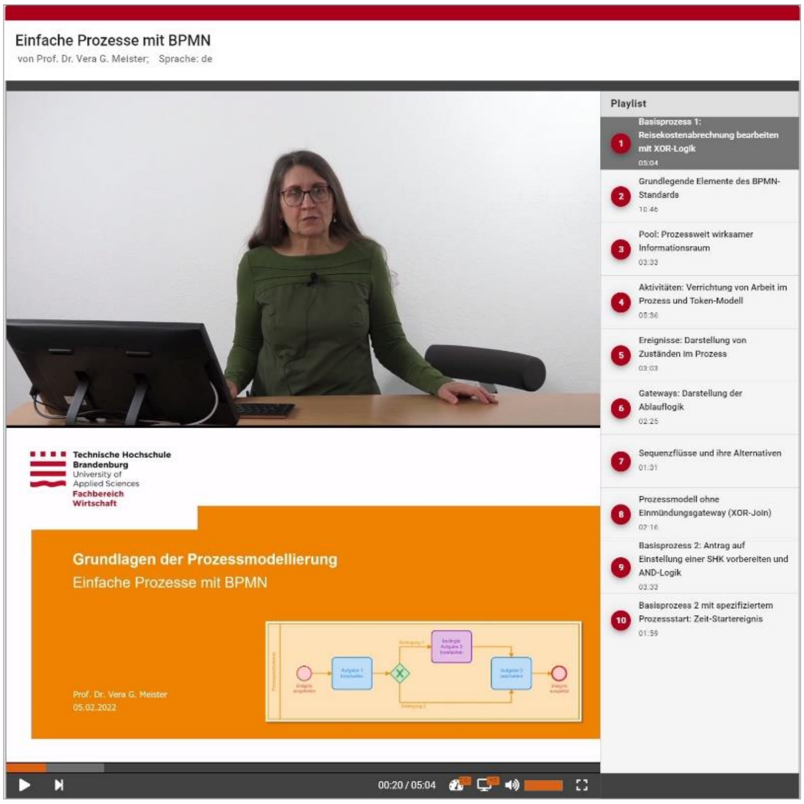


Abb. 1: Videovorlesung auf der eLectures@THB-Plattform

2 Aufbau des Kurses

Der Kompaktkurs BPMN ist substanzieller Teil eines Moduls im Bachelorstudiengang Wirtschaftsinformatik. Neben BPMN-spezifischen Kompetenzen werden auch allgemeine Themen des Geschäftsprozessmanagements und Modellierungsansätze für spezifische Sichten vermittelt. Durch das Inverted-Classroom-Format (vgl. z. B. [ZH17]) gelingt es, die Präsenzzeit für direkte Interaktionen und Kommunikation zwischen und mit den Studierenden zu nutzen. Wiederkehrende und aufeinander aufbauende Feedback-Schleifen spiegeln das Vorgehen in typischen agilen Modellierungsprojekten und erlauben zudem ein tieferes Verständnis für die Möglichkeiten und Grenzen der Notation.

2.1 Videovorlesungen

Die fünf Videovorlesungen im Kompaktkurs folgen dem grundlegenden didaktischen Prinzip vom Einfachen zum Komplizierten. Dabei erfolgt die Vermittlung durchgängig anhand praktischer Beispiele, die überwiegend der Verwaltungspraxis entstammen und im Laufe des Kurses weiterentwickelt werden. Die verwendeten Prozessmodelle wurden mit dem Camunda Modeler erstellt und als hochauflösende Grafiken eingebunden.

Jede der Vorlesungen besteht aus zehn kurzen Sequenzen von 2-10 Minuten. Sie resultieren zu einer Gesamtspieldauer von 40-55 Minuten je Vorlesung. Der Zugang erfolgt über die offene Plattform eLectures@THB, die den Open-Source-Videoplayer des Hasso-Plattner-Instituts der Universität Potsdam implementiert (s. Abb. 1). Dadurch ist es möglich, zwei Video-Objekte (Sprecher und Screencast) parallel und untereinander skalierbar abzuspielen. Links auf die Videovorlesungen können aber auch direkt in die Kurse im Lernmanagementsystem (Moodle) eingebunden werden.

2.2 Seminaraufgaben

In der Idealvorstellung sollen Lehrveranstaltungen – insbesondere an Hochschulen für angewandte Wissenschaften – einen seminaristischen Charakter haben. D. h. neben dem Vortrag des/der Lehrenden soll eine intensive Diskussion und Beteiligung der Studierenden ermöglicht werden. In der Praxis gelingt das nur selten. Wird die primäre Wissensaneignung in die Eigenverantwortung der Studierenden gelegt – wie im Inverted-Classroom-Ansatz umgesetzt – steht die Präsenzzeit vollständig für seminaristische Aktivitäten zur Verfügung.

Dafür bedarf es abwechslungsreicher und anregender Diskussions- und Interaktionsanlässe. Folgende Formen wurden für den Kompaktkurs entwickelt und eingesetzt:

- *Q&A-Forum*: Die Studierenden werden aufgefordert, vertiefende Fragen (Questions = Q) zu den Inhalten der Vorlesung in ein Forum zu posten. Die Diskussion (Answers = A) entspannt sich entlang dieser Fragen. Alternativ können die ersten Minuten einer Präsenzveranstaltung genutzt werden, um in Kleingruppen Fragen zu entwickeln, die im Nachgang gesammelt, visualisiert und diskutiert werden.
- *Fehlerdiskussion*: Die Studierenden erhalten fehlerhafte Modelle, die sie in Kleingruppen analysieren und die Fehler kenntlich machen bzw. einordnen. Grobe Fehlerkategorien sind: syntaktische Fehler, fachliche Fehler und stilistische Fehler. Die Anzahl der Fehler in den Modellen ist vorgegeben. Im Plenum stellen die Kleingruppen ihre Ergebnisse vor, die so diskutiert werden.
- *Praxisreflexion*: Die Studierenden erhalten Modellausschnitte zu typischen Mustern der Ausnahmebehandlung (angeheftete Zwischenereignisse, Throw-Catch-Muster, Ereignis-Teilprozesse) sowie der Mehrfachinstanziierung (Schleifen und Listen,

parallel und sequenziell, für Aktivitäten und Pools), die sie interpretieren und in ihrem Erfahrungsraum reflektieren. Auch hier erfolgt die anschließende Diskussion im Plenum.

2.3 Selbsttests

Selbsttests spielen im Kompaktkurs eine doppelte Rolle. Sie dienen zum einen der persönlichen Fortschrittskontrolle im Selbstlernprozess. Zum anderen bereiten sie auf eine typische Prüfungsform in Zertifikatskursen – auch im Umfeld der Prozessmodellierung – vor. Auch in den Selbsttest-Fragen wird viel mit Prozessmodellen gearbeitet. Daneben gibt es theoretische Problem- und Vertiefungsfragen.

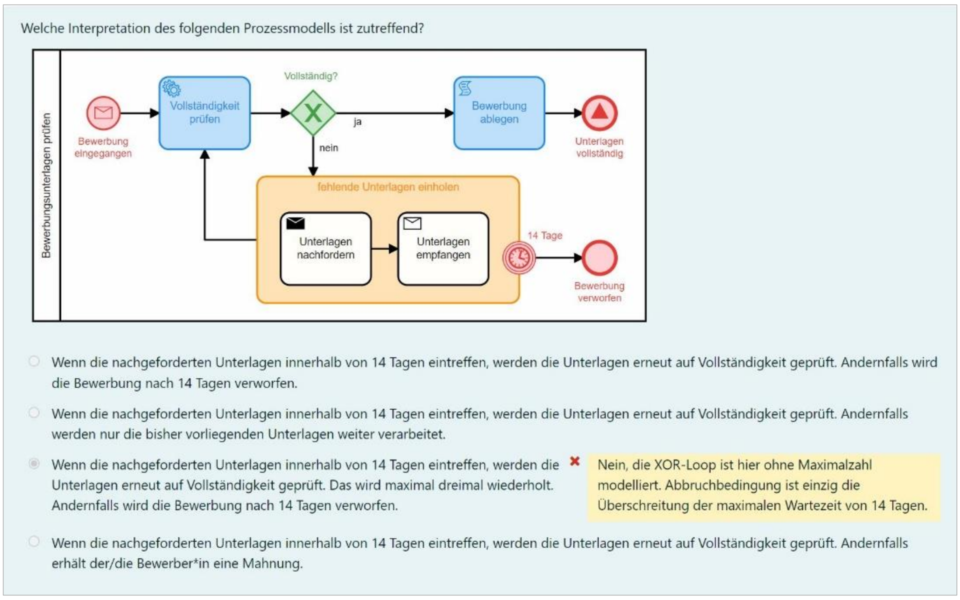


Abb. 2: Selbsttestaufgabe mit Feedback auf fehlerhafte Antwortoption

Die im Kompaktkurs eingesetzten Selbsttests bestehen aus jeweils 10 Single-Choice-Fragen. Dieses Format wird von den Studierenden als fair wahrgenommen und ist zugleich trennscharf. Alle vier Antwortoptionen werden mit Feedbacktexten ausgestattet, sodass die Studierenden ihre Resultate reflektieren können (s. Abb. 2). Die Selbsttests haben eine Zeitbeschränkung auf 15 Minuten, können aber mehrfach wiederholt werden.

2.4 Modellierungsübungen

Für den praktischen Part der Präsenzveranstaltungen erhalten die Studierenden Modellierungsübungen, die ebenfalls im Schwierigkeitsgrad ansteigen. Die erste Übung zeigt zwar einen vergleichsweise komplexen Prozess, setzt aber zunächst auf die Analyse der einzelnen Modellierungselemente sowie der Prozesslogik. Erst im Anschluss folgt eine Reflexion des Modells auf Basis einer textuellen Beschreibung. Zudem dient es der Einführung des Modellierungswerkzeugs Camunda Modeler.

In der zweiten Übung ist dann eine erste eigene Modellierung gefragt. Die Studierenden erhalten einen absichtlich sehr knapp gehaltenen Text und werden gebeten, nicht nur den im Text beschriebenen Prozess zu modellieren, sondern auch über weitere Optionen im Prozess und deren Umsetzung im Modell nachzudenken.

Für die weiteren Übungen wird eine Text-Serie genutzt, um im Schwierigkeitsgrad ansteigende Modellierungselemente und -muster praktisch anzuwenden. Diese iterative Form verdeutlicht das typische Vorgehen in praktischen Modellierungsprojekten und bietet Ansatzpunkte für vertiefende Diskussionen. Musterlösungen werden nur sparsam eingesetzt, da sie den Wahrnehmungsfokus und damit die Kreativität einschränken.

2.5 Modellierungsprojekt

Neben einem Abschlusstest, der im Format den Selbsttests ähnelt, dient ein freies Modellierungsprojekt als wichtigste Prüfungsleistung. Hier sind die Studierenden gehalten, selbständig in ihrer eigenen Domäne zu arbeiten. Kleingruppenarbeit ist nur dann zulässig, wenn mehrere Studierende über gemeinsame Prozessexpertise in einer Domäne verfügen, was praktisch selten der Fall ist.

Die Projektarbeit erstreckt sich je nach Kursdesign über mehrere Tage oder Wochen. Im Ergebnis sind die Prozessmodelle zu präsentieren und abzugeben. Die quantitativen und qualitativen Anforderungen an den Prozess und das Modell sind klar spezifiziert, um trotz unterschiedlicher Modellierungskontexte eine Vergleichbarkeit der Leistungen zu gewährleisten. Darüber hinaus ist eine Prozessdokumentation mit folgenden Bestandteilen anzufertigen:

- Prozessabgrenzung nach einem vorgegebenen Template,
- Beschreibung des Prozessumfelds (Unternehmen/Organisation/ Organisationseinheit, Branche, Rechtsform, Größe),
- Beschreibung des Prozesses – textuelle Darstellung mit wesentlichen Details,
- Motivation von mindestens fünf Modellierungsentscheidungen,
- Einschätzung des Prozesses im Hinblick auf seinen Automatisierungs- bzw. Digitalisierungsgrad sowie das diesbezügliche Verbesserungspotenzial.

3 Einsatzerfahrungen und -pläne

Das Kurskonzept wurde – wie im Titel benannt – primär für die akademische Lehre im Studiengang Bachelor Wirtschaftsinformatik erarbeitet. Es wird sowohl an der Heimathochschule als auch für Gastvorlesungen im Ausland eingesetzt. Darüber hinaus wird es in leicht modifizierter Form in Studiengängen der Betriebswirtschaftslehre angewandt.

Geplant ist, zumindest die Videovorlesungen und die Selbsttests in der betrieblichen Weiterbildung in der öffentlichen Verwaltung einzusetzen. Die inzwischen zahlreich laufenden Digitalisierungsprojekte – auch an Hochschulen – erfordern Modellierungskompetenz und darauf aufbauend die Fähigkeit aktiv in diesen Projekten mitzuwirken und zu kommunizieren.

Erfahrungen mit dem Kurskonzept wurden sowohl in der Vor-Ort- als auch in der Online-Lehre gesammelt. Beide Formate sind gut anwendbar. Die Mehrzahl der Studierenden schätzt die Flexibilität der Wissensaneignung durch Videovorlesungen. Das gesamte Anforderungspaket wird als herausfordernd, aber effektiv eingeschätzt. Dieses Feedback wurde sowohl im Rahmen der regelmäßigen, formellen Evaluation der Lehre als auch in veranstaltungsinternen, informellen Feedback-Einheiten gesammelt. Der vorliegende Beitrag ist nicht als wissenschaftliche Untersuchung des vorgestellten Konzeptes zu verstehen, sondern als Beitrag zum Erfahrungsaustausch innerhalb der Fachcommunity.

4 Zugang zu den Materialien

Alle Materialien des Kurses sind unter den in Tab. 1 angegebenen Links verfügbar.

Kursmaterial	**Format**	**Zugangslink**
Einfache Prozesse mit BPMN	Videovorlesung	BPMN-Basics
Kommunikation und Kollaboration	Videovorlesung	BPMN-Collab
Spezifikation von BPMN-Elementen 1	Videovorlesung	BPMN-Spec1
Spezifikation von BPMN-Elementen 2	Videovorlesung	BPMN-Spec2
BPMN – Weiterführende Themen	Videovorlesung	BPMN-Advanced
Vorlesungsskripte	Dokumenten-Archiv	BPMN-Skripte
Selbsttests zu allen Vorlesungen	Moodle-XML	BPMN-Selbsttests
Seminaraufgaben, Modellierungs-übungen und Projekttemplates	Dokumenten-Archiv	BPMN-Aufgaben

Tab. 1: Auflistung der Kursmaterialien mit Zugangslinks

5 Schlussbemerkungen

Die Autorin verfolgt seit jeher das Prinzip der Offenheit von Lehr-Lernmaterialien und hat ihre Ausarbeitungen bereits vielfach mit Kollegen und Kolleginnen sowie mit anderen Institutionen geteilt. Das hier im Beitrag vorgestellte Kurskonzept wurde 2022 grundlegend überarbeitet. Es sind langjährige Erfahrungen aus der Lehrtätigkeit an deutschen und ausländischen Hochschulen sowie im Kontext betrieblicher Weiterbildung eingeflossen. In dieser Form wurden die Materialien noch nicht breit veröffentlicht. Eine Weiterentwicklung in Richtung OER (Open Educational Resources) ist angedacht.

Literaturverzeichnis

[Ha20] Hamann, D. et al.: Zur Modellierungsausbildung in der Informatik: Auswertung von Modellierungsaufgaben im Zeitraum 2011 bis 2019. In (Michael, J; Bork, D., Hrsg.) Modellierung-C 2020: Short Papers, S. 33-39, 2020, URL: http://ceur-ws.org/Vol-2542/MOD20-SP4.pdf.

[IS13] ISO/IEC 19510: Information technology - Object Management Group, Business Process Model and Notation, 2013, URL: https://www.omg.org/spec/BPMN/ISO/19510/PDF.

[Ro13] Rot, E.: Kontrollierte Experimente zur Verständlichkeit, Ästhetik und Übersichtlichkeit von Prozessmodellen. Masterthesis, Hochschule Ravensburg-Weingarten, 2013, URL: http://dbis.eprints.uni-ulm.de/999/.

[ZH17] Zeaiter, S; Handke, J. (Hrsg.): Inverted Classroom – The Next Stage – Lehren und Lernen im 21. Jahrhundert. Tectum Wissenschaftsverlag, 2017.

Business Technologies

A Modeling Centered, Application Oriented Course

[1]Heinrich C. Mayr, [2]Volodymyr A. Shekhovtsov

Abstract: In this paper, we present the course "Business Technologies", which we offer at Alpen-Adria-Universität for students of the master's programs in Informatics and Information Systems to introduce them to the exciting world of modeling. It is a combination of lecture and project work that has been offered regularly since 2013, with the content being adapted and updated every year.

We intend this paper to stimulate discussion of modeling as an important component of Informatics teaching. Accordingly, not all our theses are scientifically substantiated; rather, some of them are intentionally formulated in a somewhat pointed way to provoke the discussion.

Keywords: Course content, project work, modeling, metamodeling, ontologies, process, business process, quality

1 Introduction

Since their introduction, the curricula of informatics studies have undergone massive changes: In the beginning the mathematical and telecommunication basics were in the foreground; from the mid-1970s onwards the main areas of software engineering, databases and information systems, operating systems, theoretical computer science, languages and artificial intelligence emerged. Within the framework of a curriculum, students were able at that time to specialize in sub-areas in the sense of majors. With the emergence of new application areas, new studies were developed, such as business informatics, bioinformatics, legal and administrative informatics, etc. A dam burst with the introduction of the Bologna structure: whenever a new buzzword comes up or a new hype emerges, new curricula are promptly invented and set up, differing from each other often only in nuances. For example, the website "studyCHECK.de"[3] currently lists more than 60 study programs with different names in Germany and Austria, which can be attributed to Informatics.

In all this time, models mostly played the role of a Cinderella in informatics teaching, despite their importance as tools for mastering complexity, for understanding complex issues, for specifying processes and systems of all kinds. Usually, the focus is on teaching

[1] Application Engineering, Alpen-Adria-Universität Klagenfurt, Austria heinrich.mayr@aau.at
[2] Information and Communication Systems, Application Engineering, Alpen-Adria-Universität Klagenfurt, Austria volodymyr.shekhovtsov@aau.at
[3] https://www.studycheck.de/studium/informatik-mathematik; visited 23.5.2022

a modeling language (SQL-DDL, UML, BPMN, SySML or a related one), and often this is done incidentally in a course on, for example, database systems, program development, business processes, mechatronic systems, etc. The essence of modeling, the understanding of abstraction and language hierarchies, such as one needs to develop domain-specific modeling languages, fades into the background. We suspect that this is mainly since abstracting and dealing with abstract is more difficult today for both teachers and learners, because the mathematical-algebraic foundations necessary for it have been largely banished from Informatics curricula. But the sometimes-encountered view that modeling is something imprecise and can therefore easily be done on the side also plays its part.

In this paper, we present a course that we offer at Alpen-Adria-Universität for students of the master's programs in Informatics and Information Systems to bring modeling a bit out of its Cinderella niche. The course title is "Business Technologies" and is chosen intentionally: on the one hand to arouse students' curiosity, on the other hand to express that modeling plays a central role in the technical support of entrepreneurial activities. It is a combination of lecture and project work that aims to introduce interested students to the essence of modeling. We have been offering this lecture regularly since 2015, with the content being adapted and updated every year.

The structure of this short paper is as follows: In section 2, we briefly describe what motivated us to set up this course. In section 3 we give an overview of the course content, followed by the supervised project tasks for the students including examples of solutions in section 4. The paper concludes with an evaluation and reflection of this endeavor.

2 Motivation

Already at the beginning of this millennium, modeling had a shadowy existence in many informatics curricula. Therefore, we developed and offered a lecture (with exercises) on the topic of modeling as part of the diploma program in computer science at that time. However, the colleagues in the curriculum committee had agreed to the inclusion of this lecture in the curriculum only if it was held in the 2nd semester, so that the other courses (database systems, software engineering, operating systems, etc.) could build on it. Consequently, the modeling contents from these courses had to be packed into the new modeling lecture. This endeavor failed grandiosely, because the efforts that students of the 2nd semester (who mainly struggle with programming, algorithms and data structures) understand modeling not only as a loose drawing of some graphs or networks, were unsuccessful: this was expressed not least in the fact that the students found the material easy at first, but then were very surprised that the exams had almost the highest failure rates.

After a few years, we abandoned this endeavor and instead developed an (elective) lecture for the now introduced master's and doctoral programs in computer science and information systems - i.e., for students from the 7th semester onwards. These students should be expected to be a bit above the technical stuff and to be able to understand abstract contexts. As already mentioned in the introduction, we have named this course "Business Technologies", because the modeling of structures and processes, but also the development of modeling languages are crucial for the design and development of information systems for business practice. One - not entirely selfless - goal was to attract students for projects (master's theses and dissertations), but in the foreground was the desire to give modeling a place in our Informatics curricula.

Since the student population at our faculty is very international, the course is taught in English. Its "curricular value" is 4 ECTS. It consists of a teaching part and a supervised project, which groups of students have to choose and work on.

3 Course Contents

Over time, we have repeatedly updated the course content, adjusting its presentation and sequence to reflect experience, and based on student comments. Currently, the course is organized as follows:

1. **Introduction and Motivation**: We derive the importance of modeling in computer science from its explanation as the science and technique of abstracting, analyzing, designing, and executing processes. According to this, models are the key instruments for (1) understanding, (2) analysis and measurement, (3) planning, execution, control, and support of processes by appropriate systems and infrastructure depending on the process' nature and (4) optimization w.r.t. effectiveness, efficiency (vs. resources), cost, time consumption, user satisfaction, quality of results, etc.
2. **Model and Modeling Process**: In this section, basic concepts are introduced and the essential properties of models according to [Ma15] are explained. Special attention is given to the distinction between model object, model (as mental object) and model representation (by symbols/language elements) for the purpose of communication. The conclusion is an explanation of the different types of models with respect to their purpose: from prescriptive or descriptive or explanatory to transient models, from causal to explorative to predictive models, etc. The conclusion is an overview of the "House of Model" [Th14].
3. **Metamodel Hierarchy, MCA and DSML**: Motivated by the paradigm of "Model-Centered Architecture (MCA)" [Ma17] for digital Ecosystems, model hierarchies (such as the classical Information Resource Dictionary System levels and the OMG MetaObject Facility) and the associated language hierarchies are introduced here: Attention is paid to not mixing model aspects and language aspects, as this is also important in the context of Domain Specific Modeling Languages (DSML) and their development, introduced subsequently. This chapter then also discusses the step-by-step development of a DSML according to [MM15] and illustrates it with a number of example projects.

4. **(Business) Process Modeling:** Now the essential concepts of process modeling in general and business process modeling in particular are introduced. The focus is not on teaching a concrete language (even though Event Process Chains (EPC) are sometimes used for illustration) but on the aspects and properties of processes to be modeled, as well as the different types of processes encountered in practice.
5. **BPMN, ADOxx, Adonis and Horus:** In accordance with its title, this chapter is dedicated to concrete and well-known modeling methods and languages; in addition, the metamodeling framework ADOxx®, which is used in the exercises and projects, is also introduced here.
6. **Ontologies:** The semantic foundation of conceptualizations plays a major role in modeling and, of course, especially in the development of domain-specific modeling languages. Therefore, the essence of ontology is considered in more detail here, and the different types of ontologies encountered in computer science are highlighted. To address ontology implementation, OWL (W3C Web Ontology Language) is introduced.
7. **Process Quality, Quality Modeling, Principles of Orderly Modeling:** As the name of this very broad chapter suggests, it deals with the different aspects of quality in the context of modeling: quality models and their underlying "quality ontologies", quality of models, quality of the modeling process, quality awareness in system development.
8. **Aligning Business Models with Business Processes:** This is about the important integration of models of the different facets and levels of a company.
9. **Foundations of Process Modeling**: Here, the theoretical foundations of "Model Checking" and its principles are covered in sections (1) Petri Nets, (2) Finite State Automata, (3) State Charts, and (4) Model Checking.

As one can conclude from this list, it is quite challenging to deal with all these topics in a course in the appropriate depth and with the necessary exactness in such a way that they are also comprehensible to the students. We therefore must admit that the last chapter usually comes off a bit short, also because the mathematical background required here is rather unfamiliar to the students. In view of the fact that our studies are also geared more towards "applied computer science", this is, however, justifiable.

4 Practical Tasks / Projects

As noted in section 2, the course "Business Technologies" consists of a part of frontal teaching and a supervised group work in the form of a project. In these projects, either a comprehensive business process model or a (simple) domain-specific modeling language including a modeling tool has to be developed by each group. Suggestions for topics are given by the course instructors, but students can also suggest topics from their own field of experience. The project work starts from the 4th week of the semester and lasts until the end of the lecture. During this time, students must repeatedly present interim results (analysis, drafts, etc.) - and receive advice for further work. The final result is the finished model or modeling tool, as well as a report.

Examples of topics from the area of business process modeling

"*You are requested to perform the necessary requirements elicitation and analysis, process model design and implementation for one of the following business processe*s:"

- Approval of a building project from the perspective of a municipality
- Purchase and registration of a company car
- Operation of a fast-food restaurant
- Customer acquisition by an electricity supplier (example solution see figure 1)
- Preparing a dissertation at AAU (along the study and examination regulations)
- Preparing and performing a conference participation including travel
- Organizing a scientific conference
- Handling a property damage report – from dealing with insurance to organizing repairs
- Handling bug reports and feature requests by a ticketing system like Atlassian JIRA or Bugzilla.

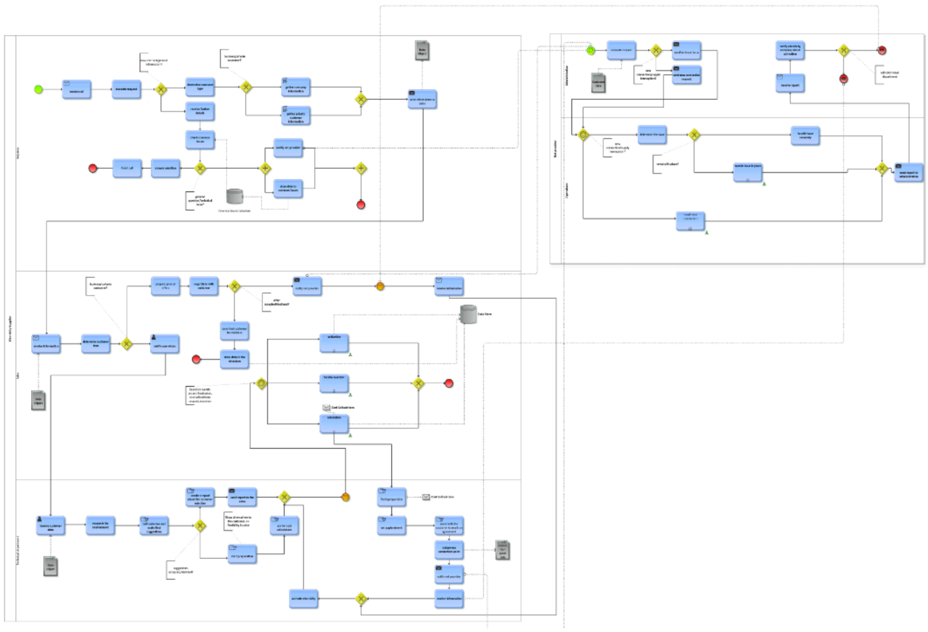


Figure 1: Business process model developed by a student group for the "Customer acquisition by an electricity supplier" assignment, represented using ARIS Express; *Note: this image is intended to give an impression of the complexity of the solution, but not its details*

Examples of topics for designing a DSML and developing a modeling tool.

