
GESELLSCHAFT
FÜR INFORMATIK

Dominik Bork,
Dimitris Karagiannis,
Heinrich C. Mayr (Hrsg.)

Modellierung 2020

19. – 21. Februar 2020
Wien, Österreich

Gesellschaft für Informatik e.V. (GI)

Lecture Notes in Informatics (LNI) - Proceedings
Series of the Gesellschaft für Informatik (GI)

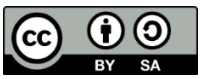
Volume P-302

ISBN 978-3-88579-696-1
ISSN 1617-5468

Volume Editors
Dr. Dominik Bork
Prof. Dr. Dimitris Karagiannis
Universität Wien, Fakultät für Informatik, Währinger Strasse 29, 1090 Wien
Email: dominik.bork@univie.ac.at, dimitris.karagiannis@univie.ac.at

Prof. Dr.Dr.h.c. Heinrich C. Mayr
Alpen-Adria-Universität Klagenfurt, Universitätsstraße 65 – 67, 9020 Klagenfurt
Email: heinrich.mayr@uni-klu.ac.at

printed by Köllen Druck+Verlag GmbH, Bonn



Vorwort

Die Fachtagung „Modellierung“ ist eine Plattform zur inhaltlichen Diskussion für eine große Anzahl von Fachgruppen in der Gesellschaft für Informatik (GI), die sich mit unterschiedlichsten Perspektiven des Themas Modellierung beschäftigen. Sie stellt somit ein zentrales Forum für den Erfahrungsaustausch zu akademischen wie auch praxisbezogenen Modellierungsansätzen dar.

Im Jahr 2020 wurde die Fachtagung „Modellierung“ an der Universität Wien ausgerichtet. Das Programm der Modellierung 2020 umfasste neben den wissenschaftlichen Fachvorträgen drei Workshops, vier Tutorien sowie ein Praxisforum und die Präsentation von Tools & Demos. Die Workshops behandelten sehr aktuelle Themen und zeigten auf, wie die Modellierung als etablierte Disziplin auch Antworten auf ganz aktuelle Fragestellungen liefern kann. Die Titel der drei Workshops waren wie folgt: „Modelle in der KI“, „Conceptual Modeling for Distributed Ledger Technologies“ und „Modellierung in der Hochschullehre“. Die vier Tutorien dienten dazu praktische Anwendungen aktueller Modellierungsansätze vorzustellen und die Teilnehmenden auch aktiv zu involvieren. Thematisch waren die Tutorien wie folgt konzipiert: „Komposition Domänenspezifischer Sprachen unter Nutzung der MontiCore Language Workbench am Beispiel SysML 2“, „Story Driven Modeling with the Fujaba Library“, „Models as Programs“, und „Multi-level Modelling with the FMML and the XModeler“.

Für das wissenschaftliche Programm der Modellierung 2020 wurden von insgesamt 23 Einreichungen die besten 11 als Langbeitrag ausgewählt. Dies entspricht einer Annahmequote von 47,8%. Weitere vier Einreichungen wurden als Kurzbeiträge akzeptiert und gemeinsam mit den Beiträgen zu den Workshops und Tools & Demos als Volume 2542 über CEUR-WS publiziert. Dieser Tagungsband umfasst alle 11 Langbeiträge sowie Kurzbeschreibungen der Tutorien. Die Begutachtung aller Einreichungen erfolgte durch ein doppelt-blindes Beurteilungsverfahren mit jeweils drei Gutachten pro Einreichung.

Die akzeptierten Beiträge behandeln aktuelle wissenschaftliche Erkenntnisse zu einer breiten Palette von Themen in den Bereichen Domänenspezifische Modellierung, Modellkonstruktion und kollaborative Modellierung, Modellierungsausbildung & Grundlagen der Modellierung, Modellsimulation und – verifikation, und Model-Driven Development.

Im Rahmen der Modellierung 2020 Tagung fand ein eingeladener Vortrag statt. Prof. Dr. Wolfgang Reisig behandelte das Thema „Komposition von Komponenten-Modellen: der Schlüssel zur Konstruktion großer Systeme“.

Wir danken allen Vortragenden für ihre Beiträge und den Mitgliedern des Programmkomitees für die zeitgerechte Erstellung der hochwertigen Gutachten. Ein großer Dank geht an die Organisatoren der Workshops und Tutorien, welche wesentlich zur Attraktivität und Vielfältigkeit der Konferenz beigetragen haben. Weiterhin bedanken wir uns bei allen Personen, die an der Organisation der Tagung in ganz unterschiedlichen Rollen beteiligt waren - insbesondere bei Victoria Döller für ihre umfangreiche Unterstützung.

Wien, Klagenfurt, im Februar 2020

Dominik Bork, Dimitris Karagiannis und Heinrich C. Mayr

Sponsoren

Wir danken den folgenden Unternehmen und Institutionen für die Unterstützung der Konferenz.

Universität Wien,
Fakultät für Informatik

Gesellschaft für Informatik

Alpen-Adria-Universität Klagenfurt

OMiLAB
www.omilab.org

Querschnittsfachausschuss Modellierung

Die Plattform der GI zur Diskussion und zum Erfahrungsaustausch über aktuelle und zukünftige Themen der Modellierungsforschung . Beteiligte GI-Gliederungen:

- ARC Architekturen
- ASE Automotive Software Engineering
- EMISA Entwicklungsmethoden für Informationssysteme und deren Anwendung
- FoMSESS Formale Methoden und Software Engineering für Sichere Systeme
- MMB Messung, Modellierung und Bewertung von Rechensystemen
- MobIS Modellierung betrieblicher Informationssysteme
- PN Petrinetze
- RE Requirements Engineering
- ST Softwaretechnik
- WI-VM Vorgehensmodelle für die betriebliche Anwendungsentwicklung
- WM Wissensmanagement

Tagungsleitung

Leitung des Programmkomitees:	Dimitris Karagiannis, Universität Wien
	Heinrich C. Mayr, Alpen-Adria-Universität Klagenfurt
Workshops:	Dominik Bork, Universität Wien
	Judith Michael, RWTH Aachen
Praxisforum:	Heinz Züllighoven, WPS-Workplace Solutions GmbH
	Harald Kühn, BOC Products & Services AG
DoktorandInnensymposium:	Ulrich Frank, Universität Duisburg-Essen
	Jörg Desel, FernUniversität Hagen
Tutorien:	Holger Giese, Hasso Plattner Institute for Digital Engineering, Universität Potsdam
	Susanne Leist, Universität Regensburg
Tools & Demos:	Hans-Georg Fill, University of Fribourg
	Agnes Koschmider, Christian-Albrechts-Universität zu Kiel
Organisationskommittee	Dominik Bork, Universität Wien
	Victoria Döller, Universität Wien

Programmkomitee

Jörg Desel	Fernuniversität Hagen
Jürgen Ebert	Universität Koblenz-Landau
Gregor Engels	Universität Paderborn
Peter Fettke	DFKI, Universität des Saarlandes
Hans-Georg Fill	University of Fribourg
Ulrich Frank	Universität Duisburg-Essen
Holger Giese	Hasso Plattner Institute an der Universität Potsdam
Martin Glinz	Universität Zürich
Florian Johannsen	Hochschule Schmalkalden
Jan Jürjens	Universität Koblenz-Landau
Gerti Kappel	Technische Universität Wien
Dimitris Karagiannis	Universität Wien
Walter Koch	Schaeffler AG
Agnes Koschmider	Universität Kiel
Thomas Kuehne	Victoria University of Wellington
Florian Matthes	Technische Universität München
Heinrich C. Mayr	Universität Klagenfurt
Jan Mendling	Wirtschaftsuniversität Wien
Judith Michael	RWTH Aachen
Daniel Moldt	Universität Hamburg
Günther Müller-Luschnat	
Friederike Nickl	Swiss Life Deutschland
Markus Nüttgens	Universität Hamburg
Andreas Oberweis	Karlsruher Institut für Technologie

Sven Overhage	Universität Bamberg
Henderik Proper	Luxembourg Institute of Science and Technology
Ulrich Reimer	FH St. Gallen
Wolfgang Reisig	Humboldt-Universität zu Berlin
Anne Remke	Universität Münster
Matthias Riebisch	Universität Hamburg
Stefanie Rinderle-Ma	Universität Wien
Bernhard Rumpe	RWTH Aachen
Kurt Sandkuhl	Universität Rostock
Klaus Schmid	Universität Hildesheim
Andy Schürr	TU Darmstadt
Christoph Seidl	IT University of Copenhagen
Elmar Sinz	Universität Bamberg
Friedrich Steimann	Fernuniversität Hagen
Stefan Strecker	Fernuniversität Hagen
Peter Tabeling	gematik, Berlin
Bernhard Thalheim	Universität Kiel
Marina Tropmann-Frick	HAW Hamburg
Andreas Vogelsang	TU Berlin
Matthias Weske	Hasso Plattner Institute an der Universität Potsdam
Manuel Wimmer	Johannes Kepler Universität Linz
Robert Winter	Universität St. Gallen
Heinz Züllighoven	Universität Hamburg

Zusätzliche GutachterInnen

Hagen Barth	Fernuniversität Hagen
Manuela Dalibor	RWTH Aachen
Victoria Döller	Universität Wien
Felix Härer	University of Fribourg
Dominik Huth	Technische Universität München
Dominik Janssen	Universität Kiel
Jörg Christian Kirchhof	RWTH Aachen
Lars Luthmann	TU Darmstadt
Benjamin Ternes	Fernuniversität Hagen

Inhaltsverzeichnis

Eingeladener Vortrag

Wissenschaftliche Beiträge

Tutorien

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 11

Komposition von Komponenten-Modellen: der Schlüssel zur Konstruktion großer Systeme

Wolfgang Reisig[1]

Abstract: Verfeinern und Komponieren sind fundamentale Prinzipien zur systematischen Konstruktion großer, rechnerintegrierter Systeme und Systemmodelle. Zu Fragen der Verfeinerung von Modellen wurden im Lauf der Zeit allgemeine Konzepte und Entwurfsprinzipien entwickelt. Zu Komposition von Komponenten gibt es bisher wenig Entsprechendes; stattdessen werden in der Literatur vielfältige spezielle Techniken und heterogene Kompositions-Operationen vorgeschlagen. In diesem Vortrag diskutiere ich Konzepte und Richtlinien zum Verständnis der Komposition von Komponenten-Modellen als generelles Prinzip des Systementwurfs. Algebraische Aspekte und weitere Eigenschaften des Komponierens zeigen, dass es sich lohnt, Komposition nach diesen Prinzipien zu gestalten. Beispiele aus unterschiedlichen Bereichen, darunter Petrinetze, BPMN, UML, interface description languages und Service-orientierten Architekturen, etc. unterstreichen den Nutzen der Konzepte.

[1] Humboldt-Universität zu Berlin, Berlin, Deutschland. reisig@informatik.hu-berlin.de

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 13

Towards Aligning Business Models with Business Processes: A Tool-based Approach

Thorsten Schoormann[1], Simon Hagen[2], Jonas Brinker[3], Sebastian Wildau[1],
Oliver Thomas[2 3], Ralf Knackstedt[1]

Abstract: In an increasingly dynamic environment, organizations need to be able to constantly adapt their business model. When making decisions in terms of adapting a business model, the operative structures that help to implement these adaptations need to be considered as well. Changes in both layers, the business model as well as the business processes, have immediate impacts on each other, and thus, should be aligned in order to allow more informed decision making. In this article, we report on a research project that explores how business models and processes can be aligned by iteratively building and evaluating a prototypical software platform. Doing this, this study contributes to the design of software that facilitates the alignment of both layers as well as provides a foundation for deriving more advanced knowledge such as in the form of design principles.

Keywords: Business model, Business process, Alignment, Software platform, Graph-database

1 Introduction

An increasing digitalization opens plenty of new avenues such as for developing novel business models but also constitutes a number of challenges like the need for constantly transforming organizational structures [KSK18]. In order to achieve sustainable advantage and to survive in uncertain and complex environments, organizations need to be able to reconsider and adapt their core business logic (e.g., [Ve14] [Bo13] [Va12])—especially in fast-moving markets like those that are concerned with evolving (digital) technologies. As adapting business models is an ambitious task, corresponding decisions are often made to late [Bo13]. One approach that can facilitate this, is an enhanced alignment of the business model with the business processes that operationalize a certain business model. Changes in the business model layer have direct impacts on the underlying business processes [Va12] and vice versa. This also in line with core elements of business process management (BPM) which argue that 'strategic alignment' (i.e., processes need to be aligned with the overall strategy of a company) and 'govervance' (i.e., design of decision-making to guide process-related actions) need to be adressed [RB10]. According to [Os04, p. 147], an "area of contribution could be the improvement of business process design due

[1] University of Hildesheim, Information Systems and Entperise Modelling, Universitätsstraße 1, 31141 Hildesheim,{thorsten.schoormann; ralf.knackstedt}@uni-hildesheim.de
[2] University of Osnabrück, Information Management and Information Systems, Parkstraße 40, 49080 Osnabrück,{simon.hagen; oliver.thomas}@uni-osnabrueck.de
[3] Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI), Smart Enterprise Engineering, Parkstraße 40, 49080 Osnabrück, jonas.brinker@uni-dfki.de

to a better understanding of the business model". Doing so, the process designer is able to do more informed decisions in terms of how to implement an abstract vision [Os04].

To support the act of adapting a business model, visual thinking is indispensable [OP10]. Accordingly, visualization has been determined as the main tool for developing and analyzing business models [TA17]. Various benefits from visualization such as an enhanced communication of a model, better generating of new business ideas as well as facilitating collaboration, reducing complexity and uncovering hidden structures within a business model are already explored in previous research (e.g., [EH11] [Os04]). However, such visualizations are mostly abstract, wherefore the integration or linkage of additional conceptual models like process models should be investigated. Business processes enable transferring business model elements into more detailed information [Al08] [Os04] by, for instance, specifying the component 'key activities' via process models that describe these activities through in-depth process flows. To link both models, two main directions are conceivable, namely (*bottom-up*) business processes as a unit of analysis that provides meaningful information for (re-)designing and planning business models [Bo13], and (*top-down*) business models as an orientation for implementing business processes [Va12].

However, although some approaches are already explored, there is still potential regarding how both layers can be connected and, especially, how these can be supported by software, which is evident by, for example, [Va12, p. 8] who called for an "analysis and verification of the role of description languages and software tools." This lack is problematic because it hinders the alignment of business models and processes as well as inhibits business model developers and process designers in making informed decisions. Hence, the question for this research study is formulated as follows: *How to design a software-supported alignment of business models with underlying business processes?*

To address this question, we first outline the research background of business models and business processes as well as the related work dealing with the alignment of both (Section 2). Following design science research (Section 3), we build and evaluate our main artefact, namely a software platform (Section 4) and demonstrate its applicability through a use case (Section 5). Finally, we discuss the results and conclude (Section 6). This article provides a concept and an explanatory instantiation of a software-supported alignment, which can be employed, for instance, by practitioners to make extensive analysis across their business models and processes to make more informed decisions.

2 Research background

2.1 Business models and their representations

Business models are often seen as an enabler for innovation and have emerged as a valuable unit of analysis and starting point for implementing new ideas [MTA17]. Accordingly, a good business model is essential for every successful organization—

regardless of whether it is a new business or an established player [Wi16] [Ma02]. The business model concept is widely understood as a (management) tool that facilitates different activities, for example: design, analysis, understanding, and evaluation of a company's core business logic [Ve14], visualization of a business logic, comprehension of the key elements in a specific domain [Os04], communication of such an understanding [Ga01], and assessment of the viability of new ideas [WV01].

Therefore, different business model modeling languages (BMMLs) are proposed that help structure business models [JKS17], for example: [Wi16] differentiate between strategy components (strategy, resources, and network), customer and market components (customer, market offer, and revenue), and value creation components (manufacturing, procurement, and financial). [OP10] describe a business model through nine *components* (value proposition, customer segments, channels, customer relationships, key resources, key activities, key partners, revenue streams, and cost structure), whose instantiations (or *elements*) serve to describe a concrete business model. However, regardless which approach or set of components is followed, one of a business model's central aspects is the processes that help to implement the core logic into a company as well as specific important activities that need to be performed for ensuring success.

2.2 Business process models

BPM research provides practice-oriented contributions by understanding, modeling, implementing, and optimizing business processes as well as providing corresponding methods such as modeling techniques and (software) tools (e.g., [Ha10] [HC93] [BR07] [SB10]). In essence, a process describes a logical, sequential order of operational events using and transforming resources to the main product or service outputs of a company [BS04]. According to van der Aalst, BPM-related tasks are often based on general phases for "(re-)design – implement/configure – run/adjust" [Aa13, p. 5]. Typically, these phases include steps for process identification, discovery, analysis, redesign, implementation, and monitoring [Du13].

For several decades now, business processes have been considered very important for the competitiveness of companies (e.g., [LDL03]). With the increasing complexity of companies from an organizational perspective and a technological perspective, the management of business processes has also gained in importance [Ja17], enabling a continuous adaptation and improvement of business activities [Tr10]. For this purpose, a number of modeling languages for the description and representation of business processes have been established [LK06] like, for instance, the *Event Driven Process Chain (EPC)* that is intended to present business processes with their activities, functions, and actors, or the *Business Process Modeling Notation (BPMN)* that has a broader set of elements and can be transformed into an executable model with the help of the *Business Process Execution Language* (BPEL).

2.3 Combining business models and business processes

In general, the business model concept can be seen as an intermediary between the strategy of a business and the business processes [Al08] [Ve14] (Fig. 1, grey cells mark the focus of this study). Whereas strategy, on a high level, attempts to explain how an organization will prevail over competitors, business processes describe the way of implementing a business model into daily business routines—the operative level transforming a set of inputs into outputs—which is important, because changes in the business model layer raise organizational questions as well [Os04]. IS research mainly lays its focus on the relationship between business models and business processes and seeks to explore how both concepts interact with each other [Ve14]. This interaction can take different forms such as *top down* (e.g., "the business model acts as the base system from which the detailed operational business process model should be derived", [Al08]) and *bottom up* (e.g., "monitor business processes in operations and to adjust the business model according to changes in business processes", [Bo13]). However, in order to enable innovation and improvement across an entire business, these layers require continuous alignment.

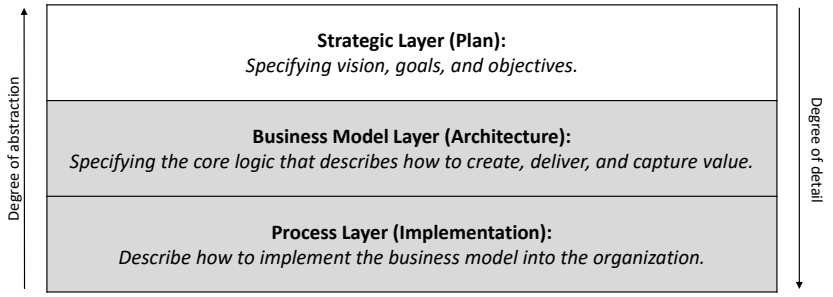


Fig. 1: Business model as a mediator between strategy and processes (adapted from [Os04])

3 Method

Design science research is appropriate to, for instance, explore IT support in the context of business model development (e.g., [Ve14]). For developing our main artefact, namely software supported alignment of business models with business processes, we first solve a specific problem (i.e., by building a concrete artefact) and seek to derive a more general solution afterward [Ii15]. Our research process is well-grounded by the method from Kuechler and Vaishnavi [KV08]. In total, we ran through several design cycles until the current state of our artefact (Fig. 2).

Following [KV08], we began with the *formulation of a concrete problem*, here the lack of software support that allows integration business models with more detailed models such as business processes (see also [Sz16]), for instance, to make use of detailed information during the analysis of business models or the dependencies between specific processes and

business decisions. To *suggest a solution* for this problem, we reviewed existing literature to obtain typical approaches for alignment of both layers. Drawing on the results, in the *development phase*, we started to implement the identified approaches in two available prototypes: A business model development tool (see [SBK18] for a more detailed description of the tool) and a service development platform that allows modeling business processes (see [Ka19] [Ha19]). For *evaluation purposes*, we accompanied and observed a project (across three months, i.e., one semester) that was performed by three students who were enrolled in a bachelor-level Information Systems program. Based on their feedback and our observation during the application, we were able to report some initial lessons learned for the current software platform.

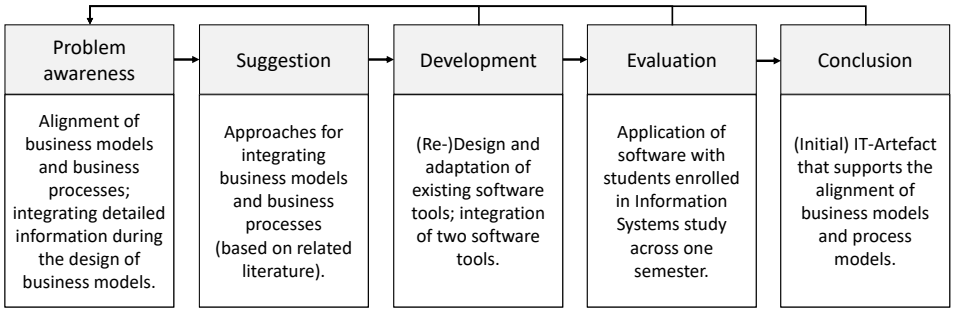


Fig. 2: Research process (adapted from [KV08])

4 A software platform for aligning business models and processes

4.1 Suggesting a solution concept

As there is only limited research on how to align, connect, or integrate business models with business processes, we started to develop an abstract architecture for structuring possible solutions that contribute to this field. Therefore, in line with other process landscape-approaches that structure different level of abstraction (e.g., [Oe03]), we seek to explore how an alignment, starting from the business model layer (i.e., top-down), can be designed. For this, a business model representation acts as an initial point to derive different approaches that support the connection of both layers (i.e., business model and process). To do so, based on a well-established modeling language for business models, the *Business Model Canvas*, an alignment on three different levels is conceivable (Fig. 3):

First, on **Level 0**, an entire business model is detailed by business processes. In doing this, the complete set on components and elements are used for deriving a business process model. For example, a Carsharing business model is specified by a process model that visualizes how value is created and how value delivered to a certain customer segment such as private households interesting in sharing mobility. In this case, the process might

also be on a rather abstract level, wherefore the value and helpfulness of one concrete process model requires further investigation.

Second, on **Level 1**, a single business model component including their elements can be detailed via a process model. For example, the component 'key activities' contains general tasks that need to be carried out in order to contribute to the value creation and value delivery of a company. These abstract activities can be specified through more detailed process models that help to represent information such as which concrete tasks need to be performed, which resources are required to execute a task, and which persons or roles are in charge of a task. As an example, the component key partners from a Carsharing business model can be specified by describing how external partners are involved that provide insurances for the shared vehicles (i.e., what task is performed by whom).

Third, on **Level 2**, a single business model element (i.e., a concrete element of a business model component) can be used to specify process models. For instance, the key activity 'cleaning cars' for a Carsharing business model can be specified by describing what types of cleanings are offered (e.g., in-cabin cleaning vs. rim cleaning vs. front shield cleaning etc.), what tools are required for a certain cleaning (e.g., brushes, sponge, soap), and which skills are demanded to execute it (e.g., staff that is trained with chemicals).

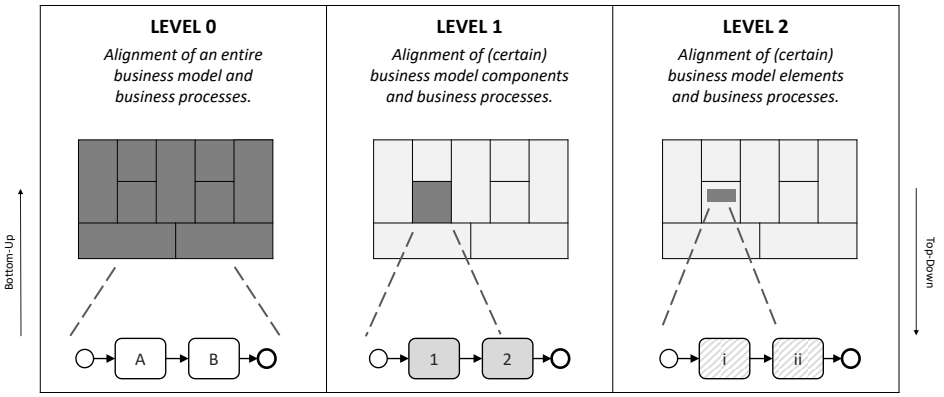


Fig. 3: Level of alignment (between the business model and business process models)

In order to respect the bottom up-perspective as well, we aim to provide a direct connection between both layers, which means an adjustment in the business model should affect the underlying process model elements and, vice versa, a change in a process model should lead to an update of the business model representation.

4.2 Developing an expository instantiation

Following the suggested solution concept, for the alignment of business models with process models, our software platform utilizes two independent web-based applications,

each covering one of the functionalities, and integrates them into one digital platform (Fig. 4). The exchange, and thus, the integration of both applications (and their underlying modeling languages) is realized with the help of an API interface, which enables merging both models in a single data-scheme.

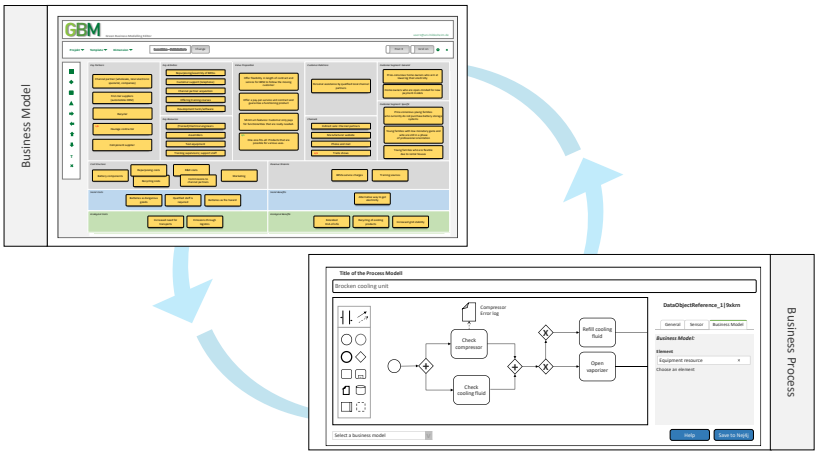


Fig. 4: Integrated business model (top) and process (bottom) modeling environments

To represent **business models**, a software-supported development environment is implemented by applying the modeling language Business Model Canvas [SBK18]. The application, a so-called business model development tool, provides several features for modeling a business model including representing the core business model elements by means of post-its. Typically, these tools allow to digitally represent and change business models, and as such, have the potential to support their users in performing certain actions more efficiently than with the 'pen & paper' versions of the modeling languages. To do so, they allow, for instance, assessing business models, filtering certain elements, versioning of business models, or collaborating within a team[4]. The information from the represented business model is initially stored in a relational database as well as is made accessible to the second application for process modeling via an API.

The **process modeling tool** is based on the freely available web application *bpmn.io*[5] and thus provides a user-friendly interface for modeling processes according to BPMN 2.0. For this study's purpose, an additional attribute has been implemented into the application, which contains the connection to the post-it's (i.e., business model elements) from the previously described business model development tool. This allows the user to assign business model elements to relevant parts of the process model. The BPMN element 'Data Object' is considered for this and can be, when selected, linked to the business model

[4] For an overview of typical features that are implemented by business model development tools, please refer to Szopinksi et al. 2019 [Sz19].

[5] *bpmn.io* is a BPMN 2.0 rendering toolkit and web modeler. For more information and downloads, please refer to: https://bpmn.io (accessed: 08.10.2019).

through a dynamic dropdown field, which is able to provide the entire set of elements from the business model (Fig. 5 on the right side). To limit the visible elements in the dropdown field, the related business model has to be chosen before modeling the process, since a process is used only within the scope of a single business model. If a process section is assigned to several business models or parts, additional data objects can be created.

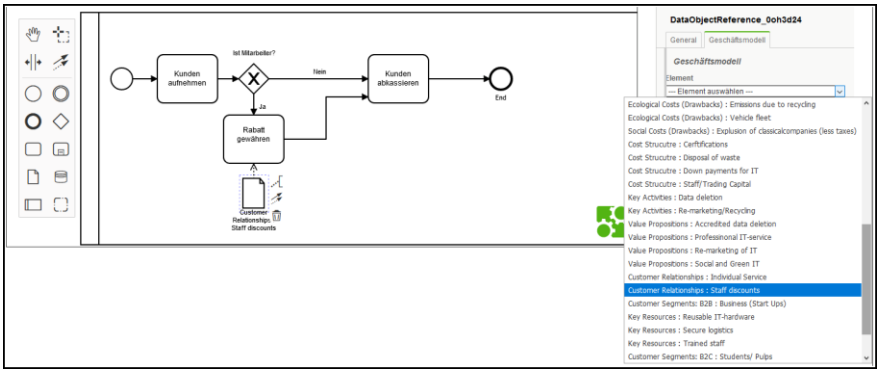


Fig. 5: Process modeling environment with linkage to the business model

Moreover, we tested a configurator that presents all elements from the business model as well as provides a possible target model element (i.e., elements in BPMN 2.0). Therefore, we stored an initial list of a possible connection between the business model components of the Business Model Canvas and standard objects of BPMN such as pools, data objects, and tasks. For example, a key partner might be transferred into a process pool or a key resource might be transferred into a data object (Fig. 6). If the user would like to specify another target element, she or he is free to select a more suitable type from the dropdown menu. In doing this, the user can select and add certain elements to the process modeling environment.

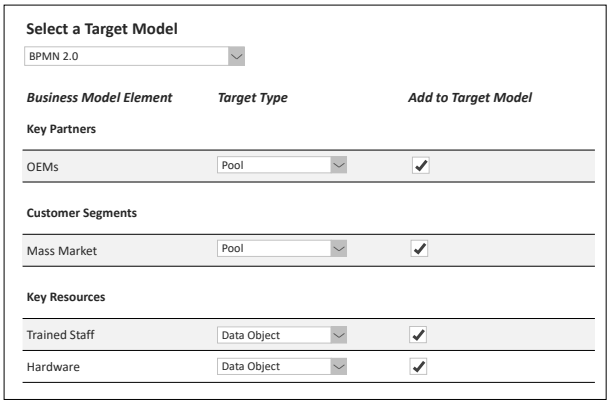


Fig. 6: Configurator to transfer business model elements into process model elements

From a technical perspective, the integration is facilitated by merging initially separated models into a single graph database. We decided to use graph-oriented storage technologies in the scope of our approach, since their implicit resemblance to our data-schemes (e.g. structural models), their examination in process related scopes [Aw07] and their fast query times, which become relevant when handling extensive processes and business models. Compared to other storage technologies, graph databases focus on the relationships between elements. The underlying structure is a graph consisting of individual nodes and edges, with each of these elements containing key-value pairs in which information is stored. The nodes can be connected arbitrarily by the edges to link the information and form mesh structures [BK14]. If a business model is imported, it is converted into such a tree structure, where the root node of the tree represents the business model and is related to its child notes, one per field of the BMC (value proposition, etc.). Each field-node holds relations to the above mentioned post-its (Fig. 7). Business processes are stored in the same way: The elements of the BPMN are represented by nodes with key-value pairs for their type, name, etc. and connected to each other via relationships. This shows the implicit proximity of process models and graph databases and speaks in favor of a shared use. The business model and process are linked via the post-it's mentioned above. For this prototype, *Neo4j*[6] was used as the underlying storage technology. As mentioned above, the graph database is addressed through an API that allows a direct connection between the modeling environment and data storage (Fig. 7).

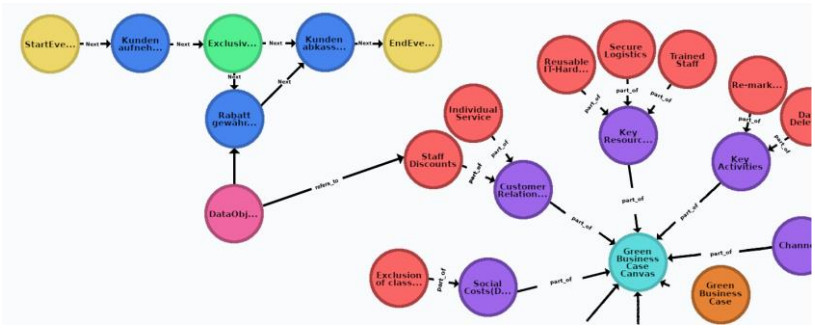


Fig. 7: Schematic representation of the graph structure
(linked process model, top-left, and business model, bottom-right)

5 Evaluation

For evaluating the software platform, we, as first step, carried out an ex ante artificial evaluation (e.g., [Ve16]) in an educational context. Artefacts that characterize the structure of a system are typically evaluated through the implementation of prototypes, which demonstrate the applicability via illustrative scenarios [Pe12]. Within a practice-oriented

[6] „Neo4j is a highly scalable native graph database, purpose-built to leverage not only data but also data relationships.“ For more information please refer to https://neo4j.com.

university course across three months, a group of three bachelor-level students from Information Systems applied the software platform to develop a business model for electrical vehicle charging stations and its underlying business processes. Moreover, three researchers from (1) business model development, (2) process design and (3) electrical vehicles act as experts for evaluating the group's ideas and providing feedback. In the following, we describe some lessons learned and justify these with statements from the project participants.

The results show that the software, in general, is a valuable asset for the integrated modeling of business processes and business models ("especially with the 'key activities' it would be appropriate to define these more precisely with processes."). The feedback is focused on the (visual) implementation of the environment. By choosing the BMC as a representation for business models and BPMN as a modeling language for processes, all users were able to employ the software easily. While the top-down approach was straightforward to apply ("we applied the top-down strategy for our models [in opposite to the bottom-up approach]"), building the business model based on the processes was perceived as difficult due to the lack of transitions from process elements to business model fields. Furthermore, direct linkage of, e.g., key activities to process steps was suggested in order to link the two layers more closely. Furthermore, through the demonstration, we could collect additional requirements for the software platform, for example: features for zooming across the process models, replacing buttons for saving a project, providing recommendations in terms of typical elements that detail other elements, as well as additional features for collaboration. Through an illustrative scenario, the students were able to iteratively specify which elements of the business model could be detailed by which elements of the BPMN. Therefore, they reported a log file in which they specified and justified which element can be detailed such as certain 'key activities' have been transferred into a process, certain 'key partners' have been transferred into pools and lanes, and certain 'key resources' have been transferred into data objects. The results from this inductive approach can be employed in further steps to specify concrete relations between specific business model and process model elements.

We aggregate the feedback into the following three essential lessons learned: First of all, the modeling language used (independent of the subject) has to be oriented on common languages and frameworks to leverage the full potential of our approach. An integration seems valuable and feasible, but a common understanding has to be met in order to benefit accordingly. Second, the users seek for support while performing the modeling, thus we propose a recommendation tool which helps the user (a) choosing the next or according elements and (b) indicate elements that need further detailing or already have similar functionality to reduce redundancies or adapt them appropriately. Third, the modeling environment should be flexible so that an individual working environment can be created.

6 Discussion and future research

For aligning business models and process models, this study introduces an abstract solution concept as well as a software platform consolidating modeling features for both perspectives (i.e., business model and process) based on graph databases. We evaluated the platform within a university course, which demonstrated the applicability of the artefact and revealed insights on missing or improvable features. Overall, the results indicate advantages of an integrated software platform for the users in terms of a coherent modeling environment and the resulting improved usability as well as an integrated data-model and storage in a shared database, enabling faster in-depth analysis.

Having the information of both models in a uniform data-scheme allows configuring and adapting existing processes into new business models and thereby generating plenty of 'meta-information' regarding the relation of processes and business models. Accordingly, this study has several implications for research and practice: First, (semi-)automatic derivation of processes or recommendations for common/standardized business processes. As there are already comprehensive catalogs of, for example, reference process models or standards and frameworks, it might be possible to assign these to specific business model elements. The software platform can be able to automatically recommend a set of processes based on a business model representation (e.g., typical processes for administrating a booking platform or for billing). Consequently, through an alignment, users can directly trace the impacts of adjusting a business model in terms of their underlying processes. These functions allow the user to leverage existing process models and to facilitate the transfer of process innovations in other domains (cross-industry innovation). Second, the unified data-structures enable the execution of business processes on information systems such as assistance systems, which enable, for example, the immediate roll-out of adjusted business processes when they are linked to the business model (e.g. as proposed by [Ka19]). Third, new opportunities and approaches for ensuring the compliance of a company are provided. For instance, because process models indicate detailed information regarding outputs of a certain task (e.g., higher negative impact on the environment by creating an increased amount of emissions), more informed decisions can be made on a business model layer. Accordingly, processes provide a valuable unit of analysis for both. Fourth, when altering the process of a key activity, this change can be reflected in the business model automatically. For example, the runtime of a business process can be measured or estimated (in terms of costs) and directly influence the overall value of the business model or new pools or lanes can lead to updated key partners in the business model.

Even though we derived some valuable insights, this study is however not free of limitations, which opens paths for future research. Fist, when integrating more elements and their representations of a business model into our software platform (e.g., pre-defined goods and CAD-models of products as key resources), even more relations between detailed information and the business model layer may be observed and managed centrally. Implementing and evaluating this potential extension will be part of our future research, in which we seek to make use of product-related models in particular that,

ultimately, help to represent business models of, for instance, entire product-service systems. Second, another limitation is that we primarily focus on the top down approach, and thus, mostly specify process elements based on a certain business model. In further investigations, the bottom up approach might be explored in order to analyze how the information provided within, for example, large process landscapes that often exist in companies, can act as a foundation for deriving further models. Third, in the current version of the platform, we focus on business process models as a type of detailed models that contribute to the representation and analysis of business models. In next steps, we aim to explore and integrate more types of information sources like, for example, by aligning further Canvas-based tools such as the *Value Proposition Canvas* [OP14] that might be fruitful for detailing the component value proposition (i.e., Level 1 alignment, see Section 4.1) or by aligning further modeling languages such as the *Entity Relationship Model* [Ch76] to represent data in its relationship for resources such as software applications. Finally, as our evaluation took place in an artificial setting, next steps should deal with observing the artifact in a naturalistic environment (e.g., by employing methods such as a case study or expert interviews). This extended evaluation should explore the usefulness of the software platform in addition to the already demonstrated applicability.

Overall, we hope that our work (i.e., a concrete software platform and an architecture for aligning business models and processes) provides orientation for practitioners and academics in terms of what approaches can be employed for integrating more detailed models such as business process models with business model representations. In doing this, we lay the foundations for further research that is concerned with, for instance, making more informed business model decisions, conducting more extensive analysis across an organization, or (semi-)automatic derivation of models.

Acknowledgments

This research is partly funded by the European Regional Development Fund and the State of Lower Saxony (NBank) in the scope of the research project SmartHybrid—Service Engineering (ID: 6-85003236) and Process Engineering (ID: ZW 6-85003451). We would like to thank them for their support.