"*You are requested to perform the process of DSML-development as described in [MM15] for one of the following domains:*"

- Family trees
- Room plans
- Petri Nets
- System Dynamics
- Static and dynamic aspects in medical practices (example solution see figure 2)
- Ambient assistance in the households of the elderly
- Garden planning with 2D elements
- Covid19 Vaccination
- Traffic planning of a city

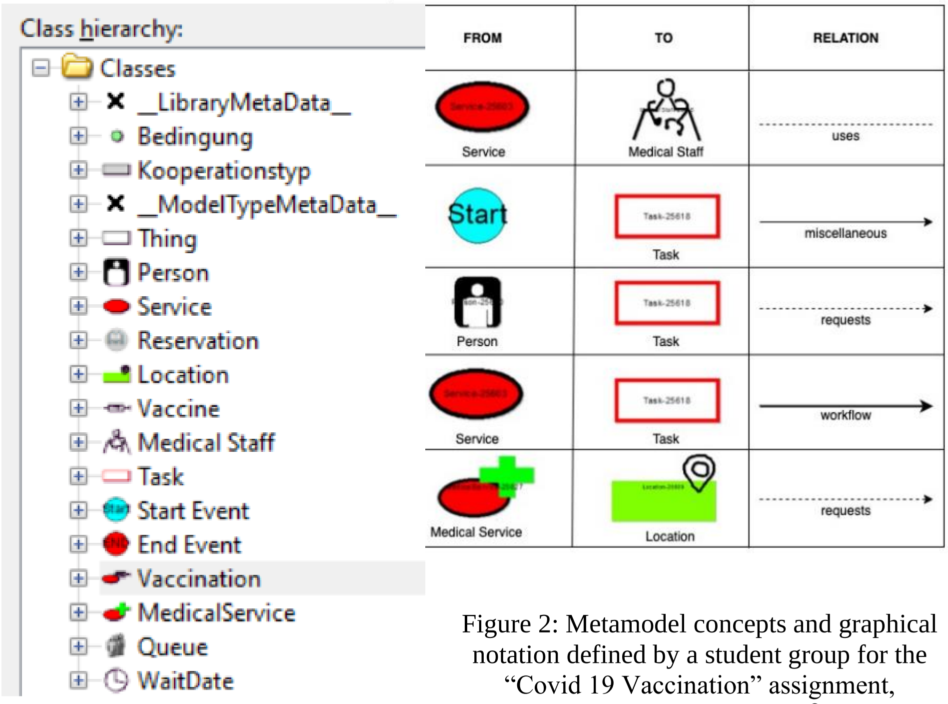


Figure 2: Metamodel concepts and graphical notation defined by a student group for the "Covid 19 Vaccination" assignment, implemented in Adoxx®

5 Evaluation

In times when it is in to look at "artificial intelligences", data analytics, deep learning etc., a course focused on modeling does not find an extremely large number of participants, but very interested and committed ones.

They all performed their respective tasks well to very well, of course under appropriate supervision, and we particularly enjoyed seeing the progress in understanding modeling and thinking at different abstract levels: that models and their representations should be separated, that model and language hierarchies are structured differently, that at each model level (M^0 to M^n in terms of MOF) there must be a semantic foundation in the form of an ontology or encyclopedia so that one can talk about conceptual models, all this the students grasped over time. And they were able, in the context of their project, to define their own modeling language and generate a modeling tool for it or to perform an extensive requirements elicitation and study, to choose a BP modeling tool and to develop an appropriate business process model. In addition, they showed a deeper understanding of what constitutes processes, why parallelism and concurrency are to be distinguished from each other, or why it makes sense to do model checking. According to statements and records of the students, they were able to complete their assignments in the planned time, i.e., in approx. 60 full working hours per person.

The self-serving goal of attracting students for master's theses and dissertations, as stated in the motivation, was achieved: A total of four dissertations and four master's theses have been started by interested students of this course, of which five have been completed and three will be completed soon.

6 Acknowledgement

We thank the unknown reviewers for their constructive and valuable comments.

7 References

Students will, of course, be given extensive references to the individual chapters as part of the course. Within the scope of this short paper, such a list would go too far, so we limit ourselves to the literature references referenced in the text.

[Th14] Thalheim, B.: The Conceptual Model: An Adequate and Dependable Artifact Enhanced by Concepts. Information Modelling and Knowledge Bases, Vol 25, IOS Press, 2014

[Ma15] Mahr, B.: Modelle und ihre Befragbarkeit - Grundlagen einer allgemeinen Modelltheorie. Erwägen-Wissen-Ethik (EWE), Vol. 26, Issue 3, pp. 329-342, 2015.

[Ma17] Mayr, H.C.; Michael, J.; Ranasinghe, S.; Shekhovtsov, V.A.; Steinberger, C.: Model Centered Architecture. In Cabot, J. et als (eds.): Conceptual Modeling Perspectives. Springer Nature, 2017.

[MM15] Michael, J.; Mayr, H.C.: Creating a Domain Specific Modelling Method for Ambient Assistance. In: Proc. Int. Conf. on Advances in ICT for Emerging Regions 2015.

A Report on Automatic Generation of Petri Net Exercise and Exam Task Instances[1]

André Brandt, Marcellus Siegburg[2] Janis Voigtländer[3] Ke Wang

Abstract: We report on generators for different task types addressing Petri net concepts from a modeling lecture for undergraduate students. A focus is on how to control difficulty and intended insights about the subject matter on the learners' side. We explain the influence of provided configuration parameters for several task types on an exemplary instance each, and comment on presentation and implementation, as well as very briefly on exam experience.

Keywords: Petri nets; E-learning; Task generation

1 Introduction

To help teaching Petri net concepts to undergraduate students, and assess their understanding, we employ an e-learning setup with diverse types of exercise and exam tasks. We have implemented task types for different Petri net concepts (mathematical representation, concurrency, conflicts) and with different answer modi such as multiple-choice and matching tasks. Each task type is equipped with a generator that can produce a multitude of task instances. Each such generator is controlled by a set of custom configuration parameters. Such parameters can involve size constraints (e. g., how large the Petri net should be), structural constraints (e. g., whether certain graph patterns may or should appear in the Petri net), and more task specific constraints (e. g., whether certain forms of distractors should appear among the alternatives presented in a multiple-choice task). These parameters control the difficulty of generated task instances and allow to steer learning and understanding of specific aspects of the Petri net concepts under consideration. Setting the configuration values for a task generation run is performed by the lecturer. Each obtained task instance comes with a correct-by-construction answer that can be used in giving immediate automatic feedback on student submissions. But we have also used the setup with non-immediate grading as part of distant online exams, where the generation facilities were instrumental to providing individual task instances to students, basically eliminating potential for plagiarism.

In what follows, we discuss our Petri net task types, with a focus on their configuration parameters and how they can be used to adjust the level of challenge for students. We also

[1] Part of the work reported here was funded via:
Projekt „PITCH – Prüfungen innovieren, Transfer schaffen, Chancengerechtigkeit fördern“ (08/2021–07/2024), Projektnummer FBM2020-EA-1190-00081, aus Mitteln der Stiftung Innovation in der Hochschullehre
[2] University of Duisburg-Essen, Faculty of Engineering, Germany, marcellus.siegburg@uni-due.de
[3] University of Duisburg-Essen, Faculty of Engineering, Germany, janis.voigtlaender@uni-due.de

briefly discuss our implementation strategy, and then conclude with some subjective as well as quantitative experience from exercises and an exam.

2 Task Type: Matching Representations of Petri Nets

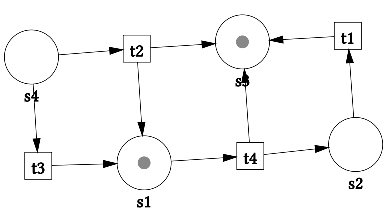
In the lecture, students are given a formal definition of Petri nets as a mathematical structure. This builds on set-theoretic constructions they have learned in a preceding discrete mathematics course. The role in our lecture is to provide a basis for formal definition of subsequent semantic concepts, but also to address frequent questions about the syntax of Petri nets, such as whether it is allowed to have an arrow directly from a place node to another place node, whether it is allowed to have a transition node without any outgoing arrows, etc. One task type used to practice the relationship between mathematical and diagrammatic representation is as follows: Students are shown a mathematical representation using the notation from the lecture, as in Fig. 1, and several Petri net diagrams, as in Fig. 3, and are asked which of the diagrams corresponds to the given mathematical rendering.

$N = (S, T, {}^\bullet(), ()^\bullet, m_0)$, where
$S = \{s_1, s_2, s_3, s_4\}$ and
$T = \{t_1, t_2, t_3, t_4\}$, as well as using the place ordering (s_1, s_2, s_3, s_4):
- ${}^\bullet t_1 = (0, 1, 0, 0)$
- ${}^\bullet t_2 = (0, 0, 0, 1)$
- ${}^\bullet t_3 = (0, 0, 0, 1)$
- ${}^\bullet t_4 = (1, 0, 0, 0)$
- $t_1^\bullet = (0, 0, 1, 0)$
- $t_2^\bullet = (1, 0, 1, 0)$
- $t_3^\bullet = (1, 0, 1, 0)$
- $t_4^\bullet = (0, 0, 1, 0)$

Moreover, $m_0 = (1, 0, 1, 0)$

Fig. 1: Mathematical representation of a Petri net

Let us discuss the main configuration parameters used for this task type, besides the obvious one controlling how many diagrams to present as possible choices. Several of them are also employed in other task types considered later on. Here we additionally mention with what concrete settings for the parameters the generator was called to obtain the task instance from Figs. 1 and 3, and the effect certain changes to those settings could have had.



parameter	value
`places`	4
`transitions`	4
`minTokensOverall`	2
`maxTokensOverall`	2
`maxTokensPerPlace`	1
`minFlowOverall`	10
`maxFlowOverall`	10
`maxFlowPerEdge`	1

Fig. 2: A Petri net abiding by the given configuration values

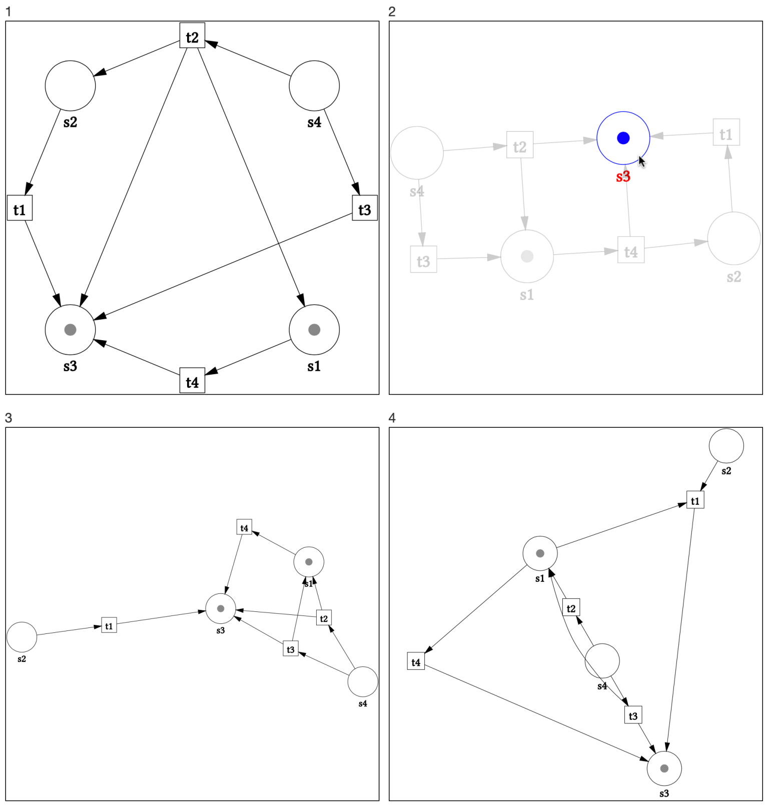


Fig. 3: Four Petri net diagrams, exactly one of which corresponds to Fig. 1

There are some self-explanatory size constraints which restrict the numbers of nodes and edges in a Petri net, as well as provide control over tokens and edge weights. These are illustrated in Fig. 2 by putting one generated Petri net alongside their specific values used. Note that we could have increased `maxTokensOverall` to `3` without changing `minTokensOverall`. This could have had two effects: First, some students might have received a task instance like the one from Figs. 1 and 3, with only two tokens in each net, while others would have seen three tokens per diagram. Second, even for a single task instance, it could in principle have been the case that some displayed diagrams contain two tokens, and others three tokens. The latter, however, would have been prevented here because we also set `tokenChangeOverall = 0` when generating Figs. 1 and 3. That makes sense in the current task type because redistributing tokens would simplify the generated task instances a lot: For example, since the initial marking is given as $m_0 = (1, 0, 1, 0)$ in Fig. 1, any diagram that does not have one token on each of s_1 and s_3 and no others, would immediately be disqualified from being the correct answer – without the student even having to look at the pre- and postcondition weights in the mathematical representation.[4] In order to actually have some differences (apart from layout) between the presented diagrams, and to ensure that only one of them corresponds to the given mathematical representation, we use a setting `flowChangeOverall = 2` (and `maxFlowChangePerEdge = 1`).

One structural constraint setting used in the example is `isConnected = Just True`. Setting that parameter to `Just False` instead would have enforced that each of the four Petri nets offered as choices in Fig. 3 would have consisted of at least two disjointed graph components, while setting it to `Nothing` would have meant that we do not care. The same kind of "three-valued logic" applies to the configuration parameters `presenceOfSelfLoops`, `presenceOfSinkTransitions`, and `presenceOfSourceTransitions`, which were all set to `Just False` for the example. By allowing, or even enforcing, self loops and sink or source transitions to occur, we can already expose students to Petri nets containing these patterns, which will have a more interesting interplay with concepts like concurrency and conflicts later in the lecture. For example, a source transition can never be involved in a conflict.

Another parameter setting used here was `atLeastActive = 1`, basically preventing that any of the displayed Petri nets is deadlocked, despite that aspect not really having much of a conceptual impact on solving instances of this task type.

Concerning display, note that no weight numbers are being shown on the arrows in Figs. 2 and 3 because we set option `hideWeight1 = True`. This is the case throughout the paper to avoid issues with readability of overlapping labels. In our online setting, interactive highlighting features such as demonstrated in net 2 in Fig. 3 are available to students: marking nodes or edges in tandem with their corresponding labels when pointing on them.

Finally, we had `useDifferentGraphLayouts = True` and `graphLayout = [Sfdp, Circo, Neato, TwoPi]`, ensuring that each Petri net is displayed using a different GraphViz layout engine from a fixed set of choices. Setting `useDifferentGraphLayouts = False` and `graphLayout = [Circo]` would have meant that all four Petri nets in Fig. 3 are displayed in the current style of the first one, thus making it much simpler to work out – just visually – what the changes between them are, and thus which one corresponds to the mathematical representation in Fig. 1.

Another task type, similar to the one discussed above, and also implemented, works the other way around: displaying one diagram and several mathematical renderings, then again asking for correspondence.

3 Task Type: Finding Concurrent Transitions in Petri Nets

An important concept introduced in the lecture, for Petri nets but also with a more general modeling outlook, is concurrency. One task type used to practice detecting concurrency in

[4] In other task types, positive values for `tokenChangeOverall` make more sense, and are then accompanied by a positive setting for `maxTokenChangePerPlace`.

the context of Petri nets is as follows: Students are shown a Petri net, as in Fig. 4, told that it contains two concurrently activated transitions, and asked to identify the relevant pair.

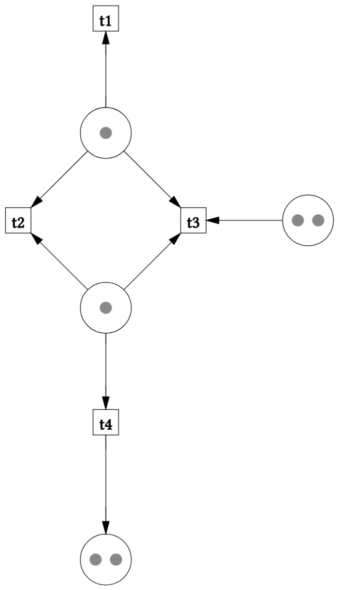


Fig. 4: A Petri net with two concurrently activated transitions

Several configuration parameters used for this task type are the same, and play the same role, as in Sect. 2. For example, there are again those mentioned in Fig. 2 as well as the `isConnected` parameter. The presence of a transition node without any outgoing arrows in Fig. 4 is not by happenstance but a consequence of setting `presenceOfSinkTransitions = Just True` in the configuration used when generating this example. Also, now `atLeastActive = 3` was used to avoid instances where only two transitions are activated at all, which would necessarily have made those the concurrently activated ones. That is, we wanted to have at least one further activated transition as a potential distractor. A new setting `hidePlaceNames = True` was used to simplify the display. In Sect. 2 that would not even have been an option because it would most likely make it impossible to always solve task instances (uniquely).

What might be called "change settings" (concretely `tokenChangeOverall = flowChangeOverall = 2` and `maxTokenChangePerPlace = maxFlowChangePerEdge = 1` in the configuration used when generating the example in Fig. 4) now play a slightly different role than before. Essentially, they control "how far away" the Petri net with concurrent transitions is from one that does not contain any concurrency. For example, we could thus deliberately generate only instances where the concurrency hinges on a single token (taking away that token would destroy the concurrency), or where it hinges on a single arrow's absence or weight (e. g., adding one arrow would destroy the concurrency).

A variant of the task type discussed above, also implemented, displays two Petri nets, one without concurrent transitions and one with a pair of concurrent transitions, and simply asks which net is which – without requiring students to pick out the specific pair of transitions.

4 Task Type: Finding and Explaining Conflicts in Petri Nets

Another important concept introduced in the lecture is the notion of two transitions in a Petri net being in conflict. One task type used to practice working with this concept is as follows: Students are shown a Petri net, as in Fig. 5, told that it contains two transitions in conflict, and asked to identify the relevant pair as well as the place(s) that is/are responsible for the conflict. That is, they also have to identify all places that are joint preconditions for the conflicted transitions while not having enough tokens to fire both transitions concurrently.

There are again several configuration parameters that are already known from Sects. 2 and 3 – we do not repeat most of them here. To generate the example in Fig. 5, we again used `atLeastActive = 3`, but also `presenceOfSelfLoops = Just True`. Concerning the "change settings", similar comments as towards the end of Sect. 3 apply. A new parameter is `uniqueConflictPlace`. Setting it to `Just True` means that where the students have to identify all places bearing responsibility for the conflict, the correct answer will actually be a singleton. Distractors for the conflict and its origin can be configured using more advanced settings:

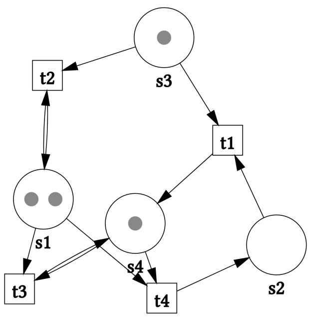


Fig. 5: A Petri net with two transitions in conflict (`t3` and `t4`)

Setting `addConflictCommonPreconditions = Just True` enforces that the two transitions in conflict have an additional place as common precondition, i. e., both having an incoming arrow from at least two same places. Such additional precondition places are meant to not be causes of the conflict. However, depending on the setting of `uniqueConflictPlace` from above, actual additional conflict-causing places could be enforced as well. Another setting that was used when generating the example in Fig. 5 is `withConflictDistractors = Just True`. It enforces the existence of at least one other pair of transitions, besides the conflicted pair, with non-disjoint preconditions. In the example, this resulted in the distractor pair `t1` and `t2` with common precondition `s3`. This pair has no additional common precondition places because `conflictDistractorAddExtraPreconditions = Just False` was set. Its common precondition place has only one token, and thus looks like a conflict, because `conflictDistractorOnlyConflictLike = True` was set.[5] What students would have to "discover" in the concrete instance from Fig. 5, in order to overcome the distractor pair, is that `t1` and `t2` are not actually conflicted, because `t1` is not even activated. But, of course, we as educators avoided having to handcraft the example to achieve this effect – instead relying on our generator and its declarative parameter settings.

In a slightly simpler version of the task type discussed above, the conflict-causing places do not have to be identified as part of the answer. Moreover, there is yet another task type, also implemented, which displays two Petri nets, one without a conflict and one with a pair of conflicted transitions, and simply asks which net is which.

[5] Setting `conflictDistractorOnlyConcurrentLike = True` instead would have resulted in more tokens on that place.

5 Implementation

While we discussed the task types on concrete examples in the preceding sections, each has an underlying generator. That is, for Sect. 2 we have a generator that from numeric and Boolean (or three-valued conditional) settings for `places`, `transitions`, etc., gives us many pairs of "Figs. 1 and 3", and likewise for Sects. 3 and 4. For example, Fig. 6 shows four random instances that were generated with the same configuration settings as Fig. 5. They are from a distant online exam, which is referred to in the next section, that employed literally hundreds of these (one per student). In preparation for the exam, similar exercise task instances had been provided to students, some generated with larger values for `places` and `transitions` and the token and flow numbers.

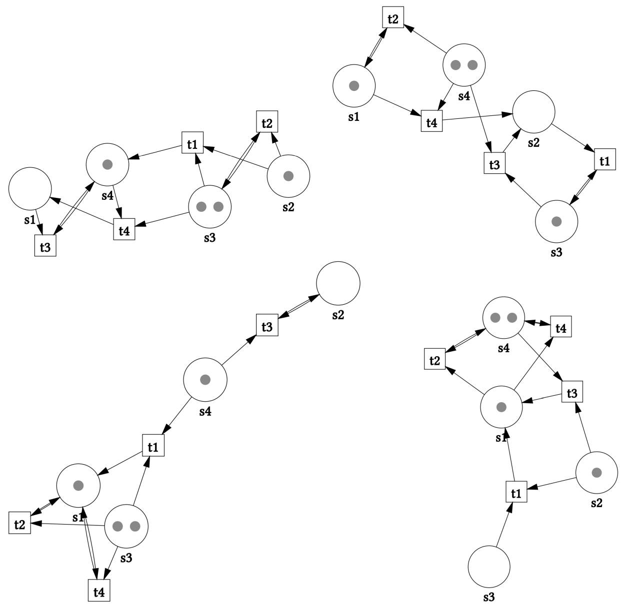


Fig. 6: Four of many different-but-alike instances used in an exam

It is probably apparent that randomness would not really do the job here. That is, a pure generate-and-test approach where Petri nets would be randomly generated (within given size constraints) and then tested whether they satisfy all the desires (like presence of certain distractors) and otherwise discarded, would not really be practical. Moreover, it would defy our correct-by-construction aspirations for task instances and feedback/sample solutions. Hence, we followed a more formal approach already adopted in previous work on generating task instances on the topic of UML class and object diagrams [KSV20, SV20]. It uses Alloy [Ja02, Ja11] to model the subject matter (back then, UML diagrams; now, Petri nets) along with various structural properties and semantic concepts on top. This gives an Alloy library, originally prototyped in [Wa19] and since then adapted and extended. Each task type can be thought of as a certain use case of that library. We let configuration parameters influence a logic formula built using predicates from the library, and sending that formula to the Alloy model checker and interpreting the returned outcomes gives us task instances. Initial experiments in this direction were the topic of [Br20], while the current work represents more fully developed task types and generators.

6 Conclusion

Subjectively, judging from forum discussions with and among students, our generated exercise tasks on UML and Petri net concepts have been successful in furthering engagement with the material as well as pointing to areas of miscomprehension and need for additional practicing before the exam. We are also gaining more experience with using such generated tasks in actual exams. Fig. 7 shows some data from our latest installment (March 2022). Tasks 09 to 14 are Petri net task types reported on in this paper, while Tasks 02 to 08 are task types reported on earlier [KSV20, SV20], and the remaining three tasks are yet different. A deeper analysis is out of scope here, but even at a superficial glance we see interesting effects, such as the difference in results between Task 13 and Task 14, both "from Sect. 4", but only the latter requiring students to identify the conflict-causing place(s).

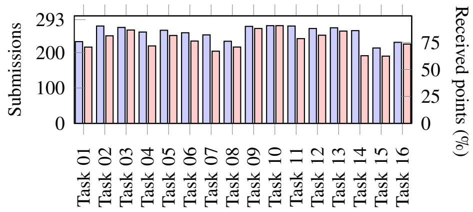


Fig. 7: Exam submissions made by students (blue) and point percentages received for those submissions on average (red).

We also would like to systematically categorize our task types and configuration options by the learning objectives they address, also using appropriate taxonomies. A first foray in this direction was undertaken in [Sc20], but not for the specific task generators discussed here.

Bibliography

[Br20] Brandt, André: Automatische Generierung von Übungsaufgaben zu Petrinetz-Konzepten mittels Alloy. Bachelor thesis, University of Duisburg-Essen, 2020.

[Ja02] Jackson, Daniel: Alloy: A Lightweight Object Modelling Notation. ACM Transactions on Software Engineering and Methodology (TOSEM), 11(2):256–290, 2002.

[Ja11] Jackson, Daniel: Software Abstractions – Logic, Language, and Analysis, Revised edition. MIT Press, 2011.

[KSV20] Kafa, Violet; Siegburg, Marcellus; Voigtländer, Janis: Exercise Task Generation for UML Class/Object Diagrams, via Alloy Model Instance Finding. In: SACLA 2019, Proceedings. Springer, pp. 112–128, 2020.

[Sc20] Schweiger, Matthias: Analyse von Übungsaufgaben zu Petrinetz-Konzepten. Bachelor thesis, University of Duisburg-Essen, 2020.

[SV20] Siegburg, Marcellus; Voigtländer, Janis: Generating Diverse Exercise Tasks on UML Class and Object Diagrams, Using Formalisations in Alloy. In: MoHoL 2020. volume 2542 of CEUR Workshop Proceedings. CEUR-WS.org, pp. 89–100, 2020.

[Wa19] Wang, Ke: Exploring Petri net concepts through formalization in Alloy. Bachelor thesis, University of Duisburg-Essen, 2019.

ALADIN – Generator für Aufgaben und Lösung(shilf)en aus der Informatik und angrenzenden Disziplinen

Paul Christ[1], Ralf Laue[2], Torsten Munkelt[3]

Abstract: Das Erlernen von Fähigkeiten zur Modellbildung ist eine grundlegende Zielstellung in vielen Studiengängen. Insbesondere in der Informatik und angrenzenden Disziplinen lassen sich viele Modellierungsaufgaben mittels Graphen repräsentieren, was das computergestützte Generieren solcher Graphen und entsprechender Aufgaben und Lösung(shilf)en auf Grundlage bestehender Graphenalgorithmen nahelegt. Dieser Beitrag stellt das Framework ALADIN vor, welches graphenbasierte Modelle und Aufgaben für Probleme aus verschiedenen Fachbereichen generiert und Studenten bei der Lösung der Probleme unterstützt. Die Generierung erfolgt parametrisiert, um dem Anforderungsprofil der Bearbeiter zu entsprechen. ALADIN ermöglicht eine zeit- und ortsunabhängige Bearbeitung von Übungsaufgaben. Zudem prüft ALADIN die Lösungen direkt auf Korrektheit, ohne Lehrpersonal zu binden. Aufzeichnungs- und Wiedergabefunktionalität erhöht den Nutzen von ALADIN in Blended-Learning-Szenarien.

Keywords: E-Learning, Aufgabengenerator, Modellierungsaufgaben, Blended Learning

1 Herausforderungen in der Modellierungslehre

Für Modellierungsaufgaben in der Informatik und angrenzenden Disziplinen stehen oft nur wenige Übungsaufgaben zur Verfügung. Diese werden zudem häufig während der Übung oder Vorlesung gemeinsam gelöst, so dass kaum unbekannte Aufgaben zum selbständigen Üben verbleiben. Da Klausuraufgaben aus einem begrenzten Reservoir von Aufgaben stammen, stellt das Lehrpersonal oft keine Musterklausuren bereit, um die vorhandenen Aufgaben nicht durch Probeklausuren bekanntzumachen. Selbst wenn Aufgaben in ausreichendem Umfange existieren, stellen die Studenten oft Fragen zum Lösungsweg oder benötigen Lösungshilfen, was die Verfügbarkeit des Lehrpersonals voraussetzt, erheblichen Aufwand bereitet und außerhalb der Lehrveranstaltungen nur begrenzt möglich ist. Der geschilderte Sachverhalt wurde u. a. in den Modulen „Grundlagen der Wirtschaftsinformatik“, „Geschäftsprozessmanagement“, „Datenbanksysteme“, „Produktionswirtschaft“ und „Geoinformationssysteme“ beobachtet.

[1] Hochschule für Technik und Wirtschaft Dresden, Fakultät Informatik/Mathematik paul.christ@htw-dresden.de
[2] Westsächsische Hochschule Zwickau, Fachgruppe Informatik ralf.laue@fh-zwickau.de
[3] Hochschule für Technik und Wirtschaft Dresden, Fakultät Informatik/Mathematik, torsten.munkelt@htw-dresden.de

2 Ansatz und Ziele von ALADIN

Das in diesem Beitrag vorgestellte Werkzeug ALADIN wurde mit dem Ziel entwickelt, computergenerierte Modellierungsaufgaben mitsamt Lösungen und Lösungshilfen zu generieren. Die Studenten können mittels ALADIN Aufgaben zeit- und ortsunabhängig selbständig lösen. Das Lehrpersonal kann ALADIN auch dazu verwenden, (individuelle) Prüfungsaufgaben zu generieren.

Der Entwicklung von ALADIN lag die Beobachtung zugrunde, dass sich viele Aufgaben der Informatik und angrenzender Disziplinen sowie ihre Lösungswege und Lösungen als Graphen und Algorithmen über Graphen auffassen lassen. Algorithmen und Softwarebibliotheken zum Erzeugen, Speichern, Traversieren, Modifizieren und Analysieren von Graphen sind allgemein bekannt bzw. liegen vor.

Die bei der Entwicklung von ALADIN verfolgten Anforderungen waren, dass

- graphbasierte Aufgaben automatisch und zufallsbasiert erzeugt werden,
- dabei je nach Vorkenntnissen oder Wünschen umfangreiche und weniger umfangreiche Aufgaben generiert werden,
- diese Aufgaben über Internet zeit- und ortsunabhängig bereitgestellt werden,
- das Werkzeug beim Lösen der Aufgaben Unterstützung in Form von Hinweisen oder schrittweisen Teillösungen liefert,
- Lösungsversuche untereinander ausgetauscht oder den Lehrkräften mit Bitte um Rückmeldungen weitergegeben werden können.

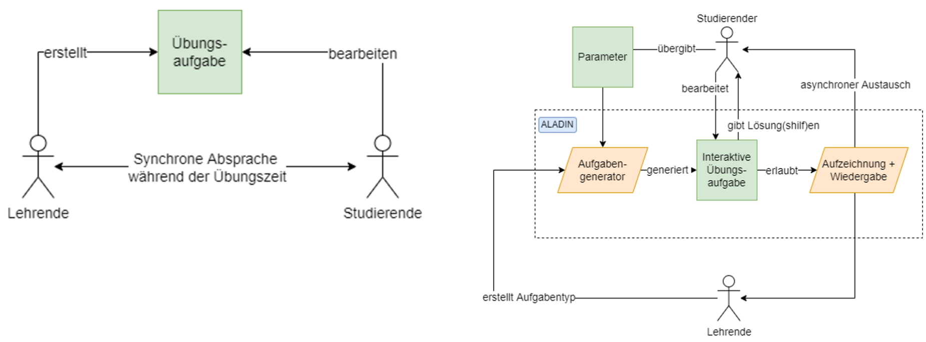
Mit ALADIN als Hilfsmittel kann die in Abschnitt 1 beschriebene Bindung von Lehrkräften reduziert werden, wie Abbildung 1 darstellt.

3 Erstellen von Aufgabentypen, Generieren und Lösen von Aufgaben und Leisten von Lösungshilfen in ALADIN

ALADIN verfügt im aktuellen Ausbau über eine Bibliothek von Funktionen, die beim Erstellen neuer Aufgabentypen genutzt werden kann. Die Bibliothek umfasst zum einen gängige Graphalgorithmen. Zum anderen stellt sie generische graphische Bedienelemente bereit, etwa zum Generieren einer Schaltfläche, einer Matrix oder einer Tabelle. Durch Komposition der Funktionen und Bedienelemente können unterschiedliche Aufgabentypen deklarativ beschrieben werden. Da die Beschreibung eines Aufgabentyps die Anordnung der Bedienelemente in Layouts in drei Größen erlaubt, unterstützt ALADIN sowohl Desktopauflösungen als auch Auflösungen mobiler Endgeräte, wie z. B. Tablets oder Smartphones.

Ein neuer Aufgabentyp kann so mittels der vorhandenen Funktionen und Bedienelemente konfiguriert werden. Sollte das Erstellen des neuen Aufgabentyps Elemente erfordern, welche nicht mit bereits vorhandener Funktionalität umsetzbar sind, kann zusätzlich benutzerdefinierter Code ausgeführt werden. So können für die Ausführung aufgabenspezifischer Algorithmen auch externe Programme eingebunden werden, deren Eingaben und zurückgelieferte Ergebnisse einer wohldefinierten Schnittstelle folgen müssen. Die Konfiguration der Logik zur Generierung des Graphen, der Lösungen und der Lösungshilfen sowie der interaktiven Benutzeroberfläche erfolgt in einer JSON-Datei, welche durch ALADIN interpretiert wird. Mit steigender Anzahl an wiederverwendbaren Funktionen in der Bibliothek wird der Bedarf an benutzerdefiniertem Code verringert und die Modellierungsmächtigkeit des Systems erhöht.

Die folgenden Schritte beschreiben die interaktive Bearbeitung einer Aufgabe in ALADIN: Die Studenten wählen einen Aufgabentyp aus und parametrisieren optional die Generierung einer Aufgabe. Auf Grundlage der Parametrisierung erzeugt ALADIN zufallsbasiert eine Aufgabe des gewählten Aufgabentyps. Die Parametrisierung erfolgt manuell durch die Studenten, oder ALADIN nimmt sie aufgrund der Nutzungshistorie automatisch vor, um die Komplexität der generierten Aufgabe an die Bedürfnisse bzw. an den Fähigkeitsstand der Studenten anzupassen. Während der Bearbeitung der Aufgabe können die Studenten Lösungshinweise anfordern. Zudem besteht die Möglichkeit, den Ablauf des Lösungsversuchs aufzuzeichnen und ihn zu einem späteren Zeitpunkt wieder abzuspielen. So aufgezeichnete Lösungsversuche können die Studenten darüber hinaus mit Kommilitonen und dem Lehrpersonal teilen, welche die bisher durchgeführten Interaktionen ebenfalls schrittweise abspielen können. Das Aufzeichnen und Teilen der Lösungsversuche ermöglicht einen asynchronen Austausch mit Anderen und individuelle Hilfestellungen, ohne Lehrpersonal (zeitlich) zu binden. Nach erfolgreicher Bearbeitung der Aufgabe wird der Nutzer entsprechend benachrichtigt.



(a) Szenario ohne ALADIN (b) Szenario mit ALADIN

Abb. 1: Interaktion zwischen Studenten und Lehrpersonal in Übungen

4 Aufgabenklassen und ihre Beziehung zu Aufgabentypen und Aufgaben in ALADIN

Die Bibliothek von ALADIN erlaubt die Modellierung verschiedener Klassen von Aufgaben, welche im Folgenden vorgestellt werden:

Für Aufgaben der Klasse „Fehler im Graphen finden“ modifiziert ALADIN einen zufallsbasiert generierten Graphen im Anschluss so, dass er syntaktische oder semantische Fehler aufweist. Die Aufgabe besteht danach darin, diese Fehler zu entdecken oder zu beheben. Da ALADIN den Graphen selbst modifiziert, sind die Modifikationen bekannt, sodass ALADIN entsprechende Lösungshilfen geben kann.

Für Aufgaben der Klasse „Berechnungen in Graphen“ generiert ALADIN ebenfalls zufallsbasiert Graphen und nutzt dann vorliegende Graphalgorithmen zum Finden der Lösung. Bei mehrschrittigen Lösungswegen können die Zwischenlösungen gespeichert und auf Anforderung der Studenten als Lösungshilfen zur Verfügung gestellt werden oder ALADIN löst die Aufgabe parallel zu den Studenten.

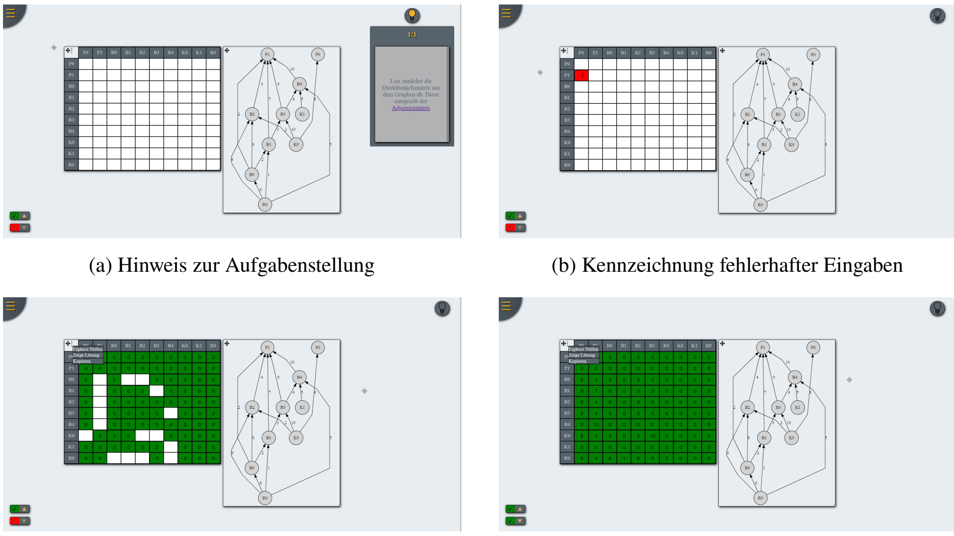
Für Aufgaben der Klasse „Graphen modellieren“ traversiert ALADIN einen zufällig generierten Graphen und erzeugt einen zu ihm äquivalenten Text. Die Aufgabe besteht nun darin, diesen Text zurück in einen Graphen zu überführen, der dem Originalgraphen isomorph ist oder mit ihm in den für die Modellierung wesentlichen Eigenschaften übereinstimmt. ALADIN identifiziert sukzessive Unterschiede zwischen dem Originalgraphen und dem von den Studenten modellierten Graphen und erstellt Lösungshilfen, die auf den Unterschieden basieren.

Für Aufgaben der Klasse „Graphen ergänzen“ entfernt ALADIN Bestandteile des Graphen, Knoten oder Kanten oder deren Bezeichnungen aus einem zufällig generierten Graphen. Die Aufgabe besteht darin, die fehlenden Bestandteile wieder zu ergänzen. Da ALADIN die entfernten Bestandteile und ihre Position im Graphen speichert, kann es auf ihrer Basis Lösungshilfen leisten.

Von einer Aufgabenklasse kann es mehrere Aufgabentypen geben, wie z. B. von der Aufgabenklasse „Graphen modellieren“ die Aufgabentypen „Datenflussdiagramm modellieren“ und „Klassendiagramm modellieren“. Von einem Aufgabentyp kann ALADIN zufallsbasiert beliebig viele konkrete Aufgaben generieren.

5 Ausgewählte Beispiele für in ALADIN realisierte Aufgabentypen

Die in diesem Abschnitt vorgestellten Aufgabentypen stehen u. a. derzeit in ALADIN zur Verfügung.



(a) Hinweis zur Aufgabenstellung

(b) Kennzeichnung fehlerhafter Eingaben

(c) Automatische Ergänzung von Null-Werten

(d) Automatisches Lösen

Abb. 2: Lösungshilfen zur Stücklistenauflösung

5.1 Aufgabentyp „Stücklistenauflösung aufgrund von Gozintographen“

Ein Gozintograph stellt den Aufbau von Endprodukten aus Baugruppen, Einzelteilen und Rohstoffen dar. ALADIN unterstützt drei Verfahren zur Stücklistenauflösung. Bei einem überführen die Bearbeiter den Gozintographen G in eine Adjazenzmatrix D (Direktbedarfsmatrix). Alsdann berechnen sie die zu D inverse Matrix I und subtrahieren I von der Einheitsmatrix E, woraus sich die sogenannte Gesamtbedarfsmatrix G ergibt. Zuletzt multiplizieren die Bearbeiter G mit dem Vektor P des ebenfalls generierten Primärbedarfs an Endprodukten und erhalten den Vektor des (Brutto)sekundärbedarfs S an Einzelteilen bzw. Rohstoffen. Beim Üben mit einer zufällig generierten Aufgabe kann ALADIN zu jedem der Schritte Lösungshilfen geben, die Korrektheit von Teillösungen bewerten und korrekte Teillösungen angeben. Abbildung 2 zeigt besagte Lösungshilfen. Der Aufgabentyp „Stücklistenauflösung“ gehört zur Aufgabenklasse „Berechnung in Graphen“, von denen ALADIN auch noch die Aufgabentypen „Geointerpolation von Messwerten“ und „Anwendung des Dijkstra-Algorithmus“ enthält.

5.2 Aufgabentyp „Netzplantechnik“

Ein Netzplan ist ein graphisches Modell zur Abbildung einer Liste an Vorgängen, ihrer jeweiligen Vorgänger und Vorgangsdauern und wird in der Projektplanung verwendet. In ALADIN abgebildete Lösungswege zur Netzplantechnik sind PERT, MPM und CPM sowie die Darstellung mittels Gantt-Diagramm. Die Bearbeiter modellieren zuerst den Netzplan, einen Graphen, und bestimmen sodann die kritischen Pfade in ihm. Der Aufgabentyp Netzplantechnik ist ein Aufgabentyp der Klasse „Graphen modellieren“.

5.3 Aufgabentyp „SQL-Abfragen“

Die Formulierung von Datenbankabfragen nach dem SQL-Standard ist eine häufige und unverzichtbare Aufgabe in Datenbank-Kursen. Der Aufgabentyp „SQL-Abfragen“ zeigt die Grenzen einfacher computergenerierter Aufgaben auf: Bei Aufgaben zur Stücklistenauflösung und zur Netzplantechnik ist es noch vertretbar, abstrakt von „Produkt A“ oder „Vorgang 3“ zu sprechen, aber bei Datenbankabfragen ist es wünschenswert oder gar erforderlich, in der Aufgabenstellung die Tabellen, ihre Felder und die Fremdschlüsseleinschränkungen (Foreign-Key-Constraints) inhaltlich sinnvoll zu bezeichnen. Schließlich besteht die eigentliche Herausforderung bei Aufgaben zum Erstellen von Datenbankabfragen gerade darin, eine Fragestellung aus der realen Welt („Finde die Hersteller aller verderblichen Produkte, die aktuell auf Lager liegen!“) in eine abstrakte SQL-Abfrage umzuwandeln. Deshalb wurde entschieden, für den Aufgabentyp „SQL-Abfragen“ nicht Datenbanken mit bedeutungsleeren Bezeichnungen zu generieren, sondern mehrere frei zugängliche Datenbanken zu verwenden, die verschiedene Anwendungsbereiche abdecken, wie z. B. Flugbuchung, ERP-System und Arztpraxis. Beim Lösen einer entsprechenden Aufgabe wählen die Bearbeiter nun, für welche dieser Datenbanken sie eine Abfrage erstellen wollen. Die Komplexität der Abfrage kann anhand der Abfragebestandteile (WHERE-Bedingung, HAVING-Klausel, Aggregrationsfunktion etc.) und der Art und der Anzahl der Joins parametrisiert werden. ALADIN generiert und präsentiert die Aufgabe in natürlicher Sprache. Zur Generierung der Aufgabe verwendet ALADIN vordefinierte Textschablonen.

Patient				
ID	Name	Vorname	Geburtsdatum	Geschlecht
0	Mustermann	Max	01.01.2000	m
1	Decker	Dirk	31.12.1999	m
2	Räubertochter	Ronja	03.02.1952	w
3	Lustig	Lea	04.05.1965	w

Patientenzustand			
ID	PatientenID	Status	Erfassungsdatum
0	0	Genesen	14.04.2020
1	1	Geimpft	01.06.2021
2	2	Geimpft	21.08.2021
3	3	Infiziert	05.12.2020
4	1	Infiziert	01.01.2022

1 n

Abb. 3: Ausschnitt aus relationalen Datenbanktabellen

Für die Datenbank aus Abbildung 3 wäre z. B. eine sinnvolle Abfrage "*Welche Patienten wurden trotz Impfung infiziert?*". ALADIN ist zwar in der Lage, die dieser Fragestellung zugrunde liegende SQL-Abfrage zu generieren, jedoch ist die Formulierung der Abfrage in natürlicher Sprache aufgrund der Anwendung der Textschablonen noch zu umfangreich und orientiert sich stark an der generierten SQL-Abfrage, wie Abbildung 4 verdeutlicht.

ALADIN reduziert den Aufwand für die Korrektur, indem es für bekannte Fehlerklassen [TSV18] gezielt Lösungshilfen leistet. Die Ausdrucksstärke von SQL, welche äquivalente

Bilde die Schnittmenge, welche die korrespondierenden Einträge der beiden Tabellen „Patient“ und „Patientenzustand“ enthält. **Gib die Spalten „Name“ und „Vorname“ aus.** Es sollen nur Daten ausgegeben werden für die „Zustand“ ‘Infiziert‘ ist und die Ausprägungen von „ID“ in einer Untermenge liegen, für die „Zustand“ ‘Geimpft‘ ist und „Erfassungsdatum“ kleiner ist als **„Erfassungsdatum“ der Obermenge.**

```
SELECT p.Name, p.Vorname FROM Patient AS p
JOIN Patientenzustand AS pz ON p.ID = pz.PatientenID
WHERE pz.Status = 'Infiziert'
AND pz.PatientenID IN
        (SELECT PatientenID FROM Patientenzustand
        WHERE Status = 'Geimpft'
        AND Erfassungsdatum < pz.Erfassungsdatum);
```

Abb. 4: Zuordnung von SQL-Abfragebestandteilen zu Textschablonen

Abfragen erlaubt, schränkt ALADIN nicht ein, da es die Korrektheit der Lösung anhand der Ergebnismenge der SQL-Abfrage verifiziert. Der Aufgabentyp „SQL-Abfragen“ ist ebenfalls ein Aufgabentyp der Klasse „Graphen modellieren“, wobei nicht das Datenbankschema, sondern der Abstrakte Syntaxbaum (Abstract Syntax Tree, AST) einer Abfrage modelliert wird.

6 Technische Realisierung von ALADIN

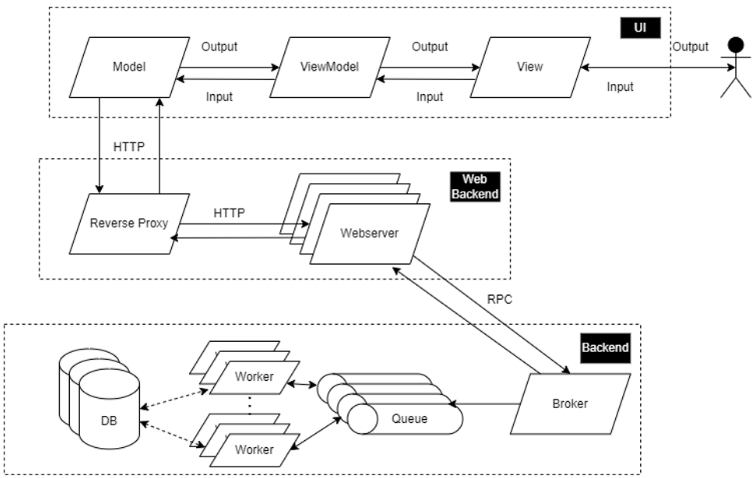


Abb. 5: Technische Realisierung von ALADIN

Bei ALADIN handelt es sich um eine quelloffene Software[4]. Ihre technische Realisierung und das Zusammenspiel ihrer Komponenten stellt Abbildung 5 dar. Das Frontend von ALADIN folgt dem MVVM-Entwurfsmuster (Model View Viewmodel), das die Umsetzung des Event-Sourcing vereinfacht, welches die Aufzeichnung und Wiedergabe von Nutzerinteraktionen, ihre nachträgliche Analyse und die Interpretation der deklarativen Beschreibung der Bedienelemente ermöglicht. Die Kommunikation zum Backend erfolgt über ein API-Gateway, welches die Anfragen an mehrere Webserver verteilt, die mittels RPC (Remote

[4] https://github.com/plc-dev/aladin

Procedure Calls) Events auslösen, die an einen Event-Broker im Backend kommuniziert werden. Das Backend selbst ist als ereignisbasierte Microservice-Architektur realisiert, wobei jeder Aufgabentyp eine eigene Queue darstellt. So kann zur Laufzeit die Anzahl der benötigten Worker erhöht werden, sofern mehr Events in den korrespondierenden Queues auflaufen. Die dargestellte Architektur erlaubt es, dank dynamischer Skalierbarkeit eine große Anzahl nebenläufiger Anfragen zu bearbeiten. Zudem ermöglichst das Publish-Subscribe-Muster das Anstoßen langlaufender asynchroner Prozesse.

7 Zusammenfassung und Ausblick

ALADIN generiert Übungs- und Prüfungsaufgaben und bietet sie Studenten digital dar, so dass sie die Aufgaben selbständig, zu beliebiger Zeit, an beliebigem Ort und in passendem Schwierigkeitsgrad lösen können. ALADIN unterstützt das Lehrpersonal bei Stellen von Übungs- und Prüfungsaufgaben, bei der Korrektur der Lösungen und der Betreuung der Studenten während der Lösung der Aufgaben. Das Aufzeichnen, Teilen und Abspielen von Lösungsversuchen ermöglicht die asynchrone Interaktion zwischen Studenten und Lehrpersonal. Für die Zukunft ist die Realisierung weiterer Aufgabentypen aus weiteren Fachgebieten geplant.

Eine interessante Herausforderung besteht darin, graphische Modelle und Aufgaben zu ihrer Modellierung zu generieren, bei denen die inhaltliche Bedeutung von Knoten- und Kantenbeschriftungen relevant ist. Bei Aufgaben zu UML-Strukturdiagrammen wird aktuell untersucht, wie Informationen etwa zu den Relationen „ist enthalten in“ oder „ist ein spezieller Fall von“ aus Ontologien oder der WordNet-Datenbank [Fe98] entnommen werden können. Als schwieriger erweist sich das Generieren inhaltlich sinnvoller Verhaltensdiagramme, wie z. B. BPMN-Diagramme. Hier ist vorgesehen, ähnlich wie bei den in Abschnitt 5.3 diskutierten SQL-Datenbanken öffentlich vorhandene Modelle (etwa von [Te18]) zu nutzen.

Literaturverzeichnis

[Fe98] Fellbaum, Christiane, Hrsg. WordNet: An Electronic Lexical Database (Language, Speech, and Communication). The MIT Press, 1998.

[Te18] Technische Hochschule Mittelhessen: , Referenzmodelle für GP, verfügbar auf http://wi-wiki.de/doku.php?id=funktionsbereiche, 2018.

[TSV18] Taipalus, Toni; Siponen, Mikko T.; Vartiainen, Tero: Errors and Complications in SQL Query Formulation. ACM Trans. Comput. Educ., 18(3):15:1–15:29, 2018.

Kollaboratives Service Blueprinting zum Erlernen von betriebswirtschaftlichen Sachverhalten

Alexander Rachmann[1], Thomas Polinski[2]

Abstract: In zwei Lehrveranstaltungen wird die Modellierungssprache Service Blueprinting eingesetzt, um verschiedene betriebswirtschaftliche Sachverhalte zu vermitteln. Die Studierenden modellieren kollaborativ verschiedene Artefakte, gemeinsamer Nenner ist die Nutzung von Service Blueprints. Anhand von Anforderungen wird diskutiert, warum Service Blueprinting eine sinnvolle Wahl als Modellierungssprache ist. Beide Lehrveranstaltungen werden vorgestellt und ein Ausblick auf die weitere Entwicklung der Veranstaltung gegeben.

Keywords: Service Blueprinting; Kollaboration; Miro

1 Einleitung

In zwei Lehrveranstaltungen am Fachbereich „Technologie“ der CBS Cologne Business School werden Service Blueprints eingesetzt um Studierende bestimmte betriebswirtschaftliche Sachverhalte zu verdeutlichen. Es werden die Modellierungssprache Service Blueprints und das Kollaborationsboard Miro benutzt.

In Kapitel 2 wird das Setting der Lehrveranstaltung als Ausgangspunkt der weiteren Lehrkonzeption vorgestellt. In Kapitel 3 werden die daraus resultierenden Anforderungen an die Modellierungssprache und die verwendete Software definiert. In Kapitel 4 werden Service Blueprints vorgestellt und verdeutlicht, wieso Service Blueprints die Anforderungen sinnvoll erfüllen. In Kapitel 5 wird der didaktische Einsatz von Service Blueprints vorgestellt. Kapitel 6 schließt das Paper mit einem Fazit und Ausblick.

2 Lehrveranstaltungen

Die Modellierungsübung wird derzeit in zwei Veranstaltungen eingesetzt. Beiden Lehrsituationen ist gemein, dass die Zielgruppen Studierende mit heterogenem Bildungshintergrund in frühen Semestern sind. Die Themen drehen sich um die betriebswirtschaftliche Einordnung von komplexen Sachverhalten:

[1] CBS Cologne Business School, Fachbereich Technologie, Hammer Landstraße 89, 41460 Neuss, Deutschland, A.Rachmann@cbs.de
[2] LUKAS-ERZETT GmbH & Co. KG, Gebrüder-Lukas-Straße 1, 51766 Engelskirchen, Deutschland, Thomas.Polinski@lukas-erzett.de

- Im Bachelorstudiengang Wirtschaftsinformatik findet ím zweiten Semester eine Überblicksvorlesung für die Vertiefung „Smart City“ statt. Hierbei werden Teilaspekte des Smart-City-Gedankens bearbeitet. Die Studierenden entwerfen eine Citizen Journey und setzen diese in Verbindung mit Technologien, Open Data und Services, die in Verbindung zum Smart-City-Gedanken stehen. Das genauere Setting wird in Kapitel 5.1 beschrieben.
- Im Masterstudiengang Digitales Projektmanagement belegen die Studierenden im zweiten Semester das Modul „Integrierte Anwendungssysteme“, bestehend aus den zwei Veranstaltungen „Außenwirksame Informationssysteme und E-Commerce“ und „Supply Chain“. Der Masterstudiengang steht einer großen Bandbreite von Studierenden offen; es nehmen Bachelorabsolventen aus der Betriebswirtschaftslehre, der Informatik bzw. “Bindestrich”-Informatiken und dem Ingenieurwesen teil. Der Kurs dient auch als Grundlegung um alle Bildungshintergründe auf denselben Stand zu bringen. Das genauere Setting wird in Kapitel 5.2 beschrieben.