Bibliography

[Aa13] van der Aalst, W. M. P.: Business Process Management. Software Engineering, pp. 1–37, 2013.

[Al08] Al-Debi, M. M.; El-Haddadeh, R.; Avison, D.: Defining the Business Model in the New World of Digital Business. In: Proc. of the Americas Conference on Information Systems (AMCIS). Toronto, Canada, 2008.

[Aw07] Awad, A.: BPMN-Q: A Language to Query Business Processes. In: EMISA 2007. St. Goar, Germany, pp. 115–128, 2007.

[Bo13] Bonakdar, A.; Weiblen, T.; Valentin, C. D.; Zeißner, T.; Pussep, A.; Schief, M.: Transformative Influence of Business Processes on the Business Model: Classifying the State of the Practice in the Software Industry. In: Proc. of the 46th Hawaii International Conference on System Science (HICCS), pp. 3920–3929, Hawaii, USA, 2013.

[BK14] Barmpis, K.; Kolovos, D. S.: Evaluation of contemporary graph databases for efficient persistence of large-scale models. Journal of Object Technology, 13(3), pp. 3-1, 2014.

[BR07] de Bruin, T.; Rosemann, M.: Using the Delphi Technique to Identify BPM Capability Areas. In: Proc. of the Australasian Conference on Information Systems (ACIS), Toowoomba, Australia, 2007.

[BS04] Becker, J.; Schütte, R.: Handelsinformationssysteme, MI Wirtschaftsbuch, 2004.

[Ch76] Chen, P. P. S.: The entity-relationship model—toward a unified view of data. ACM Transactions on Database Systems (TODS), 1(1), pp. 9-36, 1976.

[Du13] Dumas, M.; La Rosa, M.; Mendling, J.; Reijers, H. A.: Fundamentals of Business Process Management, Berlin, Heidelberg: Springer, 2013.

[EH11] Eppler, M. J.; Hoffmann, F.: Challenges and Visual Solutions for Strategic Business Model Innovation. In (Hülsmann, M.; Pfeffermann, N., eds.): Strategies and Communications for Innovations: An Integrative Management View for Companies and Networks, pp. 25–36, Berlin, Heidelberg: Springer, 2011.

[GA01] Gordijn, J.: Akkermans, H.: Designing and evaluating e-business models. IEEE Intelligent Systems, 16(4), pp. 11–17, 2011.

[Ha10] Hammer, M.: What Is Business Process Management? In (vom Brocke, J.; Rosemann, M., eds.): Handbook on BPM 1, pp. 3–16, Berlin, Heidelberg: Springer, 2010.

[Ha19] Hagen, S.; Brinker, J.; Gembarski, P.C.; Lachmayer, R.; Thomas, O.: Integration von Smarten Produkten und Dienstleistungen im IoT-Zeitalter: Ein Graph-basierter Entwicklungsansatz. HMD Praxis der Wirtschaftsinformatik, 56(6), pp. 1220-1232, Springer Fachmedien, 2019.

[HC93] Hammer, M. & Champy, J.: Re-engineering the Corporation: A Manifesto for Business Revolution. Harper Business, New York, 1993.

[Ii15] Iivari, J.: Distinguishing and contrasting two strategies for design science research. European Journal of Information Systems, 24(1), pp. 107–115, 2015.

[Ja17] Jannaber, S.; Zobel, B.; Riehle, D. M.; Thomas, O.; Becker, J.: Development of a Domain-Specific Language for Run-Time Process Modelling – Making Use of Wearables in BPM. In (Eibl, M.; Gaedke M., eds.): Lecture Notes in Informatics (LNI), Chemnitz, Germany, 2017.

[JKS17] John, T.; Kundisch, D.; Szopinski, D.: Visual Languages for Modeling Business Models: A Critical Review and Future Research Directions. In: Proc. of the International Conference on Information Systems (ICIS). Seoul, Korea, 2017.

[Ka19] Kammler, F.; Hagen, S.; Brinker, J.; Thomas, O.: Leveraging the Value of Data-Driven Service Systems in Manufacturing: A Graph-Based Approach. In: Proc. of the 27th European Conference on Information Systems (ECIS), Stockholm, Sweden, 2019.

[KSK18] Kutzner, K.; Schoormann, T.; Knackstedt, R.: Digital Transformation in Information Systems Research: A Taxonomy-based Approach to Structure the Field. In: Proc. of the European Conference on Information Systems (ECIS). Portsmouth, UK, 2018.

[KV08] Kuechler, B.; Vaishnavi, V.: On theory development in design science research: anatomy of a research project. European Journal of Information Systems, 17(5), pp. 489–504, 2008.

[LDL03] Lindsay, A.; Downs, D.; Lunn, K.: Business processes—Attempts to find a definition. Information and Software Technology, 45(15), pp. 1015–1019, 2003.

[LK06] List, B.; Korherr, B.: An evaluation of conceptual business process modelling languages. Proceedings of the 2006 ACM Symposium on Applied Computing, pp. 1532–1539. ACM, 2006.

[Ma02] Magretta, J.: Why Business Models Matter, Harvard Business Review, 2002.

[MTA17] Massa, L.; Tucci, C. L.; Afuah, A.: A Critical Assessment of Business Model Research. Academy of Management Annals, 11(1), pp. 73–104, 2017.

[Oe03] Oesterreich, B.: Objektorientierte Geschäftsmodellierung mit der UML. Heidelberg: Dpunkt-Verlag, 2003.

[OP10] Osterwalder, A.; Pigneur, Y.: Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers. John Wiley & Sons, 2010.

[OP14] Osterwalder, A.; Pigneur, Y.; Bernarda, G.; Smith, A.: Value proposition design: How to create products and services customers want. John Wiley & Sons, 2014.

[Os04] Osterwalder, A.: The business model ontology: A proposition in a design science approach (Doctoral Thesis). Université de Lausanne, Lausanne, Switzerland, 2004.

[Pe12] Peffers, K.; Rothenberger, M.; Tuunanen, T.; Vaezi, R.: Design Science Research Evaluation. In Design Science Research in Information Systems. Advances in Theory and Practice, pp. 398–410, Berlin, Heidelberg: Springer, 2012.

[RB10] Rosemann, M.; vom Brocke, J.: The six core elements of business process management. In Handbook on business process management 1, pp. 105-122, Springer, Berlin, Heidelberg, 2010.

[SB10] Scheer, A.-W.; Brabänder, E.: The Process of Business Process Management. In (vom Brocke, J.; Rosemann, M., eds.): Handbook on Business Process Management 2, pp. 239–265, Berlin, Heidelberg: Springer, 2010.

[Sc16] Schoormann, T.; Behrens, D.; Kolek, E.; Knackstedt, R.: Sustainability in Business Models–A Literature-Review-Based Design-Science-Oriented Research Agenda. In: Proc. of the European Conference on Information Systems (ECIS), Istanbul, Turkey, 2016.

[SBK18] Schoormann, T.; Behrens, D.; Knackstedt, R.: The Noblest Way to Learn Wisdom is by Reflection: Designing Software Tools for Reflecting Sustainability in Business Models. In: Proc. of the International Conference on Information Systems (ICIS), San Francisco, USA, 2018.

[Sz19] Szopinski, D.; Schoormann, T.; John, T.; Knackstedt, R.; Kundisch, D.: Software tools for business model innovation: Current state and future challenges. Electronic Markets, 2019.

[TA17] Täuscher, K.; Abdelkafi, N.: Visual tools for business model innovation: Recommendations from a cognitive perspective. Creativity and Innovation Management, 26(2), pp. 160–174, 2017.

[Tr10] Trkman, P.: The critical success factors of business process management. International Journal of Information Management, 30(2), pp. 125–134, 2010.

[Va12] Di Valentin, C.; Burkhart, T.; Vanderhaeghen, D.; Werth, D.; Loos, P.: Towards a Framework for Transforming Business Models into Business Processes. In: Proc. of the 18th Americas Conference on Information Systems (AMCIS), Seattle, USA, 2012.

[Ve14] Veit, D.; Clemons, E.; Benlian, A.; Buxmann, P.; Hess, T.; Spann, M.; Kundisch, D.; Leimeister J. M.; Loos, P.: Business Models - An Information Systems Research Agenda. Business & Information Systems Engineering, 6(1), pp. 45–53, 2014.

[Ve16] Venable, J.; Pries-Heje, J.; Baskerville, R.: FEDS: A Framework for Evaluation in Design Science Research. European Journal of Information Systems, 25(1), pp. 77–89, 2016.

[Wi16] Wirtz, B. W.; Pistoia, A.; Ullrich, S.; Göttel, V.: Business Models: Origin, Development and Future Research Perspectives. Long Range Planning, 49(1), pp. 36–54, 2016.

[WV01] Weill, P.; Vitale, M.: Place to space: Migrating to eBusiness Models. Boston, USA: Harvard Business School Press, 2001.

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 29

Model-Based Generation of Software Configurations in Mechatronic Systems

Martin Paczona[1] Heinrich C. Mayr[2] Guenter Prochart[3]

Abstract: An essential part of the mechatronic system is the software, which is responsible to bring functionality into the system consisting of mechanical, electronic and electrical parts. The software must be tailored to the specific hardware to fulfill tasks (e.g. control, monitoring) according to the system requirements. In today's industrial practice, the design is mainly done manually. First the entire architecture is drawn using drawing tools. Based on this software developers derive the low-level specification using their low-level development environments. This is error prone and time-consuming due to the fact, that a large number of hardware parameters have to be taken in account and the informal specification does not allow to derive these parameters. To improve this we present here an approach where the overall architecture of the mechatronic system is described using a Domain-Specific Conceptual Modeling Language (DSML) using the example of Electric Vehicle Testbeds. Based on this model the low-level software configurations are generated rule-based. In this paper we present the concepts of the DSML, explain the transformation rules and show the functionality of the generator by introducing a practical example.

Keywords: Modeling; Metamodel; Software; Generator; Transformation; Configuration

1 Introduction

Tuning the software for a mechatronic system is a tough job. This includes setting up the communication structure, finding the optimal control parameters, configure the IOs and implement the user interface. To develop this software and to setup the parameters cooperation of hardware developers, electrical designers and electronic designers is needed [To07]. Although comprehensive process models have been proposed for this purpose, they are hardly used in daily practice [SP09]. The reasons given include: too complicated, no time for processes, lack of tool support, lack of visible benefit, academic exercise. Furthermore, general-purpose modeling languages such as SysML and UML are only moderately accepted in the mechatronic domain because they do not provide the domain vocabulary and have a level of abstraction that discourages practitioners from using it. Instead of this process model still informal descriptions are used. Systems are described in drawing tools; based on this representation engineers collect the requirements and derive

[1] AVL List GmbH, Electrification Products, Hans-List-Platz 1, 8020 Graz,Austria martin.paczona@avl.com
[2] Alpen-Adria-Universität Klagenfurt, Application Engineering Research Group, Universitätsstraße 65-67, Austria heinrich.mayr@aau.at
[3] AVL List GmbH, Electrification Products, Hans-List-Platz1, 8020 Graz, Austria guenter.prochart@avl.com

the low-level software implementation. Now we will have an look on Electric Vehicle Testbeds (EVTs). EVTs are customized solutions for testing electric vehicles and electric vehicle components like batteries. The design of such testbeds is done by drawing the entire architecture in collaboration with the customer using a tool such as MS Visio. From this "specification", software developers, circuit plan designers and hardware developers derive detailed specifications and designs using their specific low-level design and development environments (e.g. EPLAN[EP19], Automation Studio[Bu19], SolidWorks[So19]). A further problem is that the software of a mechatronic system has a strong hardware connection, which is little considered in software development environments. Therefore, software and hardware are usually developed separately. Resulting inconsistencies often lead to system integration problems when the software is installed on the prototype system.

In addition, there is a high time pressure in the development of mechatronic systems. This applies in particular to EVTs caused by the E-Car boom [Fo18, KSK18]. Delivery times of 3-12 months, however, are common for EVT components [Ve19, Ro17, ec19]. The development therefore begins here with the specification of the hardware and its components, in particular, the long leads. Software design and development begin when the hardware has already been specified and is in production.

In order to accelerate and increase the efficiency of software development, the industry relies on the reuse of customizable software components. Software development for EVTs therefore essentially consists of configuration i.e., the selection of suitable modules, their parameterization and integration. In this paper, we propose an approach to automatically generating software configurations using EVT development as an example. This approach is based on a domain-specific conceptual modeling language (DSML) for the integrated description of hardware and software. A DSML is a language which target is a specific domain, therefore it has fewer more specific concepts rather than many generic concepts compared to a general-purpose language. The first step in the development of a DSML is the definition of the metamodel, which defines the domain concepts and the relation between the concepts. The approach is based on the "single source of truth paradigm" and avoids the inconsistencies previously mentioned as a problem. The conception of our DSML is such that the engineers can easily and intuitively understand it: to achieve this, we have conducted a series of surveys and practice workshops with stakeholders and domain experts. First experiences show a good acceptance, which is also due to the fact that our modeling language allows a very efficient and effective modeling due to its domain orientation. The further structure of the paper is as follows: Section 2 focuses on the conceptual aspects by introducing the metamodel, the DSML and the modeling tool developed. In chapter 3 we sketch the process of generator development followed by a detailed discussion of the rules for generating configurations from a model in chapter 4. Chapter 5 outlines the implementation of our approach using the metamodeling platform ADOxx. We also will show how our solution integrates into the overall development process. As a proof of concept, in chapter 6 (Evaluation) we compare the generator output with a set of manually developed artefacts.

After a short discussion of related work in chapter 7, the paper closes with a conclusion and an outlook on further work.

2 Software Configuration Model

Developing a Domain-Specific Modeling Method is a multi-stage and iterative process [MM15] comprising the development of an appropriate metamodel, the definition of one or, when required, more notations (representation languages on the model and data level according to the OMG Meta object Facility MOF [OM02]), the grounding of the metamodel concepts in an ontology, tool development, evaluation etc. [Ma18].

Since the overall metamodel and the notation principles are subject of another papers [PMP19, PM19], we limit ourselves here to the introduction of the concepts for software system and configuration modeling, as sketched in Fig. 1 Clearly, each concept comes with a set of meta-attributes that are not depicted in the figure but exemplarily mentioned in the modeling tool description below (e.g. Fig. 3). The concepts in the metamodel are connected with the generalization-relation.

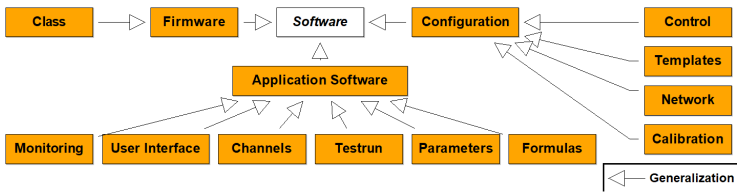


Fig. 1: Software Configuration Metamodel (excerpt).

Each of the concepts in the meta-model needs a proper representation. We refine here the basic notation by more specific elements for *Configuration*, *Firmware* and *Application Software*. To visualize the relationship between configuration concepts and other software-related concepts rectangle with tab), these elements use a rounded rectangle with a yellow color. Inside the shape a symbol indicates the type of *Configuration*. These Symbols correspond to the usual representations in automation and software. To make the meaning clearer as in the base representation, also a textual description is used [Mo09]. Fig. 2 shows the notation elements.

A modeling language without tool support has little chance of widespread use. Moreover, the automatic generation of software configurations needs a lot of attribute specifications which can only be handled efficiently by an appropriate modeling tool. An investigation of the requirements of non-software experts like Electrical Engineers, Managers, and Hardware Developers revealed that they require a clear and navigable overview over an EVT software configuration supporting intuitive understandability (see [PMP19]). Another important requirement of the domain experts is the availability of consistency checks. Based on the software configuration model (SCM), which is defined using the concepts of the metamodel (e.g. Control), the generator performs the transformations depending on the model structure

and model-element-attributes. Fig. 3 and Fig. 4 show excerpts of the attributes of the Control and Transformer elements based on the findings made during generator development (see chapter 3).

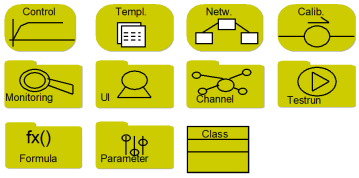


Fig. 2: Notation of the software-related concepts.

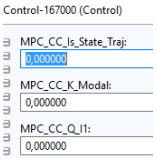


Fig. 3: Control Parameter Definition in the Circuit Plan Model.

Grounding a metamodel in an ontology strengthens it's semantic soundness. Domain ontologies are conceptualizations of aspects of a given domain shared by the respective user group [Ro11]. In a "sound "metamodeling language every modeling construct has a corresponding concept in the ontology [ES13]. Fig. 5 sketches the structure of an ontology draft which we have compiled for the domain of EVT software configuration.

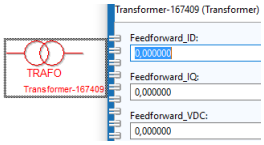


Fig. 4: Transformer Control Parameter Definition in the Circuit Plan Model.

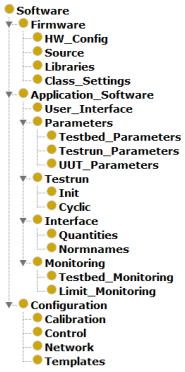


Fig. 5: EVT Software Configuration Ontology.

3 Generator Development

Since DSMLs provide conceptualizations of domain aspects, transformation rules can be directly linked to the concepts. This is an advantage over general-purpose modeling languages, for which common solutions are more difficult to find. In addition, the use of already available components can lead to faster results than building something from scratch [SK03]. If generators already have been implemented in the domain, they should be docked to the existing solution, for example by one generator generating the input for another. Before going into details, we will briefly explain the most important terms of code generation as well as the overall structure of the generator development process. The generator input, called source, is processed using transformation rules to produce the target. The generated

target is the code (or program) executed on the target platform. In our case the source is the EVT-Model (EVTM) including the circuit plan (CPM) and the SCM submodels; the target are the software configurations of the EVT components. Fig. 6 shows this process according to [Ku11]. The generator thus bridges the gap between the platform independent models and the platform model by adding platform specific information.

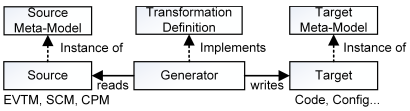


Fig. 6: Terminology of Code Generation (modified) [Ku11].

A relationship between the software and hardware elements is called cross reference [Ga10]. In Detail, a system element satisfies a requirement, if this requirement is implemented by a function which is realized by this system element. In chapter 4 we will present how this is implemented for the different EVTM transformation cases. The implementation of this cross-references and the code generation based on software and hardware models is one of the main challenges. The goal of the generation process is to increase the productivity and to increase the quality of the EVT-solution.

The generator development process can be divided into the following 7 steps:

1. *Domain Analysis*: Identifying the target (configuration files) to be generated and the information to be provided for these files (see Tab. 1).
2. *Structure Information*: dividing the information into “recurring“ information that may be compiled in templates (see step 5), and information requiring user parameterization.[SK03] propose the “piecemeal generator approach“. This means that initially most information is fixed in a template and then stepwise inserted into the models. Clearly, the number of possible generator outputs rises with the number of parameters and must represent valid configurations [Vo13].
3. *Check completeness*: Identifying information that has no correspondence in the models (source). This can be done by mapping each information unit to the corresponding model component. If an information cannot be mapped the model has to be extended accordingly (step 4).
4. *Update models*: Completing the models to cover the previously detected gaps, to ensure that complete configurations may be generated. In the EVT case, the decision, which model to complement, is basically driven by the user’s skills. For instance, if a new parameter is added to the CPM than circuit plan designers should know them.
5. *Templates*: Platform information which need not to be changed by the user is transferred into template files, the template files have to be created.
6. *Transformation rules*: The generator development starts by defining the source to target mapping (see Chapter 4).

7. *Evaluation*: In the last step the generator functionality is checked using a set of test-cases e.g. comparing with manual generated artefacts (see Chapter 6).

Description	*Type*	*Format*
Control Param., Calibration Data, Cabinet and Network Settings	Text	CSV
Version Information (Build, Package...)	Text	custom
PLC Hardware Configuration	Text	XML
Automation System Files	Script	PUMA/LYNX
Source Code Class-Definitions, Header-Files, Error Codes	Source	C++
Variable Definitions	Source	C++, ST

Tab. 1: Generator Output Overview (PUMA/LYNX are Test Autom. Systems from AVL List GmbH).

4 Transformation Rules

Transformation rules typically apply to a certain subset of model components [SK03]. In our case, we start from the EVTM, the CPM and the SCM which are defined by the domain-experts. Note that these "application models" [Ga10] together form the EVT-Model. The EVTM defines the overall structure of the EVT, the CPM the electrical part and the SCM specifies the software. The relevant components for which targets have to be produced then are *Source-Code*, *Error-Codes*, *Network-Settings*, *Control Parameters*, *Automation System Files*, and *Calibration Parameters*. Consequently, we define the transformation rules for these aspects including their scope and limitations. Fig. 7 gives an overview of the proposed transformation process. Based on the EVTM (and its submodels) a Raw Software Configuration is generated (RSC) using again the SCM notation. I.e., the RSC is an intermediate model, which the software developer may complement and refine, for example by individual code. After that the Software Configuration Generator (SCG) produces the final software configurations.

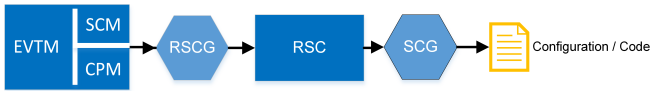


Fig. 7: Generator Process Structure Overview.

In the following we introduce the transformations for the different target types. For each type a short overview including the transformation rule and an excerpt of the generated output is given. The rules have been figured out by analysing the target artefacts in detail. A generic process for defining the rules, a formal definition of the rules and the development of generic rules for the mechatronic domain is part of future work. In the Rule definition *Output (O)* represents the result of the transformation, *Source (S)* the input of the transformation, *f()* a function and + connects two information sources. The rule definition is shown below.

Source: Source Model Source Component (Rule Notation)

Output: SCM Attribute= Source.Attribute + Platform Properties

4.1 Source-Code

The transformation rule for source-code generation assumes that the *Hardware_Component* including the *Hardware_Connections* contains most of the information needed for source code generation. During generator development it was found out, that only *Hardware_Components* need to be considered for the source-code generation which are directly connected to *IOs* of automation systems. This approach is inspired by the work presented in [Ba08] where each modeling element has a template file assigned. However, we do not use a template for each element but instead slightly different transformation rules for each modeling element. A pseudo-code version of the general rule is given below. This rule is refined for the particular elements. As an example consider a voltage sensor, which has the function to measure a voltage, and which is connected to an input channel of a control unit. The information produced by such a sensor differs from other types of elements. Since we use domain-specific models, the transformation rules can be tailored to such elements. This is a considerable advantage over the use of general-purpose modeling languages, where functions are defined based on requirements, and hardware elements based on these functions [Ga10]. In our approach, this relationship is exploited the other way round (see chapter 3). In case a sensor is connected to an analog input we can assume that the class of this sensor has a function *get_AI()*. Since the circuit model also defines the connection between the sensor and the IO we can derive the function *get_AI_+SensorName*. This makes the generated code easier to understand. We exploit this hardware relation also for generating the class and the variables. The abstract definition of the transformation shows R1 and R2. In both rules the *source (S)* is the *HW_Component* of the CPM and the Target is the generated code.

S: CPM HW_Comp (R1)
O: Class.Name= HW_Comp.Name + Platform_Info
S: CPM HW_Comp (R2)
O: Class.Prop= prefix + HW_Comp.Prop + connected HW_Comp.Prop, Platform_Info

4.2 Error-Codes

The transformation rule for error generation is based on the hypothesis that the CPM already contains most of the information needed for generating the error texts of a mechatronic system. To allow for customizing the error information of each *Hardware_Component* a string value was added to the component. Fig. 8 shows the example of the *PLC*. Errors

related to software are specified in the SCM element properties. An example is shown in chapter 6. The string values are than used to generated the Error-Codes for each *System*.

S: CPM HW_Comp (R3)
O: Errortxt=Error-Type + HW_Comp inside System + HW_Comp.Prop

4.3 Network Settings

The network setting of a testbed depends on the structure of the EVT elements, by parsing the testbed structure directly, the IDs can be calculated (R4). To keep the approach flexible the SCM Network element also provides the possibility to set the ID manually. An example is given in chapter 6.

S: EVTM Network (R4)
O: ID= f(Distribution_Box_cnt,Switch_Box_cnt) + System.Name + Platform_Info

4.4 Control Parameters

The control parameter can be divided into parameters depending on the hardware (hardware parameters) and into parameters depending on the user input (user parameters). The hardware depending parameter are calculated based on the hardware properties and hardware structure (*Inside*, *Assembly_Connection*) see (R5). The user dependent parameters are mapped from the user input in the SCM (R6).

S: EVTM HW_Comp Assembly_Connection (R5)
O: Para=f(HW_Comp.Prop, f(System elements Inside EVTM, System element Assembly_Connection)) + Platform_Info
S: SCM Control (R6)
O: Para=Control.Para+ Platform_Info

4.5 Automation System Files

The automation system controls all the systems of the EVT. The generator input are the *Application_Software* elements: *Monitoring*, *Parameters*, *Testrun*, *User_Interface*, *Channels* and *Formulas*. Each of this element provides the possibility to set properties and to define customized scripts. Fig. 9 shows a *channel* definition example. The raw software configuration generator (RSCG) accesses the EVTM to generate the RSC, this are than enhanced by the user. The SCG generates the target configuration out of this by performing a horizontal transformation. (R7) shows this relation channel definitions are generated based on the *Supply* properties and platform info.

S: SCM Channels, EVTM Supply (R7)
O: Channels.Prop= f(Supply.Prop) + Platform_Info

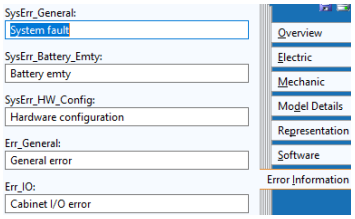


Fig. 8: PLC Component error settings.

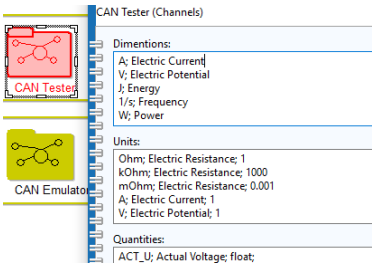


Fig. 9: Automation System, Channels.

4.6 Calibration Parameters

Each sensor of a physical system needs calibration data. This calibration data must be defined during a calibration process. Therefore, the modeling language provides a corresponding concept. The calibration data cannot be calculated, as it directly relies on the performed measurement during calibration. The main part of the generator is here a horizontal transformation which adds the platform specific content of the target-domain. The calibration data is linked to a hardware (sensor) on the CPM (using a model INTERREF). This connection is used to perform constrain checks on the calibration data and to extract further information e.g. gain values for the sensor.

S: SCM Calibration, CPM HW_Comp (R8)
O: Calibration=f(Calibration.Prop + HW_Comp.Prop) + Platform_Info

5 Tool

The complete modeling solution including metamodel, modeling tool and transformations has been developed using the metamodeling framework ADOxx. We opted for this platform because it has been intensively tested in many different research projects in science and industry [FK13]. Since the metamodel development, notation and modeling tool development are published elsewhere [PMP19, PM19] our focus in this paper is on the generation process of the software configurations for EVTs. ADOxx includes the scripting language AdoScript to perform model transformation and model-queries (transformation language support [SK03]). Consequently, there is no need to develop the model parsers manually. The transformation rules presented in chapter 4 were implemented in AdoScript like the example shown below which drives the transformation of a *SW_Assembly* based on the *HW_Assembly*.The EVTM is publicly available on the OMiLAB repository. OMiLAB is

a dedicated research and experimentation space for modeling method engineering [OM19]. We decided to make the EVTM tool there publicly available to collect further user feedback. Since the EVTM tool is developed in close cooperation with a company, not all functions are released in the public version to protect company rights. For integrating our approach into the development process of the company, we created a set of guidelines and linked the domain-specific modeling process with the existing solution so that the users can understand the difference. A starting point for this was a comparison as shown in Tab. 2.

Transformation definition in *AdoScript*:

```
SETL sClSource:("HW_Assembly")
SETL sClTarget:("SW_Assembly")
CC"Core"GET_CLASS_ID classname:(sClSource)
CC"Core"GET_ALL_NB_ATTRS classid:(classid)
CC"Core"GET_ATTR_VAL objid:(VAL sObject) attrname:"Name" SETL sNa:(val)
CC"Core"GET_CLASS_ID classname:(sClTarget) SETL idClTarg:(classid)
CC"Core"CREATE_OBJ modelid: (idTargetM) classid: (idClTarg) objname:(sNa)
CC"Modeling"SET_OBJ_POS objid: (objid) x: 5cm y: 8cm
```

Traditional	*Domain-Specific Model*	*Example*
Source-Code (Eclipse)	CPM, SCM	Class Definition
Document (MS Visio), Network Settings (Files)	EVTM	IDs Netw. Nodes
Controller Parameter (Text File / MS Excel)	EVTM, CPM, SCM	Controller gain
Error Codes (Code, MS Excel)	CPM	Error enum

Tab. 2: Comparison Table (EVT Domain Example)

6 Evaluation

To validate the correct functioning of our generator and the plausibility of the overall approach, we have taken the usual route of comparing generator outputs with manually developed artefacts from previous EVT projects. The evaluation covered the outputs: *Source-Code*, *Error-Codes*, *Network Settings* and *Control Parameters*. These outputs have been compared regarding the criteria: Completeness, Correctness, Productivity (time to produce the solution), Understandability (to the users), and Re-Usability.

Evaluation Example for a particular testbed project:
The batteries under test may have 48-1100V, max. 1700A, 550kW. The E-Motors under test may have 200-1000V, max. 900A, 250kW. The company norm requires additional safety features included in the testbed system which are: voltage display, a door contact on the supply, and a customer specific IO that becomes active when the output voltage reaches a certain level. This level should be configurable via the *Automation_System*. Further, the company norm requires to have an US safety monitor system integrated into the supply

which replaces the default device. The current rise (t90 time) should be within 1ms and the Power Distribution Unit (PDU) capacitance is increased to 1mF. In order to allow stable operation. The systems are connected to the industrial 690V/60 Hz grid. The artefacts created for this example evaluation are shown below. Fig. 10 shows the EVTM. Each of the *System* elements (blue rectangle) has a corresponding CPM, Fig. 11 shows the CPM for Supply1. For each *SW_Assembly* in the EVTM a SCM model is generated Fig. 12 shows *SW_Assembly1* and Fig. 13 shows *SW_Assembly2*. Artefact 1-4 shows the generated output based on the EVTM, CPM and SCM models shown before. The graphical models are transformed into textual representations by using the transformation rules shown in chapter 4.

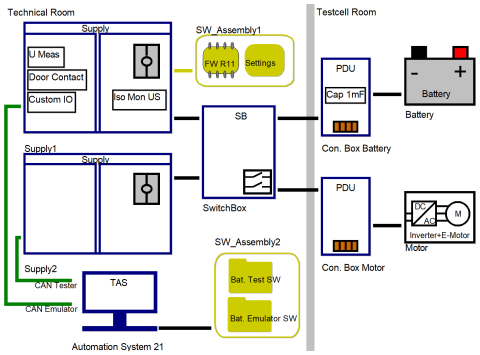


Fig. 10: EVTM Example.

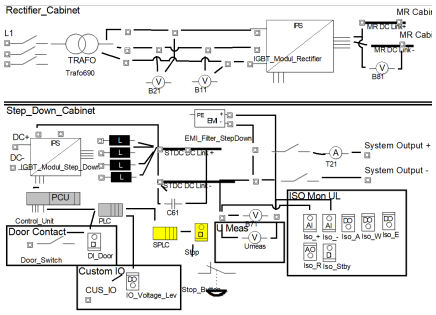


Fig. 11: Supply1 CPM.

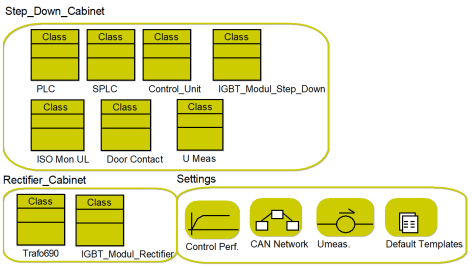


Fig. 12: SW_Assembly1 SCM.

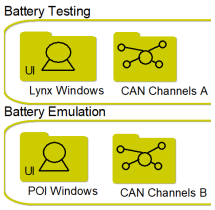


Fig. 13: SW_Assembly2 SCM.

```
WARNING_DI_DOOR, //IO warning
WARNING_TRAFO690, //Transformer temperature warning
ERROR_IGBT_MODUL_RECTIFIER, //IGBT Modul error
```

Artefact 1 Error-Codes.

```
0x11;Supply1 0x12;Supply2 0x0;SwitchBox
0x31;Con.Box Battery 0x32;Con.Box Motor
```

Artefact 2 Network Settings.

```
class Door_Contact{
public: bool get_IO_Door_Switch();
private: iDio* m_Door_Switch;};
class Custom_IO{
public: bool set_IO_Voltage_Lev();
private: iDio* m_Voltage_Lev;}
class Iso_Mon_UL{
public: Iso_Mon_UL(int16 *isoValue,
bool *enable);
virtual ~Iso_Monitor_US();
Uint16 getValue();
void setEnable(bool en);
bool isWarning();
bool isAlarm();
private: iDio* m_Enabl;
iDio* m_Iso_A;
iDio* m_Iso_W;
enum{
MAXVAL = 3000,MINVAL = 50};};
```

Artefact 3 Class definition.

```
59; 10 ; BE High Dynamic
76; 2.0E+00; REF_U_DELAY
77; 1.0E+03; SCALE_PWR
125; 2; PLANT_STRUCTURE
114; 0.00E+00; PLANT_RC1
117; 1.00E-03; PLANT_C2 (PDU)
50; 0.65; FEEDFORWARD IQ
8; 400; PWR_GRID_MX
7; -400; PWR_GRID_MN
36; 720; LIM_U_RMS_MX
94; 700 ; IT_MX
```

Artefact 4 Control parameters.

The comparison with manually developed artefacts led to the following results:

- The network settings of the generator output are easier to read because it features the names of the systems/components as used in the EVT-Model.
- The generator output is more exact regarding control parameters, as in the manual version these were not adapted to the testbed structure (e.g. cable length).
- The generator produced its output faster as only a small set of parameters had to be set manually in the model, other parameters are derived from the CPM and while other parameters were extracted from the template files. As we do not have the recorded timing data from the manual development process, we can only assume the time for defining redundant information can be saved, resulting in increased productivity.
- The generated errors are more meaningful compared to the manual developed definition as (1) they contain additional error descriptions derived from the CPM and (2) the generated names are consistent to the names used in the hardware domain.
- The generated class definition of the header file provides variable, function and enumeration definitions. Again, the difference to the manual code is that the naming is consistent and the generated artefact ensures that each hardware component has a related definition in the code.

- The connection between the model and the generated artefacts is not straightforward.

To sum up, our evaluation experiments proved that the model-based generator approach increases productivity by reusing already defined information of the hardware domain. Regarding understandability we found out that the generator has a more consistent naming and adds additional comments to make the generated parts even better understandable. A further improvement in terms of understandability would be adding information in the generated artefacts how they are linked to the model elements.

7 Related Work

Related work addresses a variety of aspects regarding software design and generation for mechatronic systems. [SP09] analyzed “successful“ industrial mechatronic system development processes and observed quite a muddle: different notations are used, developers do not have an overview over the complete system functionality, activities with the same target are done several times using different procedures and tools. In several publications the collaboration among specialists and stakeholders with different backgrounds is addressed, by presenting a model-based approach [Ga10, Le08, Ba15, HRZ14]. In the following we will give an overview about domain-specific approaches. MechatronicUML [Ho16] focuses on Requirements Engineering and Model-Driven System Development for the control software of mechatronic systems. Compared to the EVTM approach the MechatronicUML is complex, as it addresses the whole domain of mechatronic systems, where the paper focus on a special part of the mechatronic domain (EVTs). The EAST-ADL Architecture Description Language for automotive embedded systems focus is on in-vehicle systems, as it needs to be compatible with the AUTOSAR metamodel; it is not a “lightweight“ DSML compared to our solution [EA13] . Easy lab [Ba08] is a model-based programming tool which enables both the modeling of software and hardware functionality. The generator uses templates for each primitive element to produce code. Now we will have a look on the code generation literature, which is manly driven by the software engineering community. [Ga10] addresses the following purposes of code generation: (1) separation of application logic from platform details (2) improved productivity, (3) improvement in the quality of the application, and (4) increased performance of the application by generating efficient code. Well known transformation approaches are “Direct Model manipulation“ , “Intermediate representation“ and “Transformation language support“. For EVTs we are using the “Intermediate representation“ technique where the intermediate artefact is represented by the RSC. [BSM14] implements the intermediate artefact by generating an XMI file. [Fe09] defines a code generation approach for railway systems and in [CGL17] a tool to uncover model transformation problems is presented. Mechatronic systems and especially EVTs need application software and firmware. The software development of EVT systems is affected by the PLC manufacturers as they came up with powerful IDEs [Bu19, SI19]. Well known automation systems for EVTs are, as example, PUMA, LYNX

[AV19] and KS Tornado [KS19]. These systems have been analyzed, as basis for designing the software configuration metamodel.

8 Conclusion

The analysis of the literature on mechatronic system design showed that one of the greatest challenges is to deal with the complexity of the collaboration between the engineering disciplines, including management [To07]. The common way to attack this challenge is a model-based approach supporting the requirement management process which we also apply here. The collaboration is enhanced by having an overview model (EVTM) that can be understood across domains and this is further refined in the engineering disciplines. With this solution domain experts can see the big-picture of the EVT and are not lost in details. Another aspect is to ensure a smooth integration of software and hardware design by providing proper tool support. Only few publications, however, address the issue of software generation for mechatronic systems. Such systems consists of hardware components, which interact with IOs that are connected to microcontrollers and PLCs. To program these devices object-oriented programming languages like C++ and C# gain importance, in addition C and other IEC languages are still used. Manufacturers of such hardware have a high impact on the development, as they often deliver the corresponding development environments. The seamless integration into these tools is one of the cornerstones that a generator solution is accepted.