3 Anforderungen

In der Konzeption der unten beschriebenen Lehrveranstaltungen wurde frühzeitig entschieden, dass die Studierenden eine eigene Fallstudie aufbauen sollen, die sich über mehrere betriebswirtschaftliche Aspekte zieht und die in einem Team erarbeitet werden soll. Ebenso wurde früh entschieden, dass keine rein textuelle Fallstudie, wie z.B. in einer Hausarbeit, erarbeitet werden soll. Es wurde daher eine Modellierungssprache gesucht, die den folgenden Anforderungen gerecht wird:

- A1: Die Modellierungssprache soll einfach im Erlernen sein, so dass auch Studierende mit keinen Kenntnissen in der Modellierung relativ schnell agieren können.
- A2: Die Modellierungssprache soll flexibel in ihrer Anwendung sein, ggf. anpassbar für verschiedene Lehrsituationen.
- A3: Die Studierenden sollen kollaborativ modellieren, sowohl in Präsenz-, Hybrid- der Online-Veranstaltungen. Ein Zugang zur Software soll über einen Freemium-Zugang o.ä. verfügbar sein, so dass minimale Kosten entstehen.

4 Wahl der Modellierungssprache und Plattform: Service Blueprinting und Miro

4.1 Erfüllung der Anforderungen

Für die Lehre wurden Service Blueprints als Modellierungssprache ausgewählt und als Plattform Miro. Diese Wahl ergibt sich aus der Erfüllung der Anforderungen wie folgt:

- A1: In die engere Wahl wurde Business Process Model and Notation (BPMN) und Service Blueprints genommen. Service Blueprints sind bereits per se relativ schnell zu verstehen. BPMN bietet deutlich mehr Symbole an als Service Blueprint; den Studierende könnte nur eine begrenzte Auswahl von „erlaubten“ Symbolen vorgestellt werden, so dass keine lange Einarbeitungszeit in die Sprache erfolgen muss / kann.
- A2: BPMN und Service Blueprints sind beide leicht anpassbar. Service Blueprints tragen schon eine Differenzierung von betriebswirtschaftlichen Perspektiven in sich (siehe Tabelle 1). Dies macht die Anpassung an die Lehrinhalte noch einfacher als in BPMN – zumindest war dies der Eindruck der Lehrenden. Die Wahl fiel daher auf Service Blueprints.[3]
- A3: Als Modellierungssoftware wurde die Kollaborationsplattform Miro [Mi] ausgewählt. Einerseits wurde diese Plattform schon von anderen Wissenschaftler für die Modellierung genutzt [Le20], andererseits wurde sie in der Covid-19-Pandemie recht bekannt [La22]. Insbesondere der zweite Grunde führte dazu, dass diese Plattform den Vorzug bekam, da die meisten Studierende für diese Plattform schon Vorkenntnisse vorweisen konnten. Sollte sich die Gratisnutzung nicht mehr ergeben oder die Anforderungen aus sonstigen Gründen nicht mehr erfüllt sein, wäre ein Wechsel auf eine andere Plattform gut denkbar, da die Funktionalität auf verschiedenen Plattformen im Kern relativ ähnlich sind.

4.2 Service Blueprinting

Service Blueprinting ist eine Methode zur Modellierung von Dienstleistungen. Ursprünglich erdacht in den 1980er Jahren ([Sh82], [Sh84]), sind viele Varianten, sowohl syntaktisch wie auch semantisch, in Gebrauch (z.B. [BOM08], [GHS11], [LK14], [MML10]). Ein einfacher Service Blueprint in originaler Notation ist in Abbildung 1 abgebildet. Zu der „Line of Visibility“ sind verschiedene Linien hinzugekommen, nach Leimeister [Le20] sind die typischen Linien in Tabelle 1 dargestellt.

Der Einsatz von Service Blueprints soll den Studierenden in den folgend beschriebenen Settings nicht Kenntnisse in der Modellierungssprache vermitteln. Vielmehr soll die Modellierungssprache den Studierenden eine niedrigschwellige Ausdrucksform zur Verfügung stellen, mit der betriebswirtschaftliche Sachverhalte übersichtlich dargestellt werden können.

[3] Es wurde bereits dargestellt, dass ein Service Blueprint relativ leicht in ein Prozessmodell mit BPMN überführbar ist [MML10]. Es wäre also durchaus eine Vertiefung in einer Lehreinheit denkbar, in der ein Service Blueprint weiter nach BPMN überführt würde. Das hätte den didaktischen Effekt, dass Studierende niedrigschwellig modellieren können und im Verlauf der Veranstaltung ihren Modellentwurf in eine mächtigere Sprache übersetzen können.

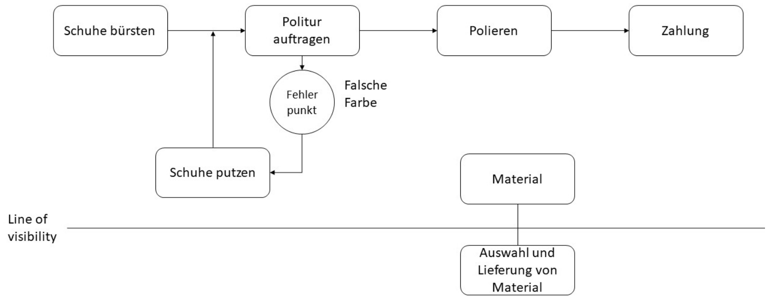


Abb. 1: Service Blueprint „Schuhe polieren“, vereinfacht nach Shostack [Sh82]

Line of . . .	**Aktivitäten über der Linie**	**Aktivitäten unter der Linie**
Interaction	Customer	Provider
Visibility	Visible Provider	Invisible Provider
Internal Interaction	Backstage Provider	Internal
Order Penetration	Support	Potential
Implementation	Preparation	Facility

Tab. 1: Linien und Aktivitäten in Service Blueprints nach Leimeister [Le20]

5 Anwendung in Lehrveranstaltungen

5.1 Veranstaltung „Überblick Vertiefung Smart City“

Die Studierenden bilden Gruppen von drei bis fünf Personen und starten mit der Beschreibung einer Citizen Journey durch die Customer Actions. Während Service Blueprints darauf ausgelegt sind, dass sie von einer Firma angeboten werden, stehen die Studierenden vor der Situation, dass ihre Journey mehrere Dienstleistungs- und Produktanbieter umfasst. Diese Differenzierung wird der Studierenden transparent gemacht, es sind bisher keine Verständnisprobleme diesbezüglich an den Dozenten getragen worden.

Jeder weitere Aspekt von Smart City wird in einer der unteren Bereiche eingetragen. Die Studierenden sind frei, in welchen Bereichen sie welchen Aspekt eingetragen. Typischerweise werden Dienstleistungen in den Bereich Visible Provider eingetragen; Technologien werden gleichmäßig verteilt in Visible Provider, Backstage Provider, Support und Preparation eingetragen. Datenquellen werden typischerweise in Support oder Preparation eingetragen.

5.2 Veranstaltung „Außenwirksame Informationssysteme und E-Commerce“

Die Prüfung wird als Modulprüfung abgelegt und umfasst ein Referat, in dem eine Fallstudie über ein erfundenes Unternehmen bzw. Produkt dargestellt wird. Teil der Fallstudie sind u.a.

eine Customer Journey, ein Service Blueprint und eine Supply Chain. Die Vorbereitung auf das Referat findet in den Veranstaltungen statt.

Im Rahmen der Veranstaltung „Außenwirksame Informationssysteme und E-Commerce“ bilden die Studierenden Gruppen von ca. drei Personen und erfinden ein Unternehmen und Produkt, welches sie in ihrer Fallstudie bearbeiten möchten. Im zweiten Schritt modellieren sie eine Customer Journey mit Hilfe der Customer Actions des Service Blueprints. Die Linien der Service Blueprints werden weitestgehend im Standard benutzt. Der Fokus der Veranstaltung liegt auf der Nutzung von außenwirksamen Informationssystemen; dementsprechend werden solche Systeme bzw. deren Prozesse typischerweise in den Aktivitäten Support und / oder Preparation modelliert.

In der Veranstaltung „Supply Chain“ werden produktionsbezogene Abläufe analysiert. Dabei wird die Produktion mit dem Service Blueprint verbunden, z.B. durch eine auftragsbezogene Fertigung.

In Abbildung Abb. 2 ist ein kommentierter Auszug aus einer Referatsprüfung zu sehen. In der obersten Zeile sind in den trapezförmigen Objekten die Phasen der Customer Journey zu sehen. Direkt darunter mit einem Post-It und Bildern sind die Customer Actions / die Customer Journey zu sehen. Hierunter sind die Aktivitätsbereiche des Service Blueprints zu erkennen. Deutlich zu erkennen ist unten die Supply Chain und, in rot markiert, die Verbindung der Customer Actions mit der Supply Chain.

5.3 Rückblick auf die Lehrveranstaltungen

Die Veranstaltung „Außenwirksame Informationssysteme und E-Commerce“ wurde einmal nach dem Muster durchgeführt, aber noch nicht formal evaluiert. Die Rückmeldung der Studierenden zu der Modellierungssituation war positiv.

Die Veranstaltung „Überblick Vertiefung Smart City“ dauert derzeit noch an. Eine explizite Bewertung der Studierenden fehlt noch bzw. konnte noch nicht methodisch korrekt erhoben werden.

In beiden Veranstaltungen wurden die Benennung der „Lines“ intensiv diskutiert. Es kann vermutet werden, dass eine Bezeichnung in deutschen Sprachgewohnheiten einfacher zu vermitteln wäre.[4]

[4] Z.B. benutzt Bitner et al. [BOM08] den Begriff der “Physical Evidence“: Die wortwörtliche Übersetzung als „physikalischer Beweis“ / „physikalische Evidenz“ erscheint unverständlich. In der Diskussion mit den Studierenden wurde der Begriff „Vorliegende Produkte, Produktbestandteile oder Produktinformationen“ als verständlicher empfunden; gleichzeitig erscheint diese Benennung aber zu lang.

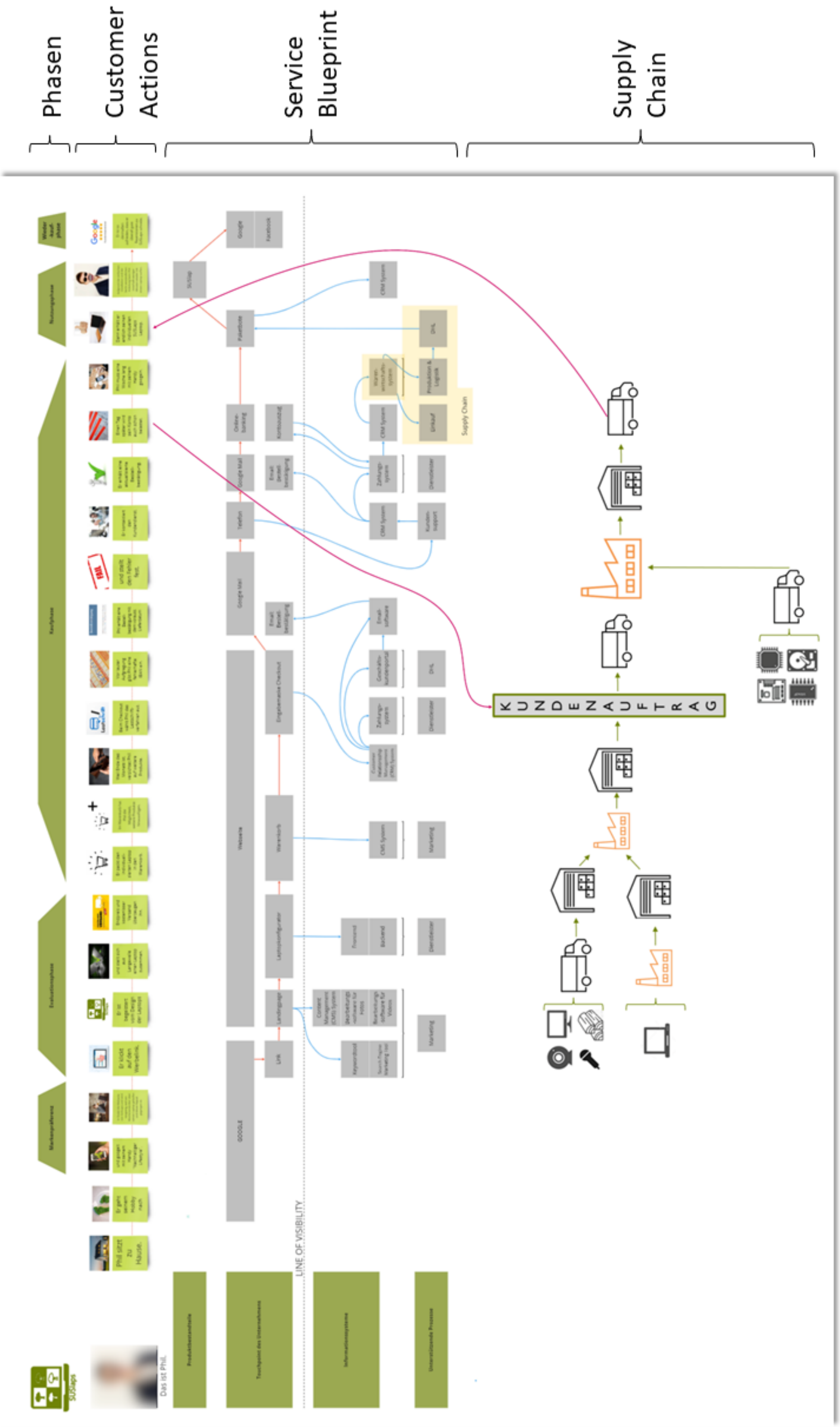


Abb. 2: Prüfungsergebnis Customer Journey, Service Blueprint und Supply Chain

6 Fazit und Ausblick

In diesem Paper wurde ein Lehransatz beschrieben, in dem Service Blueprinting als Modellierungssprache verwendet wird um mit Studierenden betriebswirtschaftliche Sachverhalte darzustellen und zu diskutieren. Der Schwierigkeitsgrad dieser Übung ist aus Sicht der Dozenten gering; die syntaktische Korrektheit der Modellierungssprache wird nicht tiefgehend kontrolliert. Die Übung ist als kreativer Akt zu verstehen, durch den die Studierende ihre Kenntnisse erweitern.

Die Stärke des Ansatzes ist seine Offenheit; Studierende können sich interessengeleitet bewegen und bleiben trotzdem in einem Framework, welches den Vergleich zu anderen Studierenden ermöglicht. Gleichzeitig bietet das Service Blueprinting einen guten Startpunkt um sich in komplexere Ablaufmodellierung zu vertiefen, z.B. durch eine Kombination mit BPMN.

Die Übung geht davon aus, dass die teilnehmenden Studierenden offen, fähig und gewillt sind, die Übung kreativ auszuführen. Studierende, die sich in einem offenen Bereich mit wenig definierten Grenzen nicht selbstständig bewegen können, werden keine gute Resultate vollbringen. Ähnlich dazu werden Studierende, die sich für die inhaltlichen Themen nicht interessieren, Schwierigkeiten haben sich kreativ in den Blueprints auszudrücken.

Der Ansatz ist von der Nutzbarkeit von SAAS-Software abhängig, d.h. damit auch von der Verfügbarkeit von Internet mit entsprechender Übertragungsbandbreite, Hardware auf Clientseite und ausreichender Erfahrung mit webbasierten Anwendungen.

Es ist angedacht die Veranstaltungen weiterhin nach dem oben beschriebenen Schema durchzuführen. Wichtige Veränderungen sind allerdings:

1. Besseres Eindeutschen der Kernbegriffe, so dass weniger Einarbeitungszeit benötigt wird.
2. Das Nutzen von besseren Vorlagen in Miro, so dass wenig Zeit für die rein syntaktische Arbeit stattfinden muss. Ggf. sollte ein eigene Vorlage erstellt werden oder eine andere Plattform genutzt werden.

Erste methodisch korrekte Evaluationen der Veranstaltungen können weiteren Aufschluss geben.

7 Danksagung

Dank gilt den Studierenden Alena Köhler, Philipp Engelfried und Sebastian Seiler für die Nutzungserlaubnis für Abbildung 2.

Literatur

[BOM08] Bitner, M. J.; Ostrom, A. L.; Morgan, F. N.: Service blueprinting: a practical technique for service innovation. California management review 50/3, S. 66–94, 2008.

[GHS11] Gersch, M.; Hewing, M.; Schöler, B.: Business Process Blueprinting – an enhanced view on process performance. Business Process Management Journal 17/5, S. 732–747, 2011.

[La22] Lawrence, L.: Armed with cash, Miro's taking on the meeting, https://www.protocol.com/workplace/miro-money, Jan. 2022.

[Le20] Leimeister, J. M.: Dienstleistungsengineering und -management. Springer-Verlag, 2020.

[LK14] Lim, C.-H.; Kim, K.-J.: Information service blueprint: A service blueprinting framework for information-intensive services. Service Science 6/4, S. 296–312, 2014.

[Mi] Miro, URL: www.miro.com.

[MML10] Meis, J.; Menschner, P.; Leimeister, J. M.: Modellierung von Dienstleistungen mittels Business Service Blueprinting Modeling. In: Dienstleistungsmodellierung 2010. Springer, S. 39–64, 2010.

[Sh82] Shostack, G. L.: How to design a service. European journal of Marketing/, 1982.

[Sh84] Shostack, L.: Designing services that deliver. Harvard business review 62/1, S. 133–139, 1984.

Message from the Modellierung'22 Tools & Demos Chairs

Simon Hacks[1] Dominik Bork[2]

Modeling is only efficiently applicable with proper modeling tool support. The conceptualization and implementation of modeling tools is consequently a long-lasting but still ongoing endeavor of the information systems engineering and conceptual modeling research communities.

At Modellierung 2022, a dedicated track was aimed for the newest modeling tools. The aim of this track was to present modeling tools that have been and are being developed by the modeling community. The Call for Papers particularly invited submissions of tools that:

- implement new interfaces for modeling tools (e.g., tangible user interfaces, virtual reality environments, web modeling tools, mobile interfaces),
- support collaborative (inter-organizational) modelling, offer novel forms of modeling support,
- are developed or offered with new technologies and platforms, or
- technically implement new modeling methods.

Each submitted tool paper underwent rigorous reviewing. Three anonymous reviewers from the Tools & Demos Program Committee were assigned and assessed the papers. Eventually, the following four tools have been accepted and were presented at the conference:

In the paper entitled *The Simplified Platform, an Overview* Mark A.T. Mulder et al. introduce a novel, cloud-based platform for modeling and metamodeling called *Simplified*. The paper describes the platform's architecture, features, and visualizations, and also reports on the possibilities to develop support for new modeling languages.

The paper *Beyond Low Code Platforms: The XModeler^{ML} — an Integrated Multi-Level Modeling and Execution Environment* by Ulrich Frank et al. introduces the multi-level modeling platform XModeler$^{\mathrm{ML}}$. The paper proposes the architecture of the tool and exemplifies its use with three use cases. A principal feature of XModeler$^{\mathrm{ML}}$ is the support for multi-level modeling languages that, on the XModeler$^{\mathrm{ML}}$ editor side, allow the creation of models that feature an arbitrary number of classification levels.

[1] University of Southern Denmark, The Maersk Mc-Kinney Moller Institute, Odense, Denmark
shacks@mmmi.sdu.dk
[2] TU Wien, Business Informatics Group, Favoritenstrasse 11, 1040 Vienna, Austria dominik.bork@tuwien.ac.at

The latest developments on an ADOxx-based modeling tool for the Multi-Perspective Enterprise Modeling (MEMO) modeling method are reported in the paper *MEMO4ADO: A Comprehensive Environment for Multi-Perspective Enterprise Modeling* by Alexander Bock and colleagues. The paper shows how the different MEMO languages have been realized and integrated with ADOxx and reports a brief use case.

Eventually, Yorck Zisgen and colleagues report on a tool which allows the generation of synthetic (sensor) event logs for conducting process mining. Their paper, entitled *Modellierungsumgebung zur Erzeugung synthetischer Ereignisprotokolle für das Process Mining*, introduces the tool and showcases its application in two cases, one regarding processes in the smart homes domain, the other regarding processes in the domain of hospitals.

Program Committee

We want to thank all members of the program committee for their valuable feedback on the submitted papers.

- Syed Juned Ali
- Hans-Georg Fill
- Jonas Friederich
- Antonio Garmendia
- Jens Gulden
- Stephan Kühnel
- Ana Nicolaescu
- Alixandre Santana
- Andreas Steffens
- Benjamin Ternes

Simon Hacks and Dominik Bork

The Simplified Platform, an Overview

dr. ir. Mark A.T. Mulder [1], Rick Mulder, Fiodor Bodnar, Mirjam van Kessel, Jorge Gomez Vicente

Abstract:

Simplified is a new approach to modelling and meta-modelling. This platform is a result of experience with a previous research tool for modelling Design and Engineering Methodology for Organisations (DEMO). It's cloud based development makes it suitable for research and business applications. The configurable notations, flexible user interface, and real-time transformation and visualisations makes the platform adaptable and understandable for every stakeholder.

Keywords: Modelling; Meta-modelling; Collaboration; Enterprise Engineering

1 Introduction

The history of the Simplified modelling platform started with the research project towards the PhD 'Enabling the automatic verification and exchange DEMO models' [Mu22]. The DEMO [Di06, DM20] method is a core method (based on a theoretically founded method*ology*) within the discipline of Enterprise Engineering (EE) [Di13]. The DEMO methodology focuses on the creation of so-called *essential* models of enterprises. The latter kind of models aim to capture the organisational essence of an enterprise by leaving out (as much as possible) details of the socio-technical implementation. The organisational essence is then expressed primarily in terms of the actor roles involved, as well as the business transactions [Re96] (and ultimately in terms of speech acts [Ha81]) between these actor roles. More specifically, an essential model comprises the integrated whole of four aspect models: the Construction Model (CM), the Action Model (AM), the Process Model (PM) and the Fact Model (FM).

As part of this work, requirements for tooling were defined in order to find the most suitable tool for DEMO modelling. Among other requirements, the tool would have to support the essential model of an organisation. Unfortunately, no suitable tooling was found that could support all the requirements.

The lack of good tooling for demo modelling prompted us to start the development of a new tool. This initiative resulted in the creation of *Plena*, an add-on to an existing Sparx modelling software. This add-on supported the import of the interview lines along with the modelling of all aspect models. During the further development of Plena, it became apparent

[1] TEEC2 BV, The Netherlands markmulder@teec2.nl

that the underlying Sparx modelling software was increasingly becoming a limiting factor which impeded the implementation of the desired features. Among other features, we wanted to support collaborative design for online team modelling, multiple notations, API and white label UI integration that would allow customers to apply their own corporate design language to the UI, and multiple languages all while applying state of the art development methods.

Considering the limitations of the Sparx software we have decided to develop a standalone cloud based modelling platform from scratch, which would allow us to implement all the required features. The resulting platform is called the *Simplified modelling platform*; or *Simplified* in short form.

It is worth mentioning that [GBDV20] concluded that our previous tool Plena scored quite high on its usability. However, Plena lacked when it came to usability criteria 'U3' *'Ease of learning'* and 'U6' *'Reduction of excess'*. This has also been addressed in Simplified and will be later discussed in more detail.

This paper describes the used architecture and the implemented interfaces. After that a series of features are introduced and respective benefits for research, business users of organisations, and modellers in general are introduced. We also provide a brief overview of all technologies used in the development and production environments. Finally, we conclude with a short summary and the current state of affairs of this platform.

2 Architecture

The platform consists of total of six layers, divided in two servers: application server (involving the layers interface, message, process, cache, and persistence), and finally database server (database layer). The server architecture is visualised in Fig. 1. The layers are visualised in Fig. 2

First, the interface layer consists of two interfaces for accessing the platform. The REST/JSON API is the simplest interface, allowing the retrieval of the generic public information that requires no authentication. No modelling information can be exchanged using this API. This information includes the installed public notations (e.g. ArchiMate, DEMO, BPMN) and the creation of free accounts. The other interface, that facilitates the access to the platform is an authenticated asynchronous web socket messaging interface that can not only receive and handle messages, but also can broadcast to all relevant connected users. This interface can be used by authorised developers to build their own user interaction. Next, the message layer handles messages either by using the process layer for partial message processing, or by forwarding the request to the distributed cache in the cache layer. The cache layer contributes to fast retrieval of information and distributes the load between multiple servers. This cache layer will, in turn, send data or initially retrieve the information from the persistence layer. The configurable persistence layer will activate the right persistence provider that triggers the correct driver to communicate information to the database layer. Lastly, in theory the database cluster realising a database layer can be any database, but at the moment it is limited to the Microsoft SQL server and to the open source MySQL server.

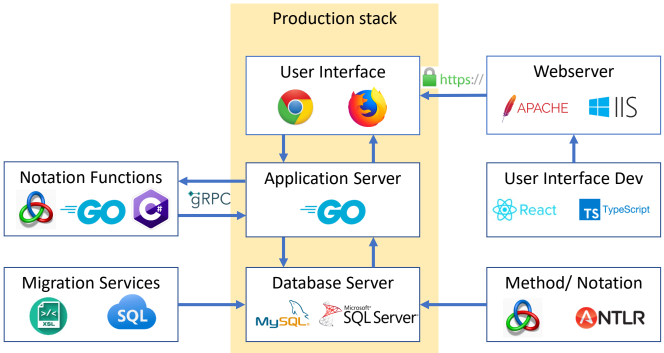


Fig. 1: Architecture of Simplified

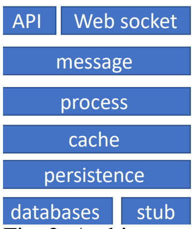


Fig. 2: Architecture Layers

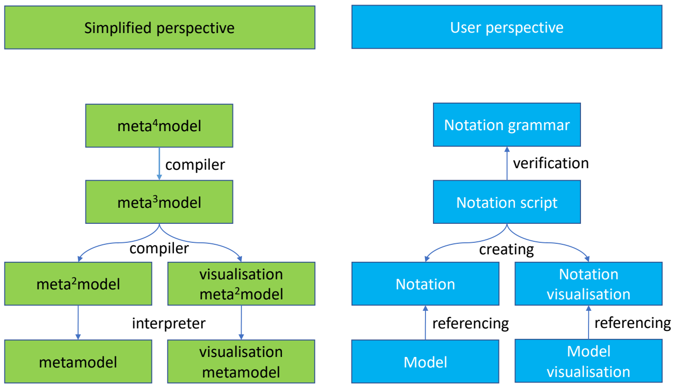


Fig. 3: Notations from different perspectives

Messaging to the asynchronous interface is done in JSON format with a dynamic payload structure, allowing for per message configuration, and developing structures during the lifetime of the platform. This messaging structure is used for the user interface in any form. Other clients can use this structure to interface with the back-end engine for notation specific operations. Every client, that is also a server, can request the platform to forward special operations upon user request. The user request results in a gRpc call to the providing server platform that returns the calculated information. This information is then processed and returned to the client UI for the visualisation.