The approach presented here builds on the work cited in chapter 7, especially the transformation approaches, the cross-references [Ga10] which define the software to hardware relation and the state-of-the-art automation systems. We showed, how to generate software configurations in the EVT domain out of hardware and software models that are represented in a DSML. The EVT domain served as an example, and we are convinced that it is also applicable in other mechatronic engineering disciplines. The work showed that it is really worth to investigate in DSML development and generator development because it results in increased productivity and quality. Compared to traditional approaches which use complex languages our approach is lightweight, which all the resulting benefits. To gain further feedback from the modeling community a version of the modeling tool has been made public available on the OMiLAB [OM19] repository. But there is still some work to do. As an example, the modeling language needs to be extended by concepts for formulating constraints over attribute values in order to allow for more pinpoint transformations. In addition the transformation rules have to be defined formal, including a definition of the general process to define such rules for the mechatronic domain. Also, we plan to enhance the tool to automatically generate circuit plans based on the developed CPM. In the end, the modeling environment should cover all the software and hardware aspects of a mechatronic system for a particular domain. Another interesting application would be to use the domain specific models of the mechatronic system as digital twin to train engineers and to monitor the state of the system.

References

[AV19] avl.com, https://www.avl.com/, accessed: 20/09/2019.

[Ba08] Barner, S. et al.: EasyLab: Model-Based Development of Software for Mechatronic Systems. In: 2008 IEEE/ASME International Conference on Mechtronic and Embedded Systems and Applications. IEEE, Beijing, China, pp. 540 – 545, 11 2008.

[Ba15] Barbierei, G. et al.: Modelling and Simulation for the Integrated Design of Mechatronic Systems. IFAC-PapersOnLine, 48(10):75–80, 2015.

[BSM14] Bardaro, G.; Semprebon, A.; Matteucci, M.: AADL for robotics: a general approach for system architecture modeling and code generation. JOSER, pp. 32–44, 2014.

[Bu19] Automation Studio, https://www.br-automation.com/en/products/software/automation-studio/, accessed: 20/09/2019.

[CGL17] Cuadrado, J. S.; Guerra, E.; Lara, J.de: Static Analysis of Model Transformations. IEEE Transactions on Software Engineering, 43(9):868–897, sep 2017.

[EA13] EAST-ADL Domain Model Specification, https://www.east-adl.info/Specification/V2.1.12/EAST-ADL-Specification_V2.1.12.pdf, accessed: 20/09/2019.

[ec19] Component Lead Times, https://www.ecianow.org/, accessed: 20/09/2019.

[EP19] EPLAN Electric P8: Power für Elektroprojektierung und Engineering, https://www.eplan.at/at-de/loesungen/elektrotechnik/eplan-electric-p8/, accessed: 21/09/2019.

[ES13] Eessaar, E.; Sgirka, R.: An Ontological Analysis of Metamodeling Languages. In: Information Systems Development. Springer, New York, pp. 381–392, 2013.

[Fe09] Ferrari, A. et al.: Modeling Guidelines for Code Generation in the Railway Signaling Context. In: Proceedings of the First NASA Formal Methods Symposium. Moffett Field, California, 2009.

[FK13] Fill, H.; Karagiannis, D.: On the Conceptualisation of Modelling Methods Using the ADOxx Meta Modelling Platform. Journal Enterprise Modelling and Information Systems Architectures, 8:4–25, 2013.

[Fo18] 2-Prozent-Marke geknackt: E-Autos boomen - aber nur mit Zwang und Förderung, focus.de, accessed: 20/09/2019.

[Ga10] Gausemeier, J. et al.: Computer-Aided Cross-Domain Modeling of Mechatronic Systems. In: INTERNATIONAL DESIGN CONFERENCE. volume 2, Dubrovnik, 5 2010.

[Ho16] Holtmann, J. et al.: The MechatronicUML Requirements Engineering Method: Process and Language. Technical report, Software Engineering Department, Fraunhofer IEM & Software Engineering Group, Heinz Nixdorf Institute, Paderborn University, 12 2016.

[HRZ14] Hackenberg, G.; Richter, C.; Zäh, M.: A Multi-disciplinary Modeling Technique for Requirements Management in Mechatronic Systems Engineering. Procedia Technology, 15:5–16, 12 2014.

[KS19] Tornado Test BedPrüfstandsautomatisierung., https://www.ksengineers.at/de/Automotive-Testing/Management-und-Automation-Tools/Tornado-Test-Bed, accessed: 20/09/2019.

[KSK18] Kuhnert, F.; Sturmer, C.; Koster, A.: Five trends transforming the Automotive Industry. techreport, pwc.at, 2018.

[Ku11] Kuester, J.: Model-Driven Software Engineering Code Generation. IBM Research, 2011.

[Le08] Lennon, Tony: Model-based design for mechatronic systems. ELECTRONICS WORLD, 114:23–26, 05 2008.

[Ma18] Mayr, H. C. et al.: A Model Centered Perspective on Software-Intensive Systems. In: Proc. 9th Int. Workshop on Enterprise Modeling and Information Systems Architectures. volume 2097, CEUR-WS.org, Rostock Germany, pp. 58–64, 2018.

[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 ICTer2015. Colombo, 2015.

[Mo09] Moody, D. L.: The "Physics" of Notation: Toward a Scientific Basis for Constructing Visual Notations in Software Engineering". IEEE Transactions on Software Engineering, 35:756–779, 2009.

[OM02] OMG: MetaObjectFacility (MOF) Specication. OMG, 2002.

[OM19] Open Models Laboratory, `http://austria.omilab.org`, accessed: 20/09/2019.

[PM19] Paczona, M.; Mayr, H. C.: Model-Driven Mechatronic System Development. In: 2019 IEEE 15th International Conference on Automation Science and Engineering (CASE). IEEE, Vancouver, Canada, pp. 1730–1736, aug 2019.

[PMP19] Paczona, M.; Mayr, H. C.; Prochart, G.: Model-based Testbed Design for Electric Vehicles. In: Proc. EMISA 2019, Lecture Notes in Informatics, GI. Tutzing, 2019.

[Ro11] Roussey, C. et al.: An Indroduction to Ontologies and Ontology Engineering. In: Ontologies in Urban Development Projects. Springer-Verlag, London, pp. 9–38, 2011.

[Ro17] Roos, G.: Power Semiconductor: Lead Times Continue to Stretch. EPSNews, 2017.

[SI19] PLC Programming with SIMATIC STEP7, `https://new.siemens.com/global/en/products/automation/industry-software/automation-software/tia-portal/software/step7-tia-portal.html`, accessed: 20/09/2019.

[SK03] Sendall, S.; Kozaczynski, W.: Model Transformation: The Heart and Soul of Model-Driven Software Development. Software, IEEE, 20(5):42 – 45, 10 2003.

[So19] SOLIDWORKS 3D CAD, `https://www.solidworks.com/product/solidworks-3d-cad`, accessed: 21/09/2019.

[SP09] Stetter, R.; Pum, U.: PROBLEMS AND CHALLANGES IN INSTUSTRIAL MECHATRONIC PRODUCT DEVELOPMENT. In: International Conference on Engineering Design (ICED 09). Palo Alto, California, 2009.

[To07] Tomiyama, T. et al.: Complexity of Multi-Disciplinary Design. CIRP Annals, 56(1):185–188, 2007.

[Ve19] Vennes, J.: Coopert Transformer Lead Times. techreport, boarderstates.com, 2019.

[Vo13] Voelter, M.: DSL EngineeringDesigning, Implementing and UsingDomain-Specific Languages. CreateSpace Independent Publishing Platform, 2013.

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 45

A Domain-Specific Modeling Method for Supporting the Generation of Business Plans

Michael Wieland [1], Hans-Georg Fill [2]

Abstract: For the formation of new companies it is indispensable to provide well-defined and comprehensive business models in order to attract investments and earn trust of relevant stakeholders. In the past, several methods have been proposed to support this complex undertaking and successfully arrive at viable business models. In this paper we propose a domain-specific modeling method for operationalizing the concepts contained in the Business Model Canvas as one of the most prominent methods in this field. The modeling method has been implemented on the ADOxx meta modeling platform and permits to generate business model templates in the form of a business plan as text documents. The usage of the approach is illustrated through an example from the business model of the telecommunications application Skype. Finally the benefits and limitations of the prototype are discussed using a strengths, weaknesses, opportunities, and threats analysis.

Keywords: Domain-specific Modeling; Business Model; Business Model Canvas; Business Plan; Modelling Tool.

1 Introduction

The market success of a company is influenced by the ability to create value for the customer [SH13]. This competence forms the basis for strategic goals as well as the underlying business processes and the definition of technological opportunities [BS96]. Digitalization as a driver of disruptive innovations or servitization increases the pressure on companies in terms of optimizing their business routines and reducing costs [Ma06, Ha91]. On the other hand, customers need to be provided with continuously enhanced products and services as part of the business model [Mv03]. Business model innovations represent the modification of an existing business model or the building of a new business model to meet customer needs in a better way [Ve17]. Thereby, methods and tools to identify business model innovations are required for example by start-ups or in incubators of already established firms to support this continuous innovation process.

When designing such methods and tools, the challenge is not to over-simplify the complexity of the way a company operates [OP10]. However, it is important that systematic assumptions are made about the contents of a business model and that an efficient and effective conception

[1] University of Bamberg, Information Systems and Applied Computer Sciences, An der Weberei 5, 96047 Bamberg, Germany, michael-dieter.wieland@stud.uni-bamberg.de
[2] University of Fribourg, Digitalization and Information Systems Group, Bd de Perolles 90, 1700 Fribourg, Switzerland, hans-georg.fill@unifr.ch

of ideas is enabled. For this purpose, we decided to focus on established approaches for creating and analyzing business models in the sense of an integrated business model management [OP10]. Although such methods and tools already exist in principle, we found that a tool that supports both the visual representation of business models as well as automatically translating them into textual documents does not exist so far. That latter aspect of textual documents is however important for startups and incubators that want to convince investors and document all aspects of their future business in form of a detailed *Business Plan*. Therefore we decided to design a domain-specific conceptual modeling method that is capable of both visually representing business models in the established notation of the so-called *Business Model Canvas* [OP10] as well as generating textual documents from these representations.

The remainder of the paper is structured as follows: In section 2 we will briefly describe work related to our approach. Section 3 will present the details of the modeling method and in section 4 we will show a small use case for demonstrating the working of the method. The paper will conclude with a discussion of the approach in section 5 and an outlook to future work in section 6.

2 Related Work

In this section, we provide an overview of methods and techniques for conceptualizing business models. In addition, we briefly characterize the components of modeling methods as we will use them later for describing our method.

2.1 Methods and Tools for Business Model Conceptualization

One well-known method for planning business strategies is the SWOT (Strength, Weaknesses, Opportunities and Threats) analysis for assessing market and environmental conditions [BAD93]. However, this approach is particularly suitable for a retrospective assessment of existing companies. Other approaches describe companies on the basis of their business model. Following the definition of Osterwalder and Pigneur a *Business Model* "describes the rationale of how an organization creates, delivers and captures value" [OP10, p.14]. Based on our literature research, we identified several approaches, each taking a different perspective. For example, the e-business model approach by Timmers takes the perspective of the market, with the focus on specific elements such as e-shop or e-procurement [Ti98]. Stähler defines a business model through the three main components: "User promise", "Architecture of value creation" and "Earnings model"and focuses on the value creation [St01]. Akkermans also describes an approach for electronic businesses where he uses an underlying value model supported by a conceptual modeling approach [AG03]. Maurya describes a problem-focused approach, the so-called "Lean Canvas" [Ma12]. The integrated business model management approach proposed by Osterwalder and Pigneur requires to review the business model from

different aspects. Their Business Model Canvas (BMC) creates the link between strategic management and business model management and is grounded on a concrete building block structure and also links to the implementation guide in form of a *Business Plan* [OP10, p.268].

For supporting the definition and analysis of business models, a range of tools have been developed in the past, especially for the BMC. As Schoormann et al. describe in a review of these tools, they are mostly oriented towards pure visual representations and lack typical features of modeling tools such as the support for a modeling procedure [SBK16]. Also, the linkages between elements and the transformation of visual representations into textual documents that are suitable for investors do not seem to have been covered so far. A first conception that takes up these deficiencies by showing linkages explicitly has been presented by Augenstein and Fleig [AF18], however the tool has not been fully implemented.

2.2 Components of Modeling Methods

In the approach described by us, we will revert to the concept of a domain-specific conceptual modeling method, which targets a specific field of application and that can be realized as an IT-based prototype [JF17, KMM16]. These methods are composed of a modeling language, a modeling procedure, and Mechanisms and Algorithms [KK02] (MA). The syntax of the modeling language defines a grammar, and an according semantics specifies the meaning of its elements. The modelling procedure describes how the modeling language and the MA are to be used. Different forms of MA enable the computer-aided and automated processing of model contents. To ensure efficient processing of information in relation to calculation and storage, the interdependence between MA, the modeling procedure and the modeling language must be taken into account in the development of modeling methods [FRK13]. Software platforms such as Eclipse, MetaEdit+ or ADOxx facilitate the implementation of such modeling methods, cf. [TK09, FK13].

3 A Domain-Specific Modeling Method for Generating Business Plans

Based on the previous elaborations we can now advance in this section by describing our modeling method. The conceptualization of the modeling method is founded on the core elements of the BMC. We thus describe this conceptualization in the form of the derivation of the modeling procedure and the different model types. After that we describe the design of our modeling language based on a meta model and outline the implementation aspects for our prototype.

3.1 Research method

The research procedure follows the process described by Johannsen and Fill, without the formalization step in FDMM [JF17] - see Figure 1. Therefore, we start with the elaboration of a concept for generating Business Plans based on the BMC. This is followed by the development of the domain-specific modeling method (DSMM) with the according model types in semi-formal notation, which are then implemented as a modeling tool.

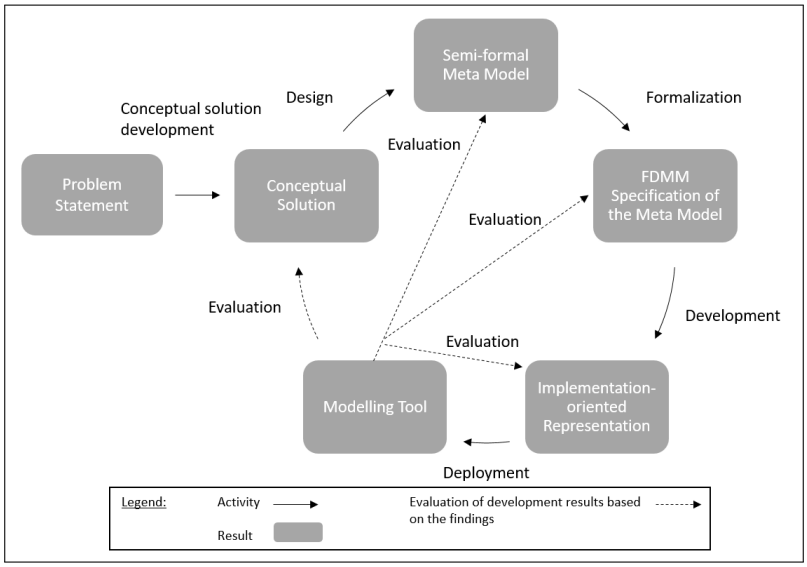


Fig. 1: Design-oriented Research Cycle for Modeling Methods [JF17, p.257]

In the activity *"Conceptual Solution development"*, the goal is to create a concept for systematic creation of *business plans* as an investor-oriented representation based on the design of a business model. The *"Design"* step contains the development of meta models to support techniques of the design process for generating business models. These model types are used to codify business concepts within a DSMM. For the semi-formal description, UML class diagrams described in natural language are used. At this stage of development, it is not guaranteed that all required information for a technical implementation is yet available. *"Formalization"* would mean that the previously created semi-formal meta models are specified in an exact notation. For example, attributes, classes, and possible restrictions would be exactly described here [FRK13]. In addition, the graphical representation of the elements is specified using the meta model [BF14]. The *"Development"* step stands for the implementation of the modeling language and the MA for generating the business plan documents using a software platform. In the *"Deployment"* phase, the tool is further advanced from the development environment to a standalone tool including installation procedures. Finally, the *"Evaluation"* phase contains the assessment of the modelling tool,

where experience gained over the entire development period is gathered. In this paper we not provide a detailed evaluation, the modeling concept is only demonstrated using a practical example, which allows to assess the accuracy and consistency with the semi-formal and formal specifications of the method, called ”Demonstration”.

3.2 Characterization of the Business Model Canvas

The aim of the BMC is to provide a simple, relevant and intuitively understandable model that captures the core business logic of all companies and does not disregard the complexity of corporate functions [OP10]. It provides a uniform approach for describing and adapting business models in the context of strategic action fields, resulting from nine building blocks which are specified in a standardized order. The BMC serves as a tool for analysis, to improve understanding, as a basis for discussion and to increase creativity. Figure 2 shows a BMC template and the building blocks according to [OP10]. These will be described in the following according to the definitions by Osterwalder and Pigneur [OP10].

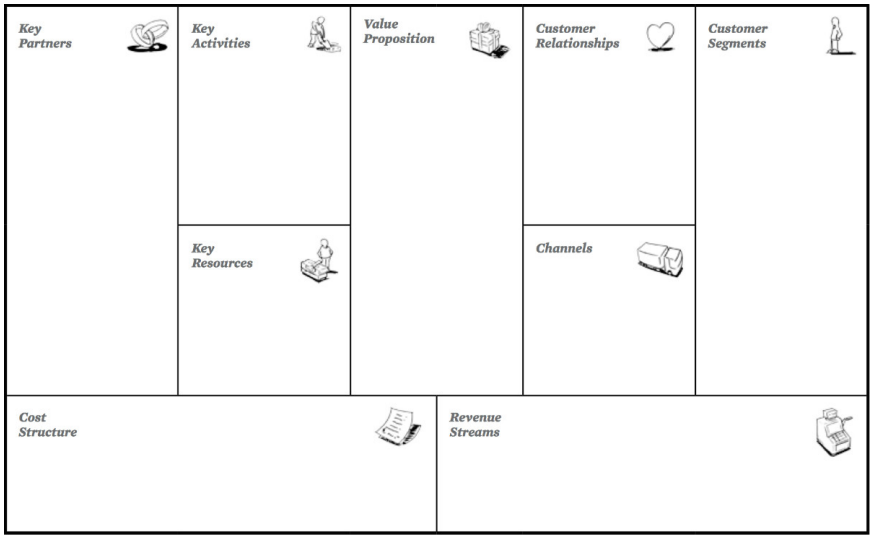


Fig. 2: The Business Model Canvas [OP10, p.44]

Key Partnerships represent a network of suppliers and partners which are necessary for a business model. *Key Activities* specify the essential activities of a company’s business model. *Key Resources* are required to operate the business model, they are needed by any company to operate a business model, deliver or create value, reach markets, maintain customer relationships and generate revenue. The *Value Proposition* building block describes a collection of products or services that provide value to a specific customer segment. The

Customer Relationship block outlines the different types of relationships between companies and a customer segment. The *Channels* building block describes how to interact with a customer segment to deliver a value proposition. *Customer Segments* represent the various customers and other groups that the company intends to address. The block *Cost Structure* describes all of the operating costs incurred within a business model. *Revenue Streams* represent cash flows from each customer segment.

Osterwalder and Pigneur describe two approaches for creating a BMC. Either, a large poster is needed where the nine building blocks are shown together with a lot of post-it notes. The core factors are then noted on the post-its in the form of keywords and assigned to the respective block. The second option is that the factors are described using software where a BMC template has to be filled with textual descriptions [OP10, p.266]. As shown by the study conducted in [SBK16], such tools often just imitate the use of post-it notes in an electronic format. Thus, they do not harness the possibilities of modeling methods for inter-linking elements and applying MA to them.

3.3 Conceptualization of the IT-based Modeling Method

Based on the previous elaborations, the next step in the research method is the actual conceptualization of an IT-based modeling method which supports the creation of business models and business plans. For this purpose we derived ten requirements that have been derived from the literature [OP10], that shall be satisfied by the modeling method. These are shown in Table 3.3 and will be explained in detail in the following.

Tab. 1: Requirements for a Domain-specific Modeling Method for Business Plan Generation

Categorization	Requirements (Rq)
Business Plan Generation	Rq1: Business plan structure support Rq2: Tool supported generation of business plan templates as a textfile
Business Model Generation	Rq3: Support the Business Model Canvas structure Rq4: BMC Design support Rq5: BMC Strategy support Rq6: Support of BMC Process Stages
Properties of techniques	Rq7: High understandability and learnability Rq8: Flexible handling
Interdependencies between techniques	Rq9: Sequential ordering of techniques Rq10: Complementary interdependencies

The category *"Business Plan Generation"* contains two requirements and describes all the requirements which must be met in relation to business plans. The solution shall serve as a structured input interface for mapping the structural design of business plans as described in the literature, for example to attract investors [CYK09] (Rq1). As a result of the data input

using the input mask, it should be possible to generate a business plan template as a text document (Rq2). The text file created in this way serves as a business plan template for further processing, which can be complemented as needed. After the document has been completed, the final business plan will be saved as text document.

The category *"Business Model Generation"* describes the requirements for the business model generation, where the BMC principles serve as a basis and the focus is on the aspect of graphical modeling. The presentation of the basic structure of a BMC on the basis of building blocks represents an important requirement [OP10] (Rq3). In the business model design, it should be possible to simply create new ideas for business model concepts, therefore a modeling approach should be used to support techniques such as Ideation or Visual Thinking by using a guided design process [OP10, p.131] (Rq4). Business models should be stored over time and it must be possible to adapt these models due to changing environmental conditions as it is also required to manage several business models at the same time (Rq5). The implemented modelling tool should also support each of the five design process phases defined by Osterwalder and Pigneur [OP10, p.249] (Rq6). In the category *"Properties of Techniques"*, implicit requirements for the properties of the different techniques must be met. Due to the complexity of business models and business plans themselves, it must be ensured that each designed model type is intuitively understandable (Rq7). It must be ensured that the techniques are flexible so that they can be used for different types of business plans (Rq8). For example, it shall be possible that the ideation process can be started from any building block and that one can freely choose suitable relations to other building blocks.

The category *"Interdependencies between the techniques"* specifies interrelationships between techniques. It should be possible to specify a distinct sequence of the techniques for achieving particular results (Rq9). The different techniques should also be mutually supportive in order to allow synergy effects that improve the end result through the combined use of each technique (Rq10), e.g the outcomes of a technique are recorded and further processed by a subsequent technique.

3.4 Derivation of Modeling Procedure and Model Types

For deriving the model types (diagram types) of our modeling method, we defined first a procedure for generating business plans in a structured way according to the requirements Rq1 and Rq2. Subsequently, the elements of the BMC for generating business models were considered (Rq3-Rq4). Finally, the additional requirements Rq7-Rq10 were taken into account. This resulted in the derivation of a 4-step procedure as shown in Figure 3.

At the beginning of the business model design process it is started by creating first ideas for business models. This is supported by the model type "Business Transaction Model" (BTM) that supports techniques such as Visual Thinking and is already based on the building blocks of the BMC.

After the modelling of the first ideas in the Business Transactions models, a first snapshot of a BMC is created. This is accomplished through a second model type, the ”Business Model Canvas Model” (BMCM). This model type provides mechanisms for an automatic synchronization of the business concepts of the BTM into the typical structure of a BMC. Thus, these first two model types fulfill the strategic requirements of an integrated management solution, where it is possible to adapt several business models to environmental changes(Rq5). In addition, the process stages of the BMC basic design process are supported (Rq6).

The third step consists of setting references to the BTM and BMCM and the addition of relevant data for a business plan document. This step is supported by the model type ”Business Plan Document Model”. Once this is completed, it is possible to extract the data as an editable text document called *Business Plan*. Sometimes, optimizations of a business model are required at certain intervals in order to adapt to changing market conditions, cf.[Mu13]. This is highlighted by the dotted arrow that points back to the first step. Finally, it is possible to edit and complete the exported Business Plan template and convert it to a non-editable text document.

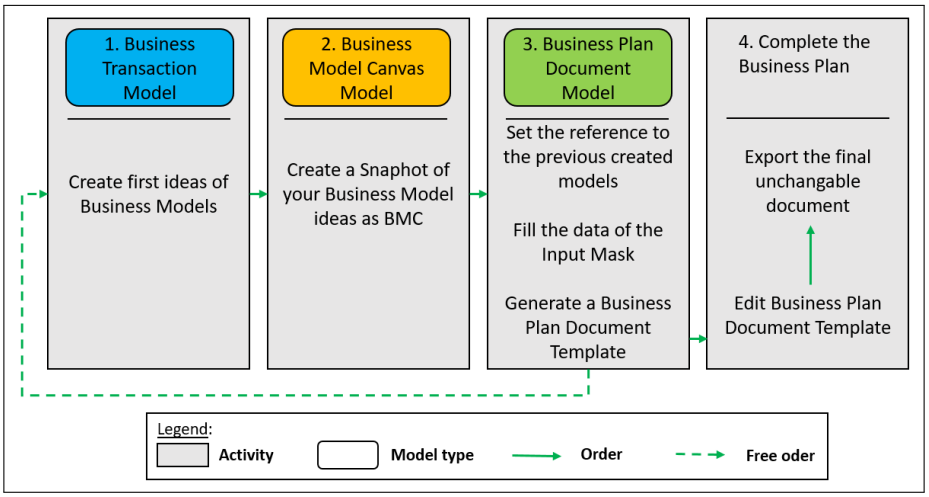


Fig. 3: Business Plan Document Design Process

3.5 Design of the Modeling Language

The techniques that support the modeling procedure were carefully examined in order to identify the core principles of each technique and to meet the previously derived requirements. This has resulted in three main model types. The model types are shown in Figure 4, their inter-linkages are depicted by ”Reference” relations, the graphical notation for the classes is

shown besides them. For some classes, multiple options for the graphical representation are available, e.g. for 'Value Proposition' to distinguish between general value propositions, products, and services. Due to limitations of space, attributes are omitted in the figure.

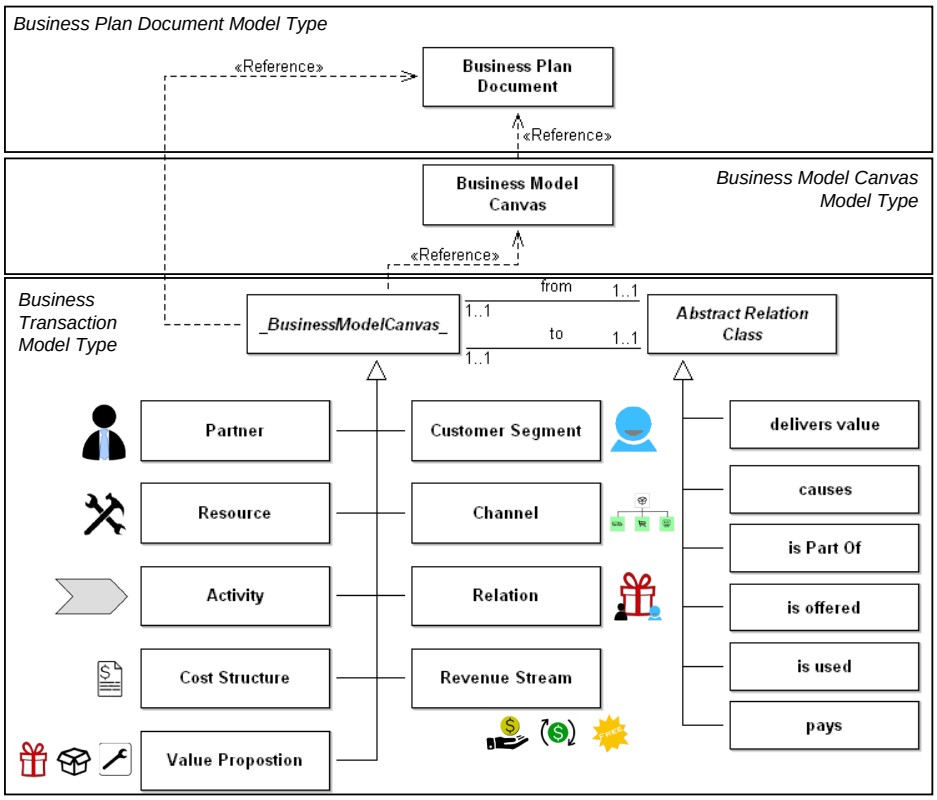


Fig. 4: Meta model of the Modeling Language and including Model Types

The *"Business Transaction Model"* type was developed with the design focus on creating ideas for business models. The model type serves as the starting point in our Business Plan creation process. Thereby, the building block elements described in section 3.2 serve as the basis for the language elements and the modeling behaviour, e.g. to support the story telling technique. Therefore each building block is represented by a class, e.g. "Channel". An essential aspect is that the instances of the classes can be linked to each other. To enable this, the relation classes "delivers value", "causes", "is Part Of", "is offered", "is used", "pays" are available.

The basis of the BMCM is a previously created instance of a BTM type. This model type serves as an overview in the known BMC layout to ensure the familiar understanding of the business models within the BMC design. The BMCM type is only used as a view in this

case, so it will not be possible to edit it. It is automatically generated from the information in a BTM. To create the BMCM, the reference to the source model from which the BMCM has to be created must be selected.

In the *"Business Plan Document Model"* the additional data necessary for business plans is specified. This is done via a range of attributes that are part of the class "Business Plan Document". These are organized along the categories "Description", "Cover Page and Table of Contents", "Executive Summary", "The Team", "The Business Model", "Financial Analysis", "External Environment", "Implementation Plan", "Risk Analyis"and the "Conclusion" and represent the business plan structure [OP10, p.269]. Due to the large number of these attributes they are not shown in Figure 4.

3.6 Implementation of the Modeling Method

The modeling method has been implemented using the ADOxx meta modeling platform [FK13]. This included the implementation of the model types including their classes, relation classes and attributes as well as corresponding graphical representations. The graphical icons were partly taken from *"Openclipart.org"*, and adapted to the specific needs of the modeling method.

In the *Business Transaction Model* we integrated a guided design process which indicates the progress within the modeling procedure, thus embedding the method in work practices of users [Sa18]. As part of the *Business Model Canvas Model* we implemented a BMC design template which automatically references elements of the assigned BTM type using an automatic update mechanism of ADOxx in the form of so-called *expression attributes*. The *Business Plan Document Model* is able to integrate the referenced BTM and BMC instances. An algorithm for the creation of business plans has been implemented in ADOscript. The algorithm collects data in the Business Plan document, which is based on the entered data on the graphical user interface. Finally, the prototype was deployed as a stand-alone modeling tool with its own setup routine for installation on Microsoft Windows 10.

4 Application of the Modeling Method in a Use Case

For the demonstration of the method we demonstrate the implementation of the various models to support business plan development and their practical usefulness. The IT-based modeling method, targets users in the practical environment who have to develop business plans or business model concepts and create or adapt them in digital change. Our method enables documentation and communication and further processing of the results (e.g. business model concept) within the framework of conceptual models. For the purpose of the demonstration we refer to a small use case to illustrate how the method works.

4.1 Use Case

To demonstrate the applicability of the Business plan creation and of our implementation which supports the generation process, the business model of Skype as case study of Osterwalder and Pigneur's basic literature was used [OP10, p.98]. The core aspect of the business model is to offer video and phone calls via internet based on peer-to-peer technology. Due to the scope of a business plan, the applicability for the creation of a business is only to be shown briefly. The focus is on the creation process and the modelling capability of business model innovation using ideation technology and the presentation as BMC. Figure 5 illustrates the steps for creating a business plan based on our prototype.

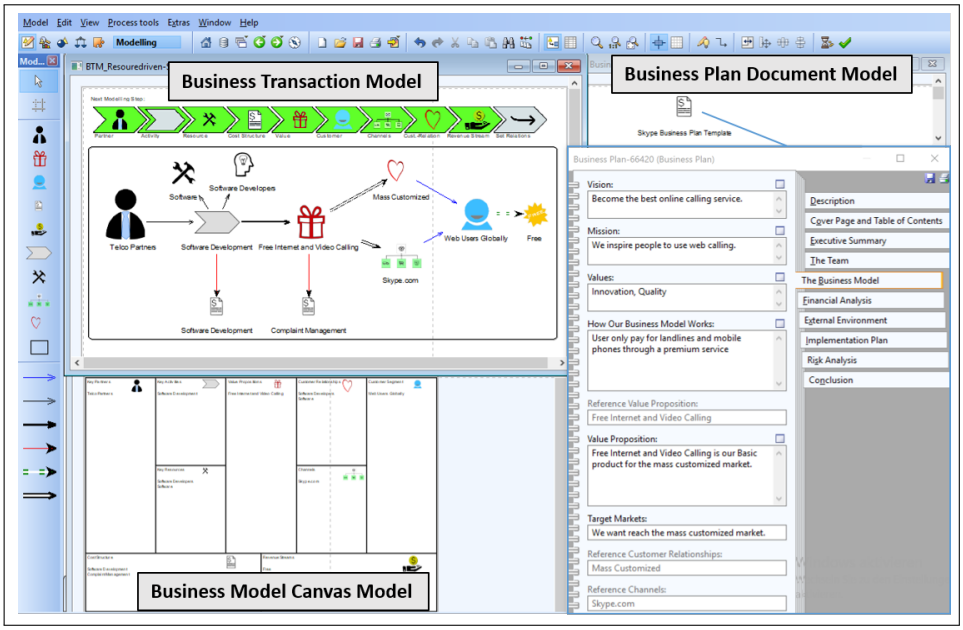


Fig. 5: Screenshot of the Prototype Showing Instances of the Model Types "Business Transaction Model", "Business Model Canvas Model", and "Business Plan Document Model" based on the Business Model of Skype

In a first step, the *"Business Transaction Model"* is used to model business transactions based on their specific relationships by using the elements of the domain-specific modeling language as described in section 3.5. For brevity, we consider only the resource-driven epicenter of a business model innovation in the business model design process[OP10, p.138]. On top of the modeling area, the modeling progress is displayed according to a recommended modeling order - see Figure 6 for the details.

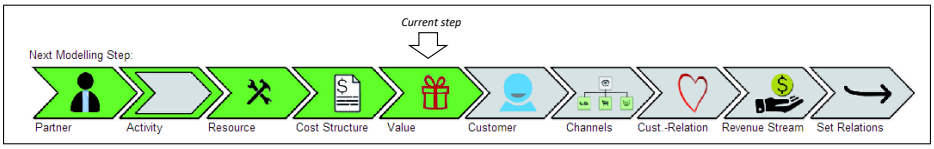


Fig. 6: Progress Bar Indicating the Completed and Next Modeling Steps

For better understanding, the graphical language elements as applied in the use case will be explained based on the modeling flow, i.e. each building block element is followed by the corresponding relation. We begin to place the ”Partner” object on the modeling area and name it ”Telco Partner”, which represents a telecommunication partner. The business partner of Skype ”delivers value” by providing a specific ”Activity”, i.e. in the example of Skype ”Software Development”. Sub elements of an activity like ”Software” and ”Software developers” are related to the activity by the relation ”is part of”, whereby cost factors are represented by the element ”Costs”, e.g. for ”Software development”. These are attached using the relation ”causes”. Following the example of Skype, the value proposition is represented as the service ”Free Internet and Video Calling” and consists mainly of the software development activity. This can be identified by the relation ”is part of”. Additionally, it is possible to identify object-related earnings or costs, for example ”Complaint management” for the offered service. The ”is offered” relation shows the used ”Channel” to provide the service and also identifies the specific ”Relation”, i.e. in our example the ”Mass Customization”. For a customer segment, e.g. ”Web Users Globally”, the object ”Customer Segment” is used. For the assignment of the ”Channel” and ”Relation” elements to a specific ”Customer Segment”, the ”is used” relation is applied. To determine the value of a customer group, e.g. ”Web Users Globally”, it is quantified with the relation ’pays’ together with a ”Revenuestream” element. As soon as all transaction-relevant objects have been modeled, the model instance is stored and it is continued with the BMCM type.

The initial task for creating a *”Business Model Canvas model”* is to place a special aggregation object named *"Business Model Canvas"*. This element has the design of a BMC template, which is a fixed representation that can not be changed by the user. To automatically complete the template, the user has to choose from which *"Business Transaction Model"* the information should be inserted. Technically, we use the references in the meta model for this purpose. After adding a reference, all data are automatically inserted in the Business Model Canvas element. To assign the respective elements, we applied expression attributes, e.g. ”Reference Contain Partner”, which automatically collects the name of each modeled partner object in one variable of a reference model.

The next step is to create the *”Business Plan Document Model”* instance and add the BTM and BMCM references. Figure 5 illustrates this step where attributes like ”Reference Value Propositon”, ”Reference Customer Relationships” or ”Reference Channel” are automatically completed with the previously modeled information from the BTM. Thereby, the object names represent only a short description, e.g. ”Free Internet and Video Calling”, which

serves as a template for the Business Plan Document model. For example ”Value Propositon”, requires a full description with all details of the service in a business plan on the basis of the listed ”Reference Value Propositon” objects.

Thus, in the next step, the user has to add additional data via the graphical interface, which represents the structure of a business plan. All attributes that require text input are part of the final business plan document. Therefore, a complete textual description is required as shown by the attributes ”Vision”, ”Mission”, ”Values”, ”How Our Business Model Works” and ”Target Markets”. After all textual descriptions have been entered in this way, the business plan document can be created with the button "Create business plan". This triggers an algorithm that collects all data and integrates them in a textual document in .docx format. In addition, screenshots of the referenced BTM and BMCM instances are added as graphics at pre-defined positions in the document. This business plan document now serves as a template and can be further completed by other users and then converted into the PDF format as a final business plan.

5 Discussion

The domain-specific modeling method was developed to create business plans based on idea generation techniques. The BTM and the BMCM types were developed for this purpose. Next, a SWOT analysis will be used to identify and discuss strengths and weaknesses based on the requirements of the method [KMM16]. In addition, we mention current limitations.

Strengths of the modeling method and the developed tool are: The method is hybrid, integrated and combines established structured and complex business plan document creation based on the BMC approach (Rq1,Rq3). The modeling method enables fast and intuitive business model concept development and adaption by supporting business model design techniques, e.g. Ideation or visual thinking (Rq4). The business model strategy is supported as the evolution of a business model over time can be documented using the modeling tool. Different business model creation process phases are supported by providing the described techniques for each phase. The DSMM in the BTM serves as a common language basis for initiating business model innovations. The prototype supports techniques of each related BMC design stage, with the focus on the design phase (Rq6). We assume that the graphical implementation of the DSMM within the prototype makes them intuitive to use and therefore easy to learn (Rq7). The structured modeling process simplifies the usage and improves the results by ensuring the completeness of the required information (Rq9). The models created serve as a basis for subsequent models and thus optimise the overall result of the business plan which needs to be created (Rq10).

Weaknesses of our method are, that as soon as the number of objects within the BTM increases, also the complexity of modeling increases and the comprehensibility decreases. A possible solution would be to focus on only one part of a business model in one transaction model, i.e. split the information across different model instances. Furthermore, in the current

version there is no error handling for incorrect relation types. This means, the user has to know which relation is adequate in which context. This may cause problems in using the modeling language consistently. A possible remedy would be to design mechanisms for inspecting the inherent semantics of elements and display hints for better guiding a user [Fi18, Fi17].