The modelling part of the back-end is designed to store the model and the metamodel of a notation or methodology. The architecture uses dynamic metamodels that restrict the models on run-time. The notation architecture structure is visualised in Fig. 3. To be able to create a model and a visualisation of that model one has to agree on a notation for both aspects. This results in creating a meta-model. To be able to describe this meta-model we have created a script language that can describe notations. This script language is described in a grammar and the levels of abstraction are reflected in the platform and the transformation between the levels are either interpretation or compilation. Below you can find an extract from the grammar for the 'DEMO 3' notation (the '...' contains a lot more definitions).

```
ScriptVersion01 Notation for DEMO version 3.7
typedef TRSRT ENUM (None, Original, Informational, Documental)...
element "Transaction Kind" TK37 ( ... , "Transaction Sort" TRSRT) ...
connection Executor Executor37e from TK37 to EAR37 ...
diagram "Org... Constr... Diagram" OCD37 contains (... TK37, ...)
table "Transaction Product Table" TPT37 select (x."Identification", ...
visual TKVisual37ocd of TK37 on (OCD37) { ...
fillcolor(255,255,255) // White Filling ...
print(10, 20, 40, 25, "{element.identification}", 0) ... } ...
```

The platform will be available in the middle of June 2022 on https://simplified.engineering/.

3 Features and Benefits

We observe that some of the features mentioned below should not be regarded anymore as distinctive tool features in the current state-of-the-art because many tools on the web have this same set of features [Lu22, Ca22, JG22, Ed22, Go22b, Nu22, Cr22].

Therefore, we conjecture that these features constitute a baseline for the modelling tools. The following items give a summary of the baseline features, but more items could be added outside the scope of this paper.

- Platform: login with username and password, activity logging, activity audit.
- User interface components: repository browser, notation toolbox, showing object properties, search option.
- User interface behaviour: remembering layout preferences per user, moving all docking components.
- Modelling behaviour: automatic saving, available notation elements and connections, drag and drop elements from source to the canvas, naming elements and connections, showing next most logical steps, making notes, moving elements and connections on the canvas (single or groups), resize the elements, start and end connections on any place of the border of an element, making anchors in a connection line, duplicate elements and connections, adding attributes to an element, aligning elements.

Next to these baseline features, Simplified has its own distinctive features described in the next paragraphs.

All users working on the same models, automatically collaborate. Simplified adheres to the concept of optimistic locking, or rather the last save wins. The 'what you see is what you saved' concept ensures that all users have the same view of the model. Given the fact that there can be multiple views of the same model catering the needs of various stakeholders, a change in the underlying model automatically leads to changes in the respective views of the model, keeping them up-to-date. Viewers can also follow changes being applied to a live version of the model from their own viewpoints.

With Simplified one can support one's own modelling methodology and associated notation. The notation can be transformed to a notation script that allows for specifying the elements, connections, visualisations and restrictions for the model. Creating a notation can be done from scratch or one can build on an existing notation. Furthermore, predefined notation scripts for DEMO, BMPN, ArchiMate, VISI and IDEF0 are available on the platform. Other notations such as PetriNet, ER, UML, etc. can be added on request.

Modelling in a multi cultural environment requires a user interface that can be used in multiple languages. Multilinguality is built into the UI and at the moment Simplified supports five languages, but this can easily be extended. Having said that, the model itself should be translatable too. Simplified allows the model and all of its elements to be manually translated to all the required languages in order to present the model to users as comprehensively as possible.

During the development process of the Simplified modelling platform we have taken into account usability requirements [Na12, GBDV20, Mu22]. Firstly, 'ease of learning' is significantly improved when compared to its predecessor Plena as it is no longer an add-on to an existing modelling tool, but is a standalone cloud based modelling environment. As a result, there is no such underlying requirements as installation of software and plugin imports which contribute to the ease of learning. Secondly, for the 'reduction of excess' we have now full control over the UI and what is going to be shown to the user. Moreover, users get a freedom of choice on how their UI layout will look like as all windows within the UI are dockable. For instance, this gives the user a possibility to place a 'repository browser' or 'toolbox' docking window to a desired position on the screen.

In the Architecture section we touched briefly upon the use gRpc [Fo22] for support of the special operations. This interface allows for cross-programming-language development and we have used it to let remote servers subscribe to our server to link to actions in the user interface. gRpc is used to create a lightweight synchronous protocol between servers instead of the asynchronous web socket. gRpc extensions can be used not only in the new models but also in the existing ones. For example, model validation logic can be imported and run against the model to check for its compliance with the underlying notation.

For the launch of the tool in the middle of June 2022 we are considering four standard and one custom licence types to be available to the users of Simplified. Standard licence offering will include the Basic, Standard, Pro, Enterprise, and Research&Education versions [2]. It is worth noting that Basic licence type will provide a free, thus low barrier entry point to the modelling environment with basic functionality to the user. We categorise Basic and Standard for personal, and the rest for professional use.

4 Visualisations

Simplified is a server based solution which supports models and visualisations of those models [Mu22]. Therefore, new technologies that want to connect to those visualisations can do so by using the messaging system. In the end, the visualisation can be summarised by a view with visual elements having a x,y,z coordinate and a size. Those properties are supported and can be extended within the platform. Systems that use views for gamification, mobiles, desktops, AR, VR, or live translations for stakeholders can potentially be connected. We already have a gamification connected to the system. The current modeller is a web page supporting all 2.5D modelling and a white label offering. White labelling helps with fast integration of Simplified platform functionalities in your own environment, leads to reduced investment and decreased time to market on the modelling features.

4.1 Modeller

Simplified Web UI is built of (web) pages in a modular fashion. Every set of features has its own page. Each page is built with modular function blocks which share a state with the other modular function blocks. Each modular block has its functions and a visual representation. Next, the modeller page consists of the visualisations of a modular function block that have a docking possibility within the page. In addition to sharing a 'session' state between some pages, pages have their own states as well. As explained in the next sections, examples of modular blocks are Toolbox, Browser and Properties. The page layout is saved as a session variable, which allows each user to have its own unique layout. Due to the modular nature, it is also possible to have stakeholder layout presets. Other pages that do not have the docking layout, still use the modular setup, which makes it possible to customise each page to specific needs of the stakeholder.

4.1.1 Browser

The browser has 2 modes: a tree mode and a list mode. The tree mode displays the structure of the project, with models at the top. Under them are the folders, which in turn contain elements, diagrams and other folders. Using the switch view button we can switch to the list mode. The list mode displays all elements and diagrams in a list. This also enables 3 extra

[2] https://teec2.nl/products/modelling-platform/

drop down fields where we can filter on elements we want to see in our list. In addition to that, we can have multiple browsers, each in different modes, and link them to specific property dockers. These browsers, each with their own view, will all have the up-to-date data displayed in their own way as shown in Fig. 4.

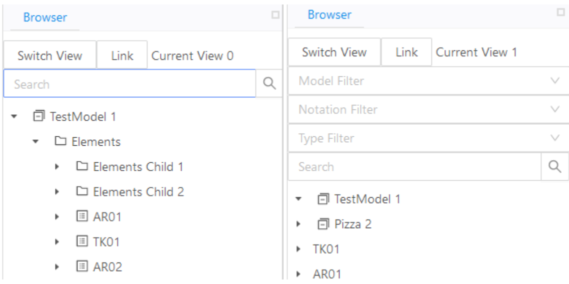


Fig. 4: Different ways to display elements in the browser

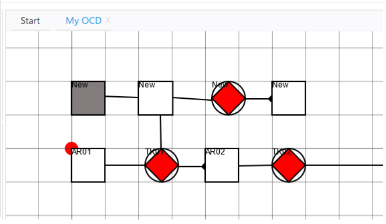


Fig. 5: The diagram view

4.1.2 Diagram

The diagram is arguably the most important part of the modeller. A diagram can be opened from the browser and represents a graphical view or visualisation of a part of the model following the defined viewpoint of the used notation as presented in Fig. 5. Element drawings are requested from the server on demand. When an element cannot be found or it does not exist, it will be displayed as a red cross with a box around it. This way you can still interact with the element with basic functionality, such as moving and scaling. All actions made by users on the diagram are displayed in real time to other people working on the same diagram. This allows for smooth collaboration and modelling together.

4.1.3 Toolbox

The toolbox contains the notations that are available to the user as visualised in Fig. 6. These notations have categories, of which two are standard, namely connections and elements. In a notation we can specify custom categories, which contain a subset of elements. By doing so we can ensure that the user can have quick access to their most important elements. The toolbox can be refreshed, which comes in handy in case of an updated notation or a new notation that is not yet shown in the toolbox. This is especially useful when a co-worker updates a notation and you want to immediately see the new updated graphics for example. Missing drawings of specific elements will be displayed in a similar fashion as in a diagram with a red cross with a box around it.

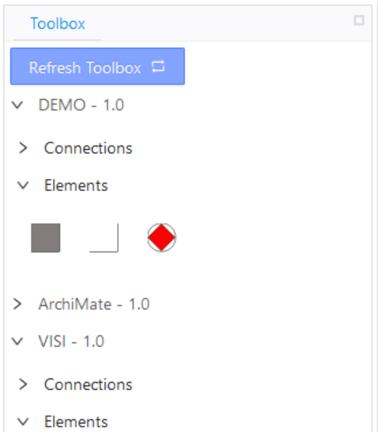


Fig. 6: Toolbox with several notations and their categories

4.1.4 Properties

The property module displays the properties of the selected element as visualised in Fig. 7. When a visual element is selected, in addition to the visual element properties, it also shows a separate tab with the element properties. This property module can be either a global or a linked property module. In case of a global property module it does not matter where an element is selected, its properties will be displayed in the property module. Whereas, with a linked property module, only the properties of the element that is selected in a specific linked browser will be shown. This way the user can have several browsers, each with their own intended usage, such as a broad overview, or a detailed focus.

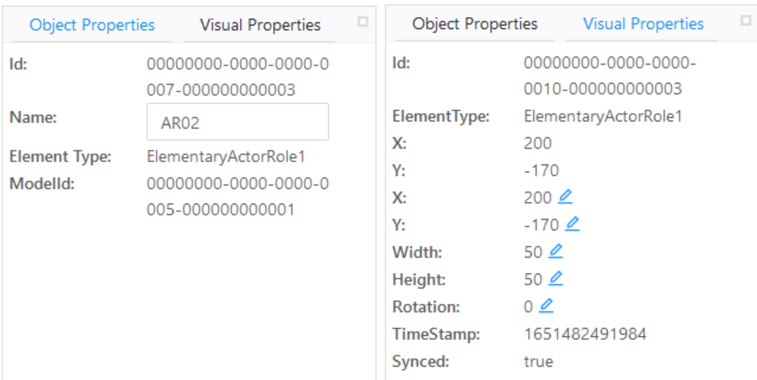


Fig. 7: The Element and Visual Element properties

4.2 Mobile & Tablet

Cross-platform is an important topic today[Sh16]. Due to the modular nature of Simplified, it is possible for mobile/tablet users to customise their interfaces in a way that works well with touchscreens. Since the website is developed mobile-first, there is no need for a separate app or download, thus further contributing to a lower barrier to start modelling from any device.

4.3 Gamification

Gamification is a way to explain information playfully while engaging the audience to participate. We have spent a considerable amount of time trying to communicate various aspect models to stakeholder groups and came to the conclusion that a different visualisation of the same model can facilitate the model understanding [MP21]. To involve the stakeholders interactively, the gamification interface connects directly to the platform. Therefore, it can operate in the same real-time fashion as the modeller does.

5 Development and production

For development we are using the IntelliJ IDE environment [Je22]. This supports the MySql [Or22] and MsSql [Mi22] development, as well as the Go [Go22a], React [Re22a], and Antlr [Pa22] languages. Even our own language, written in Antlr, can be supported during development. This IDE supports the whole team as the sources have been stored and shared using GitLab.

The web browser front-end is written in React and is using open source libraries to support the most common user interface components like docking windows and graphical shapes. React.js [Re22a] is an open-source JavaScript library, which is lightweight and it also provides a freedom to choose between different tools. React uses HTML (JSX) inside JavaScript which extends the functionalities of HTML structures into JavaScript.[Li22] JavaScript is fast, but updating the DOM makes it slow. Whereas React minimises DOM changes by monitoring the component's state with the Virtual DOM [Re22b] and finds the least expensive way to update the DOM.

The back-end is completely written in Go (for some people known as Golang). Go [Go22a] is an open-source programming language that is compiled to fast [Ga22] machine code. In addition, Go inherits the disciplined syntax of C with a feature to manage memory safely. The Go language has Goroutines. These routines are basically functions that can run simultaneously and independently, which makes it scalable. The Antlr grammar parts also compile to Go. The testing of the platform is done using unit tests in Go on the back-end, and Selenium tests procedures on the front-end. Manual testing is registered in Testmonitor[Ce22].

For production, all layers are run on open source components. The back-end is run as a docker configuration on Kubernetes. This allows for scaling of the application back-end. This application server connects to a MySql cluster for storage, thereby using the cluster principle to allow for scaling. Remote developers in companies or universities are independent of our development as they can run their servers against our platform to gain access to models and add functionality to the user interfaces.

6 Conclusion

The new platform, Simplified, takes away the limitations that we experienced during the development of Plena. It supports the collaborative design and multiple notations expressed in multiple languages. Though development has not been completed yet, the current version can hold all models for the following notations: DEMO, ArchiMate and BPMN.

Besides, up till now no restrictions have been seen that would impede the addition of more notations to the platform. The architecture layers are helpful in extending the support for more databases which we recently experienced while adding the MySql support. The extension of features can be done in a modular fashion, allowing for future feature development without disruption.

Additional research is needed to extend the features of modelling and a broader investigation of the limitations and gaps of other modelling tools is yet to be conducted. We have a road map of features that include, but are not limited to the following features:

- The referencing of models from a template, inheritance (advising or forcing) references.
- The support of visualisation concepts like swim lanes on diagrams, matrices, cubes, and automated layout features.
- User support on choosing the next step of the methodology.
- Advanced ruling that can efficiently verify models.
- Automated transformation between notations, and generation within notations and to other systems.

Finally, a lot of features that come with specific modelling methodologies will be added. We can conclude that the chosen direction of research and development is paying off as in a relatively short period of time we established a greenfield platform without restrictions for (meta)modelling support.

Bibliography

[Ca22] Camunda: , Web-based tooling for BPMN, DMN and Forms. `https://bpmn.io/`, April 2022.

[Ce22] Ceelen, Rene: , Test Management Software. Simplified. https://www.testmonitor.com/, April 2022.

[Cr22] Creately: , Business Process Mapping Tool. https://creately.com/lp/bpm-software-online/, April 2022.

[Di06] Dietz, J. L. G.: Enterprise Ontology – Theory and Methodology. Springer, Heidelberg, Germany, 2006.

[Di13] Dietz, J.L.G.; Hoogervorst, J.A.P.; Albani, Antonia; Aveiro, David; Babkin, Eduard; Barjis, Joseph; Caetano, Artur; Huysmans, Philip; Iijima, Junichi; Kervel, Steven JH Van: The Discipline of Enterprise Engineering. International Journal of Organisational Design and Engineering, 3(1):86–114, 2013.

[DM20] Dietz, J.L.G.; Mulder, H.B.F.: Enterprise Ontology: A Human-Centric Approach to Understanding the Essence of Organisation. Springer Nature, November 2020.

[Ed22] Edrawsoft: , Unlock Diagram Possibilities. https://www.edrawsoft.com/, April 2022.

[Fo22] Foundation, Cloud Native Computing: , A high performance, open source universal RPC framework. https://grpc.io, April 2022.

[Ga22] Game, Benchmark: , Go costs versus Java costs. https://benchmarksgame-team.pages.debian.net/benchmarksgame/fastest/go.html, April 2022. Accessed: 2022-05-03.

[GBDV20] Gray, Thomas; Bork, Dominik; De Vries, Marné: A New DEMO Modelling Tool that Facilitates Model Transformations. In: Enterprise, Business-Process and Information Systems Modeling, pp. 359–374. Springer, 2020.

[Go22a] Golang: , Build fast, reliable, and efficient software at scale. https://go.dev/, April 2022.

[Go22b] Google: , WebEA - New Tabbed Interface. https://sparxsystems.com/products/procloudserver/5.0/index.html#9AEC9F32-C964-4795-844E-FB9145DCDE43, April 2022.

[Ha81] Habermas, Jürgen: Theorie des kommunikativen Handelns, volume 2. Suhrkamp Frankfurt, 1981.

[Je22] JetBrains: , Why IntelliJ IDEA. https://www.jetbrains.com/idea/, April 2022.

[JG22] JGraph: , Security-first diagramming for teams. https://app.diagrams.net/, April 2022.

[Li22] Lindley, Cody: , What Is JSX. https://www.reactenlightenment.com/react-jsx/5.1.html, April 2022.

[Lu22] Lucid: , Where seeing becomes doing. https://www.lucidchart.com/pages/, April 2022.

[Mi22] Microsoft: , SQL Server 2019. https://www.microsoft.com/nl-nl/sql-server/sql-server-2019, April 2022.

[MP21] Mulder, Mark.A.T.; Proper, Henderik.A.: On the Development of Enterprise-Grade Tool Support for the DEMO Method. In: CAiSE 2021. 2021.

[Mu22] Mulder, M.A.T.: Enabling the automatic verification and exchange DEMO models. PhD thesis, Radboud University Netherlands, March 2022.

[Na12] Nassar, Victor: Common criteria for usability review. Work, 41(Supplement 1):1053–1057, 2012.

[Nu22] Nulab: , Teams who diagram together, thrive together. `https://cacoo.com/`, April 2022.

[Or22] Oracle: , MySQL. `https://www.mysql.com/`, April 2022.

[Pa22] Parr, Terence: , What is ANTLR? `https://www.antlr.org/`, April 2022.

[Re96] Reijswoud, V. E. Van: The structure of business communication: Theory, model and application. PhD thesis, TU Delft, 1996.

[Re22a] React: , React - A JavaScript library for building user interfaces. `https://reactjs.org/`, April 2022.

[Re22b] React: , Virtual DOM and Internals – React. /urlhttps://reactjs.org/docs/faq-internals.html, April 2022.

[Sh16] Shin, Dong-Hee: Cross-Platform Users' Experiences Toward Designing Interusable Systems. International Journal of Human–Computer Interaction, 32(7):503–514, 2016.

Beyond Low Code Platforms: The XModeler^{ML} — an Integrated Multi-Level Modeling and Execution Environment

Ulrich Frank[1] Luca L. Mattei[2] Tony Clark[3] Daniel Töpel[1]

Abstract: This paper presents the multi-level modeling tool XModeler^{ML}. It allows to create models that feature an arbitrary number of classification levels, thereby enabling the conjoint specification of modeling languages and their application within one diagram editor. The potential of multi-level modeling in general, of the XModeler^{ML} in particular is illustrated through three use cases.

Keywords: Language engineering; models at runtime; model-based development

1 Introduction

Multi-level modeling is motivated by the lack of expressiveness and abstraction, traditional modeling approaches suffer from [AK01, Fr14b]. Different from traditional approaches to object-oriented modeling, multi-level modeling allows for an arbitrary number of classification levels. Hence, it allows for the simultaneous design of modeling languages and models specified with them, thus enabling models that provide for a higher level of reuse, improved adaptability and integrity. For a detailed discussion of prospects offered by multi-level modeling see [Fr22].

The use cases outlined in this paper serve the description of the XModeler^{ML}, a multi-level language engineering, modeling and execution environment. They are dedicated to demonstrate specific features of the tool that set it apart from other modeling tools. The XModeler^{ML} and the models presented with the use cases described in the paper as well as screencasts that explain the use of the tool are available in the "XModeler^{ML}" section of the webpages of the project "Language Engineering for Multi-Level Modeling" (LE4MM)[4].

2 Architecture

The XModeler is a language engineering workbench whose core language is reflexive and extensible. In this way, the XModeler is both an instance of itself and a basis for defining a

[1] Universität Duisburg-Essen, [ulrich.frank,daniel.toepel]@uni-due.de
[2] Universität Duisburg-Essen, luca.mattei@stud.uni-due.de
[3] Aston University, Birmingham tony.clark@aston.ac.uk
[4] https://www.wi-inf.uni-duisburg-essen.de/LE4MM/

wide range of co-existing language variants. The FMML^x represents an extension of the XModeler's meta model, XCore, to introduce explicit levels and deferred instantiation. The resulting version of the XModeler, called XModeler^{ML}, also features a new user interface that supports multiple modes of creating, modifying and interacting with objects at any level. An extensive description of the foundational language architecture can be found in [CSW15], the FMML^x is described in more detail in [Fr22]. Further specifications and illustrations are available on the webpages of the project LE4MM.

Fig. 1 is intended to give an overview of the XModeler^{ML} from a user perspective. It shows the various components offered to users and outlines their representation within the GUI. Users of the tool have the choice to create models with a diagram editor, a model browser, or both. All classes in FMML^x models are objects at the same time. Models are executable, that is, the operations defined for objects can be executed — within a diagram editor, a model browser or through the console. In addition, a default instance browser can be generated for every class in a diagram. Like the diagram editor, the model browser allows navigating through the classes/objects of a model, and modify class properties and object states. Various code editors allow writing code to implement operations and constraints. The XModeler^{ML} features dynamic typing. Therefore, models as well as corresponding languages can be changed at runtime.

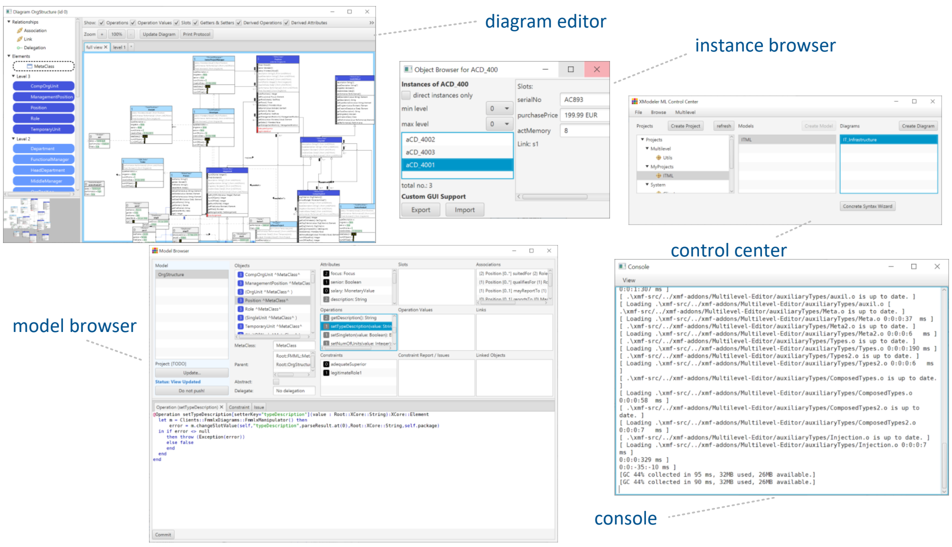


Fig. 1: Core components of the XModeler^{ML} from a user perspective

Fig. 2 shows the architecture of XModeler^{ML} from an implementation perspective. Both the user interface and the XMF virtual machine (VM) run on the Java VM. XCore and its adaptation for FMML^x[Fr14a] are run by the XMF VM and form the foundational (meta) language — XOCL — for the specification of further languages and models. The XMF clients are programmed in XOCL and can therefore work on the objects in the XMF VM.

To allow users to interact with the XMF VM, several editors have been implemented in Java which represent the key GUI components of the XModeler^{ML}. These editors communicate with their counterparts, the XMF Clients.

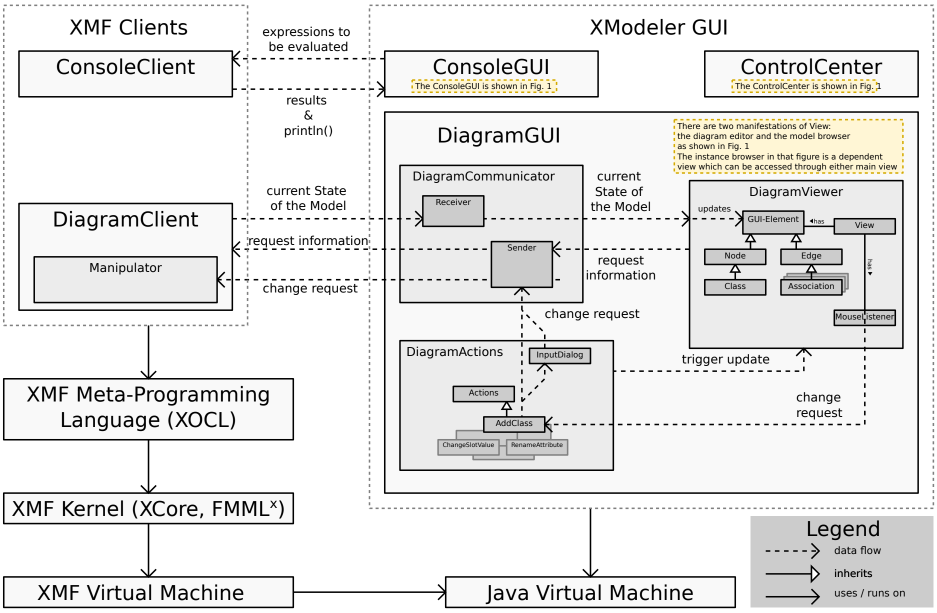


Fig. 2: Core components of the architecture

3 Use Case 1: UML++ — Integration of Executable Object Models and Classes

Multi-level modeling represents a paradigm shift that may seem irritating to some. Also, hardly any multi-level programming languages exist that would allow for a straightforward implementation of multi-level models. For these reasons, it is probably safe to assume that many modelers and software developers are reluctant to adopt the idea of multi-level modeling right away. But even those who do not want to give up on traditional object-oriented languages can benefit from a multi-level language architecture. The following use case demonstrates how the XModeler^{ML} provides extra value to the construction of UML class diagrams.