Opportunities are, that the business plan document model structure serves as a good starting point for further development. As an example, a financial plan, market- or risk analysis and also the a reimport functionality of the created docx file could be developed. Also constraints for the different relations in a BTM may be useful for a better user guidance. It could be further envisaged that the modeling approach is embedded in application workflows of investment agencies to support start-ups and business incubators in defining their business plans.

Threats may be already established and used procedures, processes and applications for the creation of business plans and the design of business models which prevent the use of the method. The introduction of new approaches is always accompanied by change and this always depends on user acceptance. In the UTAUT model of Venkatesh, these specific influencing factors are described in more detail [VMD03]. A potential solution is to show how the method aligns with established existing approaches, i.e. the BMC.

Limitations are that we finally, have to acknowledge that the prototype has so far only been tested with the mentioned use case. In order to meet the requirements of the target group, the prototype should be tested in a real environment in multidisciplinary teams. These results can be used to adapt the method more specifically to further requirements [KMM16].

6 Conclusion and Outlook

In this paper we described a modeling method for generating business plans using established concepts for business model management. The modeling method was implemented as a first prototype. The practical relevance was demonstrated through a use case, by which we identified that our prototype can serve as a good starting point for generating business plans. However, during the prototype design, we identified a lot of future development potential. A next step for further development could be the integration of a questionnaire for semi-automatically creating the models and thus further guiding the user. Other features may be the integration of a financial business plan for conducting calculations, like a break-even analysis or the combination with approaches for processing the semantic contents of business models, e.g. to further enhance user guidance.

Bibliography

[AF18] Augenstein, Dominik; Fleig, Christian: Towards increased business model comprehension – principles for an advanced business model tool. In: European Conference on Information Systems (ECIS). AIS, 2018.

[AG03] Akkermans, J. M.; Gordijn, Jaap: Value-based requirements engineering: exploring innovative e-commerce ideas. Requirements Engineering, 8(2):114–134, 2003.

[BAD93] Baker, William H.; Addams, H.Lon; Davis, Brian: Business planning in successful small firms. Long Range Planning, 26(6):82–88, 1993.

[BF14] Bork, Domenik; Fill, Hans-Georg: Formal Aspects of Enterprise Modeling Methods: A Comparison Framework. In (Bork, Domenik; Fill, Hans-Georg, eds): HICCS Conference. IEEE Computer Society, pp. 3400–3409, 2014.

[BS96] Brandenburger, Adam M.; Stuart, Harborne W.: Value-based Business Strategy. Journal of Economics & Management Strategy, 5(1):5–24, 1996.

[CYK09] Chen, Xiao-Ping; Yao, Xin; Kotha, Suresh: Entrepreneur Passion And Preparedness In Business Plan Presentations: A Persuasion Analysis Of Venture Capitalists' Funding Decisions. Academy of Management Journal, 52(1):199–214, 2009.

[Fi17] Fill, Hans-Georg: SeMFIS: a flexible engineering platform for semantic annotations of conceptual models. Semantic Web, 8(5):747–763, 2017.

[Fi18] Fill, Hans-Georg: Semantic annotations of enterprise models for supporting the evolution of model-driven organizations. Enterprise Modelling and Information Systems Architectures (EMISAJ), 13:5–1, 2018.

[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.

[FRK13] Fill, Hans-Georg; Redmond, Timothy; Karagiannis, Dimitris: Formalizing Meta Models with FDMM: The ADOxx Case. In (Cordeiro, José; Maciaszek, Leszek A.; Filipe, Joaquim, eds): Enterprise Information Systems, volume 141 of Lecture Notes in Business Information Processing, pp. 429–451. Springer Berlin, Berlin, 2013.

[Ha91] Harrington, H. James: Improving business processes. The TQM Magazine, 3(1), 1991.

[JF17] Johannsen, Florian; Fill, Hans-Georg: Meta Modeling for Business Process Improvement. Business & Information Systems Engineering, 59(4):251–275, 2017.

[KK02] Karagiannis, Dimitris; Kühn, Harald: Metamodelling Platforms. In (Bauknecht, Kurt; Tjoa, A. Min; Quirchmayr, Gerald, eds): E-commerce and web technologies, volume 2455 of Lecture Notes in Computer Science, p. 182. Springer, Berlin, 2002.

[KMM16] Karagiannis, Dimitris; Mayr, Heinrich C.; Mylopoulos, John: Domain-Specific Conceptual Modeling. Springer International Publishing, Cham, 2016.

[Ma06] Markides, Constantinos: Disruptive Innovation: In Need of Better Theory*. Journal of Product Innovation Management, 23(1):19–25, 2006.

[Ma12] Maurya, Ash: Running lean: Iterate from plan A to a plan that works. The lean series. O'Reilly, Beijing, 2nd ed., 17th release edition, 2012.

[Mu13] Mukerjee, Kaushik: Customer–oriented organizations: a framework for innovation. Journal of Business Strategy, 34(3):49–56, 2013.

[Mv03] Maxwell, D.; van der Vorst, R.: Developing sustainable products and services. Journal of Cleaner Production, 11(8):883–895, 2003.

[OP10] Osterwalder, Alexander; Pigneur, Yves: Business model generation: A handbook for visionaries , game changers, and challengers. John Willey & Sons, Hoboken, New Jersey, op. 2010.

[Sa18] Sandkuhl, Kurt; Fill, Hans-Georg; Hoppenbrouwers, Stijn; Krogstie, John; Matthes, Florian; Opdahl, Andreas; Schwabe, Gerhard; Uludag, Ömer; Winter, Robert: From Expert Discipline to Common Practice: A Vision and Research Agenda for Extending the Reach of Enterprise Modeling. Business & Information Systems Engineering, 60(1):69–80, Feb 2018.

[SBK16] Schoormann, Thorsten; Behrens, Dennis; Knackstedt, Ralf: Softwaregestützte Modellierung von Geschäftsmodellen - Vergleich und Weiterentwicklungsperspektiven am Beispiel der Business Model Canvas. In (Mayr, Heinrich C.; Pinzger, Martin, eds): Informatik 2016. Gesellschaft für Informatik e.V., Bonn, pp. 1333–1347, 2016.

[SH13] Shamma, Hamed; Hassan, Salah: Customer–driven benchmarking. Benchmarking: An International Journal, 20(3):377–395, 2013.

[St01] Stähler, Patrick: Geschäftsmodelle in der digitalen Ökonomie: Merkmale, Strategien und Auswirkungen: Zugl.: St. Gallen, Univ., Diss, 2001 u.d.T.: Stähler, Patrick: Merkmale von Geschäftsmodellen in der digitalen Ökonomie, volume Bd. 7 of Reihe. Eul, Lohmar and Köln, 2001.

[Ti98] Timmers, Paul: Business Models for Electronic Markets. Electronic Markets, 8(2):3–8, 1998.

[TK09] Tolvanen, Juha-Pekka; Kelly, Steven: MetaEdit+. In: ACM SIGPLAN conference companion on Object oriented programming systems languages and applications. ACM, p. 819, 2009.

[Ve17] Vendrell-Herrero, Ferran; Bustinza, Oscar F.; Parry, Glenn; Georgantzis, Nikos: Servitization, digitization and supply chain interdependency. Industrial Marketing Management, 60:69–81, 2017.

[VMD03] Venkatesh; Morris; Davis: User Acceptance of Information Technology: Toward a Unified View. MIS Quarterly, 27(3):425, 2003.

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 61

HCI-Patterns für kollaborative Unternehmensmodellierung am Multi-Touch-Tisch

Anne Gutschmidt[1], Valentina Sauer[2], Kurt Sandkuhl[3]

Abstract: Sollen Unternehmensmodelle kollaborativ durch die Stakeholder selbst erstellt werden, bieten sich insbesondere Multi-Touch-Tische (MTT) als nützliches digitales Werkzeug an. Für die Gestaltung von Software und insbesondere deren Benutzungsoberfläche sollte auf bewährte Lösungen - sogenannte HCI-Patterns - zurückgegriffen werden. Das von uns betrachtete Anwendungsfeld bringt jedoch besondere Anforderungen an die Modellierungssoftware für einen Multi-Touch-Tisch mit sich. In einer Studie, angelehnt an die Repertory-Grid-Technik, ließen wir acht Modellierungsexperten, die wir in drei Gruppen einteilten, vier verschiedene Modellierungsprogramme vergleichen und ermittelten so wesentliche Merkmale von Modellierungsoftware für MTT. Diese haben wir mit Katalogen von HCI-Patterns abgeglichen und die Patterns selektiert, die am besten auf die von den Experten geäußerten Bedürfnisse passen.

Keywords: Unternehmensmodellierung; kollaboratives Modellieren; Multi-Touch-Tisch; HCI-Patterns; Modellierungssoftware; Studie

1 Einleitung

Unternehmensmodelle dienen Organisationen dazu, Wissen wie z.B. eigene Strukturen, Abläufe oder Verantwortlichkeiten festzuhalten. Mit ihrer Hilfe kann sowohl der Ist- als auch der Soll-Zustand eines Unternehmens dargestellt werden [Sa14]. Angesichts sich ständig verändernder Rahmenbedingungen, insbesondere durch die Digitalisierung von Arbeit und Produkten, bietet die Unternehmensmodellierung folglich eine große Hilfestellung. Die Partizipative Unternehmensmodellierung (PUM) sieht vor, dass die Mitglieder einer Organisation die Modelle selbst erstellen. Dabei arbeiten sie gemeinsam in kleinen Teams. Die PUM gilt insbesondere dann als sinnvoll, wenn sich verschiedene Stakeholdergruppen auf eine gemeinsame Sichtweise einigen müssen. Man erhofft sich durch das kollaborative Modellieren der tatsächlich Betroffenen eine höhere Qualität und auch Akzeptanz der Modelle [SP18]. Als Werkzeuge zur Modellierung kommen viele in Frage, wie z.B. Whiteboards, wobei digitale Werkzeuge zunehmend als attraktiv empfunden werden [Sa14], da sie einfaches Speichern, Editieren und Teilen der Modelle ermöglichen [MH14, PH09].

[1] Universität Rostock, Institut für Informatik, Lehrstuhl Softwaretechnik, Albert-Einstein-Straße 22, 18059 Rostock, Deutschland anne.gutschmidt@uni-rostock.de
[2] Universität Rostock, Institut für Informatik, Albert-Einstein-Straße 22, 18059 Rostock, Deutschland valentina.sauer@uni-rostock.de
[3] Universität Rostock, Institut für Informatik, Lehrstuhl Wirschaftsinformatik, Albert-Einstein-Straße 22, 18059 Rostock, Deutschland kurt.sandkuhl@uni-rostock.de

Insbesondere mit Multi-Touch-Tischen (MTT) kann gleichzeitiges Arbeiten mehrerer Nutzer auf einer geteilten Arbeitsfläche realisiert werden. In einer früheren Studie wurde bereits gezeigt, dass der MTT für kleinere Modellierungsaufgaben ebenso gut geeignet ist wie ein Whiteboard [Gu18b, Gu18a]. Das Erstellen von Unternehmensmodellen, aber auch die besonders heterogene Gruppe von Nutzern, die wenige bis keine Modellierungsvorkenntnisse besitzt, bringen besondere Anforderungen mit sich, die an Modellierungssoftware für den MTT gestellt werden müssen. So benötigen sie vor allem Unterstützung bei der Anwendung einer Modellierungsnotation und müssen sich zudem eine gemeinsame Benutzungsoberfläche teilen. Das Modellieren stellt dabei eine spezielle Art von Aufgabe dar, bei dem das Ergebnis zwar durch Text angereichert ist, jedoch vorrangig in einem großen Diagramm besteht. Anstatt immer wieder neue Lösungsansätze für Benutzungsoberflächen zu erarbeiten, bieten HCI-Patterns bewährte Musterlösungen im Bereich der Mensch-Computer-Interaktion. Ziel dieses Papiers ist es, eine Auswahl an HCI-Patterns für die kollaborative Unternehmensmodellierung am MTT vorzustellen. Der Beitrag widmet sich daher zwei Forschungsfragen: 1) Welche Eigenschaften einer Modellierungssoftware finden Nutzer im Kontext der PUM wichtig? 2) Welche bereits existierenden HCI-Patterns helfen bei der Umsetzung der gewünschten Eigenschaften? Dazu führten wir eine Befragung unter Modellierungsexperten durch und glichen die Ergebnisse mit verschiedenen HCI-Katalogen ab. In Abschnitt 2 stellen wir zunächst Hintergründe zur Unternehmensmodellierung und zu HCI-Patterns vor. Anschließend beschreiben wir in Abschnitt 3 unsere Studie, in der wir Personen, die bereits über Erfahrung in Unternehmensmodellierung verfügen, fragten, welche Merkmale ihnen beim Vergleich mehrerer Modellierungsprogramme am MTT auffallen und wie wichtig diese für sie sind. Um nicht nur offensichtliche Merkmale zu erfassen, lehnten wir uns in der Befragungstechnik an die Repertory-Grid-Methode an und ließen die Teilnehmer verschiedene Modellierungsprogramme in der Gruppe ausprobieren und danach im Einzelgespräch vergleichen. Die daraus gewonnene Liste von Merkmalen der Software wird in Abschnitt 4 präsentiert. Auf Basis dieser Merkmale schlagen wir im selben Abschnitt HCI-Patterns vor, die nach unserer Einschätzung geeignet sind, um die Bedürfnisse der Nutzer zu erfüllen. In Abschnitt 5 diskutieren wir die Ergebnisse, gehen auf Limitationen ein und geben einen Ausblick auf zukünftige Forschung.

2 Hintergrund

2.1 Unternehmensmodellierung

Durch Unternehmensmodellierung werden verschiedene Perspektiven auf ein Unternehmen systematisch analysiert und in Modellen festgehalten. Zu solchen Perspektiven zählen die Geschäfts- und Produktionsprozesse, Organisationsstrukturen, Produkte und Dienstleistungen, IT-Systeme und vieles mehr [Ve02]. Welche Perspektiven relevant sind und welche Ausschnitte von Unternehmen modelliert werden, hängt vom Modellierungszweck ab. Es existieren zahlreiche Ansätze zur Unternehmensmodellierung, wobei die partizipative Unternehmensmodellierung (PUM) [SPS07] für das Forschungsthema dieses Papiers besondere

Bedeutung hat. Die PUM sieht vor, dass Methodenexperten und Fachexperten aus dem Unternehmen gemeinsam an der Erstellung der Modelle beteiligt sind. In einer Untersuchung von [LW12] wurde bereits gezeigt, dass das aktive Einbinden von Fachexperten in den Modellierungsprozess dazu führt, dass sich diese Fachexperten mehr einbringen, beispielsweise durch wiederholte Korrekturen. Darüber hinaus bereitet die Modellierung den Fachexperten mehr Freude, wenn sie direkt einbezogen werden. Während in [LW12] ein Methodenexperte mit nur einem Fachexperten modelliert, geht der partizipative Ansatz noch weiter. Kleine Teams von vier bis acht Stakeholdern aus dem Unternehmen erarbeiten in Diskussionen eine gemeinsame Sichtweise, die sie schließlich im Modell dokumentieren [SP18]. Die Stakeholder stellen Fachexperten dar, die über wichtiges Wissen aus ihren jeweiligen Unternehmensbereichen verfügen, das für den Modellierungszweck erforderlich ist. Von Beginn an werden also Fachexperten einbezogen, die wertvolles Wissen in die Modelle einbringen, aber oft auch verschiedene Sichtweisen vertreten, über die es einen Konsens zu erzielen gilt. Damit soll die PUM zu einer höheren Qualität und einer größeren Akzeptanz der Modelle führen. Damit sich die Fachexperten bei der Modellierung auf den Inhalt konzentrieren können, werden sie von Methodenexperten unterstützt. Diese kommen meist von außerhalb des Unternehmens. Ein Methodenexperte leitet als Moderator die Modellierungssitzungen und wird oft durch andere Methodenexperten unterstützt, wie z.B. einen Tool Operator zur Unterstützung bei der Bedienung des Modellierungswerkzeuges oder einen Protokollanten, der Ergänzungen zum Modell und Entscheidungen während des Modellierungsprozesses dokumentiert [Sa14, SP18]. Für den MTT als mögliches Modellierungswerkzeug in der PUM spielt die Modellierungssoftware eine tragende Rolle. Sie muss ermöglichen, dass sich auch die oben erwähnten Fachexperten ohne Vorkenntnisse in der Modellierung möglichst gut zurechtfinden. Die funktionalen Anforderungen umfassen dabei im Wesentlichen das Erstellen der Symbole (geometrischer Formen), die bestimmte Konzepte der Modellierungssprache repräsentieren. Diese werden in der Regel durch Konnektoren (z.B. Pfeile oder Linien) verbunden, um Beziehungen darzustellen. Modellelemente werden meist durch Beschriftungen näher spezifiziert. Wird eine formale Modellierungssprache genutzt, so schreibt diese syntaktische und semantische Regeln vor [SP18].

2.2 HCI-Patterns

HCI-Patterns, auch HCI-Designmuster genannt, beschreiben bewährte Lösungen (Best Practices) für immer wieder auftretende User Interface (UI)-Designprobleme und beeinflussen damit implizit auch die Benutzerfreundlichkeit von Softwaretools [SW07]. Diese Patterns sollen zum einen beim Entwurf von Benutzungsoberflächen unterstützen und zum anderen vermeiden helfen, dass Designer die gleichen Lösungen immer wieder neu „erfinden“. Bewährte Designlösungen werden in Form der Patterns verallgemeinert, wiederverwendbar dokumentiert, und können bei der Lösung ähnlicher Probleme verwendet werden [Wu09]. Die Entwicklung von Benutzungsoberflächen ist ein komplexer Prozess, bei dem die Wiederverwendung von Wissen und damit auch HCI-Patterns den Designern

und Entwicklern dabei hilft, effizienter zu arbeiten und die Produktivität zu verbessern [Bo01, KH10b, KH10a, Gu07].

Ein Pattern besteht im Kern aus Kontext, Problem und Lösung [Al77]. Das HCI-Pattern Breadcrumbs wird beispielsweise im Kontext großer hierarchisch strukturierter Websites eingesetzt. Nutzer sollen sich in dieser Struktur orientieren können (Problem), weshalb ihnen ein Pfad beginnend von der höchsten Hierarchieebene bis zur aktuellen Seite angezeigt wird, wobei jede Ebene im Pfad angesteuert werden kann [va08]. Ursprünglich wurde das Konzept der Patterns und deren Verwendung von Christopher Alexander für Architekturentwürfe entwickelt [Al77]. Die „Gang of Four"(GoF) übernahm das Pattern-Konzept für den Entwurf objektorientierter Software [Ga11], bevor es auch von der HCI-Community übernommen wurde. Während GoF-Patterns auch Hinweise beinhalten, wie man ein Pattern implementiert, adressieren HCI-Patterns allgemeine Designaspekte einer Benutzeroberfläche und ihren Zweck für den Benutzer.

Das Konzept der Patterns beinhaltet nicht nur die Muster selbst, sondern auch die sogenannte Pattern-Sprache. Eine Pattern-Sprache besteht aus Patterns und deren Beziehungen, dargestellt als Netzwerk von Patterns. Übergeordnete Patterns in diesem Netzwerk können durch Patterns auf der darunterliegenden Ebene konkretisiert werden [KH10a]. Da die Patterns einer Pattern-Sprache miteinander verbunden sind, ist es offensichtlich, dass eine Pattern-Sprache Patterns für eine bestimmte Familie von Designproblemen in einer definierten Domäne kombiniert [KH10b, LS04]. Nachfolger- und Vorgängerbeziehungen zwischen (übergeordneten und darunterliegenden) Patterns sind ein Schlüsselkonzept bei der Arbeit mit Pattern-Sprachen, da sie das Auffinden eng verwandter Muster ermöglichen [KH10b].

3 Forschungsmethodisches Vorgehen

3.1 Befragung

Um eine Auswahl passender HCI-Patterns für die PUM an einem MTT geben zu können, befragten wir Modellierungsexperten, welche Eigenschaften ihnen an existierenden Modellierungseditoren auffallen und welche sie in dieser spezifischen Situation als wichtig erachten. Besondere Eigenschaften dieser Situation sind, dass a) es sich um das Erstellen von Unternehmensmodellen handelt b) mit einem MTT, c) gemeinsam in der Gruppe gearbeitet wird, und d) die Nutzer nicht notwendigerweise über Modellierungserfahrung verfügen.

3.1.1 Stichprobe

An der Befragung nahmen insgesamt acht Personen teil. In der ersten Befragungsrunde nahmen nur zwei Personen teil. Um für die Erkundung der Software die Bedingung der

Gruppenarbeit herzustellen, nahm eine dritte Person aus dem Forschungsteam teil, die aber nicht in die Befragung einging. Darüber hinaus führten wir die Untersuchung mit zwei weiteren Gruppen durch, jeweils bestehend aus drei Probanden. Alle Teilnehmer hatten Vorkenntnisse in Unternehmensmodellierung und stellten damit für uns Experten dar, die die speziellen Nutzerbedürfnisse in diesem Kontext einschätzen können. Sechs der Probanden hatten bereits durch ihre studentische Ausbildung Hintergrundwissen und praktische Erfahrungen mit Unternehmensmodellierung, die anderen zwei Probanden bilden selbst Studenten in Unternehmensmodellierung aus und hatten bereits an partizipativen Modellierungssitzungen als Methodenexperten teilgenommen.

3.1.2 Erkundung verschiedener Modellierungsprogramme

Für die Befragung baten wir die Probanden, in Gruppen von drei Personen am MTT ein Zielmodell zu erstellen. Zielmodelle stellen sehr grundlegende Modelle dar, für die eine gemeinsam erarbeitete Sichtweise auf Ziele und Probleme eines Unternehmens besonders wichtig und die PUM daher besonders geeignet ist. Durch diese praktische Aufgabe wollten wir den Probanden eine konkrete Vorstellung von dieser Modellierungssituation ermöglichen. Wir wählten für die Untersuchung vier Modellierungsprogramme aus, mit denen eine Zielmodellierung möglich ist: 1) das 4EM-Tool, das speziell für die 4EM-Methode entwickelt wurde und mit dem prinzipiell sämtliche in der Methode enthaltene Modelle erstellt werden können, 2) eine von uns entwickelte Software ModelToGather, mit der Zielmodelle nach der 4EM-Notation erstellt werden können sowie 3) Enterprise Studio und 4) Archi, mit denen Modelle nach der Archimate-Notation erstellt werden können. Die Probandengruppen sollten die Softwareprogramme erkunden können, um diese danach besser beurteilen zu können. Dies war vor allem dann nützlich, wenn ein Proband eine Software vorher nicht kannte oder jetzt nach langer Zeit erst wieder damit in Kontakt kam. Im ersten Testdurchlauf stellten wir fest, dass ein Vergleich von mehr als drei Programmen zu viel Zeit in Anspruch nahm. Daher wurden im ersten Durchlauf nur Archi, Enterprise Studio und ModelToGather betrachtet. Im zweiten und dritten Versuch ersetzten wir Archi durch das 4EM-Tool, da sich Archi und Enterprise Studio sehr stark ähneln und wir möglichst unterschiedliche Programme nutzen wollten. Die Erstellung von drei kompletten Modellen während eines Versuchs war zeitlich ebenfalls nicht realisierbar. Daher bereiteten wir Modelle vor, die die Probanden editieren sollten. Um die für die Zielmodellierung benötigten Funktionen abzudecken, führten wir vorher eine Aufgabenanalyse [St06] durch und identifizierten die wesentlichen Aufgaben, die bei einer Zielmodellierung ausgeführt werden. So mussten Komponenten und Relationen erstellt, geändert und gelöscht werden. Die Probanden starteten immer mit einem Initialmodell, das bei jeder Software ein anderes Beispiel enthielt. Sie erhielten einen Zettel, auf dem gekennzeichnet war, welche konkreten Änderungen vorzunehmen waren.

3.1.3 Befragungstechnik

Im Anschluss an die Erkundung der Software führten wir Einzelinterviews durch, bei denen wir uns an der Repertory-Grid-Technik [Ke] orientierten. Dabei sollten die Probanden immer drei Modellierungsprogramme vergleichen, wobei die besondere Fragestellung lautete: Was haben zwei der Programme gemeinsam, wodurch sie sich von dem dritten unterscheiden? Diese etwas kompliziertere Fragestellung soll zu intensiverem Nachdenken anregen und bestenfalls implizites Wissen zugänglich machen [Le01, Ei93]. Die Methode sieht normalerweise eine größere Anzahl zu vergleichender Elemente vor, die nur mit einer entsprechend großen Stichprobe abgedeckt werden kann. Aus Effizienzgründen entschieden wir uns für eine Beschränkung auf drei Programme. Nachdem wir die Liste dieser Merkmale gemeinsam mit dem Probanden erstellt hatten, sollte dieser einschätzen, wie wichtig er jedes Merkmal einschätzt für die Zusammenarbeit am Multi-Touch-Tisch auf einer Skala von eins (unwichtig) bis fünf (sehr wichtig) [Le01, Ei93]. Die von den Probanden genannten Merkmale der Modellierungssoftware wurden induktiv zu Kategorien zusammengefasst analog zu einer qualitativen Inhaltsanalyse [DB16].

3.2 Vorgehen zur Auswahl von HCI-Patterns

Für die Recherche über HCI-Patterns nutzten wir als Grundlage vielzitierte Pattern-Sammlungen [Ti11, Ti99, va08] sowie weitere Pattern-Sammlungen, die für den betrachteten Kontext passend waren [La03, LHS13, CL16]. Remy et al. bieten darüber hinaus eine Sammlung von HCI-Patterns speziell für den MTT [Re10]. Durch eine vorherige Beobachtungsstudie, vorgestellt in [GSSck], formulierten wir bereits Kategorien, die groben Konzepten entsprechen, denen untergeordnete konkrete HCI Patterns dienen. In jener Studie selektierten wir bereits auf Grundlage der Analyse einer konkreten Software sowie Beobachtungsdaten Patterns, die potentiell in Modellierungssoftware zum Einsatz kommen sollten. Die hier vorgestellte Untersuchung soll unsere bisherigen Erkenntnisse nun erweitern, indem eine weitere Forschungsmethode, die Befragung von Modellierungsexperten, angewandt wird. Deshalb übernahmen wir aus dem oben genannten Beitrag Inhaltskategorien zur Einordnung unserer neuen Befragungsergebnisse. In der Befragung ermittelten wir Softwaremerkmale, aus denen wir HCI-Patterns ableiteten und diese dann den Inhaltskategorien zuordneten. Die Kategorien helfen bei der Gestaltung der Pattern-Sprache, in der Zusammenhänge zwischen den Patterns dargestellt werden. Die erste Kategorie bezieht sich auf das Einsparen bzw. effiziente Nutzen von **Platz**, da wir mit dem MTT nur über eine begrenzte Arbeitsfläche verfügen. Als zweite Kategorie definierten wir **visuelle und physische Erreichbarkeit**, da sichergestellt werden soll, dass jeder Modellierungsteilnehmer alle Elemente eines Modells immer gut erkennen und möglichst auch für Eingaben erreichen soll. Mit der dritten Kategorie, **ausgewogene Partizipation**, betonen wir das Ziel, dass sich alle an einer Modellierungssitzung in gleichem Maße einbringen können. Die Kategorie **Modellierungsunterstützung** fasst Bedürfnisse zusammen, die speziell mit der Modellierungstätigkeit zusammenhängen. Weiterhin definierten wir ein **intuitives Interface**

sowie **einfache Handhabung** jeweils als Kategorie, wobei letztere sich darauf bezieht, dass Interaktionen mit der Software so einfach und wenig aufwendig wie möglich gestaltet werden sollen. Die Kategorie **Input Devices** fasst die Möglichkeiten für Nutzereingaben am MTT zusammen. Die letzte Kategorie **Rationale dokumentieren** spiegelt das Bedürfnis danach wider, Entscheidungsprozesse einschließlich der Urheberschaft von Vorschlägen und Ideen nachvollziehen zu können. Da wir uns am Prinzip einer Pattern-Sprache orientieren, kann ein genanntes Merkmal auch mehr als einer Oberkategorie zugeordnet und damit auch dieser dienlich sein. Auf diese Weise werden die Merkmale in die Pattern-Sprache integriert.

4 Ergebnisse

Die in der Befragung aufgezählten Merkmale der Software entsprachen teilweise bereits konkreten Patterns, teilweise lassen sie sich aber auch als Konzepte auf einer Ebene zwischen den Oberkategorien und konkreten Patterns einordnen. In den folgenden Abschnitten präsentieren wir die Ergebnisse der Befragung und zeigen passende konkrete Patterns dazu auf, die wir den zuvor erwähnten Katalogen entnommen haben. Um einen Eindruck von der resultierenden Pattern-Sprache zu vermitteln, verweisen wir auf Abbildung 1, wobei die konkreten Patterns, die wir den Pattern-Katalogen entnommen haben, unterstrichen dargestellt sind. Pfeile zwischen den Patterns stellen dar, welche Patterns Bestandteil zur Umsetzung eines anderen Patterns sein können.

Eingeschränkter Platz: Vier Befragte gaben als Merkmal für Modellierungssoftware an, dass Elemente entweder problemlos angeordnet waren oder aufgrund des Platzmangels stark überlappten. Dabei wurden allerdings unterschiedliche Aspekte benannt: Das problemlose Anordnen der Komponenten wie z.B. Ziele ohne Überlappung wurde einmal genannt und mit dem Wert 5 (von fünf Stufen) als wichtig eingeschätzt. Zwei Befragte gaben mit einer Wichtigkeit von 3 bzw. 4 an, dass Elemente sich allgemein nicht überlappen, während ein Befragter mit einer Wichtigkeit von 2 nannte, dass Menüs nicht verdeckt sein sollten. Dieses Problem sollte nicht unterschätzt werden, da Unternehmensmodelle sehr groß werden können. Tidwell [Ti11] und van Welie [va08] schlagen **Collapsible Panels** vor, um Platz zu sparen, also Bedienfelder, die beliebig ein- und ausgeklappt werden können, je nach Bedarf. Darüber hinaus findet man bei Tidwell [Ti11] und van Welie [va08] als Pattern den **Drop Down Chooser**. Auf Antippen klappt sich dabei nach unten ein Menü auf, aus dem man eine Option wählen kann. **Hover Tools** [Ti11] werden eigentlich in der Mausinteraktion genutzt. Schwebt der Mauszeiger über einem Element, können dazugehörige Informationen eingeblendet werden. Diese Möglichkeit gibt es bei Touch-Interaktion nicht. Dennoch könnten Zusatzelemente, die nicht dauerhaft benötigt werden, bei Berührung für eine kurze Dauer eingeblendet werden. Es spart ebenfalls Platz, wenn Schaltflächen nur mit **Symbolen und Icons** beschriftet sind [Ti11, va08, LHS13]. Eine weitere Lösung, um dem Platzmangel entgegenzuwirken, ist der Einsatz eines größeren MTT wie bei dem **Large Collaboration**

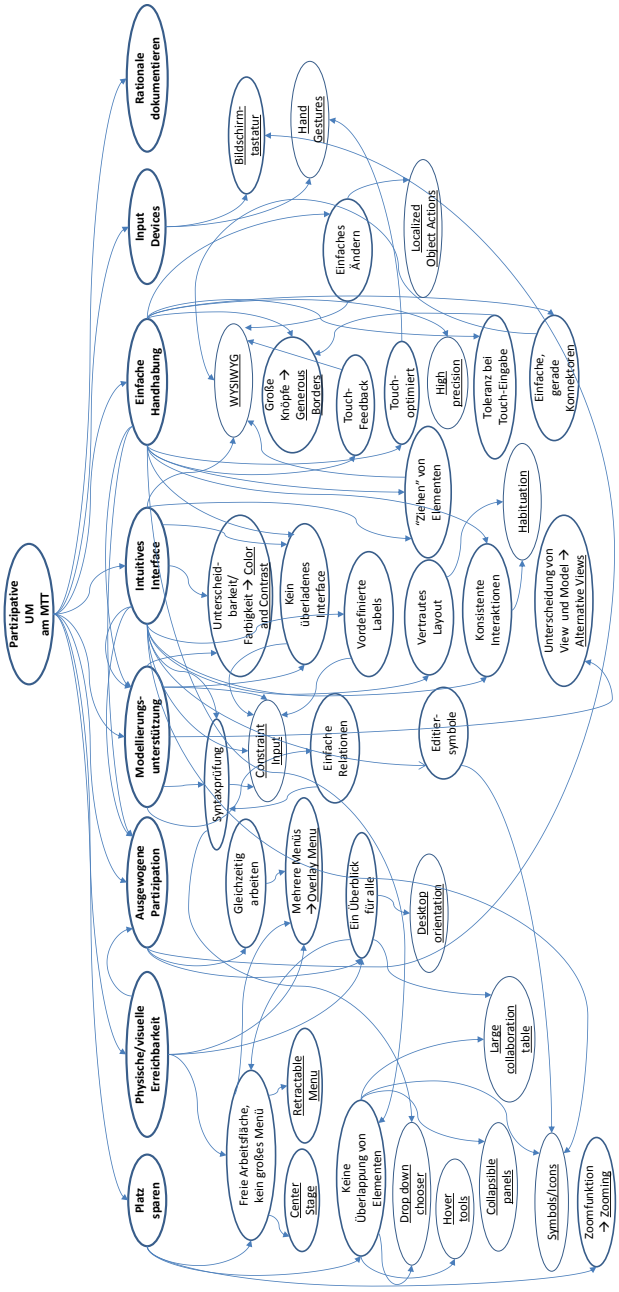


Abb. 1: HCI-Patterns für kollaborative Unternehmensmodellierung am MTT eingeordnet in eine Pattern-Sprache; konkrete Patterns aus Pattern-Katalogen wurden unterstrichen.

Table aus der Pattern-Sammlung von [Re10].
Eine **Zoom-Funktion** wurde als Software-Merkmal von drei Befragten genannt und mit einer durchschnittlichen Wichtigkeit von 3,67 bewertet. Sie entspricht einem HCI-Pattern, das in [Ti11] und [Re10] beschrieben wird. Damit könnten auch größere Modelle auf einer begrenzten Arbeitsfläche erstellt werden.
Eine freie Arbeitsfläche ohne ein fixes großes Hauptmenü, das Platz wegnimmt, nannten drei Personen und bewerteten dies mit einer Wichtigkeit von durchschnittlich 3,67. Viele Programme nutzen bereits das Pattern **Center Stage** [Ti11, va08], bei dem eine große Fläche für den Hauptzweck eines Programmes im Zentrum reserviert ist, während sekundäre Elemente weniger Platz zugeteilt bekommen und in kleineren Panels am Rand angeordnet sind. Dies ist möglicherweise jedoch bei großen Unternehmensmodellen nicht ausreichend. Van Welie führt in seiner Pattern-Liste **Retractable Menus** auf, wonach das Hauptmenü einklappbar sein soll, wenn besonders viel Platz benötigt wird [va08]. **Overlay Menus** werden an der Stelle geöffnet, an der der Mauszeiger [va08] bzw. der Touch-Punkt liegt.

Physische und visuelle Erreichbarkeit: Die freie Arbeitsfläche ohne fixes Menü dient auch der physischen Erreichbarkeit. Gibt es nur ein Hauptmenü, so bedeutet dies meist, dass nur die Personen das Menü erreichen, die diesem am nächsten stehen. Als Lösung dieses Problems kommt vor allem das **Overlay Menu** in Frage, welches an jeder gewünschten Stelle geöffnet werden kann. Dies wurde auch von zwei Personen genannt, allerdings mit einer durchschnittlichen Wichtigkeit von 2,5 bewertet. Dass eine Modellierungssoftware einen Überblick für alle bietet, wurde von einer Person genannt mit einer Wichtigkeit von 4. Hier empfehlen sich die Patterns **Center Stage** und **Desktop Orientation** an. Mit Desktop Orientation werden Inhalte auf einen jeweiligen Betrachter ausgerichtet, so dass jeder Nutzer die Chance hat, das Modell aus seiner Position „richtig herum" zu sehen.

Ausgewogene Partizipation: Sieben Befragte nannten gleichzeitiges Arbeiten als Merkmal und gaben durchschnittlich eine Wichtigkeit von 3,71 an. Das heißt, es war den Probanden tendenziell wichtig, dass sie mit mehreren Personen parallel mit dem MTT interagierten. Eng damit zusammen hängt das Merkmal, dass mehrere Elemente zur gleichen Zeit bearbeitet werden können, wozu wir drei Nennungen mit einer durchschnittlichen Wichtigkeit von 4,33 dokumentieren konnten. Remy et al. [Re10] nennen hier das allgemeine Pattern **Balanced Participation**, bei dem sich Nutzer, die in einer Gruppe arbeiten, möglichst alle einbringen können. Konkret unterstützend für ausgewogene Partizipation schätzen wir die zuvor erwähnten **Overlay Menus** ein, die mehrfach erzeugt werden können. Ausgewogene Partizipation wird ebenfalls durch das Merkmal des Überblicks für alle unterstützt. Hier bietet sich erneut das Pattern **Desktop Orientation** an. Auch ein großer MTT (**Large Collaboration Table**, [Re10]) trägt zu einem guten Überblick bei.