Common UML modeling tools allow modeling classes on M1 only. To use an object model for implementation purposes, code generation is the approach of choice. Unfortunately, with respect to a system's life cycle, code generation requires the synchronisation of models and code, which is a notorious problem. The main reason for code generation is the fact that classes in a model editor are represented as objects on M0, which does not allow for

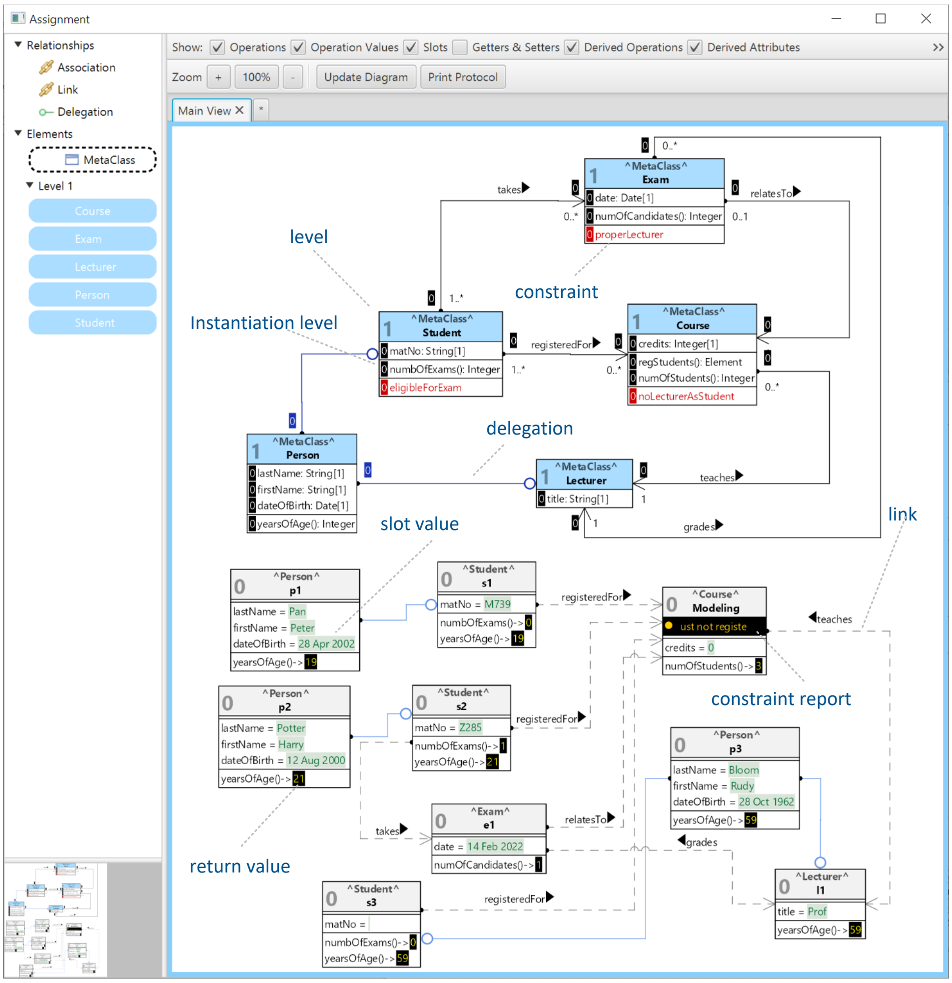


Fig. 3: Class diagram and corresponding instances within the XModeler^{ML} diagram editor

instantiating them. This is different with the FMML^x and the XModeler^{ML}. Every class is implemented at the level it conceptually represents. In the case of UML class diagrams, that means a class is represented as an object on M1. Therefore, it can be instantiated and the resulting objects can be executed within the model editor.

The object model shown in Fig. 3 widely corresponds with UML class diagrams. It was created with the FMML^x and represents a simplified view on the management of courses and exams at a university. Note that the notation is different from the UML, because it accounts for the representation of levels and deferred instantiation of class properties (for an extensive description of the FMML^x' concrete syntax see [Fr22]). The classes `Student` and `Lecturer` are modeled as roles, which is represented through a delegation association that connects both classes to their role filler class **Person**. The methods offered by a role filler object can be transparently accessed through their role objects at runtime. Additional constraints can be specified with the XOCL [CSW15]. For example, the constraint **notLectureAsStudent** of the class **Course** prevents a person to register for a course which she teaches at the same time:

```
context Course
@Constraint allowedToManipulate
   not self.regStudents()→exists(i | i.roleFiller =
self.getLecturer().roleFiller)
end
```

Constraints are checked during the construction of a model. If a constraint fails, a constraint report is presented as an alert, yellow on black, within the rectangle representing an object.

The integration of classes and corresponding executable objects in one diagram is especially useful for teaching purposes. It allows giving students an immediate feedback of their modeling decisions. At the same time, it fosters their understanding of how object models and corresponding programs are interrelated. Furthermore, it is suited to motivate true multi-level models. If one, e.g., wanted to add an operation that computes the percentage of students that took a specific course, the operation could be defined within a meta class of **Student**.

4 Use Case 2: A Multi-Level Language for Modeling IT Infrastructures

Domain-specific modeling languages (DSMLs) promise various advantages over general-purpose modeling languages (GPML). However, the design of DSMLs is challenging. First, traditional language architectures do not allow to represent all relevant domain knowledge with a DSML. Take, for example, a DSML for modeling IT infrastructures. Such a language could include a concept like “Computer”, which could be used to specify specific computer models. Even though it is known that a particular exemplar (an instance) of a computer model has a serial number and a certain main memory size, it is not possible to express this knowledge with the DSML, because it is restricted to the specification of properties of type, in this case of computer models. This restriction has serious implications. If existing knowledge cannot be expressed at the highest level of abstraction, it is required to repeat it redundantly on lower levels, which is obviously a clear threat to reuse, integrity and adaptability.

Second, the design of a DSML requires to make a clear decision for every domain concept as to whether it should be represented with the language or through the language – a decision that will often remain unsatisfactory [Fr13]. Multi-level modeling enables the creation of a hierarchy of DSMLs. At the top level, a rather generic DSML, comparable to textbook knowledge, would allow for a wide range of reuse. By using this language to define a more specific DSML, the productivity of reuse is clearly increased.

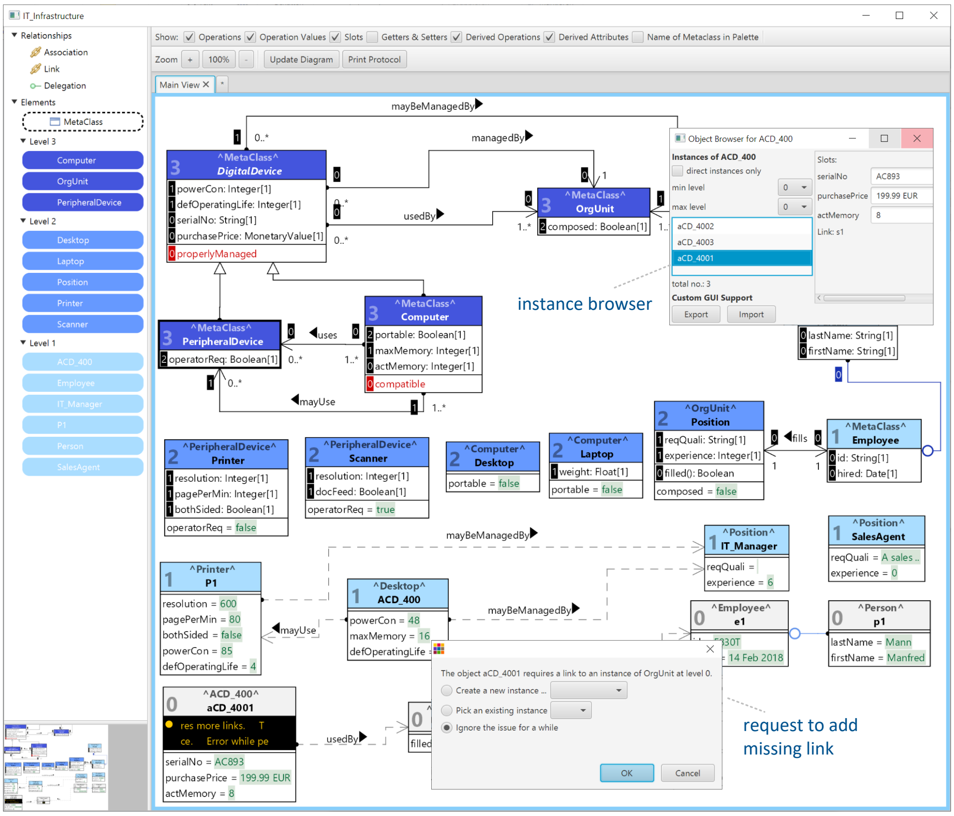


Fig. 4: Illustration of multi-level model representing multiple language layers

The multi-level model shown in Fig. 4 illustrates the structure of a language architecture that comprises DSMLs at various levels of abstraction. At the top level, mainly represented by classes on L3[5], concepts such as "Organizational Unit", "Peripheral Device", or "Organizational Unit" serve the definition of a basic language for describing an IT infrastructure and its management. Note that classes that form a DSML do not have to be on the same level. Classes like **`Person`** or **`Employee`** are generic enough to belong to the top level DSML, even though they are located at L1 only. Properties and constraints defined with this language layer may apply to objects on lower levels only. For example, the attribute **`serialNo`** of the class **`DigitalDevice`** in Fig. 4 is marked as *intrinsic* with the instantiation level L0, which means it is to be instantiated on L0 only. The constraint **`properlyManaged`** is also defined as intrinsic. It defines that a digital device may only be managed by a position that is of a type for which this eligibility was specified. Accordingly, it is checked only at L0. The next

[5] Note that we use Ln instead of Mn to refer to levels in a multi-level model, because the semantics of levels is not exactly the same as that of classification levels.

language layer comprises classes at L2 that are more specific, but should still be applicable to a wide range of organizations. In case, the number of instances of a class becomes too large to properly visualize them within a diagram, it is possible to invoke a specific browser on these instances.

The design of a model (or a DSML respectively) with an existing DSML is supported by the diagram editor. First, it offers previously defined language concepts in the palette. Second, upon the construction of a model explicit and implicit constraints specified with the language are checked and users are offered support to adequately revise the model. If, for example, the multiplicities defined with an association require an object to be linked to another, the editor presents a dialogue to guide the required completion (see Fig. 4). The XOCL allows the definition of constraints that refer to multiple levels [CF18]. Furthermore, users of the XModeler^{ML} are supported with a method for designing multi-level models [Fr21].

The integration of executable languages and models enables the realization of new application architectures that empower users to navigate from objects they use to the corresponding classes and, moreover, to the languages these classes are specified with. That does not only support users with developing a deeper understanding of the systems they use, but also with adapting them to their needs.

5 Use Case 3: Application Development with Focus on GUI

So far, the main focus of the XModeler^{ML} development was on language engineering and modeling – either with the diagram editor or with a model browser. However, for building applications, a GUI is indispensable. The implementation of a GUI builder from scratch would have required an effort beyond the available project resources. Therefore, we decided to reuse an existing tool and to integrate it widely transparently with the XModeler^{ML}. To this end, the JavaFX GluOn Scene Builder proved to be an appropriate choice because it provides for a smooth integration with the XModeler^{ML} whose GUI was also implemented in JavaFX.

The basic idea of this tool is to generate a preliminary default GUI from an object model (e.g. classes, attributes, operations, associations). The result of this default transformation is exported to the GluOn Scene Builder as an XML file specified according to FXML, an XML document type for the description of user interfaces for JavaFX. This way GluOn Scene Builder can be used to conveniently customize the layout of the default GUI, resulting in a FXML file, which is then used to overwrite the original FXML file. Subsequently, this file is loaded into the XModeler^{ML}, where it is transparently connected to the original model. The use case in Fig. 5 shows an object model and a corresponding custom GUI realized through the use of an external GUI builder.

The default GUI does not account for any descriptions besides structure and layout. As a consequence, the specific mapping of the view to the model cannot be supported by a dedicated controller. Instead, the custom GUI relies on a generic controller, which is suited to handle all conceivable custom GUIs and connect them to the model (more precisely: to a textual description of it). Currently, this is achieved by complementing the GUI description in the document (type) with additional attributes. They allow to reference particular aspects of a model or the GUI and specify corresponding actions. The generic mapping is currently described with a simple controller language. The use case in Fig. 5 shows an object model and a corresponding custom GUI that was customized through the external GUI builder.

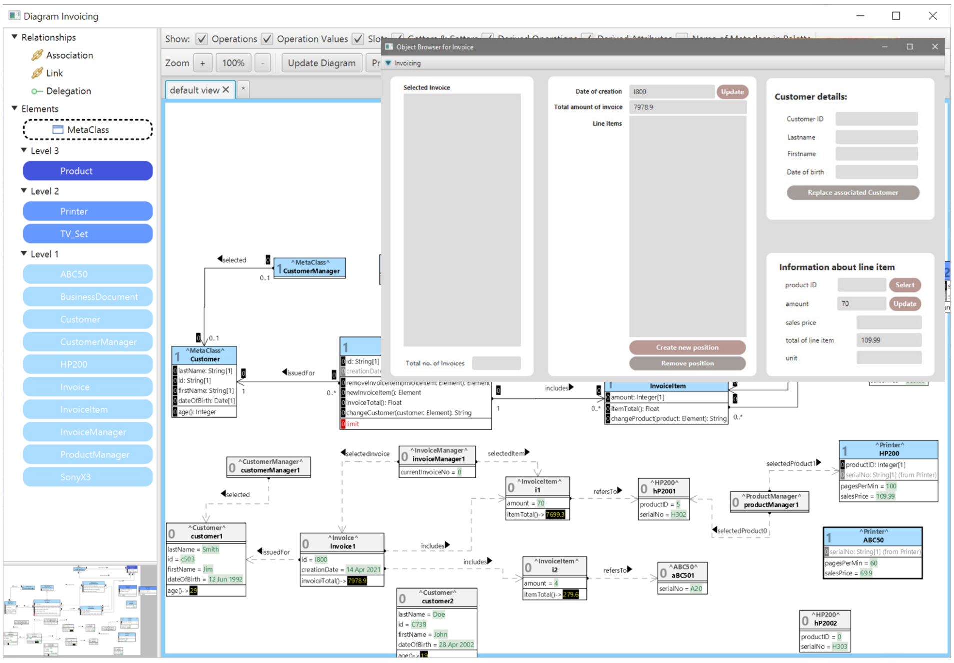


Fig. 5: GUI generated and rearranged from object model

6 Conclusions and Future Research

Various tools have been developed to support the creation and maintenance of multi-level models, e.g., [AG16, LG10]. The XModeler^{ML} differs from these tools in several respects. The XModeler^{ML} features an extensive language engineering environment including an executable multi-level constraint language, an adaptable implementation of the MVC-pattern, a compiler-compiler, and many more features that support the construction of languages, models and applications. It also features a common representation of models and programs. On the one hand, this gives developers the choice of either writing code or creating models with a diagram editor. On the other hand, it contributes to the empowerment of prospective users who may navigate and possibly change the models/applications they use at runtime. These features alone set the XModeler^{ML} clearly apart from so called “low-code” platforms, which usually do not provide elaborate DSMLs, nor allow their construction [BF21]. We plan to further develop the support for building custom GUIs, since user interfaces are usually an important part of applications. That includes the development of a more powerful controller language.

The additional abstraction enabled by multi-level modeling leads to a tighter coupling of model elements. As long as the chosen abstractions are invariant, these dependencies support the convenient and safe adaptation of models. However, if the abstractions turn out to be not invariant, changing a multi-level model becomes especially challenging. It may, e.g., happen that a class has to be changed, while its instances already exist. To support the maintenance of multi-level models, the XModeler^{ML} includes a change management system comprised of specific *change operations*[TB17]. The change operations guide the developers to change the model even on higher levels without compromising the integrity of a multi-level model. The change operations range from very basic modifications like altering a slot value, which have little to no side effects, to very sophisticated changes like altering the superclass of a class, which has a major impact on the class hierarchy. A wide range of changes is supported by the XModeler^{ML}. However, there are a few change operations, e.g., those that are comparable to class migration, that we still need to implement.

So far, the concrete syntax of models is restricted to the generic notation of the FMML^{x}. We are currently working on an editor for graphical notations that uses SVGs to allow for the design of custom notations for DSMLs. Subsequently, we plan to reconstruct all MEMO languages [Fr14a] within the XModeler^{ML}. An additional focus of our future research will be on the development of multi-level process models in order to promote reuse and adaptation of dynamic abstractions. Currently, it is only possible to define static abstractions of multi-level process models.

To demonstrate the benefits of multi-level application architectures, we will continue our work on prototypical applications that enable users to navigate and eventually change the models that form the conceptual foundation of an application and, at the same time, represent its implementation.

Bibliography

[AG16] Atkinson, Colin; Gerbig, Ralph: Flexible Deep Modeling with Melanee. In (Reimer, Stefanie Betz Ulrich, ed.): Modellierung 2016, 2.-4. März 2016, Karlsruhe - Workshopband. volume 255 of Modellierung 2016, Gesellschaft für Informatik, Bonn, pp. 117–122, 2016.

[AK01] Atkinson, Colin; Kühne, Thomas: The Essence of Multilevel Metamodeling. In (Gorgolla, Martin; Kobryn, Cris, eds): UML 2001 - The Unified Modeling Language. Modeling Languages, Concepts, and Tools, Lecture Notes in Computer Science, pp. 19–33. Springer, Berlin and London, New York, 2001.

[BF21] Bock, Alexander C.; Frank, Ulrich: Low-Code Platform. Business & Information Systems Engineering, 63(6), 2021.

[CF18] Clark, Tony; Frank, Ulrich: Multi-Level Constraints. In (Hebig, Regina; Berger, Thorsten, eds): Proceedings of MODELS 2018 Workshops. volume 2245 of CEUR Workshop Proceedings. CEUR-WS.org, pp. 103–117, 2018.

[CSW15] Clark, Tony; Sammut, Paul; Willans, James S.: Super-Languages: Developing Languages and Applications with XMF (Second Edition). CoRR, abs/1506.03363, 2015.

[Fr13] Frank, Ulrich: Domain-Specific Modeling Languages - Requirements Analysis and Design Guidelines. In (Reinhartz-Berger, Iris; Sturm, Aron; Clark, Tony; Wand, Yair; Cohen, Sholom; Bettin, Jorn, eds): Domain Engineering: Product Lines, Conceptual Models, and Languages, pp. 133–157. Springer, 2013.

[Fr14a] Frank, Ulrich: Multi-Perspective Enterprise Modeling: Foundational Concepts, Prospects and Future Research Challenges. Software & Systems Modeling, 13(3):941–962, 2014.

[Fr14b] Frank, Ulrich: Multilevel Modeling – Toward a New Paradigm of Conceptual Modeling and Information Systems Design. BISE, 6(6):319–337, 2014.

[Fr21] Frank, Ulrich: Prolegomena of a Multi-Level Modeling Method Illustrated with the FMML^{x}. In: Proceedings of the 24th ACM/IEEE International Conference on Modell Driven Engineering Languages and Systems: Companion Proceedings. IEEE, 2021.

[Fr22] Frank, Ulrich: Multi-level modeling: cornerstones of a rationale. Online First. Software and Systems Modeling, 2022.

[LG10] Lara, Juan De; Guerra, Esther: Deep Meta-modelling with MetaDepth TOOLS 2010, Málaga, Spain, June 28 - July 2, 2010. Proceedings. In (Vitek, Jan, ed.): Objects, Models, Components, Patterns. TOOLS 2010, pp. 1–20. Springer, Berlin, Heidelberg, 2010.

[TB17] Töpel, Daniel; Benner, Björn: Maintenance of Multi-Level Models An Analysis of Elementary Change Operations. In: 4th International Workshop on Multi-Level Modelling - MULTI 2017. Austin, USA, p. 14, September 2017.

MEMO4ADO: A Comprehensive Environment for Multi-Perspective Enterprise Modeling

Alexander Bock[1] Ulrich Frank[1] Monika Kaczmarek-Heß[1]

Abstract: This paper describes the construction and use of a comprehensive tool for enterprise modeling, MEMO4ADO, implemented using the meta modeling environment ADOxx. MEMO4ADO offers an extensible set of editors for various modeling languages. The paper gives an overview of the tool's architecture and describes the integration of modeling languages and diagrams. A use case serves to illustrate the design and analysis of enterprise models with MEMO4ADO. The paper concludes with a brief assessment and an outlook on the further development of the tool.

Keywords: meta modeling; DSML; enterprise modeling tool; MEMO; MEMO4ADO; ADOxx

1 Introduction

The ability of companies to compete in present-day markets depends increasingly on the effectiveness of their information systems (IS). To exploit the potential of information technology (IT), it is usually necessary to reorganize organizational processes and structures alongside the introduction of new software systems. The recognition of this interdependence of business and IT led more than three decades ago to the emergence of enterprise modeling [Za87]. An enterprise model integrates models of *information systems*, such as data models or object models, with models of a company's *action system*, like models of business processes or of organizational structures. In the decades following the publication of Zachman's pioneering, but relatively coarse high-level framework in the 1980s [Za87], a variety of more sophisticated methods for enterprise modeling have been developed [FS98, Sa14], among them *Multi-Perspective Enterprise Modeling* (MEMO) [Fr14].

MEMO is an extensive method for enterprise modeling. It supports various perspectives on the enterprise, which can be described in more detail using an extensible set of domain-specific modeling languages (DSMLs). The set of DSMLs includes a language for modeling organizational structures and business processes (OrgML) [Fr11a, Fr11b], as well as languages for describing goal systems (GoalML) [OFK15], decision processes (DecisionML) [Bo15], and IT infrastructures (ITML) [FKHdK21]. In addition, MEMO integrates general-purpose modeling languages such the ERM and UML class diagrams.

The evolution of MEMO has been accompanied by the development of modeling tools from the very beginning. Earlier modeling tools for MEMO were implemented on the

[1] Universität Duisburg-Essen, [alexander.bock,ulrich.frank,monika.kaczmarek-hess]@uni-due.de

basis of Smalltalk [Fr94] and the EMF [GF10]. *MEMO4ADO* is the latest in this series of tools. It is implemented using the meta modeling environment ADOxx [FK13], with the first version having been presented in 2015 [BF16]. Since then, MEMO4ADO has been intensively used for teaching purposes, and it has been extended and revised on a regular basis [FB20, FKHdK21]. MEMO4ADO is freely available at `https://austria.omilab.org/psm/content/memo4ado/info`.

This paper provides an overview of the tool's foundation, its latest version, and demonstrates how to use it. To this aim, first, an overview on the MEMO's language architecture and its implementation in ADOxx is provided. Then, the implemented languages, their integration as well as model editors are shortly described. Finally, a use case pointing to the way the MEMO4ADO tool may be used to analyse or design an enterprise model is provided. In opposition to already published papers on MEMO4ADO, e.g., [BF16, FB20], we focus here on the tool itself and functionalities it offers, as well as newly added languages. The paper concludes with final remarks on the tool usage and next steps.

2 MEMO's Language Architecture and its Implementation in ADOxx

As already mentioned, MEMO4ADO is implemented on the basis of the meta modeling environment ADOxx [FK13]. As indicated in the introduction, we had used EMF for a previous implementation of a MEMO modeling environment, and struggled with the effort required to implement a new DSML and to maintain the tool. In contrast, ADOxx turned out to be better suited to satisfy our requirements. In particular, it offers the ability to efficiently and effectively implement new DSMLs and corresponding modeling editors. The possibility to easily create stand-alone modeling tools has also influenced our choice.

Fig. 1 illustrates the language architecture of MEMO and the implementation in ADOxx. As is seen on the left-hand side, the various MEMO modeling languages are specified with a common meta modeling language, the MEMO MML [Fr11c]. The different languages are integrated through common concepts, such that an integrated object model can be generated for the implementation of modeling tools. Adaptations to the original meta models are, consequently, propagated consistently to the unified object model of the modeling tool.

As is seen on the right-hand side of Fig. 1, the ADOxx environment enabled us to follow a largely analogous scheme in the practical implementation. Each MEMO meta model was reconstructed using the ADOxx meta modeling editor (called 'ADOxx Development Toolkit'). The common concepts that integrate the different MEMO languages were specified using the ADOxx's capacity of cross-diagram links, cf. *interref* [FK13, p. 7]. These links enable the modeler to define references from elements in one diagram (e.g., a business process diagram) to elements in another (e.g., an IT infrastructure diagram). Using this feature, the close integration of the MEMO language could be successfully carried over to the MEMO4ADO tool.

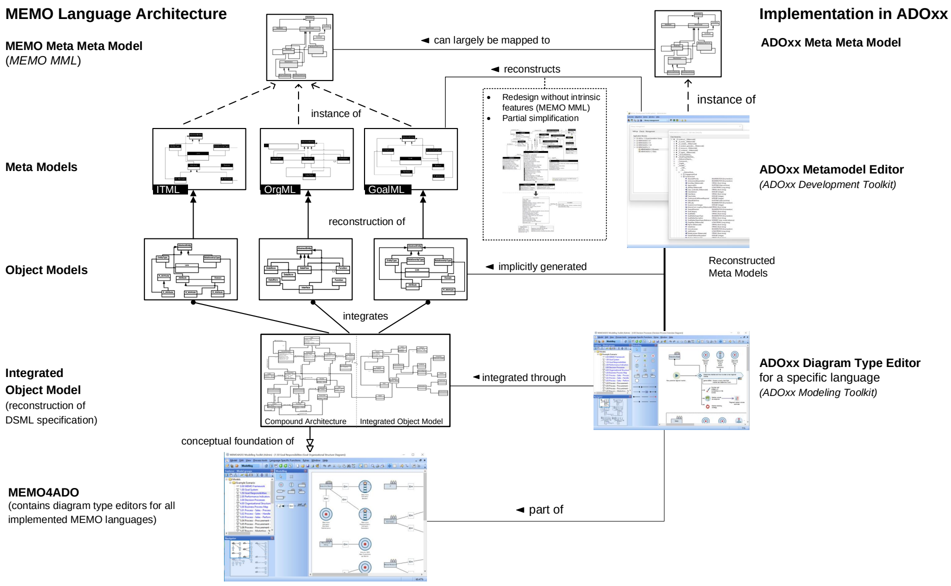


Fig. 1: Language architecture and mapping to tool architecture

Only two kinds of modifications were required during the reconstruction of the MEMO meta models in ADOxx. First, the MEMO MML [Fr11c] features so-called ‘intrinsic features’. These are used to mark metaclasses, attributes, or associations which are specified at level M2 but instantiated only at level M0. The modeling environment of ADOxx, however, is confined to level M1; it is not possible to instantiate elements again into instances at level M0. Accordingly, we had to partially redesign the MEMO languages without intrinsic attributes. The second group of modifications covers some minor simplifications in the service of accessibility. For example, the complex MEMO GoalML originally required separate definitions of goals and their objects, but it turned out to be more practical to reunite these aspects into one concept.

On the basis of the reconstructed meta models and certain other specifications in the ADOxx Development Toolkit (including the definition of the concrete syntax), the diagram editors for all implemented languages could be generated automatically. The resulting tool, MEMO4ADO, integrates all implemented MEMO languages in the form of different diagram types (e.g., ‘Goal System Diagrams’, ‘Organizational Structure Diagrams’, and others). These diagram types and their use will be sketched in the following sections. For a more comprehensive description of MEMO4ADO and its implementation, see [BF16, FB20].

3 Implemented Languages, Integrated Diagram Types and Features of Model Editors

The current version of MEMO4ADO (version 1.10) implements, among others, the MEMO GoalML [OFK15] to model organizational goal systems, the MEMO OrgML to model organizational structures [Fr11a], the MEMO OrgML to model business processes [Fr11b], the MEMO DecisionML [Bo15] to model organizational decision processes, the MEMO MetricML [St12] to model performance indicator systems, and the MEMO ITML to model IT infrastructures [FKHdK21]. Each of these DSMLs allows for the creation of dedicated diagrams of specific types. All DSMLs are integrated through a common meta model and through common concepts. Thus, elements of one model may reference elements of other models, cf. Sec. 2. This enables a targeted navigation through an enterprise model. While each DSML is supported by a specific model editor (e.g., 'Business Process Control Flow Diagram' created using the OrgML, bottom right in Fig. 2, or 'IT Infrastructure Diagram' created using the ITML, top right in Fig. 2), it is also possible to design diagrams that represent an integrated view on models created with different DSMLs, cf. [BF16, FB20]. For example, in the integrative diagram type 'Goal-Organizational Structure Diagram' it is possible to link goals defined in a 'Goal System Diagram' and organizational units defined in an 'Organizational Structure Diagram' (bottom left in Fig. 2).