Modellierungsunterstützung: Drei Befragte nannten Syntaxprüfung als ein Merkmal der Software, wobei sie dem eine Wichtigkeit von durchschnittlich 2 zusprachen. Eine der Personen bewertete es mit 2, wenn korrekte und inkorrekte Syntax angezeigt würden. Vordefinierte Labels können ebenfalls beim Modellieren unterstützen. Eine Person erwähnte mit der Wichtigkeit 4 das einfache Beschreiben von Relationen zwischen Komponenten und mit gleicher Wichtigkeit die Nutzung von Popups. Ein Befragter nannte vordefinierte Typen für Relationen, zwei Befragte vordefinierte Typen für Komponenten. In allen Fällen wurde eine Wichtigkeit von 4 angegeben. Als hilfreiche HCI-Patterns identifizierten wir hier **Constraint Input** [va08] sowie **Drop Down Chooser** [va08, Ti11, Ti99]. Beispielsweise könnte die Software zur Laufzeit prüfen, welche Relationslabels für die Beziehung zwischen zwei Komponenten gerade zulässig sind. Soll eine neue Komponente erstellt werden, so könnte dies über ein Drop-Down-Menü geschehen, das nur die Komponententypen enthält, die das Modell vorsieht.
Einige genannte Merkmale sind abhängig von der Modellierungssprache. So scheinen Farben (fünf Nennungen, Wichtigkeit durchschnittlich 4) und allgemein die Unterscheidbarkeit der Modellkomponenten (eine Nennung, Wichtigkeit 4) wichtige Merkmale zu sein. Passendes Pattern ist hier **Color and Contrast** [LHS13]. Darüber hinaus wurden einfache Relationen als Merkmal zweimal genannt mit einer Wichtigkeit von durchschnittlich 4. Bei den möglichen Labels für eine Relation sollte es nicht zu viele Auswahlmöglichkeiten geben, die die Nutzer überfordern. Dies liegt jedoch in der Sprache begründet. Allerdings könnte durch eine Syntaxprüfung zur Laufzeit die Anzahl der Labels reduziert und die Auswahl damit erleichtert werden.
Sieben Befragte nannten ein nicht überforderndes im Gegensatz zu einem überladenen Interface und gaben durchschnittlich eine Wichtigkeit von 3,43 an. Ein nicht überladenes Interface macht es intuitiver und unterstützt ebenfalls das Modellieren. Im Zusammenhang damit wurde auch ein Fokus auf wesentliche Komponenten von zwei Personen genannt mit einer Wichtigkeit von durchschnittlich 3. Sowohl Archimate als auch 4EM sehen vor, dass die verschiedenen Modelle, die mit diesen Notationen angefertigt werden können, miteinander verbunden werden können. In einigen Editoren ist es möglich, nicht nur die Elemente der aktuellen Modellart zu sehen, sondern den gesamten Umfang der Notationselemente. Auch hier könnte das Pattern **Constraint Input** eingesetzt werden. Dem gegenüber stehen jedoch vier Nennungen, die sich auf das Anzeigen aller möglichen Komponenten beziehen, allerdings mit einer Wichtigkeit von durchschnittlich lediglich 2,5. Zwei Personen nannten als Merkmal, dass die Software viele Funktionen anbietet im Gegensatz zu einer kleinen Auswahl von Basisfunktionen, bewerteten dies aber mit einer Wichtigkeit von durchschnittlich nur 1,5. In unserer Stichprobe scheint damit eher ein Wunsch nach Reduktion der Auswahlmöglichkeiten vorzuherrschen.
In einigen Modellierungsprogrammen ist es üblich, zwischen „Ansicht" (View) und „Modell" zu unterscheiden. Wird in diesem Falle eine Komponente in der Ansicht gelöscht, so verschwindet sie dort zwar, ist aber noch im Modell enthalten. Dies wurde von einer Person genannt und mit dem Wert 2 als weniger wichtig eingeschätzt. Bei [Ti11] findet man das Pattern **Alternative Views** für verschiedene Ansichten.

Intuitives Nutzer-Interface: Sieben Personen nannten intuitive Bedienbarkeit als Merkmal einer Modellierungssoftware mit einer durchschnittlichen Wichtigkeit von 4,7. Viele der in der vorherigen Kategorie zur Modellierungsunterstützung genannten Merkmale und Patterns beziehen sich auch auf die intuitive Bedienung der Software. Dazu zählen eine nicht überladene Bedienoberfläche, Unterscheidbarkeit und Farben der Modellelemente und vordefinierte Labels. Dementsprechend sind für intuitive Bedienbarkeit auch dieselben Patterns nützlich, vor allem **Constraint Input** und **Color and Contrast**. Auch die allgemeine Syntaxprüfung macht die Bedienung intuitiver.
Ein vertrautes Layout, das von anderen Anwendungen bekannt ist, wurde von einem Befragten mit der Wichtigkeit 2 angegeben. Tidwell [Ti11] beschreibt ein Muster für Nutzerverhalten, das sie Habituation nennt. Es beschreibt, dass Nutzer erwarten, dass bestimmte Interaktionen so funktionieren, wie sie es von anderen Programmen gewohnt sind. Dass Interaktionen für ähnliche Aktionen gleich und damit konsistent über die gesamte Anwendung, insbesondere das Öffnen von Menüs, gleich sein sollten, erwähnte eine Person mit der Wichtigkeit 1. Auch dies setzen wir in Zusammenhang mit dem Bedürfnis nach **Habituation**, wenngleich dies für die Probanden von geringerer Bedeutung zu sein schien.
Auch das „Herüberziehen" von Objekten, wenn z.B. eine Komponente erzeugt (eine Nennung, Wichtigkeit 4) oder eine Relation von einer zur anderen Komponente gezeichnet wird (Vier Nennungen, durchschnittliche Wichtigkeit 3,75), zählt zur intuitiven Bedienung. Dies entspricht der Idee des Patterns **WYSIWYG** (What you see is what you get) [Ti99], bei dem der Nutzer in Echtzeit das Resultat seiner Interaktionen mit dem Modell sehen kann.
Eine Person erwähnte Symbole, die die Möglichkeiten des Editierens signalisieren, mit einer Wichtigkeit von 3, was durch das Pattern **Symbols/Icons** umgesetzt wird.

Einfache Handhabung: Wie bereits angedeutet, gibt es einige Überschneidungen zwischen intuitiver Bedienbarkeit und einfacher Handhabung. Dazu zählen eine nicht überladenen Bedienoberfläche, das „Ziehen" von Objekten und konsistente Interaktionen bei Menüs. Entsprechend sollten die oben bereits erwähnten Patterns hilfreich sein. Auch sollten sich Elemente nicht zu stark überlappen.
Darüber hinaus sollten nach der Meinung von zwei Personen mit einer durchschnittlichen Wichtigkeit von 3,5 Knöpfe groß sein. Bei drei Befragten fanden wir sehr unterschiedliche Aussagen zur Präzision von Touch-Eingaben bzw. zur Toleranz bei „vorschnellen" Eingaben. Einerseits kann es frustrierend sein, wenn eine Software schwerfällig oder gar nicht auf Eingaben reagiert. Andererseits kann es ebenso den Arbeitsablauf stören, wenn das Nutzer-Interface auf jede Berührung übersensibel reagiert. Das Pattern **Generous Borders** [Ti11] entspricht dem Bedürfnis nach Knöpfen, die groß genug sind, und sorgt für eine Toleranz bei Eingaben durch größere Abstände zwischen Knöpfen. Remy et al. [Re10] führen das Pattern **High Precision Input** an, unter dem sie verschiedene Maßnahmen zusammenfassen, um Touch-Eingaben möglichst genau zu erfassen, z.B. den Einsatz von Gesten oder Algorithmen, die die Präzision der Eingabeerfassung verbessern.
Zwei Personen nannten mit einer durchschnittlichen Wichtigkeit von 2,5 Touch-Feedback. Das Nutzer-Interface soll entweder optische oder haptische Rückmeldung bei Berührung

geben. Dies entspricht der Idee des **WYSIWYG**. Drei Personen bewerteten durchschnittlich mit dem Wert 3, dass die Software für Touch-Eingaben ausgelegt oder optimiert sein sollte. In diesem Zusammenhang steht möglicherweise auch eine Aussage in unserer Befragung über ein eigenes Bedienkonzept, das mit einer Wichtigkeit von 3 eingeschätzt wurde. Wenn eine Software am MTT genutzt wird, sollte sie auch Touch-Gesten nutzen (**Hand Gestures**, [Re10]).
Ein weiteres Merkmal ist das einfache Ändern des Modells. Sechs Personen gaben mit einer durchschnittlichen Wichtigkeit von 4 die Möglichkeit zur Änderung des Komponententyps an. Im Gegensatz dazu müsste ein Nutzer eine Komponente löschen, neu erstellen und eventuell dazugehörige Relationen aktualisieren. Auch das Ändern von Relationen wurde von drei Personen mit einer durchschnittlichen Wichtigkeit von 4 genannt. Für diese Änderungen erscheinen **Localized Object Actions** [Ti99] geeignet. Sie sehen vor, dass Bedienelemente zum Bearbeiten eines Objektes gruppiert und in der Nähe dieses Objektes angeordnet werden. Zwei Nennungen bezogen sich allgemein auf nutzerfreundliche Änderbarkeit mit einer Wichtigkeit von durchschnittlich 3,5. Bei drei Nennungen ging es um das schnelle Editieren von Bezeichnungen ohne vorherige umständliche oder unklare Interaktionen, was mit einer durchschnittlichen Wichtigkeit von 2,3 bewertet wurde. Eine Umsetzung des Patterns **WYSIWYG** wäre, dass, ohne sich durch Menüs oder andere zusätzliche Schritte arbeiten zu müssen, ein Textfeld sofort geändert werden könnte.
Ein Befragter störte sich an Konnektoren, die nicht gerade sondern kompliziert zwischen Komponenten verliefen. Mit einer Wichtigkeit von 4 bewertete der Befragte das Merkmal einfacher gerader Konnektoren. Hier müsste sich eine Software stärker am Pattern **WYSIWYG** orientieren, damit der Verlauf der Konnektoren für die Nutzer berechenbarer ist.
Auch das Herüberziehen von Komponenten und Konnektoren unterstützt die einfache Handhabbarkeit und setzt, wie oben bereits angeführt, das Pattern **WYSIWYG** um.

Input Devices: Vier Personen erwähnten die Bildschirmtastatur als Merkmal einer Modellierungssoftware und bewerteten dies mit einer Wichtigkeit von durchschnittlich 4,5. Sofern diese erreichbar gemacht wird für alle, z.B. durch die Möglichkeit mehrere Instanzen, die beliebig platziert werden können, dient dies auch der ausgewogenen Partizipation. Die genannten Merkmale, dass eine Anwendung für Touch optimiert ist oder ein eigenes Bedienkonzept hat, passen zum Pattern **Hand Gestures**.
In den Aussagen der Probanden fanden wir keine Hinweise zur Kategorie „Rationale dokumentieren".

5 Diskussion

5.1 Zusammenfassung

Ein Pattern oder Muster ist erst dann ein Muster, wenn es sich um eine wiederholt eingesetzte Lösung für ein wiederkehrendes Problem handelt [Ga11]. Dies gilt auch im Bereich der

Design-Lösungen für Benutzungsoberflächen. In diesem Papier haben wir eine Studie vorgestellt, in der nach Patterns für Software gesucht wurde, die speziell in der partizipativen und kollaborativen Unternehmensmodellierung am MTT eingesetzt werden soll. Ziel war es also nicht, neue Patterns zu entwickeln, sondern aus vorhandenen Pattern-Sammlungen diejenigen herauszusuchen, die die besonderen Nutzerbedürnisse in diesem speziellen Kontext erfüllen helfen. Um ein schärferes Bild einer Modellierungssoftware und auch von den Nutzerbedürfnissen zu erhalten, nutzten wir eine Befragungstechnik angelehnt an die Repertory-Grid-Technik. Im Vergleich von verschiedenen Modellierungsprogrammen nannten uns die Probanden Merkmale entsprechender Software. Einige dieser Merkmale lassen sich bereits als Muster in Pattern-Sammlungen finden, wie z.B. Zooming, Color and Contrast oder Symbole. Andere Merkmale wurden eher als allgemeine Eigenschaften oder Anforderungen formuliert, die wir als Konzepte in eine Pattern-Sprache einordneten, wie z.B. das Bedürfnis, dass sich Elemente nicht überlappen sollten, oder das Bedürdnis, dass Teile des Modells möglichst einfach geändert werden können. Diese Merkmale brachten wir mit konkreten Patterns in Verbindung, die wir für die Erfüllung der dazugehörigen Bedürfnisse für passend halten.

Bei genauerer Betrachtung der Patterns fällt auf, dass es zahlreiche Patterns gibt, die positiv zusammenwirken oder einander unterstützen. So ist Desktop Orientation, also das flexible Ausrichten des Inhalts auf wechselnde Betrachter, auch eine Möglichkeit, visuelle Erreichbarkeit für alle zu gewährleisten. Allerdings ergeben sich auch zahlreiche Widersprüche zwischen Patterns. So kann die Desktop Orientation die Zusammenarbeit und damit auch die ausgewogene Partizipation stören. Die Nutzer müssen sich einig sein über eine Veränderung der Ausrichtung. Dabei kann es zu Unterbrechungen im Arbeitsfluss kommen und ohne klare Organisation können bestimmte Ausrichtungen und damit auch Nutzer blockiert werden. Das Overlay Menu wurde als Möglichkeit vorgeschlagen, um große Hauptmenüs zu ersetzen und damit Platz zu sparen und ausgewogene Partizipation zu unterstützen. Werden allerdings mehrere Menüs gleichzeitig genutzt, die sehr groß sind, kann dies wieder Platz wegnehmen.

Einige der von den Befragten genannten Merkmale sind auf die Modellierungssprache zurückzuführen, wie z.B. der Einsatz von Farben, welcher von mehr als der Hälfte der Probanden tendenziell als wichtig eingestuft wurde, oder die Anzahl zur Verfügung stehender Labels, die potentiell überfordern kann. In der Praxis sollte vor Durchführung eines Modellierungsprojektes auf diese Aspekte geachtet werden, wenn eine Modellierungssprache ausgesucht wird.

In den Interviews wurde deutlich, dass Platz auf der Arbeitsfläche sowie intuitive Bedienbarkeit, einfache Handhabung und physische und visuelle Erreichbarkeit unabdingbare Voraussetzungen für ausgewogene Partizipation und Modellierung sind. Obwohl nach der Wichtigkeit der genannte Merkmale für die Zusammenarbeit gefragt wurde und sie mit Zusammenarbeit eigentlich nicht direkt in Zusammenhang stehen, wurden besonders diese Kategorien häufig genannt. Sie stellen damit eine Art Hygienefaktoren für die gemeinsame Modellierung dar. Fehlen sie, ist Zusammenarbeit nicht möglich, ihre besondere Förderung

steigert die Qualität der Zusammenarbeit an sich vermutlich jedoch nicht. Die Software muss jedoch darüber hinausgehende Merkmale aufweisen, um dann tatsächlich Zusammenarbeit zu fördern. So wurde mit tendenziell hoher Wichtigkeit gleichzeitiges Arbeiten bewertet. Ein Pattern, das das gleichzeitige Bearbeiten mehrerer Elemente beschreibt, haben wir in den Pattern-Sammlungen bisher nicht gefunden. Eventuell könnte sich hier in Zukunft ein neues Pattern fomulieren lassen, welches das eher unspezifische Pattern Balanced Participation von [Re10] etwas konkreter fortführt.

5.2 Limitationen und Ausblick

Bei der vorgestellten Studie handelt es sich nur um eine kleine Untersuchung mit wenigen Probanden und wenigen Modellierungsprogrammen. Eine Ausweitung der Untersuchung wäre wünschenswert. Dabei möchten wir ein breiteres Spektrum von Nutzern einbeziehen. Dazu gehören neben Experten in der Modellierung auch Nichtexperten, sowie Nutzer, die sowohl erfahren als auch unerfahren mit Touch-Interaktionen sind. In einem realistischeren Kontext ergeben sich vielleicht andere Schwerpunkte, wie z.B. die Dokumentation der Entscheidungsfindung oder Urheberschaft (Rationale), die in der aktuellen Befragung keine Rolle spielte. Darüber hinaus streben wir an, mehr Softwareprodukte zu vergleichen, die speziell für MTT ausgerichtet sind, wie z.B. metasonic Process Touch. Zukünftige Studien müssten dabei über die bisher betrachtete Zielmodellierung hinaus gehen und andere Modellarten wie z.B. Prozessmodelle einbeziehen. Dennoch liefert uns die Untersuchung bereits einige Erkenntnisse über die Nützlichkeit bestimmter HCI-Patterns. Ob die Liste der Patterns vollständig ist, können wir nicht mit Sicherheit beantworten. Unsere Ergebnisse sind als Hypothesen zu verstehen, die es zu prüfen gilt. Erst mit zunehmender Nutzung von MTT für kollaborative Modellierung und der Untersuchung weiterer Modellierungsprogramme können mehr und mehr zuverlässige Aussagen über wiederholt eingesetzte Musterlösungen getroffen werden. Unser Beitrag stellt daher einen ersten Ansatz zur Entwicklung einer Pattern-Sprache für PUM an MTT dar. Vorausgesetzt, dass diese wie oben beschrieben weiterentwickelt wird, kann sie als Wissens- und Kommunikationsbasis für User-Interface-Entwickler in diesem spezialisierten Bereich dienen.

Literaturverzeichnis

[Al77] Alexander, Christopher: A pattern language, Jgg. 2 in Center for Environmental Structure series. Oxford Univ. Pr, 1977.

[Bo01] Borchers, Jan O.: A pattern approach to interaction design. 15:359–376, 2001.

[CL16] Experiences - A Pattern Language for User Interface Design. http://www.maplefish.com/todd/papers/Experiences.html#Interaction.

[DB16] Döring, Nicola; Bortz, Jürgen: Forschungsmethoden und Evaluation in den Sozial- und Humanwissenschaften. Springer-Lehrbuch. Springer, 5. vollständig überarbeitete, aktualisierte und erweiterte auflage. Auflage, 2016. Döring, Nicola (VerfasserIn) Bortz, Jürgen (VerfasserIn) Pöschl-Günther, Sandra (MitwirkendeR).

[Ei93] Einführung in die Repertory Grid-Technik. Bd. 1. Grundlagen und Methoden, 1993. Scheer, Jörn W. (Hrsg.) Catina, Ana (Hrsg.).

[Ga11] Gamma, Erich; Helm, Richard; Johnson, Ralph; Vlissides, John: Design patterns. Addison-Wesley professional computing series. Addison-Wesley, 39. printing. Auflage, 2011.

[GSSck] Gutschmidt, Anne; Sauer, Valentina; Sandkuhl, Kurt: Identifying HCI Patterns for the Support of Participatory Enterprise Modeling on Multi-Touch Tables. The Practice of Enterprise Modeling, in Druck.

[Gu07] Guerrero-García, Josefina; González-Calleros, Juan Manuel; González-Monfil, Adelaida; Pinto, David: A method to align user interface to workflow allocation patterns. In (González Calleros, Juan Manuel; Collazos Ordoñez, Cesar A.; Guerrero-García, Josefina, Hrsg.): Proceedings of the XVIII International Conference on Human Computer Interaction. ICPS. ACM, S. 1–8, 2007.

[Gu18a] Gutschmidt, Anne: Empirical Insights into the Appraisal of Tool Support for Participative Enterprise Modeling. In: Proceedings of the 9th International Workshop on Enterprise Modeling and Information Systems Architectures, Rostock, Germany, May 24th - 25th, 2018. S. 70–74, 2018.

[Gu18b] Gutschmidt, Anne: On the Influence of Tools on Collaboration in Participative Enterprise Modeling–An Experimental Comparison between Whiteboard and Multi-Touch Table. In: ISD 2018. 2018.

[Ke] Kelly, George Armstrong: The psychology of personal constructs. Routledge. Kelly, George Armstrong (VerfasserIn).

[KH10a] Kruschitz, Christian; Hitz, Martin: Analyzing the HCI design pattern variety. In (Hanyuda, Eiichi, Hrsg.): Proceedings of the 1st Asian Conference on Pattern Languages of Programs. ACM, S. 1, 2010.

[KH10b] Kruschitz, Christian; Hitz, Martin: Human-Computer Interaction Design Patterns. 3, 2010.

[La03] User Interface Design Patterns. `https://www.cs.helsinki.fi/u/salaakso/patterns/index.html`, Accessed: 2019-05-02.

[Le01] Leach, Chris; Freshwater, Kate; Aldridge, Jan; Sunderland, Joanne: Analysis of repertory grids in clinical practice. 40:225–248, 2001.

[LHS13] Lockton, Dan; Harrison, David; Stanton, Neville A: Exploring design patterns for sustainable behaviour. The Design Journal, 16(4):431–459, 2013.

[LS04] Lukosch, Stephan; Schümmer, Till: Communicating Design Knowledge with Groupware Technology Patterns. In (de Vreede, Gert-Jan, Hrsg.): Groupware: Design, Implementation and Use, Jgg. 3198 in Lecture Notes in Computer Science, S. 223–237. Springer Berlin Heidelberg, 2004.

[LW12] Luebbe, Alexander; Weske, Mathias: Determining the Effect of Tangible Business Process Modeling. In (Plattner, Hasso; Meinel, Christoph; Leifer, Larry, Hrsg.): Design Thinking Research: Studying Co-Creation in Practice. Springer Berlin Heidelberg, Berlin, Heidelberg, S. 241–257, 2012.

[MH14] Mercier, E.; Higgins, S.: Creating joint representations of collaborative problem solving with multi-touch technology. Journal of Computer Assisted Learning, 30(6):497–510, 2014.

[PH09] Piper, Anne Marie; Hollan, James D.: Tabletop Displays for Small Group Study: Affordances of Paper and Digital Materials. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. CHI '09, ACM, New York, NY, USA, S. 1227–1236, 2009.

[Re10] Remy, Christian; Weiss, Malte; Ziefle, Martina; Borchers, Jan: A Pattern Language for Interactive Tabletops in Collaborative Workspaces. In: Proceedings of the 15th European Conference on Pattern Languages of Programs. EuroPLoP '10, S. 9:1–9:48, 2010.

[Sa14] Sandkuhl, Kurt; Stirna, Janis; Persson, Anne; Wißotzki, Matthias: Enterprise modeling. The Enterprise Engineering Series. Springer Berlin Heidelberg, 2014.

[SP18] Stirna, Janis; Persson, Anne: Enterprise Modeling - Facilitating the Process and the People. Springer International Publishing, 2018.

[SPS07] Stirna, Janis; Persson, Anne; Sandkuhl, Kurt: Participative Enterprise Modeling: Experiences and Recommendations. In (Krogstie, John; Opdahl, Andreas; Sindre, Guttorm, Hrsg.): Advanced information systems engineering, Lecture Notes in Computer Science 4495. Springer, 2007.

[St06] Stanton, Neville A.: Hierarchical task analysis. 37:55–79, 2006. Journal Article Research Support, Non-U.S. Gov't Review.

[SW07] Specker, Markus; Wentzlaff, Ina: Exploring Usability Needs by Human-Computer Interaction Patterns. In (Winckler, Marco; Johnson, Hilary; Palanque, Philippe, Hrsg.): Task Models and Diagrams for User Interface Design, Jgg. 4849 in Lecture Notes in Computer Science, S. 254–260. Springer Berlin Heidelberg, 2007.

[Ti99] Common Ground. `http://www.mit.edu/~jtidwell/common_ground_onefile.html`, Accessed: 2019-05-02.

[Ti11] Tidwell, Jenifer: Designing interfaces. Safari Tech Books Online. O'Reilly, 2nd ed.. Auflage, 2011.

[va08] Welie.com: Patterns in Interaction Design. `http://www.welie.com/patterns/index.php`, Accessed: 2019-07-30.

[Ve02] Vernadat, F.B.: Enterprise modeling and integration (EMI): Current status and research perspectives. Annual Reviews in Control, 26(1):15 – 25, 2002.

[Wu09] Wurhofer, Daniela; Obrist, Marianna; Beck, Elke; Tscheligi, Manfred: Introducing a Comprehensive Quality Criteria Framework for Validating Patterns. In (Dini, Petre, Hrsg.): Computation world. IEEE, S. 242–247, 2009.

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 77

Understanding individual processes of conceptual modeling

A multi-modal observation and data generation approach

Kristina Rosenthal[1], Benjamin Ternes[1], Stefan Strecker[1]

Abstract: How conceptual modeling is performed by modelers, how modeling processes proceed, which modeling difficulties modelers face and why, and how to overcome these difficulties by tailored modeling support has received limited attention in conceptual modeling research so far. Studying individual modeling processes by observing modelers during conceptual modeling contributes to identifying modeling difficulties, and to understand whether modelers require tailored modeling support. Based on TOOL, a modeling tool and research observatory for studying modeling processes, we design a multi-modal observation and data generation approach and report its application to exploratory studies of individual modeling processes, and show how complementary modes of observation are integrated during data analysis for a richer and more complete understanding of modeling processes. Moreover, we discuss how the multi-modality of observations contributes to understanding modeling processes and modeling difficulties, and how observations during modeling feed back into the software development of TOOL.

Keywords: Conceptual modeling; Modeling process; Modeling tool; Research methodology

1 Introduction and rationale

Conceptual modeling as an activity involves an intricate array of cognitive processes and performed actions including abstracting, conceptualizing, contextualizing, associating, visualizing, interpreting & sense-making, judging & evaluating, and, in group settings, communicating, discussing and agreeing [RS19; RTS19]. Conceptual modeling involves mastering theoretical foundations, modeling languages and methods, applying them to practical problems, and, while performing a modeling process, critically thinking and reflecting upon an application domain in terms of its technical languages, the natural languages in use in the domain, their imprecision, ambiguities and related challenges [WO80]. Learning and performing conceptual modeling is, hence, construed as a complex task based on codified as well as tacit knowledge [SS17] guided by theoretical underpinnings and refined by reflecting practical experience.

Despite its obvious relevance, the process (or 'act') of conceptual modeling has received limited attention so far [HLP06; HPv05]. How conceptual modeling is performed by

[1] University of Hagen, Enterprise Modelling Research Group, Universitätsstr. 41, 58084 Hagen, Germany
{kristina.rosenthal,benjamin.ternes,stefan.strecker}@fernuni-hagen.de

modelers, how modeling processes proceed, which modeling challenges and difficulties modelers experience and why, and how to overcome these difficulties by tailored modeling support has been subject to only a few studies [e. g., Se16; Ve96]. At large, surprisingly little is known about the actual act of conceptual modeling, about the reasoning of modelers and their deliberations (e. g., about modeling decisions), and whether different (idealized) types of modelers can be identified, e. g., by identifying patterns of modeling processes and/or difficulties, and whether these modeler types require different modeling support. Similarly little is known about how individuals learn conceptual modeling, about their learning progress, need for learning support and tool assistance [RTS19].

Studying individual modeling processes promises to shed further light on these largely unknowns, and, ultimately, to enable us to design targeted (tool) support for modelers at different stages of their learning and mastering of conceptual modeling. Observing the reasoning of modelers and the multifaceted cognitive processes carried out while conceptual modeling, however, faces a number of methodological challenges: Neither the relevant cognitive processes nor the corresponding deliberations on modeling decisions are directly observable. Hence, research on conceptual modeling processes has reverted to observable aspects of modeling processes such as modelers' interactions with software tools, modelers' eye movements, or their verbalizations of their thought processes ('think aloud' [ES93]).

Following this trajectory, this paper presents the design of a multi-modal observation and data generation approach to conduct studies of individual modeling processes—based on TOOL, a webbrowser-based modeling tool and modeling observatory we have been developing for the past six years [Te17; Te19; TS18]. Rather than relying on a single mode of observation, the multi-modal observation approach presented in this paper complements different modes of observation to provide a more complete picture of the phenomena under investigation [VBB13; VBS16]. Underlying our research is the fundamental assumption that modeling processes demand and deserve study from several complementary perspectives following Berger and Luckmann's remarkable view that "the object of thought becomes progressively clearer with this accumulation of different perspectives on it" [BL67, p. 10]. The richness of cognitive processes involved in conceptual modeling and the complexity of conceptual modeling as an activity is the main rationale for combining multiple modes of observation for studying individual modeling processes [cf. RS19]. Complementing different modes of observation is a research strategy common to mixed methods research designs [CP18], and assumed to allow us to identify a wider range of facets about the modeling process assisting us in better understanding individual modeling processes.

Analyzing think aloud protocols has shown promising results for understanding cognitive processes of subjects working on problem-solving tasks in general and modeling tasks in particular [e. g., BD92; ES93], and we consider verbalization of thoughts as the best available means of expression for achieving insights into modelers' reasoning as our spoken language provides us with a rich and flexible tool to express our thinking. However, reasoning of modelers will not always be observable from think aloud protocols alone, since modelers' movements and gestures entail important additional cues about their modeling challenges

and difficulties, for instance. Hence, the approach presented in this paper complements think aloud (verbal) protocols with recording interactions of modelers with the modeling tool as well as videotaping modelers to allow for additional visual clues, e. g., regarding interaction with pen & paper or modelers' movements during modeling. Additionally, subjects are surveyed about their modeling process before and after they work on a (controlled) modeling task. This specific mix of observation modes is tailored to observing modeling processes using a modeling tool and is assumed to provide a more complete picture of modeling processes and to allow for identifying a wider range of modeling challenges and difficulties.

Indeed, to ask subjects to think aloud and to complement verbal protocols with observations from such recordings is a second-best approach, warranted only because it is not possible to directly access and capture cognitive processes and, thus, modeler reasoning while modeling. Modelers may have difficulties verbalizing their reasoning while modeling on principle accounts (because verbalizing own thoughts can be difficult) or on modeling-related accounts (e. g. because of the difficulty of finding the right words to express oneself). However, among all possible alternative modes of observation, think aloud verbalization promises the richest insights into important non-directly observable aspects of the modeling process.

Section 2 discusses related work before we introduce TOOL in Section 3. In Section 4, we explain the multi-modal observation and data generation approach and report on the accompanying data analysis strategy. First exploratory studies are reported in Section 5, followed by a discussion and conclusion in Section 6.

2 Related work

Prior work has investigated individual modeling processes (e. g., data or process modeling processes) using different modes of observation and different data generation approaches. Early contributions primarily focus on data modeling processes. Using verbal protocols, [BD92] identify similarities and differences between non-experienced and experienced modelers when constructing a conceptual data model. In a subsequent study, errors of novice modelers are investigated in two laboratory experiments [BA94]. This study evaluates errors in data models constructed by novices complemented with analyzing verbal protocols with the aim to achieve insights into why the novices committed the errors [BA94, p. 64]. The two studies led to suggestions for supporting novices in data modeling with immediate feedback in supportive tools [BA94, pp. 66f]. The behavior of modelers while constructing conceptual data models is investigated in [ST95]. Based on two laboratory studies using think aloud protocols, insights into how individuals use heuristics to perform their modeling processes are achieved. A more recent study investigates how ontological modeling guidelines assist modelers in constructing UML class diagrams [Be11]. In a laboratory setting, verbal data protocols of subjects creating UML class diagrams are collected and analyzed for numbers of encountered modeling difficulties.

This stream of research primarily strives to better understand how humans construct process models and how the result of the modeling process, i. e., the process model, is affected by different modeling styles [e. g., Cl15a; Pi15]. One prominent approach in this research stream is to record modeler-tool interactions, e. g., placing a representation of a notation symbol on a virtual canvas in a software tool, and to analyze the recorded modeling processes using data mining techniques and cluster analysis [Pi15, p. 1061]. The study leads to identifying three distinct styles of modeling that could be observed in modeling processes [Pi15, p. 1059]. An accompanying modeling environment supporting the analysis of modeling processes named Cheetah Experimental Platform (CEP) has been suggested in early contributions to the discussion on the process of process modeling [e. g., PZW10]. CEP is, in particular, designed to support the workflow of (controlled) experiments—each step in the experimental workflow uses components provided by CEP [PZW10, p. 15]. The tool offers different analysis functionalities, e. g., the Cheetah Analyzer that records the process of creating process models with Cheetah Modeler [PZW10, p. 17] and allows to replay the modeling processes with the integrated step-by-step execution functionality. While the step-by-step execution allows for reconstructing the process of process modeling, important thoughts and cognitive performances are not observed, e. g., when creating synchronizations/parallelizations. For further exploring behavior patterns incorporating modeler-specific and task-specific factors, it is envisioned to track modeler-tool interactions, complemented with think aloud protocols and eye movement tracking data [Pi14].

Further related work identifies cognitive process modeling techniques by observing and tracking modeler-tool interactions while working on process modeling tasks [Cl15a]. An approach visualizing how process models are created is introduced allowing for further analysis as, for example, specific pattern reconstruction, e. g., to identify patterns of delete operations [Cl15b, p. 158]. This approach bases on process mining and collects data of modelers while working on modeling tasks. Tracked data is analyzed using step-by-step replays of the modeling process and visualized in so-called PPMCharts [Cl15a, pp. 1404f]: Every interaction of a modeler with the modeling canvas constructing a process model is visualized in a specific color and shape depending on the type of interaction, e. g., change operation [Cl15a, p. 22]. The icons are positioned on horizontal timelines, each of which refers to a model element. The complementary use of replaying and visualizing modeler-tool interactions in [Cl15a] inspired the observation approach for modeling processes reported in the present work.

Hoppenbrouwers et al. take a communication-theoretic perspective and investigate the modeling process as a dialogue [HLP06; HPv05]. To capture modeling processes, especially, modeling decisions, a part of the meta-model of a modeling laboratory aimed at gathering empirical data on the details of modeling processes is proposed [HLP06]—closely related to the observation approach presented in this work. However, we were not able to find an application of the laboratory suggested in [HLP06] so far. In further work by Hoppenbrouwers and co-authors, business process modeling processes are analyzed with a focus on collaborative modeling [WHB17]. The work aims at generating insights into psychological

mechanisms of modeling skills and related cognitive processes while modeling. For data generation, modeling processes are recorded on video while modelers are asked to think aloud in these studies.

A further related approach from the perspective of learning outcomes and with a focus on enterprise modeling investigates modeling processes of learners and develops a feedback approach based on analyzing individual modeling processes [e. g., SSD14]. JMermaid is used as experimental environment to analyze modeling behavior utilizing process mining analysis and pattern identification to study conceptual modeling behavior of novices. With a focus on improving teaching practices in the field of conceptual modeling, the approach aims to provide process-oriented feedback [e. g., Se16].

In a more distantly related approach, modeler-tool interactions are tracked and subsequently used to investigate software usability aspects of modeling tools [Th15, p. 153]. More specifically, process execution data is used for so-called usability mining. In addition to interactions of modelers with the modeling tool, interactions with the graphical user interface are recorded [Th15, p. 155]. The resulting approach is evaluated in the context of a modeling scenario with the ARIS Designer of Software AG.

3 TOOL – A webbrowser-based modeling tool and observatory

As prerequisite for understanding the multi-modal observation and data generation approach, this section provides background information, technical design considerations, and fundamental functions of TOOL. As part of the long-term research program, we have been developing a web-based modeling tool integrated with a modeling observatory for studying modeling processes since 2013 to explore design and implementation strategies for studying modeling processes, e. g., tracking interactions of learners with graphical editors (see Fig. 1) [Te17; Te19; TS18]. Two essential requirements drive the development: (1) Platform independence to the greatest extent economically viable as well as (2) usability (especially an intuitive user interface of the modeling tool based on the application scenario). Concerning the multi-modal observation approach, we implemented and combined selected complementary observation approaches ranging from conducting surveys to tracking modeler-tool interactions to recording verbal protocols—based on our fundamental assumption that modeling processes demand study from different, complementary angles. For the prototype, we opted for a web application with a JavaScript-driven browser frontend and a Java EE (Enterprise Edition)-based backend (see [Te19; TS18] for further background on TOOL). The modeling tool (frontend) is intended to support modelers, e. g., with ad-hoc syntax checking while constructing conceptual models.

The modeling observatory (backend) implements tracking algorithms and analytic components, including analysis and administration user interfaces. More specifically, the observatory implements an algorithm which tracks modeler-tool interactions while working on modeling tasks, by storing every modeler-tool interaction with the graphical editor as

time-discrete event. Studying individual modeling processes is supported in the modeling observatory by providing multiple modes of observations and complementary analysis tools (see Sect. 4) as well as configurable parameters for the observation setup that, for instance, allow to select observation modes.

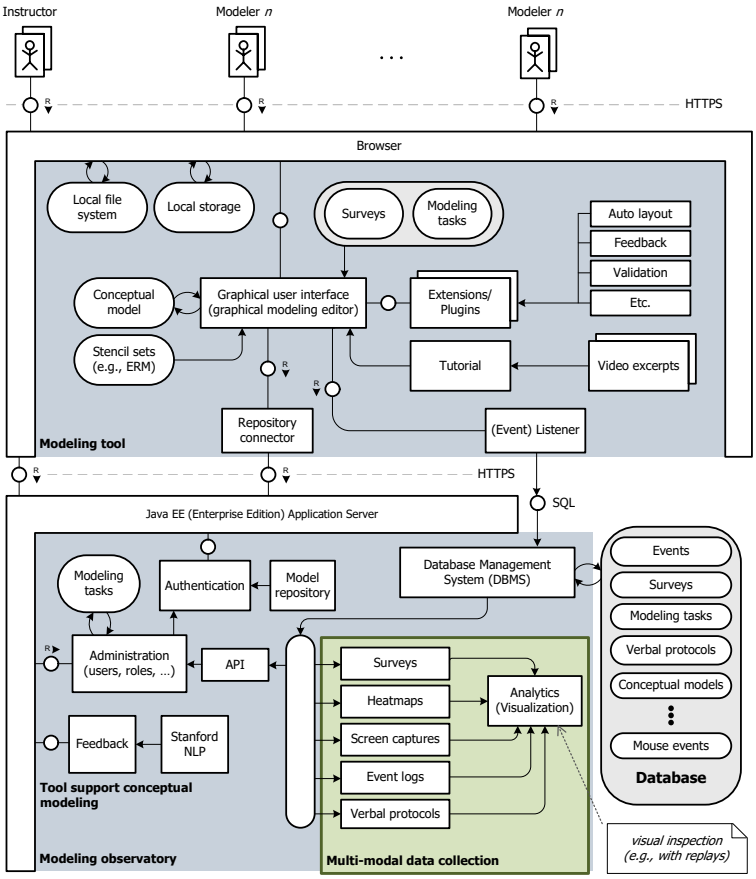


Fig. 1: Software architecture at the conceptual design level depicted using FMC (Fundamental Modeling Concepts [KGT06]). The green area represents the technical components of the multi-modal observation and data generation approach.

4 Multi-modal observation of conceptual modeling processes

In contrast to prior work, the observation approach to study individual modeling processes presented in this work aims to purposefully integrate complementary modes of observation, including the integration of verbalization of modelers during model construction (think aloud protocols, e. g., [ES93]). At present, the observation approach combines four modes

of observation (see Fig. 2) complementing and tying in with prior approaches to studying individual modeling processes [e. g., Pi14; SSD14; WHB17].

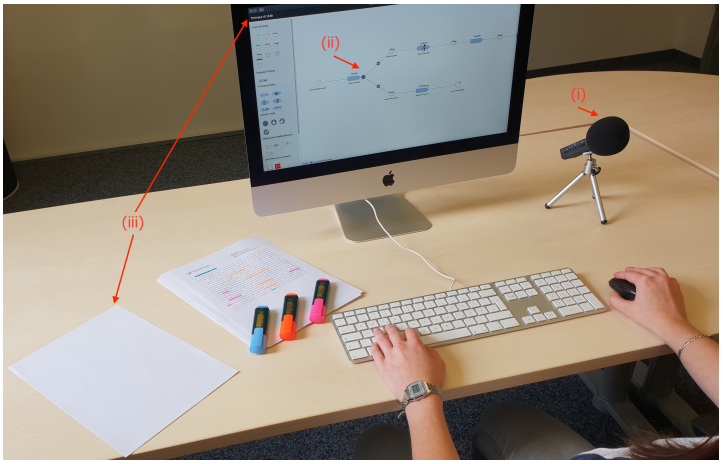
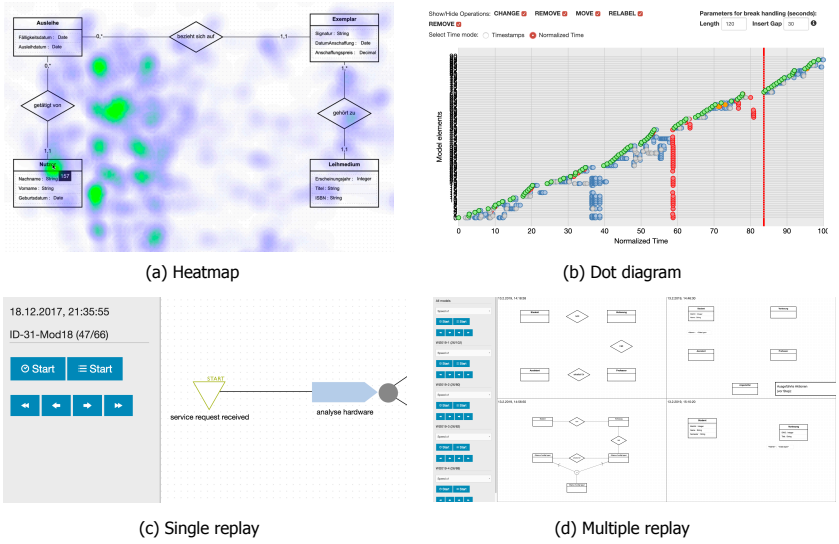


Fig. 2: Overview from the 'over-the-shoulder' perspective used for videotaping modelers (iv).