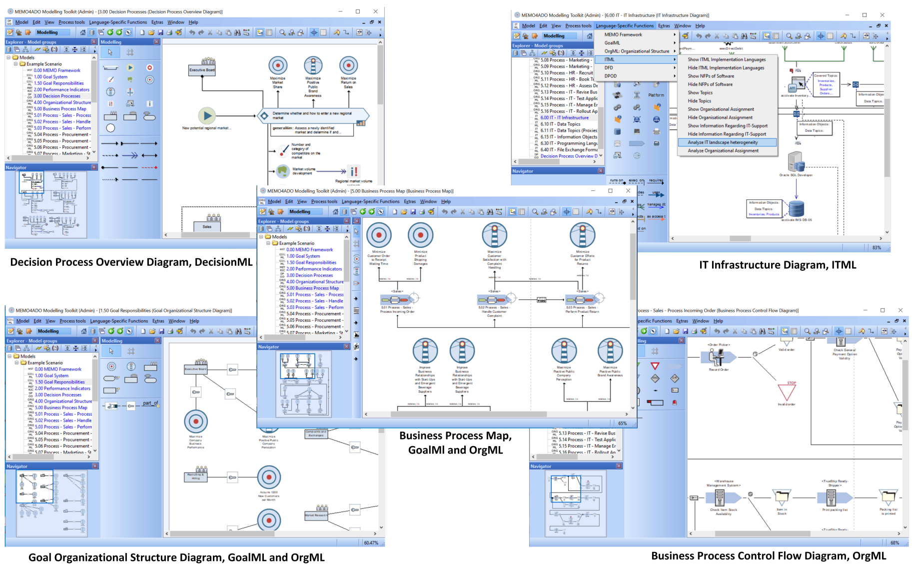


Fig. 2: MEMO4ADO Model Editor and Selected Diagram Types

All DSMLs implemented in MEMO4ADO offer rich sets of modeling concepts that allow for the design of sophisticated models. Let us exemplary consider an 'IT Infrastructure Diagram', cf. Fig. 3, being the essential MEMO ITML diagram type, used to design, re-design or analyze an organization's IT infrastructure. As such, this diagram type permits to describe

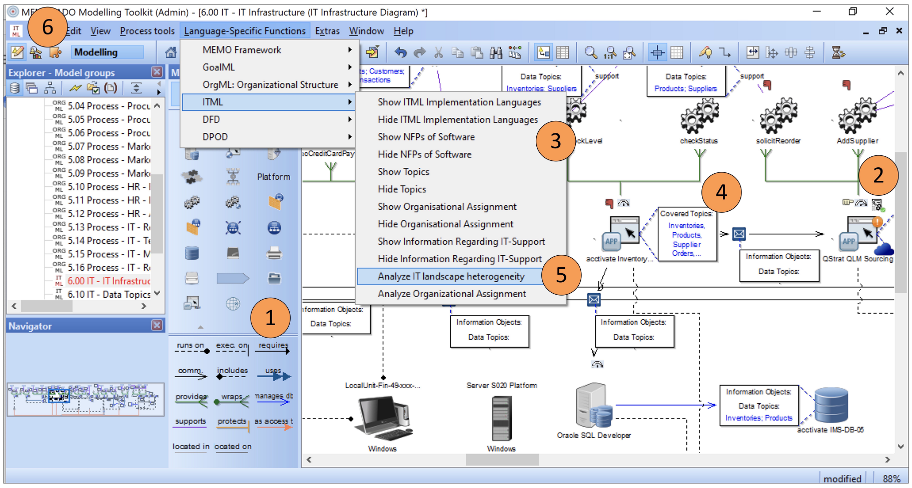


Fig. 3: MEMO4ADO: Exemplary IT Infrastructure Diagram

platforms (both physical as well as virtual ones), hardware artifacts, software systems, services, as well as various dependencies among those, cf. (1). Each kind of artifact comes with a variety of attributes. For many attributes, dedicated notational symbols are provided, whose appearance is automatically adjusted based on the current attribute value (e.g., mission critical, user satisfaction, runs in cloud, cf. (2)). A variety of relationships between modeling concepts, e.g., 'runs on', 'communicate', serve the differentiated description of dependencies between software and/or hardware artefacts.

In order to ease language use and to facilitate additional analyses for different target groups, further functions have been added. Examples of those are, cf. also [BF16, FB20]: (a) *Different Levels of Notational Details*, cf. Fig. 3, (3) – as some diagrams exhibit a level of notational detail which might be considered too complicated for some purposes, functions have been implemented that allow to switch between different levels of details. (b) *Overview Textboxes*, cf. Fig. 3, (4) – attribute values of model elements can be specified and investigated using the ADOxx notebook dialog. Sometimes, it may also be considered helpful to see values of attributes at a glance while interpreting a diagram. For this purpose, we implemented various additional text box views for each language that attach a basic text box to each model element in which the values for selected attributes are listed textually. (c) *Dedicated language-specific functions*, cf. Fig. 3, (5) – for selected languages, a various additional functions allowing to conduct specific kinds of analysis, e.g., analysis of heterogeneity of IT landscape[2], of organizational assignment, have been implemented. (d) *Ad-hoc Queries*, cf. Fig. 3, (6) – note that in addition to the above-mentioned features, the generic query

[2] Here the modeled elements are analysed to identify such aspects as, among others, types of platforms used, types of operating systems, programming languages used to implement software artifacts. Users are then presented with a corresponding table detailing the variety of those aspects and the number of elements identified.

mechanism offered by ADOxx may be also used to answer questions/conduct analyses of interest, also spanning multiple diagrams. Queries can be executed on models using the ADOxx Query Language (AQL) through the "Analysis" component. In short, the AQL query language allows to formulate queries on models in a style similar to SQL, cf. [FK13, p. 18], thus allowing to retrieve elements fulfilling some criteria, e.g., elements of some type, elements having some attribute's value, and/or related to some other element(s). The AQL queries can either be entered manually by a user or can be pre-defined, cf. [FK13, p. 18], to support specific analysis scenarios as required by the given modeling method being implemented. When accessing the pre-defined query, a user has then the option to select required elements from the lists showing all available elements matching the criteria defined in the query definition, and then execute the query.

4 Use Case: Analysis and Design of a Partial Enterprise Model

The basic way of using the MEMO4ADO tool comes down to (1) creating different core diagrams that describe selective abstractions of the enterprise in question (i.e., goal systems, organizational structures, business process control flows, and IT infrastructure), (2) to subsequently interrelate them, and finally (3) conducting integrative, reflective analysis on both action system and information system in tandem. Possible application scenarios range from strategic goal planning processes to integration of organisational IT landscape. For details, see [BF16, FB20].

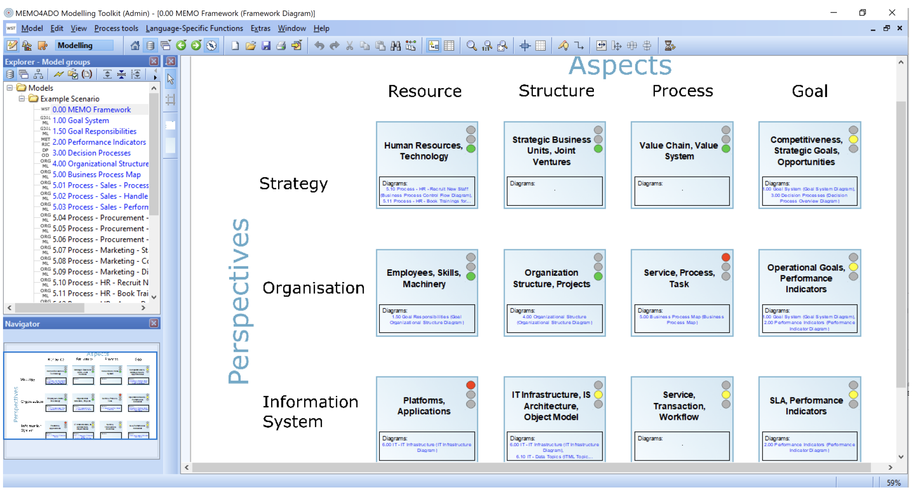


Fig. 4: MEMO4ADO: MEMO Generic Framework

The following example serves to demonstrate how the MEMO4ADO tool can be used to design and analyze an enterprise model. Due to the limited extent of the paper, the example is highly simplified. Let us assume that in order to stay competitive, a wholesale company that sells tools and material mainly to handicraft businesses has to analyze and eventually redesign its services and processes. First, a common understanding of the company and the problems it is facing needs to be established. To that end, MEMO4ADO provides a generic framework that allows for a high-level overview of a company's current or future situation. For instance, the framework presented in Fig. 4 structures an enterprise along two dimensions. The first dimension encompasses three perspectives: (1) strategic, (2) organizational, and (3) information system. The second dimension encompasses different generic aspects found in each perspective, such as structure, process, and resources. The generic framework provides a a common starting point for identifying key problem areas and for defining priorities at the beginning of a project. The representation of the framework in a two-dimensional table allows assigning specific topics to each focus, and thus, not only provides a first overview of the enterprise and the current situation, but also allows to identify areas in need of a more elaborate analysis. To further investigate specific topics, diagrams can later be assigned to each focus. This way, the generic framework can be used as a central starting point to navigate an enterprise model, cf. Fig. 4.

The company for instance, may decide to increase its competitiveness by adapting its business model (e.g., by focusing on highly customizable solutions, as well as offering additional services to improve customer experience) with a specific focus on the automation of its processes. The design of a preliminary goal model (referenced from the MEMO framework) serves to analyze the implementation of such a strategy, cf. Fig. 5. The corresponding strategy may in turn require re-engineering (sales) processes. The efficient execution of these processes requires accounting for the supporting IT infrastructure as well as for the corresponding organizational structure. To this end, the company first designs models of targeted business processes, like detailing the 'Sales - Incoming Order Process' (see Fig. 7).

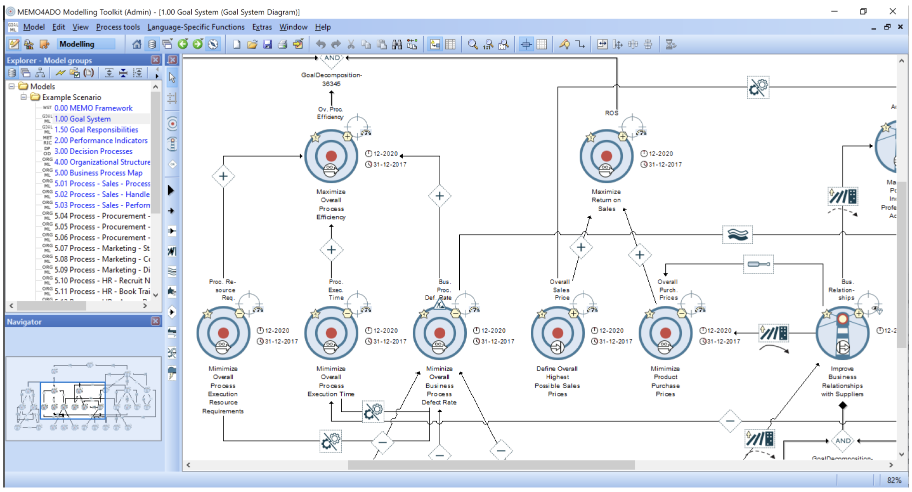


Fig. 5: MEMO4ADO: Goal System Diagram

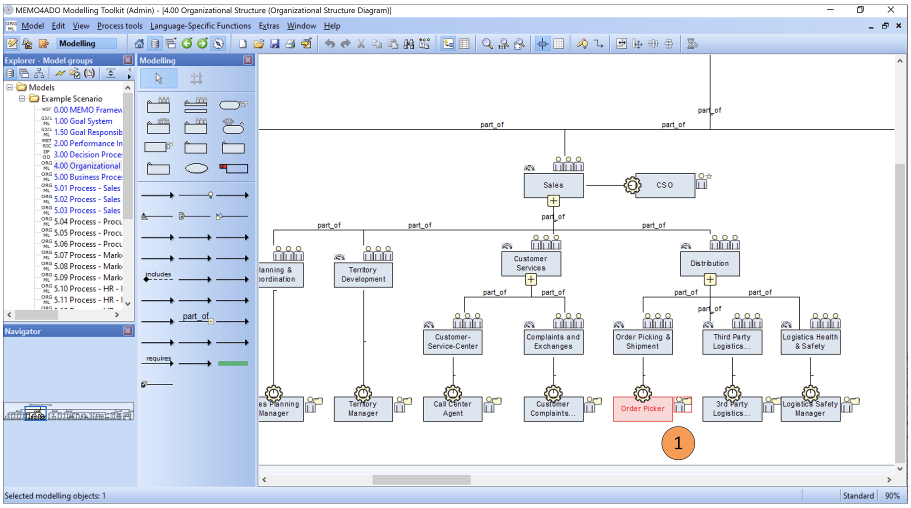


Fig. 6: MEMO4ADO: Organizational Structure Diagram

Those process models are used to guide the incremental development of corresponding models of the IT infrastructure and the organizational structure.

Thus, the process model in Fig. 7 does not only show an excerpt of a model that describes a customer order process, but it also indicates how a business process model is integrated with other models, cf. (1) and (2). On the one hand, links to services enable the integration

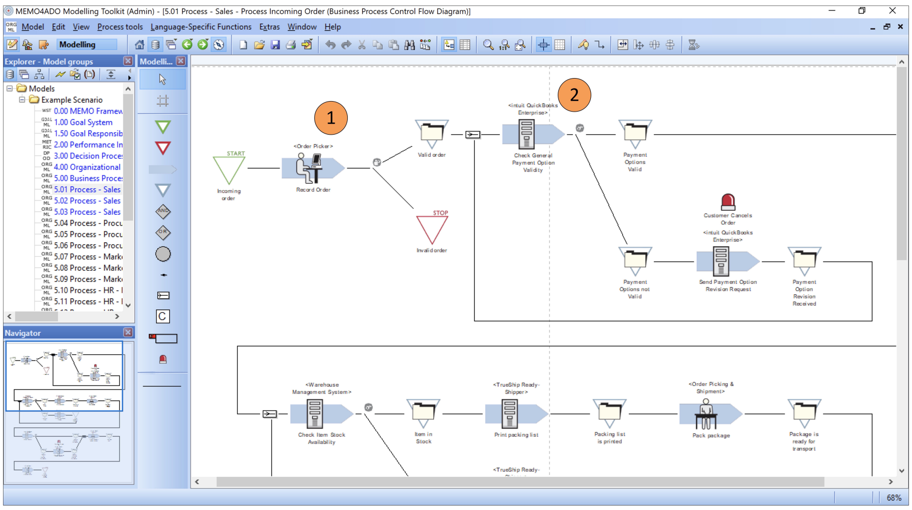


Fig. 7: MEMO4ADO: Business Process Control Flow

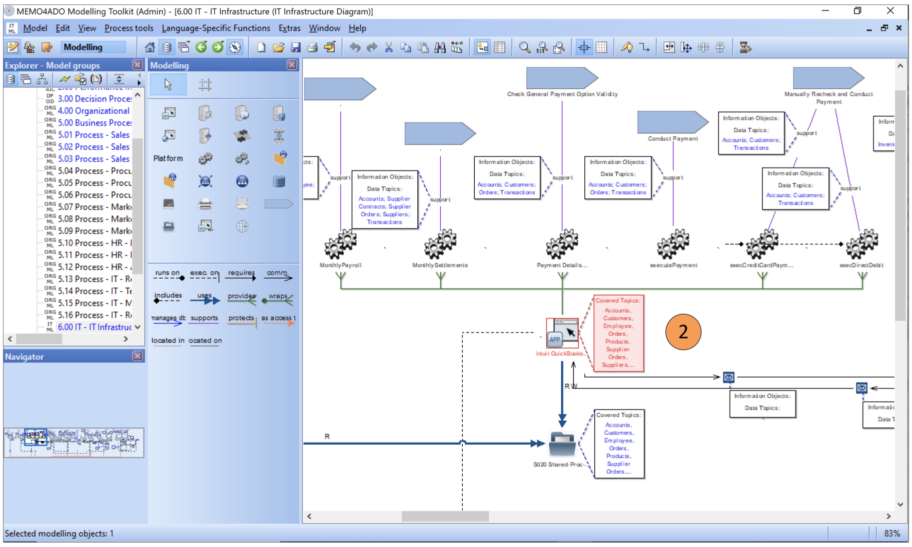


Fig. 8: MEMO4ADO: IT Infrastructure Diagram

of the business process model with a model of the IT infrastructure, cf. 'intuit QuickBooks Enterprise' Application Software, Fig. 8, supporting the activity 'Check General Payment Option Validity', Fig. 7, (2). On the other hand, the organizational units that are assigned to each activity in the process model serve as links to corresponding elements in a related model of the organization structure, cf. 'Order Picker', Fig. 6, (1) performing the activity 'Record Order', Fig. 7, (1).

The level of detail of models will often gradually increase as a modeling project progresses. Take, for example, the analysis of investments into IT required to support the prospective business model. In this case, it might be reasonable to first create a high-level model of the current IT infrastructure, in order to assess its potential to realize future IT services. In a next step, a more elaborate model of a revised IT infrastructure could be developed. Specific kinds of analysis can be supported either by the dedicated language-specific functions (e.g., analysis of IT landscape heterogeneity), ADOxx's query mechanism, or by manual exploration and navigation through the created diagrams.

5 Conclusions

MEMO4ADO implements a selected and revised subset of the various MEMO DSMLs. It enables the user to develop an integrated, multi-perspective model of an enterprise, covering business processes, goal systems, IT infrastructures, and several other areas. To improve ease of use, and to overcome certain discrepancies between the architecture of MEMO and

ADOxx, several simplifications of the original design had to made in the implementation of MEMO4ADO. But the general character of MEMO has been retained in the tool.

Since the tool implements large portions of MEMO, it provides a platform to construct rich, detailed, and closely integrated multi-domain models of organizations and corresponding information systems. As illustrated previously, these models serve as a basis to answer a host of multidisciplinary analysis questions, especially pertaining to the integration of business and IT. Therefore, MEMO4ADO represents a versatile tool for IS professionals to adress the needs of a large variety of projects. So far as we know, its conceptual scope exceeds that of any other available enterprise modeling tool.

A further target user group of MEMO4ADO are students at the bachelor's and master's level. MEMO4ADO provides an accessible environment for students to familiarize themselves with the notion of multi-perspective enterprise modeling, and to explore the benefits of an integrated view of the organization. Another feature primarily directed to students is the integration of the 'basic' languages like ERM, DFD, and UML class diagrams. This capacity helps illustrate the advantages of using several modeling languages in tandem, and in the context of a more extensive enterprise model. We have been successfully using MEMO4ADO in different bachelor's and master's courses for several years now, with app. 200 students using the tool per year.

However, it is fair to say that precisely this conceptual complexity also remains the primary challenge to the applicability of MEMO4ADO. While it has been a general design goal of every MEMO language to reflect the subtleties and nuances of the subject, the cost has been a still rather long learning curve. To relax this conflict remains an objective of future work. Already existing features bearing on this issue include, for example, the ability to choose between different diagram detail levels.

MEMO4ADO remains under active development. We are currently working on a new version encompassing additional modeling languages, such as a new version of ITML, as well as new analysis functionalities.

Bibliography

[BF16] Bock, Alexander; Frank, Ulrich: Multi-perspective Enterprise Modeling—Conceptual Foundation and Implementation with ADOxx. In (Karagiannis, Dimitris; Mayr, Heinrich Christian; Mylopoulos, John P., eds): Domain-specific conceptual modeling, pp. 241–267. Springer, Cham, 2016.

[Bo15] Bock, Alexander Christopher: Beyond Narrow Decision Models: Toward Integrative Models of Organizational Decision Processes. In: 17th IEEE Conference on Business Informatics (CBI 2015). IEEE Computer Society, pp. 181–190, 2015.

[FB20] Frank, Ulrich; Bock, Alexander: Conjoint Analysis and Design of Business and IT: The Case for Multi-Perspective Enterprise Modeling. In (Kulkarni, Vinay; Reddy, Sreedhar; Clark, Tony; Barn, Balbir, eds): Advanced Digital Architectures for Model-Driven Adaptive Enterprises, pp. 15–45. IGI Global, 2020.

[FK13] Fill, Hans-Georg; Karagiannis, Dimitris: On the Conceptualisation of Modelling Methods Using the ADOxx Meta Modelling Platform. Enterprise Modelling and Information Systems Architectures, 8(1):4–25, 2013.

[FKHdK21] Frank, Ulrich; Kaczmarek-Heß, Monika; de Kinderen, Sybren: IT Infrastructure Modeling Language (ITML): A DSML for supporting IT management. Technical report, ICB-Research Report, 2021.

[Fr94] Frank, Ulrich: Multiperspektivische Unternehmensmodellierung: Theoretischer Hintergrund und Entwurf einer objektorientierten Entwicklungsumgebung. Oldenbourg, München, 1994.

[Fr11a] Frank, Ulrich: MEMO Organisation Modelling Language (1): Focus on Organisational Structure. ICB-Research Report 48, University of Duisburg-Essen, Essen, 2011.

[Fr11b] Frank, Ulrich: MEMO Organisation Modelling Language (2): Focus on Business Processes. ICB-Research Report 49, University of Duisburg-Essen, Essen, 2011.

[Fr11c] Frank, Ulrich: The MEMO Meta modeling Language (MML) and Language Architecture. 2nd Edition. ICB-Research Report 43, University of Duisburg-Essen, Essen, 2011.

[Fr14] Frank, Ulrich: Multi-Perspective Enterprise Modeling: Foundational Concepts, Prospects and Future Research Challenges. Software & Systems Modeling, 13(3):941–962, 2014.

[FS98] Ferstl, Otto K.; Sinz, Elmar J.: Modeling of business systems using the Semantic Object Model (SOM): A methodological framework. In (Bernus, Peter; Mertins, Kai; Schmidt, Günter, eds): Handbook on Architectures of Information Systems, International Handbooks on Information Systems, pp. 339–358. Springer, Berlin and Heidelberg, New York, 1998.

[GF10] Gulden, Jens; Frank, Ulrich: MEMOCenterNG – A full-featured modeling environment for organisation modeling and model-driven software development. In: Proceedings of the 2nd International Workshop on Future Trends of Model-Driven Development (FTMDD 2010). 2010.

[OFK15] Overbeek, Sietse; Frank, Ulrich; Köhling, Christian Alexander: A Language for Multi-Perspective Goal Modelling: Challenges, Requirements and Solutions. Computer Standards & Interfaces, 38:1–16, 2015.

[Sa14] Sandkuhl, Kurt; Stirna, Janis; Persson, Anne; Wißotzki, Matthias: Enterprise Modeling: Tackling business challenges with the 4EM method. The enterprise engineering series. Springer, Berlin [u.a.], 2014.

[St12] Strecker, Stefan; Frank, Ulrich; Heise, David; Kattenstroth, Heiko: MetricM: a modeling method in support of the reflective design and use of performance measurement systems. Inf. Syst. E Bus. Manag., 10(2):241–276, 2012.

[Za87] Zachman, John A.: A framework for information systems architecture. IBM Systems Journal, 26(3):276–292, 1987.

J. Michael, J. Pfeiffer, A. Wortmann (Hrsg.): Modellierung 2022 Satellite Events,
Digital Library, Gesellschaft für Informatik e.V. 256

Modellierungsumgebung zur Erzeugung synthetischer Ereignisprotokolle für das Process Mining

Yorck Zisgen[1], Dominik Janssen[2], Christian Imenkamp, Agnes Koschmider[3]

Abstract: Process Mining hat in verschiedenen Bereichen einen erheblichen praktischen Nutzen erlangt. Der Ausgangspunkt von Process Mining ist ein Ereignisprotokoll, das die Ausführung von Aktivitäten beinhaltet, die einem Geschäftsprozess zugeordnet werden können. Somit hat die Qualität von Ereignisprotokollen erhebliche Auswirkungen auf das Process Mining-Ergebnis. Die Verwendung von Process Mining in neuartigen Anwendungsszenarien oder experimentellen Umgebungen scheitert aktuell daran, weil kaum geeignete Ereignisprotokolle öffentlich zugänglich sind. Dieser Beitrag stellt ein Werkzeug zum Generieren synthetischer (Sensor-)Ereignisprotokolle vor. Im Vergleich zu bestehenden synthetischen Protokollgenerator-Werkzeugen erzeugt der IoT-Prozessprotokollgenerator Daten auf nicht deterministische Weise. Benutzer können Rauschen kontrolliert hinzufügen und Prozesse mit IoT-Daten erweitern. Auf diese Weise ermöglicht das Werkzeug die Generierung synthetischer Daten für IoT-Umgebungen, die individuell konfiguriert werden können. Somit leistet unser Werkzeug einen Beitrag zu einem verstärkten Einsatz von Process Mining in Umgebungen, die auf (IoT-) Sensorereignisdaten angewiesen sind.

Keywords: Prozessmodellierung; Internet of Things; Ereignisprotokoll; Synthetische Daten; Geschäftsprozess-Simulation; Process Mining.

1 Einleitung

Process Mining und Internet-of-Things (IoT) sind zwei Gebiete, die sich sehr gut ergänzen. Denn IoT-Umgebunden produzieren große Datenmengen, die die Methoden des Process Mining für eine gute Prozessanalyse benötigen [Ja20]. Process Mining wiederum erlaubt die effiziente Analyse dieser Daten und ermöglicht Einblicke in IoT-gestützte Prozesse. Allerdings weisen aufgezeichnete IoT-Daten eine Herausforderung im Hinblick auf Datenqualität auf. Durch das Fehlen von Daten, oder eine unvollständige Aufzeichnung wird die direkte Anwendbarkeit von Process Mining Methoden erschwert. Semantisch gesehen sind Sensorereignisdaten auf einem viel niedrigeren Niveau als Ereignisdaten, die üblicherweise auf einem wesentlich höheren Abstraktionsniveau aufgezeichnet werden. Somit können Sensorereignisdaten nicht direkt von Process Mining Werkzeugen verarbeitet werden.

[1] yzi@informatik.uni-kiel.de
[2] dominik.janssen@informatik.uni-kiel.de
[3] ak@informatik.uni-kiel.de
Christian-Albrechts-Universität zu Kiel, AG Process Analytics, Institut für Informatik,
Hermann-Rodewald-Str. 3, Kiel

In diesem Beitrag stellen wir den *IoT-Prozessprotokollgenerator* vor, der es erlaubt Ereignisprotokolle aus simulierten IoT-Umgebungen in verschiedenen Qualitätsstufen für Process Mining Verfahren zu erstellen. Das Werkzeug kann sowohl einfache Ereignisprotokolle ohne IoT Bezug generieren, als auch IoT-Sensorereignisprotokolle wie sie für IoT-Umgebungen typisch sind. Um den Erstellungsprozess aus Anwendersicht zu vereinfachen, wurde die Simulations-Engine mit einer benutzerfreundlichen grafischen Oberfläche entwickelt. Benutzer können Prozesse grafisch modellieren und die Aktivitäten im Modell im Bezug auf Dauer und Häufigkeit konfigurieren. Zusätzlich können verschiedene Arten von Rauschen in beliebiger Menge dem Ereignisprotokoll hinzugefügt werden. Optional kann eine IoT-Umgebung mit verschiedenen Arten von IoT-Sensoren dem Prozessmodell zugeordnet werden. Die hieraus resultierenden Ereignisprotokolle können schließlich für verschiedenste IoT-Umgebungen, mit unterschiedlichen Sensor- und Fehlertypen erstellt werden und für experimentelle Zwecke verwendet werden. So können zum Beispiel Bewegungssensoren mit diskreten (d.h., `EIN` und `AUS` Zuständen) simuliert werden, aber auch Temperatursensoren mit stetigen Sensorwerten. Eine solche Simulation kann beispielsweise die Frage beantworten, ob das Ausstatten einer Umgebung mit IoT-Sensoren überhaupt empfehlenswert ist. Engpässe in Produktionskapazitäten lassen sich durch das Aufzeigen von Überlastungen in der Simulation aufdecken. Es wurde bereits gezeigt, dass synthetisch generierte Daten nicht nur ein valider Ersatz für echte Daten sind [Ch19, PWV16], sondern auch einen Einblick in domänenspezifische Forschung geben kann [Tr18]. Wir sind davon überzeugt, dass unser IoT-Prozessprotokollgenerator die Anwendung von Process Mining Methoden in Fällen erleichtert, in der es erschwerten Zugang zu Daten gibt, oder die Datenqualität die Aussagekraft der Analyse einschränkt.