(i) Recording verbal protocols: This mode of observation targets the reasoning of modelers while modeling via verbalization (think aloud)—an approach that has shown promising results in observing subjects performing problem-solving and, particularly, modeling tasks [e. g., BD92; ES93]. It is aimed at understanding cognitive processes of subjects during conceptual modeling, e. g., by identifying cognitive breakdowns indicating modeling difficulties [RS19]. A component of the modeling observatory allows to collect verbal data by recording voice while subjects are working on a modeling task. Before the beginning of a task, subjects are instructed and informed to verbalize all their thoughts during the modeling. Subjects' comments during the modeling are recorded and can be downloaded as WAV file in the analysis interface of the modeling observatory. If a modeler stops speaking, a notification in the top right corner reminds the modeler to continue speaking—as suggested for gathering think aloud protocols [ES93, pp. 82f, 256].

(ii) Tracking modeler-tool interactions: This mode of observation aims to observe modelers' interactions with the graphical editor of the modeling tool. Every interaction during the construction of a conceptual model is recorded as a time-discrete event. This mode of observation is supported by the modeling observatory and implemented with the modeling tool [see Te19] with which the subjects construct the conceptual model. We developed three complementary analysis interfaces for visualizing and analyzing modeler-tool interactions based on tracked data: (1) real-time and step-by-step replays showing the construction of the conceptual model on the drawing area (s. Fig. 3c–d): The step-by-step replay visually shows every step of model construction whereas the automatic replay shows model construction in real-time, both allowing for visual inspection of modeling behavior; (2) detailed and time normalized dot diagrams (s. Fig. 3b): The dot diagrams provide several configuration parameters that allow the visual appearance of the diagram to be tailored to the

needs of the analysis, e. g., time normalization. The vertical axis indicates the consecutively numbered model elements that are created (green circle), changed (blue circle), deleted (red circle) or relabeled (orange circle). The red vertical line is a normalized time gap that can be adjusted. This visualization is inspired by the PPMCharts visualizing the process of process modeling [Cl15b] and the Dotted Charts suggested in [SA07]; (3) visualizing mouse-movements and mouse clicks performed by modelers in the modeling tool (heatmaps showing dwell time and mouse clicks, s. Fig. 3a). In addition, heatmaps allow insights into user behavior in the modeling tool, e. g., uncontrolled clicking. The analysis interfaces for the tracked modeler-tool interactions allow for further exploring situations identified as deviant or unclear in the audio and video protocols of modeling processes and for identifying anomalous modeler-tool interactions by manual inspection.



(a) Heatmap (b) Dot diagram

(c) Single replay (d) Multiple replay

Fig. 3: Visualization of modeler-tool interactions.

(iii) Surveying subjects pre- and post-modeling: The modeling observatory supports online surveys that the subjects can be asked to complete within the modeling tool before and after modeling. The aim is to gather additional data from subjects performing a modeling task, e. g., on experience and knowledge of conceptual modeling, encountered modeling difficulties and challenges, or the perceived familiarity and difficulties with the domain of the modeling task, or demographic information of the subjects [cf. RS19]. The surveys provide support for closed-ended and open-ended questions. In the modeling observatory, survey results are linked to the constructed conceptual models to support analysis, e. g., for identifying modeling difficulties.

(iv) Videotaping modelers: Subjects are videotaped from an 'over-the-shoulder' perspective (see Fig. 2) to capture the overall interaction with the modeling tool during the work on a modeling task and the subjects' behavior as movements and gestures entailing additional

information on their modeling process, e. g., cues about modeling challenges and difficulties. Recording modelers' behavior outside of the modeling tool supports resolving ambiguous situations in the verbal protocols [Zu13].

Depending on the needs of a study, an observation workflow user interface allows configuring the selection and sequence of observation modes: (i) recording verbal protocols, (ii) tracking modeler-tool interactions, and (iii) surveying subjects pre- and post-modeling. Additionally, modeling tasks, display elements of the user interface, general information and privacy statements can be configured in line with the requirements of the respective study.

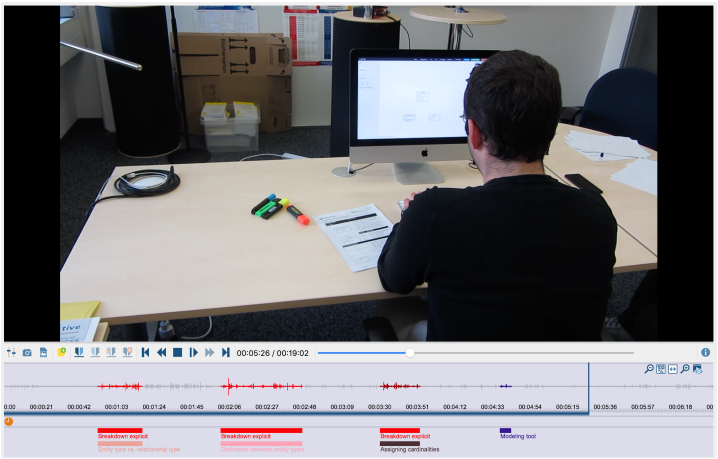


Fig. 4: Coding audio-visual protocols (screenshot from MAXQDA [VE18]).

The multi-modal observation approach integrates with the data analysis strategy. In the following, it is explained how the collected data from the complementary modes of observation is technically combined and how analyses proceed. The verbal think aloud protocol are linked and time-synchronized to the video recordings into audio-visual protocols comprising both observations. To add structure to the data, we code these videos by systematically assigning codes to video segments/clips (following, e. g., [MHS14, pp. 81f]) using a qualitative data analysis software (see Fig. 4 for an example from a pre-test). Directly coding the audio-visual protocols rather than transcribing the verbalizations allows to benefit from the complementary angles provided by the respective mode of observation. Following problem-solving research and the Cognitive Load Theory [e. g., Sw88], we use the notion of cognitive breakdowns [e. g., Be11; NS72] to identify modeling difficulties which modelers experience while constructing a conceptual model and, hence, use cognitive breakdown as deductive code. We define a cognitive breakdown as a cognitive difficulty which a modeler experiences when constructing a conceptual model based on a natural language description [Be11, p. 4]—"when a line of thought fails" [BM08, p. 768]. Such a cognitive breakdown can manifest itself in a modeler explicitly verbalizing a difficulty while modeling or in interrupting or terminating a modeling activity, e. g., a modeling activity which is not completed, but instead the modeler switches to another activity [Be11, p. 4].

Segments in which a subject encounters a difficulty or an obstacle are marked with the code 'cognitive breakdown', i. e., when the subject explicitly verbalizes a difficulty experienced during modeling or when the subject interrupts or terminates a modeling activity. This code is complemented with codes generally anticipated in think aloud protocols as, e. g., talking about non-task-related issues, evaluation of the task at a meta-level, silent periods and actions outside of the modeling tool (following, e. g., [SBS94, p. 122]). During coding, the coding scheme is complemented with emerging codes and sub codes—allowing for refinements according to the actual behavior exhibited in the modeling processes. We supplement coding audio-visual protocols with analyzing timed modeler-tool interactions allowing us to identify peculiar situations. Segments identified as unclear in the videos are submitted to closer inspection by analyzing the recorded modeler-tool interactions in the respective time period to better understand the observed situation, and to decide on assigning a code: Dot diagrams visualizing modeling processes and replays of model construction are used for further exploring situations identified as deviant or unclear in the audio-visual protocols. Vice versa, anomalous data in the recorded modeler-tool interactions is identified and further investigated through analyzing the audio-visual protocols: Modeler-tool interactions in the specific time frame are step-wise visually replayed as performed by the modeler, and, hence, visually analyzed (see Fig. 3c–d). In addition, dot diagrams are used for identifying anomalous modeler-tool interactions by manual inspection. Data integration is taken one step further by reviewing the pre- and post-modeling surveys and, thus, by supplementing another mode of observation (self-disclosure). This coding step proved valuable especially as the perceived difficulties can serve as indication for closer inspecting and deciding on assigning a code in the audio-visual protocols. We provide a more detailed description of the data analysis strategy and coding steps as applied in an exploratory study, see [RS19, pp. 8–10]. Please note that data analysis involving coding of videos, think aloud protocol analysis and visual inspection of different representations of modeling processes is recognized as a time-consuming and labor-intensive approach and, hence, is accompanied by relatively small sample sizes [e. g., Ni94].

5 Exploratory studies

The multi-modal observation and data collection approach has been applied in a first exploratory study in January 2019 into modeling difficulties individuals experience when constructing a conceptual data model [RS19]. Involving eight learners of conceptual modeling working on a data modeling task using TOOL, the multi-modal observation approach has been applied to observe the data modeling processes including recording verbal protocols, videotaping modelers, tracking modeler-tool interactions and surveying subjects before and after modeling. To achieve insights into the individual modeling processes, the collected data has been purposefully integrated and analyzed to identify modeling difficulties and to understand modelers' reasoning: Audio-visual protocols comprising the video recordings of the modelers' behavior and the think aloud protocols were coded for cognitive breakdowns serving as an indication for cognitive difficulties in performing the

modeling task (using the software MAXQDA [VE18] that provides support for coding of integrated audio and video segments). In addition, the analysis was substantially deepened by dot diagrams and replays of the modeling processes. Exploring modeler-tool interactions and the audio-visual protocols revealed that the used modeling task shows a (surprising) complexity posing challenges to the participants though the task was deliberately designed to balance demand on subjects and modeling complexity and does not presuppose any particular domain knowledge (see Fig. 5 for a reference solution).

Data analysis led us to identify five types of modeling difficulties the observed subjects faced while performing the data modeling task: The types of difficulties relate to different aspects of constructing conceptual data models, i. e., entity types, relationship types, attributes, and cardinalities. The majority of difficulties encountered by the participants relates to modeling relationship types, i. e., deciding between modeling an entity type or a relationship type, developing sensible identifiers for relationship types and determining cardinalities [RS19, pp. 11f]. These exploratory results are intended as a starting point for developing a taxonomy of modeling difficulties—in the sense of a taxonomic theory (following [Gr06])—over the course of multiple studies as theoretical foundation for design science research on developing (tool) support for conceptual modeling.

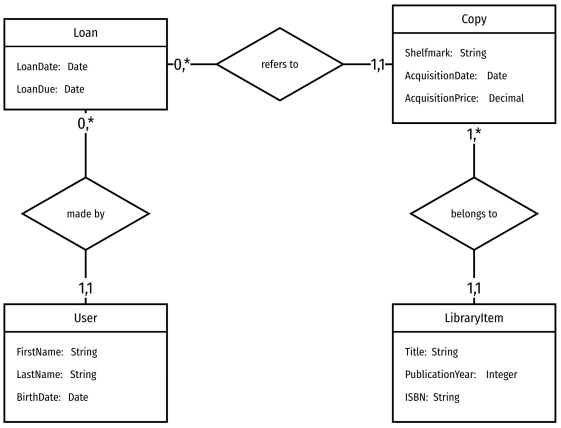


Fig. 5: Reference solution for the data modeling task from the library domain used in the first exploratory study (using a variant of the ER Model).

In a further exploratory study in May to June 2019, conceptual data modeling processes of experienced modelers have been observed following the same multi-modal observation approach as in the first exploratory study. The analysis of the modeling processes is still in progress. In both studies, participants were asked closed- and open-ended questions on problems they encountered when operating the modeling tool and regarding usability of the user interface of the tool after constructing the conceptual data model. Informed by helpful and constructive answers and suggestions of the participants, we implemented several adjustments to the modeling tool: For example, participants pointed to difficulties with

regard to displaying model elements on the drawing area and in changing and deleting model elements. Altogether, participants' answers in both studies show a positive assessment of the operability and usability of the modeling tool for constructing conceptual data models.

6 Discussion and conclusion

Principle limitations relate to the suggested multi-modal observation and data generation approach: Regarding the analysis of verbal protocols it is assumed that thinking aloud does not interfere with thought processes. But as the modeling task includes a visual, non-verbal perceptual component, thinking aloud may slow down the thought processes and/or modeling performance [e. g., ES93]. Another principal limitation is that other, presumably equally essential aspects of the modeling processes, e. g., modeler motivation or the use of additional tools outside of the modeling tool, e. g., online tutorial videos, etc. are neglected in the present data collection design. Hence, we view it as an open question of whether and how to further enrich modes of observation on modeling processes with, for instance, eye tracking [e. g., We16] respectively video recording via webcam to be able to capture modelers' facial expressions [e. g., BM08, p. 768]. At present, the modeling tool supports constructing conceptual models in a variant of the Entity-Relationship (ER) model and for creating MEMO control-flow and decomposition diagrams [Fr11]. However, the graphical modeling editors can be extended by stencil sets, and we are preparing for future studies observing and analyzing, e. g., business process modeling processes with the Business Process Modeling and Notation (BPMN)—in consideration of the specifics of the modeling language used [e. g., HPv05].

Conducting two small-scale studies with experienced and non-experienced modelers demonstrated that the modeling observatory assists in observing conceptual modeling processes in an adequate way, e. g., to identify modeling difficulties (for first results, see [RS19]). In both studies we conducted so far, no principal or technical problems regarding data collection or data analysis occurred. In contrast to using a combination of existing tools, the modeling observatory of TOOL allows to integrate not only the supported data collection approaches but also data analysis and, hence, promises to keep data collection and analysis efforts to a lower level. However, for future studies we plan to technically combine the data collection for the observation modes of (i) recording verbal protocols and (iv) videotaping modelers to directly generate audio-visual protocols comprising both observations—in order to further reduce data collection and analysis efforts.

Analyzing the collected data in the first exploratory study showed that *only* the integration of the complementary observation modes allowed us to identify a wide range of modeling difficulties by identifying cognitive breakdowns: Supplementing the analysis of audio-visual protocols with analyzing tracked modeler-tool interactions allowed us to better understand ambiguous situations and to identify further deviant interactions while reviewing the post-modeling survey proved valuable especially as perceived difficulties served as indication for closer inspection of the audio-visual protocols [RS19, pp. 8–10]. Hence, our preliminary

experience suggests that combining multiple modes of observation to study individual modeling processes contributes to achieving a richer and more complete understanding of modeling processes and modeling difficulties encountered during model construction. Informed by these exploratory insights, we are preparing for further small-scale and large-scale studies (e. g. as part of a university course with several hundred students) aimed at deepening and substantiating our understanding of modeling difficulties and identifying patterns of modeling processes. In addition, we intend to further investigate how modelers operate the modeling tool and—informed by modelers' feedback—aim to further enhance usability of the tool by aspiring a more intuitive user interface and by designing and implementing modeling support tailored to facilitate overcoming recognized modeling difficulties [cf. Te19].

References

[BA94] Batra, D.; Antony, S. R.: Novice errors in conceptual database design. European Journal of Information Systems 3/1, pp. 57–69, 1994.

[BD92] Batra, D.; Davis, J. G.: Conceptual data modelling in database design: similarities and differences between expert and novice designers. International Journal of Man-Machine Studies 37/1, pp. 83–101, 1992.

[Be11] Bera, P.: Situations That Affect Modelers' Cognitive Difficulties: An Empirical Assessment. In: 5th Americas Conference on Information Systems (AMCIS). Detroit, MI, Research paper 254, 2011.

[BL67] Berger, P. L.; Luckmann, T.: The Social Construction of Reality. Anchor Books, New York, NY, 1967.

[BM08] Burton-Jones, A.; Meso, P.: The Effects of Decomposition Quality and Multiple Forms of Information on Novices' Understanding of a Domain from a Conceptual Model. Journal of the Association for Information Systems 9/12, pp. 748–802, 2008.

[Cl15a] Claes, J.; Vanderfeesten, I.; Gailly, F.; Grefen, P.; Poels, G.: The Structured Process Modeling Theory (SPMT) a cognitive view on why and how modelers benefit from structuring the process of process modeling. Information Systems Frontiers 17/6, pp. 1401–1425, 2015.

[Cl15b] Claes, J.; Vanderfeesten, I.; Pinggera, J.; Reijers, H. A.; Weber, B.; Poels, G.: A visual analysis of the process of process modeling. Information Systems and e-Business Management 13/1, pp. 147–190, 2015.

[CP18] Creswell, J. W.; Plano Clark, V. L.: Designing and Conducting Mixed Methods Research. Sage, Los Angeles, CA, 2018.

[ES93] Ericsson, K. A.; Simon, H. A.: Protocol analysis: Verbal reports as data. MIT Press, Cambridge, MA, 1993.

[Fr11] Frank, U.: MEMO Organisation Modelling Language (2): Focus on Business Processes, ICB-Research Report No. 49, Institute for Computer Science and Business Information Systems (ICB), University Duisburg-Essen, 2011.

[Gr06] Gregor, S.: The Nature of Theory in Information Systems. MIS Quarterly 30/3, pp. 611–642, 2006.

[HLP06] Hoppenbrouwers, S. J. B. A.; Lindeman, L.; Proper, H. A.: Capturing Modeling Processes – Towards the MoDial Modeling Laboratory. In (Meersman, R.; Tari, Z.; Herrero, P., eds.): On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops. Lecture Notes in Computer Science. Vol. 4278, Springer, Berlin, Heidelberg, pp. 1242–1252, 2006.

[HPv05] Hoppenbrouwers, S. J. B. A.; Proper, H. A.; van der Weide, T. P.: A Fundamental View on the Process of Conceptual Modeling. In (Delcambre, L.; Kop, C.; Mayr, H. C.; Mylopoulos, J.; Pastor, O., eds.): 24th International Conference on Conceptual Modeling (ER). Lecture Notes in Computer Science. Vol. 3716, Springer, Berlin, Heidelberg, pp. 128–143, 2005.

[KGT06] Knöpfel, A.; Gröne, B.; Tabeling, P.: Fundamental Modeling Concepts: Effective Communication of IT Systems. J. Wiley & Sons, Hoboken, NJ, 2006.

[MHS14] Miles, M. B.; Huberman, A. M.; Saldaña, J.: Qualitative Data Analysis: A Methods Sourcebook. Sage, Thousand Oaks, CA, 2014.

[Ni94] Nielsen, J.: Estimating the number of subjects needed for a thinking aloud test. International Journal of Human-Computer Studies 41/3, pp. 385–397, 1994.

[NS72] Newell, A.; Simon, H. A.: Human problem solving. Prentice-Hall, Englewood Cliffs, NJ, 1972.

[Pi14] Pinggera, J.; Zugal, S.; Furtner, M.; Sachse, P.; Martini, M.; Weber, B.: The Modeling Mind: Behavior Patterns in Process Modeling. In (Bider, I.; Gaaloul, K.; Krogstie, J.; Nurcan, S.; Proper, H. A.; Schmidt, R.; Soffer, P., eds.): Enterprise, Business-Process and Information Systems Modeling. BPMDS 2014, EMMSAD 2014. Lecture Notes in Business Information Processing. Vol. 175, Springer, Berlin, Heidelberg, pp. 1–16, 2014.

[Pi15] Pinggera, J.; Soffer, P.; Fahland, D.; Weidlich, M.; Zugal, S.; Weber, B.; Reijers, H. A.; Mendling, J.: Styles in business process modeling: an exploration and a model. Software and Systems Modeling 14/3, pp. 1055–1080, 2015.

[PZW10] Pinggera, J.; Zugal, S.; Weber, B.: Investigating the Process of Process Modeling with Cheetah Experimental Platform. In (Mutschler, B.; Recker, J.; Wieringa, R.; Ralyté, J.; Plebani, P., eds.): CAiSE 2010 Workshop ER-POIS. CEUR-WS. Vol. 603, Hammamet, Tunisia, pp. 13–18, 2010.

[RS19] Rosenthal, K.; Strecker, S.: Toward a Taxonomy of Modeling Difficulties : A Multi-Modal Study on Individual Modeling Processes. In: 40th International Conference on Information Systems (ICIS). Munich, Germany, 2019.

[RTS19] Rosenthal, K.; Ternes, B.; Strecker, S.: Learning Conceptual Modeling: Structuring Overview, Research Themes and Paths for Future Research. In: 29th European Conference on Information Systems (ECIS). Stockholm, Sweden, Research Paper 137, 2019.

[SA07] Song, M.; van der Aalst, W. M. P.: Supporting process mining by showing events at a glance. In: 17th Annual Workshop on Information Technologies and Systems (WITS). Montreal, Canada, pp. 139–145, 2007.

[SBS94] van Someren, M. W.; Barnard, Y. F.; Sandberg, J. A. C.: The Think Aloud Method: A Practical Guide to Modelling Cognitive Processes. Academic Press, London, 1994.

[Se16] Serral, E.; De Weerdt, J.; Sedrakyan, G.; Snoeck, M.: Automating Immediate and Personalized Feedback: Taking Conceptual Modelling Education to a Next Level. In: 10th International Conference on Research Challenges in Information Science (RCIS). IEEE, Grenoble, France, pp. 1–6, 2016.

[SS17] Sedrakyan, G.; Snoeck, M.: Cognitive Feedback and Behavioral Feedforward Automation Perspectives for Modeling and Validation in a Learning Context. In (Hammoudi, S.; Pires, L.; Selic, B.; Desfray, P., eds.): Model-Driven Engineering and Software Development. 4th International Conference, MODELSWARD 2016, Rome, Italy. Vol. 692, Springer, Cham, pp. 70–92, 2017.

[SSD14] Sedrakyan, G.; Snoeck, M.; De Weerdt, J.: Process mining analysis of conceptual modeling behavior of novices – Empirical study using JMermaid modeling and experimental logging environment. Computers in Human Behavior 41/, pp. 486–503, 2014.

[ST95] Srinivasan, A.; Te'eni, D.: Modeling as Constrained Problem Solving: An Empirical Study of the Data Modeling Process. Management Science 41/3, pp. 419–434, 1995.

[Sw88] Sweller, J.: Cognitive Load During Problem Solving: Effects on Learning. Cognitive Science 12/2, pp. 257–285, 1988.

[Te17] Ternes, B.: Design and evaluation of a web-based modeling platform to support the learning of conceptual modeling and of studying the corresponding learning processes. In (Gulden, J.; Nurcan, S.; Reinhartz-Berger, I.; W, G.; Bera, P.; Guerreiro, S.; Fellmann, M.; Weidlich, M., eds.): 8th International Workshop on Enterprise Modeling and Information Systems Architectures (EMISA). CEUR-WS. Vol. 1859, Essen, Germany, pp. 138–142, 2017.

[Te19] Ternes, B.; Strecker, S.; Rosenthal, K.; Barth, H.: A browser-based modeling tool for studying the learning of conceptual modeling based on a multi-modal data collection approach. In: 14. Internationale Tagung Wirtschaftsinformatik (WI). Siegen, Germany, pp. 1984–1988, 2019.

[Th15] Thaler, T.; Maurer, D.; Angelis, V. D.; Fettke, P.; Loos, P.: Mining the Usability of Business Process Modeling Tools: Concept and Case Study. In (Mendling, J.; vom Brocke, J., eds.): Industry Track at the 13th International Conference on Business Process Management 2015. CEUR-WS. Vol. 1439, Innsbruck, Austria, pp. 152–166, 2015.

[TS18] Ternes, B.; Strecker, S.: A web-based modeling tool for studying the learning of conceptual modeling. In (Schaefer, I.; Karagiannis, D.; Vogelsang, A.; Méndez, D.; Seidl, C., eds.): Modellierung 2018. Lecture Notes in Informatics. Vol. 280, Gesellschaft für Informatik e.V., Braunschweig, Germany, pp. 325–328, 2018.

[VBB13] Venkatesh, V.; Brown, S. A.; Bala, H.: Bridging the Qualitative-Quantitative Divide: Guidelines for Conducting Mixed Methods Research in Information Systems. MIS Quarterly 37/1, pp. 21–54, 2013.

[VBS16] Venkatesh, V.; Brown, S. A.; Sullivan, Y. W.: Guidelines for Conducting Mixed-Methods Research: An Extension and Illustration. Information Systems 17/7, pp. 435–494, 2016.

[VE18] VERBI Software: MAXQDA Standard 12, `https://www.maxqda.com`, Accessed: 2019-10-09, 2018.

[Ve96] Venable, J. R.: Teaching Novice Conceptual Data Modellers to Become Experts. In: International Conference Software Engineering: Education and Practice. IEEE, Dunedin, New Zealand, pp. 50–56, 1996.

[We16] Weber, B.; Pinggera, J.; Neurauter, M.; Zugal, S.; Martini, M.; Furtner, M.; Sachse, P.; Schnitzer, D.: Fixation Patterns During Process Model Creation: Initial Steps Toward Neuro-adaptive Process Modeling Environments. In: 49th Hawaii International Conference on System Sciences (HICSS). IEEE, Piscataway, NJ, pp. 600–609, 2016.

[WHB17] Wilmont, I.; Hoppenbrouwers, S.; Barendsen, E.: An Observation Method for Behavioral Analysis of Collaborative Modeling Skills. In (Metzger, A.; Persson, A., eds.): Advanced Information Systems Engineering Workshops. CAiSE 2017. Lecture Notes in Business Information Processing. Vol. 286, Springer, Cham, pp. 59–71, 2017.

[WO80] Wedekind, H.; Ortner, E.: Systematisches Konstruieren von Datenbankanwendungen: Zur Methodologie der angewandten Informatik. Carl Hanser Verlag, Munich, 1980.

[Zu13] Zugal, S.; Haisjackl, C.; Pinggera, J.; Weber, B.: Empirical Evaluation of Test Driven Modeling. International Journal of Information System Modeling and Design 4/2, pp. 23–43, 2013.

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 93

Konstruktion eines Referenzmodells für den Wissenstransfer in und aus Hochschulverbünden

Claudia Doering[1], Christian Seel (†)[1]

Abstract: Neben der Lehre und dem Forschungsauftrag der Hochschulen gewinnt die sog. Third Mission in Form eines Wissenstransfers zwischen Hochschulen, Unternehmen und der Gesellschaft zunehmend an Bedeutung. Des Weiteren ist ein Trend von Wissenstransferaktivitäten in Hochschulverbünden zu konstatieren. Solche Verbunde aus mehreren Hochschulen, wie UAS7, bieten einzelnen Hochschulen neben Synergieeffekten auch die Möglichkeit größere Themenfelder gemeinsam zu besetzen und Drittmittel einzuwerben. Dem entgegen steht jedoch die Notwendigkeit auf der Verbundebene weitere Koordinations- und Harmonisierungsaktivitäten durchführen zu müssen. Um sowohl die notwendigen Koordinationsaktivitäten auf Ebene einzelner Hochschulen als auch des Verbundes zu identifizieren und den Aufbau von Hochschulverbünden zu erleichtern, wird in diesem Beitrag ein Referenzmodell für den Wissenstransfer in und aus Hochschulverbünden vorgestellt.

Keywords: Modellierung, BPMN, Third Mission, Referenzmodellierung, Ordnungsrahmen, Wissenstransfer, Hochschule, Universität, Transfer

1 Motivation

Eine Third Mission in Form eines Wissenstransfers zwischen Hochschulen, Unternehmen und der Gesellschaft gewinnt neben der Lehre und dem Forschungsauftrag der Hochschulen zunehmend an Bedeutung [RDH15]. Die theoretischen Rahmenbedingungen für diesen Transfer wurden bereits in 80er Jahren beschrieben und finden sich in den Konzepten der „Entrepreneurial Universities“ [Cl98], „Triple Helix“ [EL00] und „Mode 2“ [Gi94] wieder. Diesen Konzepten gemein ist die Auffassung, dass Universitäten nicht mehr als „Elfenbeintürme“, in welchen Forschung vom Rest der Gesellschaft abgeschnitten ist, wahrgenommen werden, sondern sich vielmehr als Institutionen mit einem ausgeprägten Wissenstransfer etablieren. Einhergehend mit diesem Wandel ist eine verstärkte Drittmittelorientierung der Forschungseinrichtungen und Hochschulen, welche sich nun immer mehr auf notwenige Industriekooperationen fokussieren [Ra13]. Um verstärkt Drittmittel einwerben zu können und eine größere Sichtbarkeit zu generieren, gehen Hochschulen miteinander Allianzen ein und gründen Verbünde. Der Nutzen eines Referenzmodells für den Wissenstransfer in und aus Hochschulverbünden bezieht sich nicht nur auf diese Punkte, sondern dient ebenfalls einer besseren Dokumentation und Wiederverwendbarkeit von Prozessen, was zu einer Vermeidung von Fehlern und somit auch geringeren Kosten

[1] Hochschule für angewandte Wissenschaften Landshut, Institut für Projektmanagement und Informationsmodellierung, Am Lurzenhof 1, 84036 Landshut, claudia.doering@haw-landshut.de

und einer verbesserten Prozessqualität beitragen kann.

Um die Arbeit in Hochschulkooperationen zu erleichtern, sind strukturelle und organisatorische Veränderungen an den einzelnen Universitäten erforderlich. Obwohl Transfer schon seit geraumer Zeit an Hochschulen und Universitäten verankert ist, werden diese Institutionen immer noch über Lehre und Forschung definiert und haben oftmals interne organisatorische Barrieren ausgeprägt, welche Wissenstransfer behindern. Hier ist ein Mangel an administrativer Unterstützung auch bei grundlegenden Aspekten des Wissenstransfers, z. B. in der Vertragsgestaltung, Unterstützung bei rechtlichen Fragestellungen oder der Bereitstellung von Ressourcen zu nennen [JBG04]. Bisher konzentrierte sich die Forschung vor allem auf den Wissenstransfer von oder zu einzelnen Universitäten und nicht auf Wissenstransfer innerhalb und mit Hochschulverbünden. Diese unterscheiden sich jedoch deutlich von einzelnen Universitäten und benötigen eine stärkere Unterstützung durch Koordination und Harmonisierung.

Es lassen sich in der aktuellen Gestaltung des Wissenstransfergeschehens in Hochschulverbünden sechs Defizite feststellen: Zunächst ergeben sich Forschungslücken im Bereich des organisationsübergreifenden, prozessbasierten Wissenstransfers, welcher beispielsweise in Form von Kooperationen von KMU mit Hochschulverbünden, noch weitestgehend unerforscht bzw. bisher nur exemplarisch in Einzelfallbetrachtungen beschrieben ist [BF00], [GM12]. (1) Bislang gibt es *keine vernetzte, neutrale und allgemeingültige Darstellung* der hochschulübergreifenden Prozesse des Wissenstransfers, (2) vielmehr werden nur Einzelfälle mit *niedrigem Formalisierungsgrad* publiziert [BF00], [GM12]. Gerade um eine Nachvollziehbarkeit und Adaption zu ermöglichen, ist eine konsistente und aufeinander bezogene Darstellung von Vorteil. (3) Im Transfergeschehen in Hochschulverbünden werden bislang viele Prozesse über *Gesetztestexte* definiert, welche durch die Komplexität des Rechts oftmals nur rechtswissenschaftlich-objektiv und nicht adressatenorientiert-subjektiv zu verstehen sind [To09]. Dies hat mitunter dazu geführt, dass Prozesse an den einzelnen Hochschulen in einem Hochschulverbund zwar modelliert werden, (4) hierbei allerdings *keine Modellierungssprache* eingesetzt wird und auch keine Bestrebungen einer Prozessstandardisierung zu vermerken sind. (5) Oftmals beruht die Prozessmodellierung auf *verbalen Vereinbarungen* an den einzelnen Universitäten und nicht auf einem hochschulübergreifenden Standard. (6) Hierbei lassen sich ebenfalls *unterschiedliche Reifegrade* in der Implementierung von Third Mission Arbeitsabläufen vermerken. In diesem Zusammenhang lassen sich vor allem Unterschiede zwischen Universitäten und Hochschulen feststellen. Gerade an Hochschulen, welche sich noch stärker über Lehre und Forschung definieren, sind Defizite in der Modellierung der Transferprozesse identifizierbar [SD14].

Insgesamt lassen sich nun also sechs Defizite im Themengebiet der Modellierung des Transfergeschehens in Hochschulverbünden vermerken. Die Referenzmodellierung bietet sich sehr gut an um ein intersubjektiv nachvollziehbareres und eindeutiges Verständnis der Thematik zu schaffen. Um die Sachverhalte des Wissenstransfers zu visualisieren, sind konsistente Prozesse, abgestimmte Organisationsformen und harmonisierte Dokumente erforderlich, die in dieser Veröffentlichung in einem Referenzmodell definiert werden.

Dementsprechend liegen diesem Beitrag folgende Forschungsfragen zu Grunde:

RQ1: Welche Prozesse sind für den Wissenstransfer in Hochschulverbünden relevant?

RQ2: Welche dieser Prozesse werden an einzelnen Hochschulen und welche im Verbund durchgeführt?

RQ3: In welchen Ebenen lassen sich die Prozesse des Transfergeschehens gliedern?

Der vorliegende Beitrag ist in sechs Abschnitte gegliedert. Zunächst wird ein Einblick in den aktuellen Stand der Forschung gegeben und anschließend die genutzte Forschungsmethodik dargestellt. Im folgenden Abschnitt erfolgt die Konstruktion eines adaptiven Referenzmodells für hochschulübergreifenden Wissenstransfer in einem top-down-Ansatz ausgehend von einem Ordnungsrahmen über die darunterliegenden Wertschöpfungskettendiagramme bis hin zu BPMN-Prozessmodellen. Daran schließt sich die Evaluation des adaptiven Referenzmodells und ein Fazit mit Ausblick an.

2 Stand der Forschung

Wissenstransfer steht im traditionellen Verständnis für die Schnittstelle zwischen Wissenschaft und Wirtschaft [Fr14]. Heute steht Wissenstransfer allerdings auch für jegliche Kommunikation zwischen einem Experten und einem Laien [Pi14]. THIEL geht indes noch weiter und definiert Wissenstransfer als eine "zielgerichtete Übertragung von Wissen von einem Transferpartner (Sender) zu einem anderen Transferpartner (Empfänger), wobei die Transferpartner Individuen oder Kollektive sein können und die Rollen Sender und Empfänger in einer Transfersituation wechseln können." [Th02]. Ein bereits existierendes Vorgehensmodell für Wissenstransfermaßnahmen in der Wirtschaftsinformatik stellt den Wissenstransfer zwischen Hochschulen und Unternehmen als mehrstufiges Phasenmodell dar [SD14]. Aus heutiger Sicht ergibt sich aus der Analyse dieses Vorgehensmodells die Schlussfolgerung, dass das Vorgehen für den Wissenstransfer zwischen Hochschulen und Unternehmen zwar exemplarisch beschrieben ist, aber noch kein weitverbreitetes, anerkanntes und erprobtes Vorgehensmodell zum Wissenstransfer existiert. Die Thematiken rund um die Prozesse des Wissenstransfers innerhalb von Organisationen sind bereits in vielen Werken bspw. von [No16] oder [Th02] umfangreich behandelt worden, wohingegen interorganisationaler, prozessbasierter Wissenstransfer bspw. bei Kooperationen von KMU und Forschungseinrichtungen bislang nur exemplarisch in Einzelfallbetrachtungen, bspw. in [BF00] oder [GM12], untersucht worden ist. Da Wissensmanagement ebenso vielfältig aufgefasst werden kann, wie das Verständnis von Wissen selbst, verlangt gerade die Unternehmenspraxis nach übersichtlichen Vorgehen und einfachen Darstellungen bei Wissenstransferprozessen [KMS16], [BK02]. Zusätzlich zu der Gestaltung der konkreten Durchführung von Wissenstransfer, wie bspw. in Form von Mitarbeiterschulungen, stellt sich in Hochschulverbünden ebenfalls noch die Frage, wie Wissenstransfer mit den Forschung und Lehre der Hochschulen verbunden werden kann [SD14]. Gerade um eine Nachvollziehbarkeit und Adaption zu ermöglichen, ist eine konsistente und aufeinander

bezogene Darstellung von Vorteil. Referenzmodelle haben sich zur Beschreibung betrieblicher Informationssysteme etabliert und dienen dazu, die Komplexität von Informationssystemen beherrschbar zu machen [Th06]. Wann ein Modell als Referenz gilt wird in der Praxis und in der Forschung häufig unterschiedlich betrachtet [Th06]. Die Evaluation in dieser Publikation stützt sich auf das Referenzmodellverständnis nach THOMAS, welches ein nutzerorientiertes Verständnis von Referenzmodellen postuliert [Th06]. Demnach obliegt es dem Nutzer ein Modell als Referenz anzuerkennen. Nichts desto trotz ist idealerweise mindestens ein, dem Konstrukteur bekannter, Anwendungsfall von Nöten, um das Modell als Referenzmodell deklarieren zu können [Th06].

Die Abstinenz ausreichend empirischer Literatur begründet somit mit dem Bedarf nach strukturierten Modellen die Themenfindung dieses Beitrages.