Der Beitrag ist folgendermaßen strukturiert: Kapitel 2 fasst die verwandten Arbeiten zusammen. Kapitel 3 stellt die allgemeine Struktur unseres Werkzeuges vor. In Kapitel 4 wird die Implementierung des Werkzeugs vorgestellt. Kapitel 5 demonstriert die Nützlichkeit unseres Werkzeuges am Beispiel von zwei Anwendungsfällen. Der Beitrag endet mit einer Zusammenfassung und einem Ausblick auf zukünftige Forschungsfragen.

2 Verwandte Arbeiten

Die folgenden Forschungsbereiche sind für unseren IoT-Prozessprotokollgenerator relevant: (1) IoT-Log-Generatoren und (2) Ereignisprotokoll-Generatoren.

Im Hinblick auf Kategorie 1) beschränken sich die verfügbaren Arbeiten bzw. Werkzeuge auf einen einzigen Sensortyp [Gi13, Pa13], eine spezielle Anwendungsdomäne [Ah19, Pa13], oder sie stellen eine IoT-Simulation nur konzeptuell vor [AMK18, Ch18, SSN17]. Zum Beispiel ist der Ansatz aus [Gi13, Pa13] beschränkt auf GPS Sensoren und deren Signalstärke und erlaubt keine weiteren Arten von Sensoren. Unser IoT-Prozessprotokollgenerator hingegen erlaubt das Hinzufügen von zusätzlichen Sensortypen, wie zum Beispiel An/Aus-, Bewegungs-, Licht-, oder Temperatursensoren.

IoT-Ereignisprotokoll-Generatoren wurden bereits auf verschiedene Anwendungsdomänen angewendet, zum Beispiel auf Mobilgeräte, Drahtlosnetzwerke oder cyberphysische Systeme. Kertesz et al. [KPG19], zum Beispiel, stellen einen Simulator für die Cloud-Kommunikation von mobilen IoT-Geräten generierten Sensordaten vor. Papadoupolos et al. [Pa13] beschäftigen sich mit der Signalstärke von Drahtlosnetzwerken. Ramprasad et al. [Ra19] schlagen einen Simulator für virtuelle IoT-Architekturen (EMU-IoT) vor, der eine Ende-zu-Ende Evaluation eines IoT Netzwerkes simulieren kann. Giménez et al. [Gi13] testen die Veränderung von Positionsdaten in Lagerhäusern, um Kollisionsszenarien zu testen. Ahmad et al. [Ah19] stellen eine Simulationsarchitektur vor, die sich mit der Kommunikation in Echtzeit IoT-Umgebungen befasst. Verfügbare IoT-Simulatoren sind demnach üblicherweise auf ein sehr enges Anwendungsfeld beschränkt. Eine Übertragbarkeit auf andere Domänen ist nicht vorgesehen. Unser IoT-Prozessprotokollgenerator erlaubt jedoch Simulationen in variablen Umgebungen, zum Beispiel Smart-Homes, Smart-Factories oder etwa Krankenhausumgebungen.

Grundsätzlich können synthetische Ereignisprotokolle mit den gängigen Process Mining Werkzeugen wie CPN tool [JKW07], ProM [Va05] oder WoPed [EF08] generiert werden. So wurde auch in [Ci15] ein Log-Generator vorgestellt, der deklarative Prozessmodelle erstellen kann. Diese Werkzeuge erstellen jedoch deterministische Ereignisprotokolle die direkt aus dem Prozessmodell erstellt werden (d.h. es können keine Häufigkeiten oder Wahrscheinlichkeiten für bestimmte Traces spezifiziert werden). ProM erlaubt es zwar, dem Ereignisprotokoll Rauschen hinzuzufügen, eine Analyse der verfügbaren Werkzeuge zum Filtern von Rauschen in Ereignisprotokollen hat jedoch gezeigt, dass die Werkzeuge weder in der Lage waren, Rauschen dem Ereignisprotokoll adäquat hinzuzufügen, noch angemessen das Rauschen herauszufiltern [Ko21]. Unser IoT-Prozessprotokollgenerator hingegen erlaubt es, unterschiedliche Arten und Häufigkeit von Rauschen dem Ereignisprotokoll hinzuzufügen und gleichzeitig ein rauschfreies Ereignisprotokoll zur Verfügung zu stellen.

3 Implementierung

Das in diesem Beitrag vorgestellte Werkzeug lässt sich über einen Webbrowser erreichen. Benutzer können Geschäftsprozesse als Petri-Netze grafisch modellieren oder in Form von .PNML-Dateien importieren. Zusätzlich lassen sich Simulationseinstellungen, wie die Dauer von Aktivitäten, den simulierten Zeitraum oder verschiedene Arten und die Häufigkeit von Fehlern (*Noise*) festlegen. Darüber hinaus lässt sich mit dem IoT-Umgebungsmodellierungswerkzeug die betrachtete Umgebung (Fabrik, Wohnhaus, Stadt) mit ihren vorhandene Sensoren und deren Ausfallwahrscheinlichkeiten per Drag-and-Drop modellieren. Hieraus ergeben sich in der Simulation zwei unterschiedliche Arten von Ereignisprotokollen. Zum einen ein Ereignisprotokoll mit Zeiten und Aktivitäten, wie es üblicherweise im Process Mining betrachtet wird, zum anderen ein Sensorereignislog mit Sensorwerten. Die Simulationskomponente erwartet lediglich als Eingabeparameter ein Prozessmodell, die modellierte IoT-Umgebung und Simulationseinstellungen. Hierbei wird die Annahme gemacht, dass jeder Prozessschritt an einem oder mehreren möglichen Orten stattfindet. Aus der Verknüpfung der Modelle generiert die Anwendungsschicht ein Ereignisprotokoll, welches online betrachtet oder als .TXT- oder .CSV-Datei heruntergeladen werden kann.

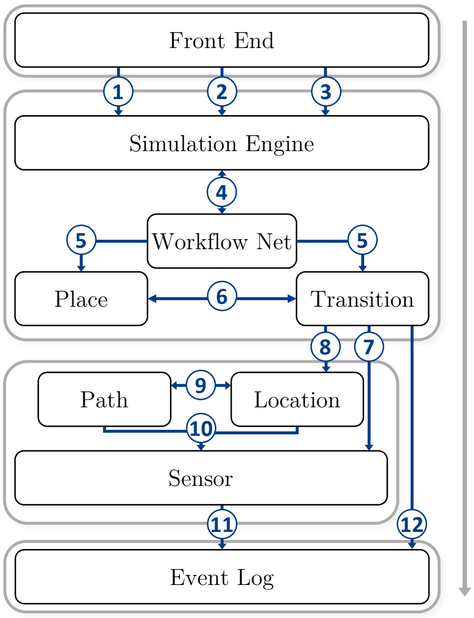


Abb. 1: Von der Modellierung über die Simulation zum Ereignisprotokoll

Zunächst werden die modellierten Geschäftsprozesse, die Simulationseinstellungen und die Wahl des Ausgabeformats an die Simulationsengine übermittelt (Abb. 1, Markierungen **1**, **2**, **3**). Die Simulationsengine erzeugt anschließend Prozessinstanzen aus den Geschäftsprozessmodellen (**4**). Die Prozessinstanzen (Workflow-Netze bestehend aus Transitionen und Stellen) werden unabhängig voneinander verarbeitet. Die Simulation von Geschäftsprozessen folgt hierbei den Regeln für Petri-Netze (**5**, **6**). Transitionen repräsentieren Aktivitäten, welche Orten in der IoT-Umgebung zugeordnet sind (**8**) und lösen Sensoren aus (**7**). Bewegungen innerhalb der IoT-Umgebung (**9**) können ebenfalls zu Sensorauslösungen führen (**10**). Sensormesswerte (diskrete, stetige oder zustandsbasierte Werte) werden gemäß den IoT-Einstellungen generiert und zusammen mit Zeitstempel und optionalem Rauschen an das Sensorereignislog übermittelt (**11**). Soll nur ein Ereignisprotokoll erzeugt werden, können Sensoren und IoT-Umgebung ausgelassen und die Prozessausführungen direkt ausgegeben werden (**12**).

Unser Werkzeug unterscheidet sich damit von Verarbeitungsskripten durch die Vermeidung deterministischer Vorgehensweisen, beispielsweise bei der Wahl, Ausführung und Reihenfolge modellierter alternativer oder paralleler Prozessschritte. Darüber hinaus werden

Aktivitätendauer und Sensorwerte zufällig gemäß angegebener Verteilung innerhalb eines modellierten Intervalls gesetzt. Daraus resultiert, dass alle möglichen Prozessausführungssequenzen im Ereignisprotokoll beobachtet werden können. Sensorauslösungen können zu unterschiedlichen Zeitpunkten innerhalb der Dauer einer Aktivität stattfinden, sowohl in einer vorgegebenen als auch in einer randomisierten Reihenfolge. Zusätzlich erlaubt unser Werkzeug die Definition von *'Regelfällen'* und *'Ausnahmefällen'* mittels unterschiedlicher Gewichtung von Wahrscheinlichkeiten an Kanten im Prozessmodell.

4 Modellierung und Simulation

Im Gegensatz zu anderen Werkzeugen zur Geschäftsprozessmodellierung beschränkt sich unser Werkzeug nicht nur auf die Modellierung, sondern ermöglicht, Prozesse benutzerorientiert zu konfigurieren. Somit lassen sich verschiedene Varianten von Ereignisprotokollen erzeugen. Abb. 2 zeigt einen Ausschnitt aus unserem Petri-Netz Editor. Stellen, Transitionen und Marken können per Drag-and-Drop bearbeitet werden. Die hier erstellten Petri-Netze bilden die Grundlage für die spätere Simulation und Erstellung des Ereignisprotokolls.

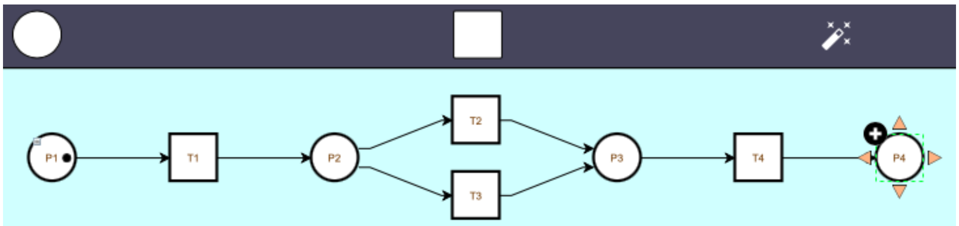


Abb. 2: Oberfläche des Petri-Netz Editors

Durch weitere Benutzereingaben können die zeitliche Dauer von Prozessschritten oder gewichtete Pfadverzweigungen im Prozessmodell definiert werden. Um neben einem Ereignisprotokoll auch ein Sensorereignisprotokoll erstellen zu können, müssen hierfür Sensoren und das IoT-Umfeld modelliert werden. In Abb. 3 ist eine solche IoT-Umgebung dargestellt. Die Ovale stehen für Orte bzw. Geräte und Maschinen an bestimmten Orten in der IoT-Umgebung, die Dreiecke sind verschiedene Sensortypen die mit den betreffenden Orten in Verbindung stehen.

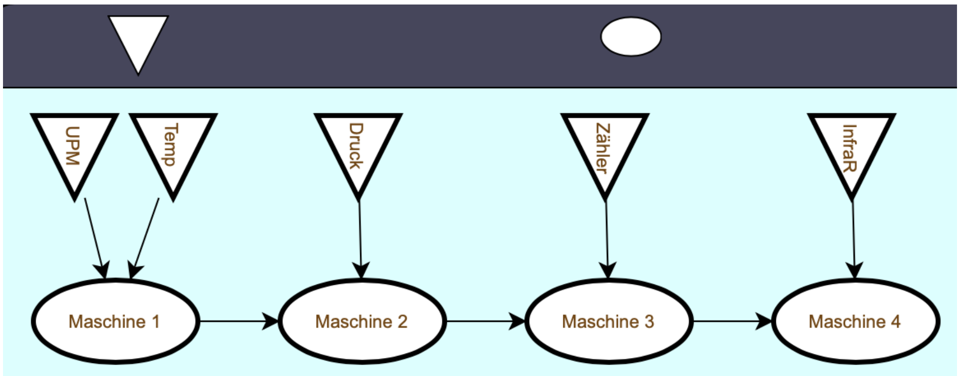


Abb. 3: Oberfläche des IoT-Editors

Für jeden Sensor kann die Art des Sensors festgelegt und eine Wahrscheinlichkeitsverteilung für die auftretenden Werte angegeben werden (Abb. 4).

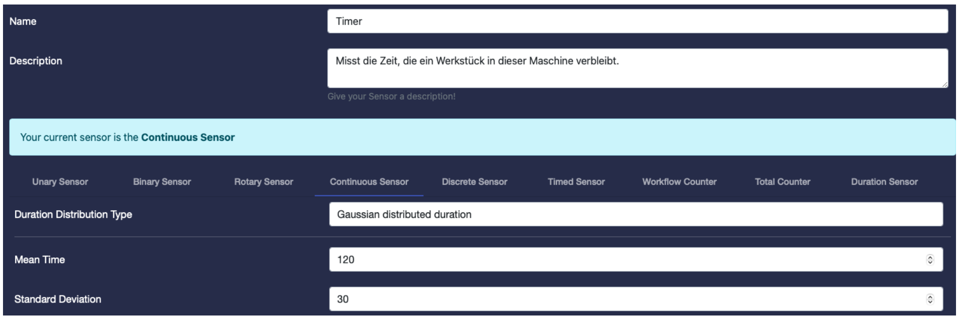


Abb. 4: Festlegung der Sensoreigenschaften

Auf Basis der modellierten Geschäftsprozesse und des IoT-Umfelds kann nun eine Zuweisung von Aktivitäten an Orte erfolgen. Hierzu kann der Nutzer aus einer Vielzahl vordefinierter Sensorarten wählen, um beispielsweise eine Smart-Home oder eine Smart-Factory Umgebung digital abzubilden.

Abhängig vom beabsichtigten Simulationszweck kann der Benutzer die Anzahl der Prozessdurchläufe, die benötigten Ausgabefelder, das z.B. Auftreten von Rauschen im Protokoll steuern. Abb. 5 und 6) zeigen die Konfiguration der Simulation und des Rauschens.

5 Beispielhafte Anwendung des IoT-Prozessprotokollgenerators

Im folgenden Abschnitt wird die Anwendbarkeit des Werkzeugs anhand zweier Anwendungsbeispiele demonstriert. Hierzu werden sowohl Krankenhausprozesse als auch Smart-Home

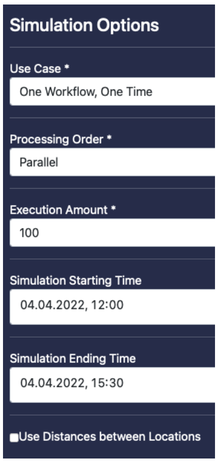


Abb. 5: Konfiguration: Simulation

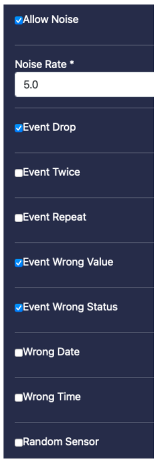


Abb. 6: Konfiguration: Rauschen

Prozesse simuliert. Um ein Ereignisprotokoll für Krankenhausprozesse zu generieren, wurde der Prozess von Elkoumy et al. [El22] angepasst. Das BPMN-Modell wurde in ein Petri-Netz übersetzt, Konfigurationen bezüglich der Dauer von Aktivitäten hinzugefügt und mit Rauschen erweitert. Anschließend wurde der Prozess simuliert. Basierend auf dieser Simulation wurde ein Ereignisprotokoll, siehe Tab. 1, erstellt. Die linke Tabelle zeigt hierbei das tatsächliche Ereignisprotokoll („Ground Truth“), während die rechte Tabelle ein um Rauschen verfälschtes Ereignisprotokoll darstellt.

Log - Clean		
Case ID	**Date Time**	**Activity**
846	2022-02-24 08:23	Register
846	2022-02-24 09:07	Hospitalize
846	2022-02-24 10:46	Blood Test
846	2022-02-24 11:18	Blood Test
846	2022-02-24 12:18	Visit
846	2022-02-24 13:12	Discharge

(a) Ursprüngliches Ereignisprotokoll

Log - Noise			
Case ID	**Date Time**	**Activity**	**Noise Type**
846	2022-02-24 20:23	Register	Wrong Time
846	2022-02-24 09:07	Hospitalize	Event Twice
846	2022-02-24 09:07	Hospitalize	Event Twice
846	2022-24-02 10:46	Blood Test	Wrong Date
			Event Lost
846	2022-02-24 12:18	Visit	Multi Recordings
846	2022-02-24 12:21	Visit	Multi Recordings
846	2022-02-24 12:22	Visit	Multi Recordings
846	2022-02-24 13:12	Register	Wrong Event

(b) Durch Rauschen verfälschtes Ereignisprotokoll

Tab. 1: Synthetisches Ereignisprotokoll zu Krankenhausprozessen

Das zweite Anwendungsbeispiel bezieht sich auf Smart-Home Umgebungen. Zu diesem Zweck wurde auf ein Sensorereignisprotokoll aus [CSE09] zurückgegriffen, welches

Aktivitäten von Bewohnern in einer Smart-Home Umgebung aufzeichnete. Es wurde eine entsprechende Umgebung modelliert, Sensoren hinzugefügt und alltägliche Aktivitäten im Haushalt wie *kochen*, *aufräumen*, oder *Frühstück zubereiten* simuliert. Die resultierenden Ereignisprotokolle mit und ohne Rauschen sind in Tab. 2a und 2b dargestellt.

Log - Clean		
Sensor ID	**Date Time**	**Value**
S1	2022-03-04 08:13	Off
S2	2022-03-04 08:17	On
S3	2022-03-04 08:25	Off
S4	2022-03-24 08:36	On
S5	2022-03-04 08:58	Off
F26	2022-03-04 09:33	96.22
S6	2022-03-04 09:42	On
F27	2022-03-04 09:56	0.493

(a) Ursprüngliches Sensorereignisprotokoll

Log - Noise			
Sensor ID	**Date Time**	**Value**	**Noise Type**
S1	2022-03-04 20:13	Off	Wrong Time
S2	2022-03-04 08:17	On	Event Twice
S2	2022-03-04 08:17	On	Event Twice
			Event Lost
S4	2022-03-24 08:36	On	Multi Recordings
S4	2022-03-24 08:36	On	Multi Recordings
S4	2022-03-24 08:37	On	Multi Recordings
S5	2022-04-03 08:58	Off	Wrong Date
S2	2022-03-04 09:33	On	Wrong Sensor
S6	2022-03-04 09:42	Off	Wrong Status
F27	2022-03-04 09:56	0.557	Wrong Value

(b) Durch Rauschen verfälschtes Sensorereignisprotokoll

Tab. 2: Synthetisches Sensorereignisprotokoll einer Smart Home-Umgebung

6 Ausblick

Dieser Beitrag stellte den IoT-Ereignisprotokollgenerator vor. Das Werkzeug ermöglicht die Erzeugung von syntethischen (Sensor)Ereignisprotokollen für das Process Mining. Die synthetisch erzeugen Ereignisprotokolle können als "Ground-Truth"für z.B. die Kombination von Process Mining und maschinellen Lernverfahren verwendet werden. Nutzer können außerdem festlegen, wie viel Rauschen sie einem Ereignisproktoll hinzufügen möchten. Auf diese Weise können die synthetisch generierten Ereignisprotokolle dazu verwendet werden Process Discovery Algorithmen zu validieren, die Qualität von entdeckten Prozessmodellen zu erhöhen und neue Anwendungsszenarien für IoT-Sensorereignisdaten zu nutzen.

Bisher ist unser Generator auf die Erzeugung von Ereignisprotokollen auf Ein-Personen-Umgebung beschränkt. Wir planen den IoT-Modellierer dahingehen zu erweitern, dass dieser Mehr-Personen-Umgebungen unterstützt und eine rollenbasierte Aufgabenzuweisung ermöglicht. Außerdem ist es geplant, weitere Prozessmodellierungssprachen und Ausgabeformate zu unterstützen und die Prozessvisualisierung zu verbessern. Die nächste Version des Werkzeugs wird es ermöglichen, Prozesse mit BPMN 2.0 zu modellieren. Neben den Ausgabeformaten .csv und .txt, planen wir die Ereignisprotokolle auch als .xes-Dateien auszugeben. Ebenfalls planen wir die Integration sogenannter Seeds, um eine kontrollierte

Reproduzierbarkeit von Ereignisprotokollen zu ermöglichen. Um die visuelle Darstellung der Prozessmodelle zu verbessern, planen wir einen 3D-Modellierungsumgebung zu integrieren, um synthetische Daten für 3D-Umgebungen zu generieren. Die entsprechende 3D-Umgebung wurde in [WK22] vorgestellt.

Literaturverzeichnis

[Ah19] Ahmad, Shabir; Malik, Sehrish; Ullah, Israr; Park, Dong-Hwan; Kim, Kwangsoo; Kim, DoHyeun: Towards the Design of a Formal Verification and Evaluation Tool of Real-Time Tasks Scheduling of IoT Applications. Sustainability, 11(1):204, Januar 2019.

[AMK18] Ahmad, Shabir; Malik, Sehrish; Kim, Do-Hyeun: Comparative Analysis of Simulation Tools with Visualization based on Realtime Task Scheduling Algorithms for IoT Embedded Applications. International Journal of Grid and Distributed Computing, 11:1–10, Februar 2018.

[Ch18] Chernyshev, Maxim; Baig, Zubair; Bello, Oladayo; Zeadally, Sherali: Internet of Things (IoT): Research, Simulators, and Testbeds. IEEE Internet of Things Journal, 5(3):1637–1647, Juni 2018. Number: 3.

[Ch19] Chen, Junqiao; Chun, David; Patel, Milesh; Chiang, Epson; James, Jesse: The validity of synthetic clinical data: a validation study of a leading synthetic data generator (Synthea) using clinical quality measures. BMC Medical Informatics and Decision Making, 19(1):44, März 2019.

[Ci15] Ciccio, Claudio Di; Bernardi, Mario Luca; Cimitile, Marta; Maggi, Fabrizio Maria: Generating event logs through the simulation of declare models. In: Workshop on Enterprise and Organizational Modeling and Simulation. Springer, S. 20–36, 2015.

[CSE09] Cook, Diane; Schmitter-Edgecombe, Maureen: Assessing the Quality of Activities in a Smart Environment. Methods of information in medicine, 48:480–5, Juni 2009.

[EF08] Eckleder, A; Freytag, T: Woped a tool for teaching, analyzing and visualizing workflow nets. Petri Net Newsletter, 75:3–8, 2008.

[El22] Elkoumy, Gamal; Fahrenkrog-Petersen, Stephan A.; Sani, Mohammadreza Fani; Koschmider, Agnes; Mannhardt, Felix; von Voigt, Saskia Nuñez; Rafiei, Majid; von Waldthausen, Leopold: Privacy and Confidentiality in Process Mining – Threats and Research Challenges. ACM, 13(1):1–17, März 2022. Number: 1 arXiv: 2106.00388.

[Gi13] Gimenez, Pablo; Molina, Benjamin; Palau, Carlos E.; Esteve, Manuel: SWE Simulation and Testing for the IoT. In: 2013 IEEE International Conference on Systems, Man, and Cybernetics. IEEE, Manchester, S. 356–361, Oktober 2013.

[Ja20] Janiesch, Christian; Koschmider, Agnes; Mecella, Massimo; Weber, Barbara; Burattin, Andrea; Di Ciccio, Claudio; Fortino, Giancarlo; Gal, Avigdor; Kannengiesser, Udo; Leotta, Francesco; Mannhardt, Felix; Marrella, Andrea; Mendling, Jan; Oberweis, Andreas; Reichert, Manfred; Rinderle-Ma, Stefanie; Serral, Estefanía; Song, Wenzhan; Su, Jianwen; Zhang, Liang: The Internet of Things Meets Business Process Management: A Manifesto. IEEE Systems, Man, and Cybernetics Magazine, 6:34–44, 10 2020.

[JKW07] Jensen, Kurt; Kristensen, Lars Michael; Wells, Lisa: Coloured Petri Nets and CPN Tools for modelling and validation of concurrent systems. International Journal on Software Tools for Technology Transfer, 9(3):213–254, 2007.

[Ko21] Koschmider, Agnes; Kaczmarek, Kay; Krause, Mathias; van Zelst, Sebastiaan J.: Demystifying Noise and Outliers in Event Logs: Review and Future Directions. In: Business Process Management Workshops. Jgg. 436 in Lecture Notes in Business Information Processing. Springer, S. 123–135, 2021.

[KPG19] Kertesz, A.; Pflanzner, T.; Gyimothy, T.: A Mobile IoT Device Simulator for IoT-Fog-Cloud Systems. Journal of Grid Computing, 17:529–551, September 2019. Number: 3.

[Pa13] Papadopoulos, Georgios Z.; Beaudaux, Julien; Gallais, Antoine; Noël, Thomas; Schreiner, Guillaume: Adding value to WSN simulation using the IoT-LAB experimental platform. In: 2013 IEEE 9th WiMob. S. 485–490, Oktober 2013. ISSN: 2160-4894.

[PWV16] Patki, Neha; Wedge, Roy; Veeramachaneni, Kalyan: The Synthetic Data Vault. In: 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA). S. 399–410, Oktober 2016.

[Ra19] Ramprasad, Brian; Fokaefs, Marios; Mukherjee, Joydeep; Litoiu, Marin: EMU-IoT - A Virtual Internet of Things Lab. In: 2019 IEEE International Conference on Autonomic Computing (ICAC). S. 73–83, Juni 2019.

[SSN17] Sharif, Mohammad; Sadeghi-Niaraki, Abolghasem: Ubiquitous sensor network simulation and emulation environments: A survey. Journal of Network and Computer Applications, 93:150–181, September 2017.

[Tr18] Tremblay, Jonathan; Prakash, Aayush; Acuna, David; Brophy, Mark; Jampani, Varun; Anil, Cem; To, Thang; Cameracci, Eric; Boochoon, Shaad; Birchfield, Stan: Training Deep Networks With Synthetic Data: Bridging the Reality Gap by Domain Randomization. In: 2018 IEEE/CVF CVPRW. S. 969–977, 2018.

[Va05] Van Dongen, Boudewijn F; de Medeiros, Ana Karla A; Verbeek, HMW; Weijters, AJMM; van Der Aalst, Wil MP: The ProM framework: A new era in process mining tool support. In: International conference on application and theory of petri nets. Springer, S. 444–454, 2005.

[WK22] Wetzel, Manuel; Koschmider, Agnes: Entwicklung einer VR-Umgebung zur Exploration von Process-Mining. HMD Prax. Wirtsch., 59(1):37–53, 2022.