3 Forschungsmethodik

Die ausgewählten Forschungsfragen und die daraus abgeleiteten Ziele begründen unmittelbar die Auswahl der Forschungsmethodik. Sie ist somit nicht von vornherein definiert, sondern muss individuell für jedes Forschungsvorhaben selektiert werden [Se10]. Die Wirtschaftsinformatik unterscheidet zwei Gruppen von Forschungsfragen. Die erste Gruppe beschreibt Forschungsfragen, welche ein deskriptives oder explikatives Erkenntnissinteresse besitzen und sich mit dem Einsatz und der Entwicklung von Informationssystemen sowie der eingesetzten Methoden beschäftigen [Se10]. Die zweite Gruppe fokussiert, im Sinne einer „Design Science“ auf die Entwicklung neuer Methoden und Artefakte [Se10], [HC10]. Um den derzeitigen Bestand an Ordnungsrahmen im hochschulübergreifenden Wissenstransfer zu ermitteln, wurde die Literaturrecherche und –analyse methodisch nach VOM BROCKE durchgeführt [Vo09]. Die identifizierte Forschungslücke impliziert die Entwicklung eines Ordnungsrahmens für hochschulübergreifende Prozesse des Wissenstransfers. Da dies eine Schaffung von Artefakten darstellt, wird bei der Konstruktion den sieben Richtlinien des „Design Science Research (DSR)“- Ansatzes nach HEVNER et. al gefolgt. Diese sind im Folgenden dargelegt:

1. *Artefakt*e: Als Ergebnis des „Design Science“-Prozesses liegt ein Ordnungsrahmen für die Referenzprozessmodellierung vor. Die Modellierung anwendbarer neuer Artefakte soll durch das iterative Vorgehen der Referenzmodellierungsforschung von FETTKE et. al gewährleistet werden [FL04].
2. *Problemrelevanz:* Die identifizierte Forschungslücke und die aktuelle Problemstellung zeigen die Relevanz des Problems auf.
3. *Evaluation:* Die Evaluierung der Forschungsergebnisse erfolgt im Rahmen der Forschungstätigkeiten. Die Forschung wird in diesem Beitrag als Suchprozess verstanden [HC10]. In einer fortlaufenden Evaluation in einem Verbundprojekt der

ostbayerischen Hochschulen und Universitäten zum Thema Wissens- und Technologietransfer, wird somit gemäß des DSR-Ansatzes, die empirische Aussagekraft der gefundenen Ergebnisse dargelegt.

4. *Beitrag zur Forschung:* Aufgrund der identifizierten Forschungslücke stellt der entwickelte Ordnungsrahmen einen Forschungsbeitrag dar.

5. *Stringenz der Forschungsmethode:* Die Entwicklung des Ordnungsrahmens nach dem Vorgehensmodell von MEISE stellt die Stringenz der Forschungsmethode sicher [Me01].

6. *Forschung als Suchprozess:* Der iterative Suchprozess wird durch die Gegenüberstellung der deduktiven und induktiven Forschungsergebnisse gewährleistet.

7. *Kommunikation von Forschungsergebnissen:* Der vorliegende Beitrag dient zur Veröffentlichung und somit der zielgruppengerechten Kommunikation der Forschungsergebnisse.

4 Konstruktion eines adaptiven Referenzmodells für hochschulübergreifenden Wissenstransfer

Aufgrund der hohen Komplexität der Prozesse des hochschulübergreifenden Wissenstransfers wurde nach dem Vorgehensmodell von MEISE der in Abbildung 1 dargestellte Ordnungsrahmen erstellt [Me01]. Das Struktur-Design orientiert sich am Referenzdesign „Haus“ [Me01]. Durch den Bekanntheitsgrad dieser Darstellung wird die Interpretation durch die Zielgruppen erleichtert. Das vorliegende Referenzmodell ist primär für interne Zielgruppen im Hochschulverbund vorgesehen. Dabei stehen hauptsächlich die einzelnen Transferstellen, Forschungsreferate, administrativen Abteilungen und die Forschenden der Verbundhochschulen im Fokus. Noch dazu wird durch die Anordnung der Management-, Kern- und Supportprozesse in dem Dach, Körper und Fundament des Hauses ein einprägsames Bild geschaffen, denn „Das Design eines Ordnungsrahmens trägt entscheidend zum Verstehen der durch ihn beschriebenen Organisationsstruktur bei.“ [Me01]. Der Ordnungsrahmen soll den Überblick für Zusammenhänge im Transfergeschehen fördern und den Mitarbeitenden den Einstieg in die spezifischen Prozesse erleichtern.

4.1 Ordnungsrahmen für die Prozesse des Wissenstransfers

Der Ordnungsrahmen setzt sich aus zwei Strukturdimensionen, dem Beschreibungsinhalt und der Beschreibungssicht (Prozesse, Organisation, Dokumente), zusammen (vgl. Abb. 1). Der Beschreibungsinhalt des Ordnungsrahmens legt die einzelnen Prozesse für hochschulübergreifenden Wissenstransfer und eine verbundübergreifende Kooperationsstruktur dar (**RQ1**). Hierbei lassen sich drei verschiedene Arten von rekerenden Prozessen unterscheiden (**RQ2**):

- Prozesse, welche nicht im Verbund, sondern an den einzelnen Hochschulen durchgeführt werden (I)
- Prozesse, welche zunächst an den einzelnen Hochschulen und dann im Verbund durchgeführt werden (II)
- Prozesse, welche ausschließlich im Verbund durchgeführt werden (III)

In einem Hochschulverbund wird jede einzelne Verbundhochschule einen adaptiven Transferprozess von der Forschung in die Lehre durchführen (I). Gewonnene Erkenntnisse aus bspw. Transferprojekten fließen, durch die an Projekten beteiligten Forschenden, wiederum in die Lehre ein, was in dem vorgelegten Ordnungsrahmen durch den Kernprozess „Rekursive Transferprozesse“ gekennzeichnet ist. Die Anbahnung von Kooperationsprozessen ist ein Beispiel für einen Kernprozess, welcher zunächst an einer Hochschule durchgeführt und danach im Verbund bearbeitet wird. Dies liegt an dem Vorgehen, bei welchem zunächst Mitarbeitende der Universitäten und Hochschulen auf mögliche Projektpartner zugehen um Projekte zu akquirieren und anschließend verbundintern nach möglichen Partnern gesucht wird (II). Ein Beispiel für Prozesse, welche ausschließlich im Verbund durchgeführt werden (III), ist die Nutzung eines gemeinsamen Methodenrepertoires. Dieses besteht aus dem, im Verbund erstellten, Methodenkatalog aller zum Transfer zur Verfügung stehenden Kompetenzen (bspw. Problemlösungs- oder Modellierungstechniken).

Das zu transferierende Wissen in dem Referenzmodell hat zunächst nur einen expliziten Charakter. Es wird in Folge dessen nur zu dokumentierendes Wissen im Referenzmodell dargestellt. Implizites Wissen, also Wissen welches dem Träger nicht bewusst ist und nicht oder nur schwierig in sprachlicher Form weitergegeben werden kann, soll zukünftig durch einen Prozess der Externalisierung in die Prozessmodellierung einfließen [Po16].

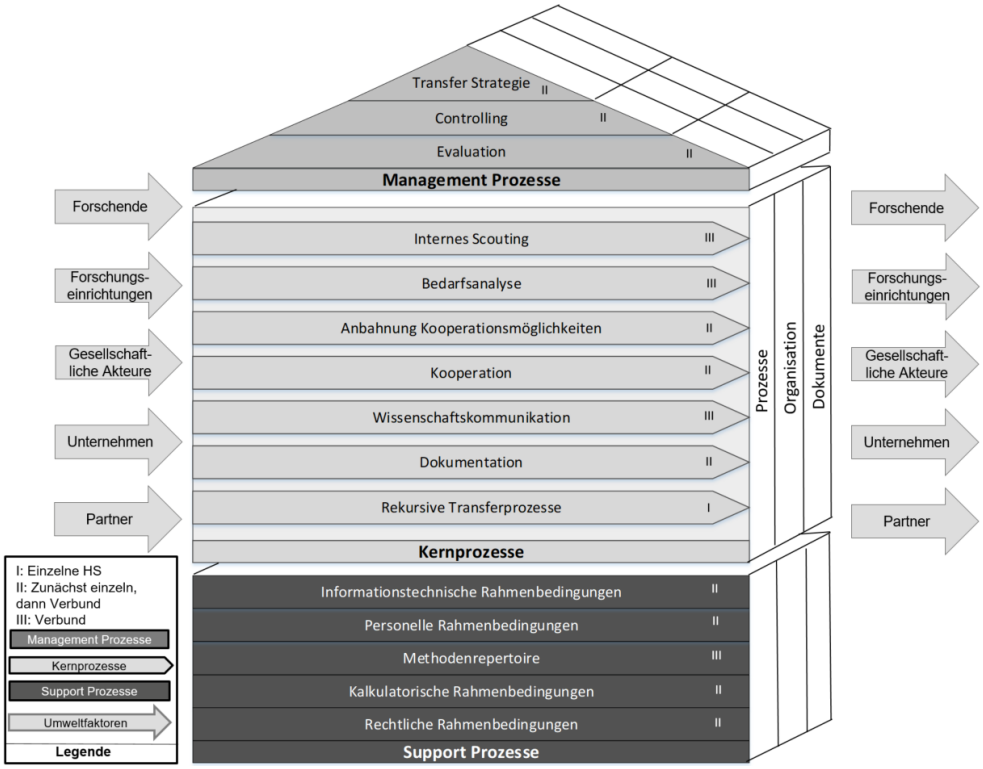


Abb.1: Ordnungsrahmen für Prozesse des Transfers

Die Umwelt des Ordnungsrahmens wird durch die für den Wissenstransfer relevanten Stakeholder beschrieben. Dies sind Forschende im Hochschulverbund, Forschungseinrichtungen, gesellschaftliche Akteure, Unternehmen und Partner. Die Stakeholder können über diesen Ordnungsrahmen mit dem Hochschulverbund zusammenarbeiten und die vorgeschlagenen Strukturen nutzen, um bspw. Forschungs- oder Industrieprojekte durchzuführen.

Empfehlungen für die Organisationsgestaltung können, im Gegensatz zu anderen referenzhaften Modellen, gegeben werden, da die Organisationsgestaltung der Verbundhochschulen in ihren Grundzügen vorgegeben ist. Bspw. nennt das bayrische Hochschulgesetz in Art. 19 alle zentralen Organe und Organisationseinheiten und schafft somit eine gemeinsame Ausgangsbasis [Ba06].

Die Dokumentensicht stellt den Dokumentenfluss im Hochschulverbund dar. Es können somit Empfehlungen gegeben werden, welche Dokumente in welchen Prozessen erstellt oder genutzt werden sollen. Gerade bei der Forschung in Hochschulverbünden ist die Modellierung der Dokumentensicht von großer Bedeutung, da alle Prozesse einer genauen

Abstimmung und einheitlichen Vorgaben bedürfen.

Zusätzlich besteht der Ordnungsrahmen aus mehreren Detaillierungsebenen, welche die wesentlichen Kernprozesse und Funktionen der Transferprozesse beschreiben (**RQ3**). Der Ordnungsrahmen (Ebene 0) dient als Strukturierungsmittel für alle darunterliegenden Prozesse. Die Kernprozesse (Ebene 1) beschreiben mithilfe von Wertschöpfungskettendiagrammen die Kontroll- und Datenflüsse (Ebene 2) der benötigten Funktionen, welche detailliert in BPMN 2.0 modelliert und zum vereinfachten Nutzerverständnis in Prozessablaufbeschreibungen dargelegt werden.

Auf allen Ebenen und Sichten des Referenzmodells wurden harmonisierte Bezeichnungen der Prozesse verwendet. Die Begriffe wurden durch einen verbundinternen Workshop definiert und sollen in zukünftigen Forschungstätigkeiten in weiteren Hochschulverbünden evaluiert werden.

4.2 Wertschöpfungsketten

Die Struktur des Ordnungsrahmens stellt den Prozess der Wertschöpfung in den Mittelpunkt. Die Darstellung in Wertschöpfungsketten bietet sich bei der Modellierung der Kernprozesse des Wissenstransfers an, da in diesen Prozessen, analog zu produzierenden Unternehmen, der Wert, also das Wissen, generiert wird. Die Kernprozesse stehen somit im Mittelpunkt der Struktur des Ordnungsrahmens und ermöglichen so die Ausrichtung aller Prozesse hin zur Wertschöpfung. In der aktuellen Version des Ordnungsrahmens sind sieben Kernprozesse verzeichnet. Diese sind: „Internes Scouting“, „Bedarfsanalyse“, „Anbahnung Kooperationsmöglichkeiten“, „Kooperation“, „Wissenschaftskommunikation“, „Dokumentation“ und „Rekursive Transferprozesse“. Jeder dieser Kernprozesse besteht aus weiteren einzelnen Prozessen. Beispielsweise setzt sich der Prozess „Internes Scouting“ wiederum aus sieben einzelnen Prozessen zusammen (siehe Abb.2).

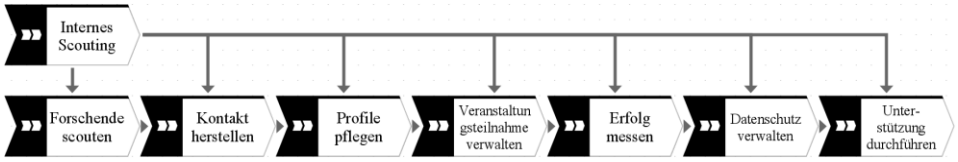


Abb. 2: Wertschöpfungskette des Prozesses „Internes Scouting“

Der Kernprozess „Internes Scouting“ definiert alle internen Tätigkeiten, um verbundübergreifend die Profile von Forschenden mit transferrelevanten Ergebnissen laufend zu erfassen und zu aktualisieren. Hierbei werden auch die Prozesse der Teilnahme an transferrelevanten Veranstaltungen, Unterstützungsformate bei der Durchführung von Transferprojekten oder die Key Performance Indicators (KPIs) von Transferprojekten definiert.

4.3 Prozessebene in BPMN 2.0

Um einen tieferen Einblick in die Prozesse des Transfers zu ermöglichen, stellt die Diagrammebene alle Prozesse mit den dazugehörigen Prozessbausteinen und Abläufen dar. Bei der Wahl der Modellierungssprache der Diagrammebene, soll der Zielstand der Modellierung festgelegt werden [Ga17]. Der Anspruch bei der Wahl einer Modellierungssprache besteht darin nicht nur fehlerfrei zu modellieren, sondern auch eine zielgruppenorientierte Darstellung zu wählen [Ga17]. Die Zielgruppe bzw. die späteren Nutzer der Prozesse in der Prozessebene benötigen detaillierte Informationen über die Prozesse, bspw. Arbeitsanweisungen oder Dokumente. Da die Prozessmodellierung ebenfalls die Möglichkeit einer späteren Automatisierung bieten soll, wurde die Diagrammebene des Ordnungsrahmens in der formalen Sprache BPMN 2.0 modelliert. Diese Notation wurde außerdem präferiert, da sie zunehmend von Open Source Anbietern implementiert wird, sich mittlerweile als internationaler Standard etabliert hat und die Möglichkeit einer Prozessautomatisierung unterstützt [Se16], [In13], [FR17]. In Abb. 3 wird exemplarisch ein Auszug des Prozesses „Forschende scouten“ aus dem Kernprozess „Internes Scouting“ aufgezeigt. Hierbei geht es um die Aufnahme der Daten aller Forschenden im Hochschulverbund in eine zentrale Forschendendatenbank. Ziel ist hierbei die Profile aller Forschenden im Verbund miteinander zu teilen und somit die Möglichkeit zur gemeinsamen Bearbeitung von Transferprojekten zu bieten. Bei diesem Prozess bietet sich beispielsweise in spätere Automatisierung an, da Profildaten aus vorhandenen Systemen, wie einem Forschungsinformationssystem, automatisch übernommen werden können.

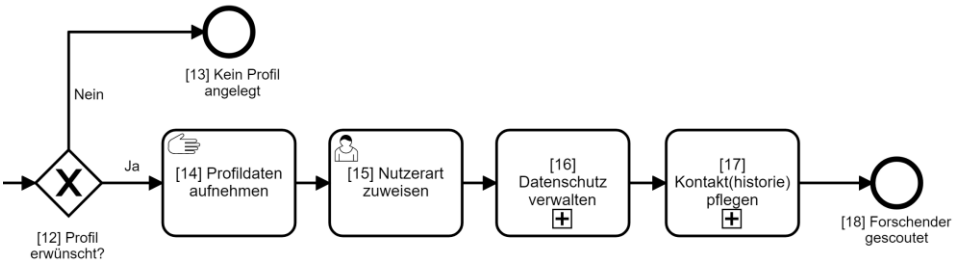
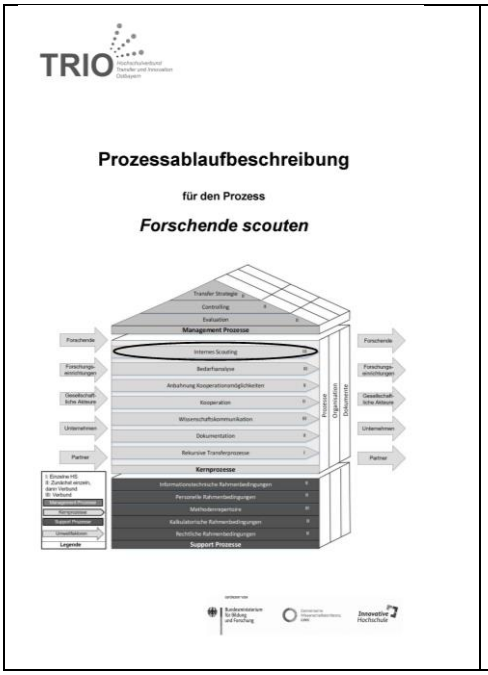


Abb. 3: Auszug der Modellierung des Prozesses „Forschende scouten“ in BPMN 2.0

Zusätzlich zu den modellierten Prozessen der Diagrammebene werden Prozessablaufbeschreibungen bereitgestellt, die das Nutzerverständnis weiter erhöhen sollen (siehe Abb. 4).



1.4 Anmerkungen zu Prozessschritten

PROZESSSCHRITT	ANMERKUNGEN	FUNKTIONS-BEREICH
[1] Info Einstellung neuer MA*in	Der Prozess wird angestoßen, sobald die Information über eine*n oder mehrere*r neue*n Mitarbeiter*innen in die Transferstelle eingeht. Zu dieser Zeit ist bereits geklärt, wie und in welchen Umfang die Information an die Transferstelle übermittelt wird. Jede Hochschule legt für sich fest, wie sie dies gestalten möchte. Bei der Profilaufnahme des Bestandspersonals startet der Prozess bei [4].	TS
[2] Potentiell Forschende!?	Handelt es sich bei dem*der neu eingestellte*n Mitarbeiter*in um eine*n Mitarbeiter*in, die potentiell an Forschung Interesse zeigen könnte? Ja → [4] Nein → [3]	TS
[3] MA*in erhält kein Profil	[O] Endpunkt, im Falle, dass MA*in als kein Forschender identifiziert wurde. Es wird kein Profil erstellt.	TS
[4] Ersten Kontakt aufnehmen	Kontaktaufnahme (z. B. telefonisch) mit neuen MA*in, um generelles Interesse an Forschung zu erfragen. Des Weiteren kann ein Termin für ein weiterführendes Gespräch / zur Vorstellung von TRIO / Transferportal vereinbart werden.	TS
[5] Interesse an Profil?	Ja → [10] Nein → [6]	TS
[6] Erneute Kontaktaufnahme erwünscht?	Ja → [8] Nein → [7]	TS
[7] MA*in erhält kein Profil	[O] Endpunkt, im Falle, dass von Seiten des Forschenden kein Interesse besteht; ggf. kann diese Information hochschulintern dokumentiert werden.	TS
[8] Wiedervorlage erstellen	Der Forschende wünscht sich einen Kontakt zu TRIO, eine gemeinsame Terminfindung ist aktuell jedoch nicht möglich. Der TRIO-MA/Transferstelle erstellt sich eine Wiedervorlage und wird zur gegebenen Zeit den Forschenden erneut ansprechen.	TS
[9] Wiedervorlage tritt ein	Die Wiedervorlage liegt vor. Somit ist der Prozess Forschende scouten mit [5] fortzufahren.	
[10] Den Forschenden ist TP detailliert bekannt?	Ja → [19] Im Gespräch stellte sich heraus, dass dem Forschenden das Transferportal bereits bekannt ist. Nein → [11] Dem Forschenden ist das Transferportal nicht bekannt.	TS
[11] TP vorstellen	Das Transferportal wird dem Forschenden vorgestellt.	TS
[12] Profil erwünscht?	Ja → [14]	TS

Abb. 4.: Auszug aus der Prozessablaufbeschreibung des Prozesses „Forschende scouten“

Die Prozessablaufbeschreibungen bestehen aus einzelnen Komponenten, welche detailliert über den Zweck, Input, Output, Kennzahlen, zugehörige Dokumente und Prozessverantwortlichkeiten Auskunft geben. Zusätzlich dazu sind alle einzelnen Prozessbausteine textuell beschrieben und der jeweilige ausführende Funktionsbereich vermerkt.

5 Evaluation des adaptiven Referenzmodells für hochschulübergreifenden Wissenstransfer

Die Evaluation des adaptiven Referenzmodells erfolgte in einem zweistufigen Vorgehen. Zunächst wurden bereits während der Konstruktion des Modells iterativ Evaluationen durchgeführt. Das erstellte Modell wurde anschließend nach dem Referenzmodellverständnis nach THOMAS als Referenzmodell deklariert [Th06]. Das Vorgehen wird im Folgenden beschrieben.

Der „Design Science“ Prozess zielt darauf ab, Artefakte zur Lösung praktischer Probleme zu schaffen [HC10]. Die Evaluierung der Ergebnisse und Artefakte ist eine der Kerntätigkeiten des Design Science Prozess und zielt darauf ab, die geschaffenen Artefakte unter Beweis zu stellen. Um die Zweckmäßigkeit der Artefakte nachzuweisen, wurden im Rahmen eines Transferprojekts interdisziplinäre Experteninterviews durchgeführt. An dem

Transferprojekt sind zwei Universitäten und vier Hochschulen in der Region Ostbayern beteiligt. Diese Hochschulen und Universitäten haben sich im Januar 2018 zu einem regionalen Hochschulverbund zusammengeschlossen mit dem Ziel den Wissens- und Technologietransfer in der Region zwischen Unternehmen, gesellschaftlichen Akteuren und dem Hochschulverbund zu intensivieren. Die ausgewählten Experten verfügen über unterschiedliche Erfahrungen und Wissen in der Abwicklung von Transferprojekten, da sie aus unterschiedlichen Positionen innerhalb der Universitäten und Hochschulen stammen. Ausgewählte Experten sind Mitarbeiter von Technologie- und Wissenstransferabteilungen, Forschungsförderungsabteilungen, Finanz- und Rechtsabteilungen sowie Forschende, welche in Verbundprojekten tätig sind. Alle Experten wurden aufgrund ihrer Verantwortung und Erfahrung im hochschulübergreifenden Wissenstransfer und ihres Besitzes von privilegierten Wissens ausgewählt [MN09]. Durch die Experteninterviews konnten umfassende Erkenntnisse im Wissenstransfer in und aus Hochschulverbünden gewonnen werden. Ziel war es sicherzustellen, dass das erstellte Informationsmodell und die dargestellten Prozesse die Realität des kollaborativen Wissensmanagements angemessen abbilden.

Ein erster Indikator für die Richtigkeit des erstellten Modells ist, dass es einen allgemeinen Überblick und harmonisierte Prozesse bietet, aber gleichzeitig hochschulindividuelle Anpassungen ermöglicht. Die Nutzung allgemein verständlicher Bezeichnungen der Prozesse ermöglicht es, dass sich alle Universitäten und Hochschulen eines Verbunds mit den Prozessen identifizieren können, auch wenn sie ihre eigenen Prozesse unterschiedlich benennen oder ausprägen. Ein Beispiel dafür ist der Prozess "Wissenschaftskommunikation", welcher in den einzelnen Universitäten und Hochschulen des Verbunds auch als "Service Presse" oder gar "Marketing" bezeichnet wird. Die Nutzung allgemein verständlicher Bezeichnungen gewährleistet eine einfachere Zusammenarbeit und hilft, Schnittstellen und Abteilungen an anderen Universitäten und Hochschulen im Verbund zu identifizieren. Da die Organisationsstrukturen der einzelnen Universitäten und Hochschulen im Verbund sehr heterogen sind, kann es mitunter schwierig sein, entsprechende Organisationseinheiten oder Prozesse an anderen Universitäten im Verbund zu identifizieren. Allgemein verständliche Bezeichnungen und das Matchen von Geschäftsprozessen und Organisationseinheiten im Verbund unterstützen und erleichtern die Zusammenarbeit und den Wissenstransfer. Die dokumentierte mangelnde administrative Unterstützung bei grundlegenden Tätigkeiten des Wissenstransfers ist derzeit an den meisten Universitäten und Hochschulen der Status quo [JBG04]. Dokumente, wie bspw. Vertragsdokumente, welche schwer erhältlich sind, können nun durch den Verbund bereitgestellt werden. Durch das Teilen von Best Practices innerhalb eines Verbunds, kann der Wissenstransfer weiter vereinfacht werden.

Die Struktur und Gestaltung des Referenzmodells trägt entscheidend zum Verständnis bei und liefert einen hohen Wiedererkennungswert. Aufgrund seiner Ähnlichkeit mit anderen Modellen wie z.B. dem Handels-H von [BM08] ist es leicht nachvollziehbar. Die Ebenen des Modells unterstützen das Verständnis der Inhalte, da jede Ebene eine Detail-

lierung der darüber liegenden Ebene ermöglicht. Zukünftige Nutzer können diese Detailebenen auch zur Restrukturierung ihrer eigenen Prozesse nutzen, da sie harmonisierte Verfahren und Best Pratices des Verbundes darstellen.
Dem Konstrukteur des Ordnungsrahmens und der dazugehörigen Prozesse des Wissenstransfers sind zum aktuellen Zeitpunkt zwei Anwendungsfälle bekannt. Zunächst wurde das Informationsmodell in einem *Transferverbundprojekt* an den beteiligten Hochschulen und Universitäten zur Übersicht der Wissenstransferprozesse implementiert. Bei der Erstellung wurden bereits alle bekannten zukünftigen Nutzer einbezogen und das Modell iterativ gemäß dem DSR-Ansatzes entwickelt [He07]. Die Evaluation erfolgte somit ebenfalls iterativ, um die empirische Aussagekraft der gefundenen Ergebnisse darzulegen. Ein weiterer Anwendungsfall ergibt sich aus der Implementierung der erstellten Prozesse an einer *einzelnen Hochschule* des Verbundprojekts. Hierbei wurden einzelne Teile des Referenzmodells in der Verwaltung der Hochschule implementiert. In diesem Zusammenhang wurden Prozesse aus den Kernprozessen „Anbahnung Kooperationsmöglichkeiten", „Kooperation", „Dokumentation" und die Supportprozesse „Kalkulatorische Rahmenbedingungen" und „Rechtliche Rahmenbedingungen" implementiert.

Durch die Nutzung des Informationsmodells konnte herausgefunden werden, dass das Modell und die erstellten Prozesse die Realität des kollaborativen Wissensmanagements angemessen abbilden. Nach dem Referenzmodellverständnis nach THOMAS kann das erstellte Informationsmodell somit als Referenzmodell deklariert werden [Th06].

6 Fazit und Ausblick

Third Mission in Verbundprojekten nimmt in der Forschungslandschaft einen immer größeren Stellenwert ein [RDH15]. Argumente dafür sind nicht nur die großen Forschungsprojekte in diesem Bereich wie die Förderlinie Horizon 2020 der Europäischen Kommission oder der Zusammenschluss von sieben forschungsorientierten deutschen Fachhochschulen zu UAS7, sondern auch der bestehende Bedarf von Unternehmen oder gesellschaftlichen Akteuren an Transfer mit Hochschulen [Am13], [La19], [IH17]. Gerade auch die aktuellen Großforschungsprogramme, wie die oben genannten Projekte, fordern explizite Wissenstransfermaßnahmen als Bestandteil jedes geförderten Forschungsvorhabens [SD14], [Am13], [Eu11].

Die Analyse des aktuellen Stands der Forschung hat ergeben, dass Wissenstransfer in und aus Hochschulverbünden weitestgehend unerforscht bzw. bisher nur exemplarisch in Einzelfallbetrachtungen beschrieben ist. Bestehende Modelle sind von einem *niedrigen Formalisierungsgrad* und einer fehlenden *vernetzte, neutrale und allgemeingültige Darstellung* der hochschulübergreifenden Prozesse des Wissenstransfers geprägt.

Ein Referenzmodell kann eine vernetzte, neutrale und allgemeingültige Darstellung liefern und somit den verbundübergreifenden Wissenstransfer erleichtern. Das adaptive Referenzmodell zeigt alle, für den Wissenstransfer relevanten, Prozesse auf (RQ1). Eine Unterscheidung in kaskadierende Prozesse verdeutlicht, welche Prozesse an einzelnen Hochschulen oder im Verbund durchgeführt werden (RQ2). Die Modellierung des Ordnungsrahmens, der Wertschöpfungsketten und der Diagrammebene zeigt die drei Ebenen, in welchen sich die Prozesse des Transfergeschehens gestalten lassen (RQ3).

Zukünftige Forschungstätigkeiten umfassen die weitere Detaillierung der Kernprozesse des Transfers und das Auffinden weiterer Prozessbausteine. Im Rahmen dieser Forschungsarbeiten erfolgt unter anderem die weitere Anwendung des Modells in einem Verbundprojekt der ostbayerischen Hochschulen und Universitäten, um weitere Erkenntnisse zu evaluieren. Hierbei soll ebenfalls eine Anwendung und Evaluierung in weiteren Hochschulverbünden, sowohl national als auch international, erfolgen. Das Konzept des Referenzmodells ist hierbei einfach auf weitere Verbünde oder auch auf andere Länder übertragbar, da die Grundstrukturen des Transfers und die Organisation von Hochschulen sich ähneln. Bei der weiteren Modellierung des Referenzmodells sind darüber hinaus geeignete Adaptionsparameter für die Gestaltung von verbundübergreifendem Wissenstransfer zu identifizieren. Das Referenzmodell soll über Ausdrücke verfügen, welche eine semi-automatische Adaption an kooperationsindividuelle Vorgehen des Wissenstransfers erlauben. Hierbei soll nach DELFMANN die Elementselektion über Konfigurationsterme betrachtet werden, bei der einzelnen Modellelementen Adaptionsparameterausprägungen zugeordnet werden [De06]. Außerdem sollen ausgewählte automatisierte Konfigurationsmechanismen nach BECKER et al. herausgearbeitet werden, um eine perspektivengerechte Konfiguration des adaptiven Referenzmodells zu ermöglichen [BK02].

7 Danksagung

Entstanden im Transferprojekt „Transfer und Innovation in Ostbayern“ das aus Mitteln der Innovativen Hochschule von 2018 bis 2022 gefördert wird. Förderkennzeichen: 03IHS078D.

8 Literaturverzeichnis

[Am13] Amtsblatt der Europäischen Union: Verordnung (EU) Nr. 1291/2013 des Europäischen Parlaments und des Rates vom 11. Dezember 2013 über das Rahmenprogramm für Forschung und Innovation Horizont 2020 (2014-2020) und zur Aufhebung des Beschlusses Nr. 1982/2006/EG, 2013.

[Ba06] BayHSchG: Bayerisches Hochschulgesetz, 2006.

[BF00] Blume, L.; Fromm, O.: Wissenstransfer zwischen Universitäten und regionaler Wirtschaft: Eine empirische Untersuchung am Beispiel der Universität Gesamthochschule Kassel. In Vierteljahrshefte zur Wirtschaftsforschung, 2000, 69; S. 109–123.

[BK02] Becker, J.; Knackstedt, R. Hrsg.: Wissensmanagement mit Referenzmodellen. Konzepte für die Anwendungssystem- und Organisationsgestaltung. Physica-Verl., Heidelberg, 2002.

[BM08] Becker, J.; Meise, V.: Strategie und Ordnungsrahmen: Prozessmanagement: ein Leitfaden zur prozessorientierten Organisationsgestaltung. Springer, 2008.

[Cl98] Clark, B. R.: The Entrepreneurial University: Demand and Response. In Tertiary Education and Management, 1998; S. 5–16.

[De06] Delfmann, P.: Adaptive Referenzmodellierung. Methodische Konzepte zur Konstruktion und Anwendung wiederverwendungsorientierter Informationsmodelle. Logos-Verl., Berlin, 2006.

[EL00] Etzkowitz, H.; Leydesdorff, L.: The dynamics of innovation: from National Systems and ‘‘Mode 2’’ to a Triple Helix of university–industry–government relations, 2000.

[Eu11] European Commission: Regulation of the European Parliament and of the Council laying down the rules for the participation and dissemination in Horizon 2020 – the Framework Programme for Research and Innovation (2014-2020), 2011.

[FL04] Fettke, P.; Loos, P.: Referenzmodellierungsforschung. In WIRTSCHAFTSINFORMATIK, 2004, 46; S. 331–340.

[Fr14] Froese, A.: Wissenschaftliche Güte und gesellschaftliche Relevanz der Sozial- und Raumwissenschaften, 2014.

[FR17] Freund, J.; Rücker, B.: Praxishandbuch BPMN. Mit Einführung in CMMN und DMN, 2017.

[Ga17] Gadatsch, A.: Grundkurs Geschäftsprozess-Management. Analyse, Modellierung, Optimierung und Controlling von Prozessen. Springer Vieweg, Wiesbaden, 2017.

[Gi94] Gibbons, M.: The New Production of Knowledge: The Dynamics of Science and Research in Contemporary Societies.

[GM12] Grothe, A.; Marke, N.: Nachhaltiges Wirtschaften in Berliner Betrieben: Neue Formen des Wissenstransfers zwischen Hochschulen und Unternehmen. Working Papers of the Institute of Management Berlin at the Berlin School of Economics and Law (HWR Berlin),, 2012.

[HC10] Hevner, A. R.; Chatterjee, S.: Design Research in Information Systems Theory and Practice. In Integrated Series in Information Systems Volume 22, 2010.

[He07] Hevner, A. R.: A Three Cycle View of Design Science Research. In Scandinavian Journal of Information Systems, 2007.

[IH17] IHK Bayern: Innovationsreport Bayern, 2017.

[In13] International Organization for Standardization: ISO/IEC 19510:2013. Information technology -- Object Management Group Business Process Model and Notation. https://www.iso.org/standard/62652.html, 13.08.2018.

[JBG04] Jacobson, N.; Butterill, D.; Goering, P.: Organizational Factors that Influence University-Based Researchers' Engagement in Knowledge Transfer Activities. In Science Communication, 2004, 25; S. 246–259.

[KMS16] Kohl, H.; Mertins, K.; Seidel, H.: Wissensmanagement im Mittelstand. Grundlagen - Lösungen - Praxisbeispiele, 2016.

[La19] Lange, C.: UAS7: German Universities of Applied Sciences. http://www.uas7.de/Start.2.0.html, 12.03.2019.

[Me01] Meise, V.: Ordnungsrahmen zur prozessorientierten Organisationsgestaltung. Modelle für das Management komplexer Reorganisationsprojekte.

Münster, Univ., Diss., 2000, 2001.

[MN09] Meuser, M.; Nagel, U.: Das Experteninterview - konzeptionelle Grundlagen und methodische Anlage. In (Pickel, S. et al. Hrsg.): Methoden der vergleichenden Politik- und Sozialwissenschaft, 2009.

[No16] North, K.: Wissensorientierte Unternehmensführung. Wissensmanagement gestalten. Springer Fachmedien Wiesbaden, Wiesbaden, 2016.

[Pi14] Pircher, R. Hrsg.: Wissensmanagement, Wissenstransfer, Wissensnetzwerke. Konzepte, Methoden, Erfahrungen, 2014.

[Po16] Polanyi, M.: Implizites Wissen. Suhrkamp, Frankfurt am Main, 2016.

[Ra13] Rauter, R.: Interorganisationaler Wissenstransfer. Zusammenarbeit zwischen Forschungseinrichtungen und KMU, 2013.

[RDH15] Roessler, I.; Duong, S.; Hachmeister, C.-D.: Welche Missionen haben Hochschulen? In CHE gemeinnütziges Centrum für Hochschulentwicklung, 2015.

[SD14] Seel, C.; Dreifuß, F.: Induktive Entwicklung eines Vorgehensmodells für Wissenstransfermaßnahmen in der Wirtschaftsinformatik, 2014.

[Se10] Seel, C.: Reverse Method Engineering. Methode und Softwareunterstützung zur Konstruktion und Adaption semiformaler Informationsmodellierungstechniken, 2010.

[Se16] Seel, C. et al.: Vergleichende Analyse von Open-Source-Modellierungswerkzeugen als Basis für Forschungsprototypen. In (Barton, T. et al. Hrsg.): Prozesse, Technologie, Anwendungen, Systeme und Management 2016. Angewandte Forschung in der Wirtschaftsinformatik, 2016; S. 35–44.

[Th02] Thiel, M.: Wissenstransfer in komplexen Organisationen. Effizienz durch Wiederverwendung von Wissen und Best Practices, 2002.

[Th06] Thomas, O.: Das Referenzmodellverständnis in der Wirtschaftsinformatik. Historie, Literaturanalyse und Begriffsexplikation. In Institut für Wirtschaftsinformatik (IWi), 2006.

[To09] Towfigh, E. V.: Komplexität und Normenklarheit - oder: Gesetze sind für Juristen gemacht. In Der Staat Berlin, 2009, 48; S. 29–74.

[Vo09] Vom Brocke, J. et al.: Reconstructing the giant. On the importance of rigour in documenting the literature search process. In ECIS 2009 Proceedings, 2009.

Dominik Bork, Dimitris Karagiannis, Heinrich C.Mayr (Hrsg.): Modellierung 2020
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 109

Model-based Software Cost Estimation

Calculating Time and Effort for Software Evolution Projects

Harry Sneed[1], Wolfgang Prentner[2]

Abstract: Estimating the costs of an evolution project differs from development project estimation and must follow its own rules. When estimating development project costs the whole system is taken into consideration. When estimating evolution costs only those parts of the system are considered that have to be changed or added. The rest is left as it is, but must be included in the test. The mixing of changed components with new components and old components presents several challenges to software product management. The main challenge is how to recognize those features that have to be added or changed – feature analysis. Together they make up the change domain. The extent of this change domain is the key factor in estimating the costs of change in each new release. It is measured by means of one or more size metrics such as function-points, data-points and object-points in order to convert size into effort. The approach used here was to model the change requirements and then compare the change model with the original requirements model to ascertain the scope of the change. To this end, both the original and the current requirements had to be extracted from the requirement text and then modelled. This approach was applied here to calculate the costs of expanding a national health record system. The preliminary results are presented in this short paper.

Keywords: Software evolution, Requirement Modelling, Natural Language Processing, Software sizing, Requirement Metrics, Function-Points, Data-Points, Object-Points, Model-based Estimation.

1 Introduction

The IT system to be evolved here is the security subsystem of an information system for managing health records of state insured citizens. There are more than 5 million health records in this particular system. The system has already been in operation for more than 7 years. It was developed by a local software shop specialized in health information systems and the same shop is still maintaining it. Recently additional security requirements have come up which were not considered in the original requirement analysis, mostly as a result of new data protection legislation. Many of them deal with access rights, user protection and system security levels. They were not foreseeable at the time the system was first conceived and if they were, it was decided to postpone their implementation. Now the law requires for them to be adhered to. The

[1] SoRing Kft, Kapitanutca 6, H1123 Budapest, Hungary, Harry.Sneed@SoRing.hu

[2] ZT-Prentner-IT, Kagranerplatz 40, A1221, Wien, Austria, Prentner@ZTP.at

national health agencies have no choice but to build them into their current health administration systems. The alternative would be to build a completely new system which satisfies all the old requirements plus the new ones, i.e. a complete redevelopment. In such a redevelopment the entire requirement model must be made again from scratch. In an evolution project only the new requirements need to be re-modelled. The old requirements are assumed to be already implemented. If the new requirements are to be compared with the old ones, then these too must be modelled as they were here. The reverse engineering of the old requirement document took more than four times the effort required to model the new requirements.

2 Levels of System Modelling

In any evolving IT system there are at least two levels to be modelled. These are the requirements and the test. If the evolution team is working according to the book there will be a design model and a test model. Model-based testing presupposes a test model from which system test cases can be derived. Model-based cost estimation presupposes a requirement model from which system size metrics can be derived [Selb09].

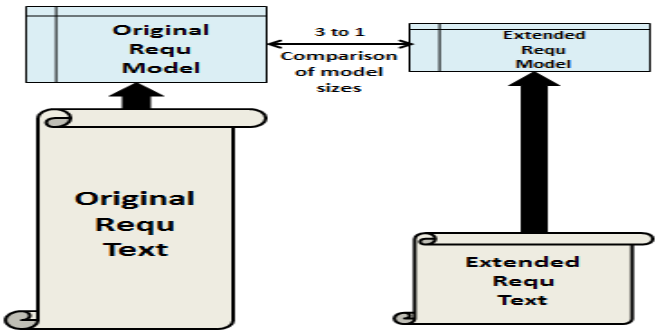


Figure 1: Modelling the Requirements

In this evolution project the requirement documentation consisted of two separate documents in German language:

- A detailed description of the existing security system requirements (Pflichtenheft)
- A general description of the new security system requirements (Lastenheft).

2.1 Existing System Requirements

The specification of the existing system requirements is a 390 page word document containing a combination of texts, tables and diagrams. The texts are for the most part

short paragraphs prescribing some security feature of the current system such as the authentication of users who log into the system. The nouns in those paragraphs are often acronyms defined in an acronym table at the end of the document. Otherwise they are a mixture of German and English medical terms such as “treatment”, “prescription”, “medication”, “Arzt”, “Behandlung” and “Ordination”. Often English and German terms are used alternately to denote the same object types, for instance “treatment” and “Behandlung”. To comprehend the texts the reader needs to be familiar with both German and English medical terms as well as with the acronyms used in this context. A sample from the text might look like this:

„Dem Gesetz entsprechend ist ein Aggregiertes Audit Record Repository (A-ARR) als zentrales Service zu errichten, das es den Teilnehmern (Bürgern) ermöglicht, Einsicht in die aufgezeichneten Protokolldaten, die ihre eigenen Gesundheitsdaten betreffen, zu ermöglichen. Die Protokolldaten werden von der ZGF bzw. von ETS und PAP erstellt und an das A-ARR mittels https weitergeleitet. Dem Bürgerportal steht eine lesende SOAP Schnittstelle zum Abrufen aller Protokolleinträge eines Bürgers zur Verfügung.“

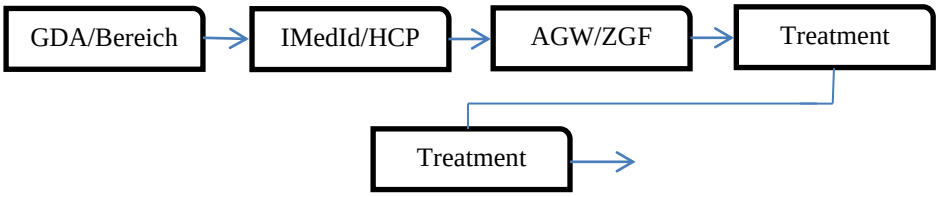
Sample 1: Specified Requirement

The texts are enhanced by tables and diagrams. If the text is describing an object there is often a table defining the attributes of that object, for instance for the term “Authentication” there is a list of authentication types equivalent to an enumeration.

# Authentication	R	
# @NotBefore	R	Time instant from which the assertion is useable. It is set as the issue instant
# @NotOnOrAfter	R	Time instant at which the assertion expires. Value is set to 4 hours
# @AudienceRestriction	R	This element contains the list of Audiences, e.g., the *contexts* (services) for whom the STS issued the assertion. Identity Assertion is used only with ETS (https://elga-online.at/ETS).
# AuthnStatement	R	
# @AuthnInstant	R	Time instant of authentication in UTC Format:yyyy'-'MM'-'dd'T'HH':'mm':'ss'.'fff'Z'
# AuthnContext	R	
# AuthnContextClassRef	R	urn:oasis:names:tc:SAML:2.0:ac:classes.*
# AttributeStatement	R	HCP identity attributes and permissions (Attribute der Identity Assertion)
# ds:Signature	R	Enveloped XML signature of the issuer of the Identity Assertion

Sample 2: Specified Data Attributes

If the text is describing a process then a process diagram is included to depict the sequence of steps within that process. The description of a process often turns out to be recursive as the steps of a process may be themselves sub-processes for which further diagrams are needed.



Sample 3: Specified Process

To measure the size and complexity of a requirement model it is necessary to distinguish between objects and actions. An object is a data entity or data set. In the code entities are implemented as structures or classes. An action can be a process, a use case or an elementary function. A process is actually a sequence of procedurally related functions. In the code processes are implemented as procedures, but in object-oriented systems, procedures are not explicitly defined, thus they are not statically recognizable. This leads to a semantic gap between requirement specifications and code. Processes or sequences of elementary functions can only be recognized by dynamic analysis [GDG06].

The detailed requirement model is detailed because what it is describing already exists and can be observed in operation. The authors of the detailed requirements are describing what happens when the system is executed. They are describing what they can physically observe. The model may not be totally accurate but it is accurate enough to be used as a basis of measurement. The only truly accurate description of a system is the code itself. By comparing the requirements with the code one can detect where they deviate from one another, but only if they are at the same level of abstraction [ChGa01]. In comparing one abstract model with another we are actually comparing two shadows of the real system.

2.2 Projected System Requirements

The specification of the new system requirements for the security system is not a single all-inclusive document like that for the current system as a whole, but a set of seven related documents prescribing what additional features the new security subsystem should have:

- An 11 page description of the proposed collaboration platform
- A 5 page description of the build process
- A 7 page description of the documentation classes
- A 10 page description of the virtual system architecture
- A 15 page description of the application container
- A 36 page description of the injection passport pilot application
- A 62 page description of the proposed x-Ray exchange service.

That adds up to 146 pages of requirement specifications for the enhanced security

system. Of course these requirement specifications are less detailed than the current system specification because the authors are not describing what they see but prescribing what they would like to have. They cannot physically observe those features but only imagine how they might be implemented. The objects and actions are therefore less precisely defined or their exact definitions are left open to be defined at implementation time (tbds) [Mugr08].

3 Modelling the Security System Requirements

The primary goal of this particular modelling process was to measure the size and complexity of the proposed system change. Since there was neither a design nor code for the extended system, the only thing that could be measured was the requirements of the new security subsystem. These could be compared with the requirements for the current system as a whole. System cost estimation implies measurement. To make an objective comparison one must compare numbers and that requires measuring the objects to be compared – in this case the two requirement documents. In order to measure a requirement text document the text of that document must first be converted to a given requirement model that coincides with the requirement-based estimation methods. Then the two models can then be matched with one another to identify the differences between them. Thus, the first step in comparing the new requirements with the existing ones is to extract a numeric model from the requirement text.

Once a numeric model has been created from the descriptive text, the model elements can be counted and the counts compared. The prerequisite to comparing models is to identify the model elements described in the text. At the current state of artificial intelligence this task can only be performed by an intelligent human being. The person assigned to this work must examine each and every text passage to decide what that text is all about and what model element is being specified by it.

Based on his many years of experience in analyzing requirement documents in different natural languages, combined with the knowledge acquired from the literature on requirement modelling, the first author identified some 30 model types ranging from use-cases to elementary function steps and from logical data entities to elementary data items, including services and user-interfaces. A full list of the model element types was published in a paper by the authors on requirements reengineering [SnPr18].

Based on his or her knowledge of the application and his or her familiarity with the terms used in the target document, the analyst assigns the text passages to a selected model type. It is human analyst who decides what a text passage is prescribing. Actually the analyst is associating the terms used in the text with the terms associated with the model types. The experienced analyst will soon recognize that the same text patterns are used again and again to describe model elements of the same type. With these patterns in the back of mind, the analyst will be able to rapidly scan through the requirement text and assign the text paragraphs to the appropriate model element types [Lefi11].

Having recognized a model element, the analyst inserts the element type identifier in a separate line before the text passage or paragraph to which that type applies.

&FREQ-011 Patientenidentifikationen

*Alle Patientenidentifikationen werden im Format **CX HL7 V2.5** gemäß IHE XDS.b Profil erwartet.*

Sample 4: Requirement Definition

&Regel Die Signatur der Assertion wird gemäß W3C XMLDSig geprüft.

Sample 5: Rule Definition

This action is referred to as marking up the text. It is similar to marking up an XML document, but much less formal. The end delimiters are left out. A model element ends where the next one begins. Considering that a page of requirement documentation will contain on average 6 model elements, it should be possible to mark up a page of requirement text within 20 minutes, or at the rate of 18 model elements per hour. A document of 100 pages would require 2000 minutes = 33 hours or some 4 working days. This is exactly what it cost to mark up the new security requirement document in this cost estimation project. The marking up of the original security document took more than 16 working days. This was not only due to the volume but also to the density of the descriptions [Bria17].

&UseCase: HCP_Assertion_anfordern

Läuft eine Benutzersession ab oder wurde invalidiert, muss auch die HCP Assertion beim ETS invalidiert werden.

&MainPath: Steps =

1) Ein GDA System oder ein Gateway eines System-Bereichs sendet eine WS Trust RST Issue Transaktion an die ZGF des System-Bereichs, um sich bei dem System wieder anzumelden.
2) Im Security Header der SOAP Nachricht befindet sich die Identity Assertion des lokalen IDPs.
3) Der Apache der lokalen AGW leitet die RST Nachricht zum zentralen ETS weiter.
4) Eventuelle Fehler werden in einer WS Trust Tabelle zurückgeliefert.
5) Fehlerfälle werden gemäß der Fehlerbehandlungsrichtlinie behandelt.

Sample 6: Use Case Definition

Once a requirement document has been marked up it can be automatically measured. The text parser will not only be able to recognize and count the nouns but also to count the model element types, i.e. it can count individual functional and non-functional requirements, use cases, actors, inputs, outputs, interfaces, data objects and data attributes. The tool for analyzing the marked-up text and counting the model entities has

been described in previous papers [Sned05]. From these counts it can derive data-points, function-points, object-points and usecase-points. These are the essential size metrics for determining the extent of a requirement model. The model sizes will not necessarily coincide with the code sizes as the model is only a shadow of the actual system. However by measuring the size of a shadow one can make some assumptions about the size of the real object depending upon the time of day. In the case of a requirement model, it depends upon the state of the requirements at the time when the model is measured. Unfortunately we cannot assume that the requirement document will be consistent with the live system. Some requirements may have never been implemented or were only partially implemented. Unfulfilled requirements make up for a good part of the technical debt [Ster10].

This does not hold for the model of the projected requirements, i.e. those that are yet to be fulfilled. They reflect how big the system extension would be if all of the new requirements were fulfilled. One can get an impression of the relation between the size of the original model and the new enhanced model by comparing their size metrics with one another [Sned10]. In table 1 the number of entities in the current requirement model are listed out next to the number of entities which are to be added in the extended requirement model. The new requirement documentation only includes the additional entities, i.e. the delta of the extended model. Of course the existing model will also be affected by the enhancements made, so additional costs have to be planned for adapting and testing existing artifacts impacted by the system enhancement.

Require. Entity Type	Current Entity Count	Added Entity Count	Total Count
User interfaces	80	8	88
System interfaces	154	30	184
Data Objects	133	61	194
Data Attributes	1929	193	2022
Object States	406	192	598
Actions	1139	217	1356
Business Rules	892	197	1089
Business Processes	98	19	117
Actors	12	6	18
Use Cases	101	20	121
Use Case Paths	40	9	49
Use Case Steps	484	56	540
Logical Test Cases	101	20	121
Function-Points	3868	1134	5002
Data-Points	15347	4971	20318

Tab. 1: Requirement Model Entity Counts

4 Estimating System Enhancement Costs

By comparing the size metrics of the existing requirement model with those of the projected model it was possible to determine the degree of change. The size of a system can be viewed from two points of view:

- Functional point of view
- Data point of view [CMPB05].

The functional point of view is reflected in the number of function-points and use case-points. The data point of view is expressed in data-points and object-points.

4.1 Counting Function-Points in the new Requirement Model

Function-Points are actually weighted counts of the model inputs and outputs plus the weights of the data entities and internal interfaces defined in the model. When the requirement model is extracted from the requirement text all the user interfaces, incoming messages and service requests are marked as being inputs and all the user interfaces, reports, outgoing messages and service responses are marked as being outputs. Inputs weigh from 3 to 6 points, outputs weigh from 4 to 7 points, data entities weigh from 5 to 15 points and internal interfaces weigh from 5 to 10 points, depending on the complexity of the requirement model as a whole. These weighted counts are added up to give the number of requirement function-points [IFPG99]. Of course the counting rules have to be simplified in order to automate the counting. The complexity of individual inputs and outputs cannot be measured. Instead the complexity of the model as a whole is applied to every input and output. If the complexity of the model is greater than 0.6 all of the outputs of that model will be assigned 7 points and all of the inputs 6 points. This simplification is already made in the Cosmic Function-Point method where all inputs and outputs have the same weight of 1.

In the model of the projected security subsystem 1134 function-points were counted, indicating a difference of 3.4 to 1 between the size of the original requirement model for the system as a whole and the size of the new model of the security subsystem. Carrying this difference over to the effort involved, it means that the extended security features should cost no more than 30% of what the original system had cost.

4.2 Counting UseCase-Points in the new Requirement Model

Use Case-Points are weighted counts of the use cases and actors defined in the model. The weight of a use case is determined by the number of paths and steps specified for that use case. Every use case must have at least one path and each path must have a least one step. There is no limit to the number of paths and steps a use case may have, but normally there is no more than two paths – the normal path and the exceptional path – each with some 3 to 10 steps. The authors of the use-case method assign 5 points to the

cases with up to 3 steps, 10 points to use cases with 4 to 7 steps, and 15 points to use cases with over 7 steps. In addition system actors are weighted from 1 to 3 points depending on their type of interface, whether batch, message or dialogue. The weights of the use cases and actors are summed up to give the total number of use case-points [Karn93].

In the model of the existing system as a whole there were 2035 use case points. In the model of the projected security system 220 use-case-points were counted. That indicates a ratio of 9:1 from the existing entire requirement model to the proposed partial one. That indicates that the system extension would only cost 10% of what the original system cost. This cannot be true. It results from the fact that the use-case method was intended for estimating frontend development. The planned security subsystem is mainly a backend solution. Thus, the use-case method does not apply here.

4.3 Counting Data-Points and Object-Points in the Requirement Model

The data point of view is reflected in the number of data-points and object-points. At the requirement model level these two counting procedures are very similar. Data elements, i.e. nouns, are weighted by 1, data objects by 2, user interfaces, reports and messages by 4 to give the number of data-points [Sned05b]. To compute object-points the number of data groups and messages are weighted by 4, the number of use cases by 8, and the number of actions by 3 [Sneed96]. In the model extracted from the original requirement document there were 15.347 data-points counted whereas in the model extracted from the requirement specification of the new security subsystem 4971 were counted. With object-points, it was 8669 object-points in the original model for the system as a whole as opposed to 2043 in the new partial model for the security subsystem. This gives a relation of 3.0 to 1 for data-points and 4.2 to 1 for object-points.

Model-Type	Funct-Points	Data-Points	UC-Points	Object-Points
Original	3868	15347	2035	8669
Extension	1134	4971	220	2043
Ratio	3.4:1	3.0:1	9.2:1	4.2:1

Tab. 2: Size of System Change Model relative to Original System Model

It is notable that function-points and data-points come to a similar ratio although the two methods are counting quite different model types. Function-points are counting data flows whereas data-points are counting data-elements. Use case points are counting paths and steps whereas object-points are counting data-groups. Object-points are more difficult to identify at the requirement level as data groups are not always explicitly defined. Thus the ratio of 4.2 to 1 deviates somewhat from the ratios of 3.0 and 3.4 to 1. Object-Points are more easily countable at the design level where the data structures are closer to the implementation in a programming language with well-defined model types such as packages, modules, classes, interfaces and methods. The use-case point count deviates significantly from the other counts. This is because there are only 20 use cases which can be assigned specifically to the security subsystem. The other use cases which

pass thru the security subsystem actually belong to other subsystems.

Unfortunately a requirement model is by definition fuzzy. The size counts cannot be more precise than the requirement text. If the text is inexact, inconsistent and incomplete the model extracted from that text will also be inexact, inconsistent and incomplete. The point to be made here is that even fuzzy, informal and imprecise requirement specifications can be captured in a requirement model and that this model can be used to measure the approximate size of the software to be developed. The question is whether it is better than nothing at all. Here an attempt was made to capture the size of the planned change relative to the system as a whole using fuzzy size metrics [Cohn06].

5 Estimating absolute Costs of System Enhancement

An alternative approach to estimating the costs of enhancing the security system is to convert the requirement size metrics into effort based on a benchmark productivity table. Such a benchmark table was provided by the David Consulting Company for users in the USA [BuDe08]. Here one can see that productivity varies between 9 und 25 Function-Points per person month. This range coincides with the productivity of the GEOS Project in Vienna where the developers produced an average of 1,1 Function-Points per work-day [SnHa05].

Projektart	Produktivität
Client/Server Entwicklung	17 Function-Points pro PM
Mainframe Weiterentwicklung	13 Function-Points pro PM
Websystementwicklung	25 Function-Points pro PM
E-Business Systementwicklung	15 Function-Points pro PM
Standard-Softwareentwicklung	18 Function-Points pro PM
Data Warehouse Entwicklung	9 Function-Points pro PM

Tab. 3: Function-Point Productivity in the USA

Every software development organization should have its own productivity table of past projects giving the relation of requirement size metrics to the effort required to implement them. From this table the organization can project the effort required for future projects by matching the sizes. If it took n person months to implement m function-points in the past, it can be assumed that it will take a similar effort to implement m function-points in the future. Deviations can be traced to different influence factors which are included in the cost calculation. This is the standard approach to converting function-points into effort as propagated in the literature since

1983. The tool used for converting function-points into effort was SoftCalc, a tool originally developed by the first author in 1990 [DuFo96].

SoftCalc does not assume a linear relation between points and effort. On the contrary, there is a separate productivity table for each estimation method in which the relation of points to person-months is recorded. It is via this table that the point count is related to the effort as shown in the following table.

Project Type	Project Technology	Project Subject	Nr Units	Person Months	Units perPM	Total Costs (in €)
Redevelop	4GL-Synon	Logistic	32.383	1200	30	16.000.000
Reimple	Object	Transport	12.067	694	17	11.104.000
Reimple	Object	Public	9.298	270	34	4.330.498
Redevelop	Procedural	Payroll	17.898	520	34	6.240.000
Convert	Cob2Java	Insurance	31122	513	60	615.600

Tab. 4: Function-Point Productivity in Austria

For the security system extension 1134 function-points were counted in the new requirement model. Based on the productivity from the original system, this amounted to 72 person months. Adjusting this raw estimate by the

- Influence, resource, risk and overhead factors

lead to a final effort estimation of 88,6 person months for the proposed system enhancement. The influence factors used here are those proposed by the IFPUG convention plus those added later on by the OMG.

- data communication (original IFPUG)
- system type (has been added)
- system performance (original IFPUG)
- multi mandate (has been added)
- transaction rate (original IFPUG)
- product reliability (has been added)
- product usability (original IFPUG)
- business criticality (has been added)
- process complexity (original IFPUG)
- system reusability (original IFPUG)
- migration necessity (original IFPUG)
- system security (has been added)
- installation multiplicity (original IFPUG)
- system adaptability (original IFPUG)

System quantities:		Effort estimates	
UseCases	22	Undjusted Effort:	72,0 PMs
Objects:	51	☑ Influence Factor	1,07
Interfaces:		Influence adjusted Effort:	77,0 PMs
Components:		☑ Resource Factor	1,00
Test Cases:		Resource adjusted Effort:	77,0 PMs
Change_Rate	0,20	☑ Risk Factor	1,00
		Risk adjusted Effort:	77,0 PMs
		☑ Use Overhead Factor	0,15
		Final Effort:	88,6 PMs

Size measurement		Project estimates	
Unadjusted Size:	1134,00	Minimum Effort:	77,0 PMs
Quality Factor:	1,00	Minimum Time:	15,6 Months
Quality adjusted Size:	1134,00	Minimum Cost:	1.155.472
Complexity Factor:	0,67	Actual Cost:	1.328.792
Complexity adjusted Size:	757,51	Optimal Staff:	4,9 Prs.
Final adjusted Size:	909,01	Annual Maint. Effort:	18,5 PMs

Sample 7: Absolute Function-Point Estimation

The absolute estimation by data-point converted the 4971 data-points of the extended requirement model to 56 person months which were then adjusted to 61,4 PMs by the data-point influence-factors and the project overhead. The 10 data–point influences were defined by Sneed in the original data-point paper.

System quantities:		Effort estimates	
UseCases	22	Undjusted Effort:	56,2 PMs
Objects:	51	☑ Influence Factor	0,95
Interfaces:		Influence adjusted Effort:	53,4 PMs
Components:		☑ Resource Factor	1,00
Test Cases:		Resource adjusted Effort:	53,4 PMs
Change_Rate	0,20	☑ Risk Factor	1,00
		Risk adjusted Effort:	53,4 PMs
		☑ Use Overhead Factor	0,15
		Final Effort:	61,4 PMs

Size measurement		Project estimates	
Unadjusted Size:	4971,00	Minimum Effort:	53,4 PMs
Quality Factor:	1,00	Minimum Time:	13,6 Months
Quality adjusted Size:	4971,00	Minimum Cost:	801.440
Complexity Factor:	0,67	Actual Cost:	921.656
Complexity adjusted Size	3320,63	Optimal Staff:	3,9 Prs.
Final adjusted Size:	3984,75	Annual Maint. Effort:	12,8 PMs

Sample 8: Absolute Data-Point Estimation

6 Estimating relative Costs of System Enhancement

The final step in estimating the costs of the security system enhancement was to compare the absolute effort estimation with the relative effort estimation and to reconcile the two estimations with one another. The absolute effort estimation obtained by joining the actual function-point count of the new requirement model with the current function-point productivity table was 88.6 person months. The relative effort estimation obtained by comparing the size of the extended function model with the size of the original function model was 85 person months, almost exactly the same as with the absolute effort estimation of 86.5 person months. The absolute effort estimation with the current data-point productivity table was 61.4 person months. The relative effort obtained by comparing the extended data model with the original data model was 94 person months, a difference of 50% to the absolute cost estimation. This indicates that for this particular requirement document the function-point estimation is more reliable.

7 Justifying the Costs of model-based Estimation

It cannot be denied that this estimation could have been done with less cost. Constructing a requirement model just to estimate enhancement costs is indeed somewhat extravagance. Therefore, the total effort of 32 person days (18 PDs for marking up the original requirement document + 4 PDs for marking up the new requirement document + 8 PDs for adjusting the analysis tools + 2 PDs for summarizing and reporting the results). An expert on Java programming with experience in developing security software might have read through the requirements and come to a conclusion on the costs within a day. On the other hand, such an estimate cannot be readily explained. There are neither facts nor numbers behind it. It is based on intuition and human judgement. For smaller projects it may work. The requirement document here is too big and too complex for an average developer. He or she needs a model to structure their thoughts and to make meaningful associations. Besides the requirement model provides other advantages above and beyond cost estimation. Sizing a system is only one reason for modelling. There are several other reasons such as establishing a baseline to trace requirements through a finished system.

8 Comparing estimated with actual Costs

All papers on cost estimation are confronted with the demand to compare the estimated costs with the actual costs. This author would be more than glad to satisfy that demand but there are two major barriers to overcome. One barrier is that of timing. In the case of large scale projects like this one there is a major time gap between the time the project is estimated and the time it is completed. This project is still pending, meaning it has yet to start. Therefore, there is no data to compare. The purpose of the estimation project was to collect data on whether the project would be feasible. The calculated data was

compared with the costs projected by the responsible health department managers. When they perceived that the effort calculated was within plus/minus 20% of what they projected they were satisfied. The cost estimation had served their purpose.

What could have been done was to transpose the costs of developing the current system to the planned new subsystem based on the ratio of their sizes. According to a comparison of the two requirement models, the new system should cost circa 1/3 of the original system as a whole. If the original system had taken 240 person months to development then the new subsystem would take at least 80 person months, provided the same persons are working on the project. This is a good example of cost estimation by analogy [ShSc97]. But then follows the other barrier.

The other barrier is that of organizational confidentiality. No organization is particularly open about their development costs, their productivity and their product quality. These topics are highly sensitive. In this respect industry is more secretive than the military. Thus, even if the project had been completed it would still be very difficult to get data on the accumulated costs. Managers are concerned that the data will in some way be used against them. For this reason project consultants are obliged to sign several non-disclosure agreements before they are allowed to take part in a project. Unfortunately, this also applies to projects in the public domain like this one.

9 Conclusions and further Work

The effort estimation of a system enhancement project reported on in this paper shows how difficult it is to predict the costs of evolution projects on the basis of fuzzy requirement specifications. The estimators were confronted with two fuzzy requirement documents – one for the existing security system and one for the proposed security system. One solution might have been to measure the size of the existing code, which was actually done, and to match the code metrics to the actual costs. Then one might have guessed how much less the system extension is compared to the original system. As measured here by comparing the two requirement models, it was 1/3 of the original system. Then it should cost no more than 1/3 of what the original system had cost. Unfortunately, you cannot compare an informal model with a formal one. They simply do not match, especially if they are at different semantic levels. At the code and for the most part at the design level, the model entity types are of a technical nature, e.g. modules, procedures, tables and files. At the requirement level model entity types are more of a business nature, e.g. business processes, use cases, business rules, logical entities, data attributes and business interfaces. In some cases they may match to corresponding technical model types. In many cases they do not match. The reason for having two models is that there are two classes of users – business users and technical users, each with their own language. They could in fact be united, only business-oriented persons insist on speaking their own language. So the IT world remains divided with all the consequences that go along with a divided world – redundancy, inconsistency and lack of traceability [Parn77].

Bibliography

[AB00] Abran,A.; Khelifi,A.; Buglione,L.: A System of Reference for Software Measurement with ISO 19761 (COSMIC FFP) in: Abran: Software Measurement – Research and Application, Shaker Publ., Aachen, 2004, pp. 89-108

[Anda01] Anda, B.: Comparing Effort Estimates based on Use-Case Points with Expert Estimates, Proc. of IEEE ICSE 2001, Computer Society Press, Toronto, Oct. 2001, S. 218

[Bria17] Briand, L.: "Analyzing Natural Language Requirements – The not too sexy but curiously difficult research that industry needs", 23rd Int. Conference on Requirements Engineering, Foundation for Software Quality, LNCS 10153, Springer, 2017, p. 131

[BuDe08] Bundschuh, M./ Dekkers, C.: The IT Measurement Compendium, Springer Verlag, Berlin, 2008

[CMPB05] Chen, Z.; Menzies,T.; Port, D.; Boehm, B.: „Finding the right Data for Software Cost Modelling", IEEE Software, Nov. 2005, p. 38

[ChGa01] Chechik, M.; Gannon, J.: „Automatic Analysis of Consistency between Requirements and Designs", IEEE Trans. on S.E., Vol. 27, Nr. 7, July 2001, p. 651.

[Cohn06] Cohn, M.: Agile Estimating and Planning, Prentice-Hall, Upper Saddle River, N.J., 2006, p. 35

[DuFo96] Dumke, R.; Foltin, E.: Softwarequalität durch Meßtools, Vieweg Verlag, Wiesbaden, 1996, p. 97-104

[GDG06] Greevy, O.; Ducasse, S.; Girba, T.: "Analyzing Software Evolution through Feature Views", Journal of Sw Maintenance&Evolution, Vol. 18, No. 6, Dec. 2006, p. 425

[IFPG99] International Function-Point Users Group, Function-Point Counting Practices, Release 4.1, IFPUG, Westerville, Ohio, 1999

[Jantz08] Jantzen, K.: "Verfahren der Aufwandsschätzung für komplexe Softwareprojekte von heute", InformatikSpektrum, Band 31, Nr. 1, 2008, p. 35

[Karn93] Karner, G.: Metrics for Objectory, Master's Thesis, University of Linköping, Sweden, Nr. Lith-IDA-Ex-9344:21, Dez. 1993, p. 21

[Lefi11] Lefingwell, D.: Agile Software Requirements, Addison-Wesley, UpperSaddle River, N.J., 2011, p. 258

[Mugr08] Mugridge, J.: „Managing Agile Project Requirements with StoryTest-driven Development", IEEE Software, Jan. 2008, p. 68

[Parn77] Parnas, D.: "The use of precise Specifications in the development of Software", Proc. of IFIP Congress-77, North-Holland, Amsterdam, 1977, p. 860

[Selb09] Selby, R.: „Analytics-driven Dashboards enable leading Indicators for Requirements and Designs of Large-Scale Systems", IEEE Software, Jan. 2009, p. 41

[Sned96] Sneed, H.: „Estimating the Development Costs of Object-oriented Software", Proc. of

7th European Software an Control Metrics Workshop, Wilmslow, GB, 1996, June 1996, p.135

[Sned05] Sneed, H.: “Reverse Engineering deutschsprachiger Fachkonzepte”, Software-Technik Trends, Vol. 25, No. 2, May, 2005, p. 45

[Sned05b] Sneed, H.: Software Projektkalkulation–Praxiserprobte Methoden der Aufwandsschätzung verschiedener Projektarten, Hanser Verlag, München, 2005, p. 41

[Sned10] Sneed, H.: „Vergleich zweier Aufwandsschäztungen nach Function-Point und UseCase Point“, DASMA Metrikon2010, Shaker Verlag, Kaisersläutern, Nov. 2010, p. 52

[SnHa05] Sneed,H.; Hasitschka,M.; Teichmann,M.-T.: Software-Produktmanagement, dpunkt Verlag, Heidelberg, 2005.

[SnPr18] Sneed,H.; Prentner, W.: ”Requirements Reengineering”, Objektspektrum, Nr. 6, Dec. 2018, p30

[ShSc97] Shepperd, M.; Schofield,C: “Estimating Software Project Effort using Analogies”, IEEE Trans. on SE, Vol. 23, No. 12, Nov. 1997, p. 736

[Ster10] Sterling, C.: Managing Software Debt – Building for inevitable Change, Addison-Wesley, Boston, 2011

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 125

Modellbasierte Methode zur Ableitung nicht-funktionaler Anforderungen im Kontext der Softwaremodernisierung

Lukas Strey[1], Christian Hein[1], Tom Ritter[1], Christian Knauer[2] und Angelika Ganz[2]

Abstract: Komplexe softwarebasierte Systeme müssen sowohl funktionale als auch nicht-funktionale Anforderungen (NFA) erfüllen, um von den Nutzern akzeptiert und als Unterstützung der Arbeitsaufgaben angenommen zu werden. Wir stellen in diesem Artikel eine modellbasierte Methode zur systematischen Ableitung nicht-funktionaler Anforderungen aus einem harmonisierten Katalog von Qualitätskriterien dar. Diese Methode wurde im Kontext der Modernisierung der Haushaltsverfahren des Bundes entwickelt und angewendet. Zunächst wurde ein Katalog von Eigenschaften erstellt, in dem generische nicht-funktionale Anforderungen aufgeführt sind. Der NFA-Katalog ist nach den Qualitätskriterien der ISO-Normenreihe 25000 strukturiert und erweiterbar. Dieser Katalog steht für die konkreten Entwicklungs- oder Modernisierungsvorhaben bspw. für die Erstellung von Ausschreibungen zur Verfügung. In einem methodischen Vorgehen wird mit Hilfe von Mustererkennung in Systemmodellen ein Softgoal-Modell aufgebaut, das als Filtermechanismus für die Auswahl der relevanten nicht-funktionalen Eigenschaften dient. Durch eine Anreicherung mit Systemkontextinformationen werden so konkrete und auf das System bezogene Anforderungen abgeleitet, welche sich für die Systementwicklung nutzen lassen.

Keywords: Nicht-funktionale Anforderungen; Modellierung; Modernisierung; Softgoal; BPMN

1 Einleitung

Bei der Entwicklung umfangreicher und verteilter IT-Systeme kommt es auf die korrekte Umsetzung der geforderten Funktionalität an. Bereits hierbei ergeben sich große Herausforderungen insbesondere in der Modellierung der funktionalen Anforderungen. Für den realen Einsatz solcher Systeme besitzen nicht-funktionale Anforderungen (NFA), welche auch als Qualitätsanforderungen bezeichnet werden, ebenfalls eine große Bedeutung. Ein funktional korrekt implementiertes System ist unbenutzbar, wenn bspw. das Antwortzeitverhalten nicht akzeptabel ist.

Im Rahmen der Modernisierung der Haushaltsverfahren des Bundes ist die Qualitätssicherung in allen Modernisierungsschritten von Anfang an fest eingeplant. Es wird ein einheitliches methodisches Vorgehen bereits in der Anforderungserhebung verfolgt und der Berücksichtigung von NFAs hohe Priorität gegeben. Das Ziel ist es, in Ausschreibungen eine hohe Abdeckung bzgl. NFAs und einen hohen Grad an Wiederverwendbarkeit für

[1] Fraunhofer FOKUS, SQC, Kaiserin-Augusta-Allee 31, 10589, Berlin, Deutschland {lukas.strey,christian.hein, tom.ritter}@fokus.fraunhofer.de
[2] Bundesministerium der Finanzen, Referat II A 6, Wilhelmstraße 97 , 10117, Berlin, Deutschland

ähnliche Vorhaben zu erreichen. Zugleich sollen Fachexperten der Haushaltsverfahren intensiv in die Modernisierung eingebunden werden. Weiterhin wird in der Modernisierung konsequent auf möglichst vollständige Modellierung gesetzt, um auch auf diese Weise ein hohes Qualitätsniveau bspw. bzgl. der Konsistenz der Anforderungen zu erreichen.

Die Spezifikation nicht-funktionaler Anforderungen ist eine zur Spezifikation funktionaler Anforderungen vergleichbar komplexe und schwierige Aufgabe. So müssen NFAs auf den unterschiedlichen Abstraktionsebenen der logischen und technischen Architektur- und Prozessmodelle jeweils adäquat definiert werden. Orientierung bieten vorhandene Qualitätsnormen und -standards, die Qualitätskriterien in Form von Modellen beschreiben. Für ein konkretes System muss betrachtet werden, welches Qualitätskriterium in welcher Ausprägung für welches Element der funktionalen Architektur relevant ist, um dafür konkrete NFAs zu formulieren.

In der praktischen Anwendung ergeben sich dafür Erfahrungswerte und es lassen sich Muster erkennen, woraus wir eine Methode entwickelt und diese im Kontext der Modernisierung der IT-Systeme der Haushaltsverfahren angewendet haben. Mit dieser Methode lassen sich systematisch Qualitätskriterien für die Bestandteile der Architektur- und Prozessmodelle finden und daraus auf Basis eines umfassenden Katalogs nicht-funktionale Anforderungen ausformulieren. Zugleich ist diese Methode leichtgewichtig und lässt sich mit anderen Ansätzen kombinieren.

Überblicksartig stellen wir in diesem Beitrag diese modellbasierte Methode zur Ableitung nicht-funktionaler Anforderungen im Kontext der Softwaremodernisierung vor. Die Methode umfasst **(1)** den Aufbau und Pflege eines Qualitätskriterienkatalogs, in dem generische formulierte NFAs vorliegen, **(2)** der Aufbau eines Softgoal-Modells als essentieller Filtermechanismus, **(3)** die systematische Untersuchung von Systemmodellen bzgl. der Relevanz von NFAs mithilfe des Softgoal-Modells und **(4)** die Ableitung einer strukturierten Menge konkreter NFAs für ein gegebenes Systemmodell, mit der Systemanforderungen bspw. für Ausschreibungen eindeutig spezifiziert sind. Die Methode stellt eine Unterstützung der Fach- und Softwareexperten dar und wird als Ergänzung zu anderen etablierten Verfahren benutzt. So lassen sich die Zwischenergebnisse bspw. in Anforderungsanalyse- und Brainstorming-Phasen benutzen bzw. verfeinern.

2 Der NFA-Katalog

Das in diesem Artikel beschriebene Vorgehen stützt sich auf die Verwendung eines NFA-Katalogs, der einen Wissensspeicher von vordefinierten bzw. zu verfeinernden NFAs darstellt, die jeweils bestimmten Qualitätskriterien zugeordnet sind. Der NFA-Katalog enthält in allgemeiner Form generisch beschriebene NFAs, die für ein konkretes System mit Kontextinformationen angereichert werden. Als Grundlage für die Struktur des Katalogs wurde die ISO-Normenreihe 250xx herangezogen. Dennoch ist die Struktur flexibel erweiterbar und kann mit neuem Wissen iterativ angereichert werden.

2.1 ISO-Normenreihe 250xx

Die ISO-Norm 25000 bildet den Einstieg in die Normenreihe für "Qualitätskriterien und Bewertung von Softwareprodukten (SQuaRE)"[IS14]. ISO-Norm 25010 beschreibt die Qualitätsmodelle für *Product Quality* (Produktqualität) und *Quality in Use* (Anwendungsqualität) [IS11]. Außerdem beschreibt die ISO-Norm 25012 ein Modell für *Data Quality* (Datenqualität) [IS08]. Diese Qualitätsmodelle nennen einzelne Charakteristika, welche den jeweiligen Qualitätsaspekt des Modells betreffen. Für *Product Quality* und *Quality in Use* ist jedes Charakteristikum in einzelne Sub-Charakteristika unterteilt. Zusammengenommen beschreiben die Charakteristika der drei Modelle umfassend die Qualität eines Softwareprodukts. ISO-Normen liefern außerdem Messgrößen für quantifizierbare Charakteristika der drei Qualitätsmodelle, ISO-Norm 25022 für *Quality in Use* [IS16a], Norm 25023 für *Product Quality* [IS16b] und Norm 25024 für *Data Quality* [IS15].

2.2 Vorgehen zum Aufbau des Katalogs

Abbildung 1 zeigt die Struktur des NFA-Katalogs. Erkennbar ist die Unterteilung entsprechend der drei Qualitätsmodelle in einzelne Teilbereiche, sowie die Struktur aus Charakteristika und Sub-Charakteristika. Die vollständige Auflistung der Charakteristika ist in den entsprechenden ISO-Normen zu finden. Als Grundlage bei der Formulierung einer

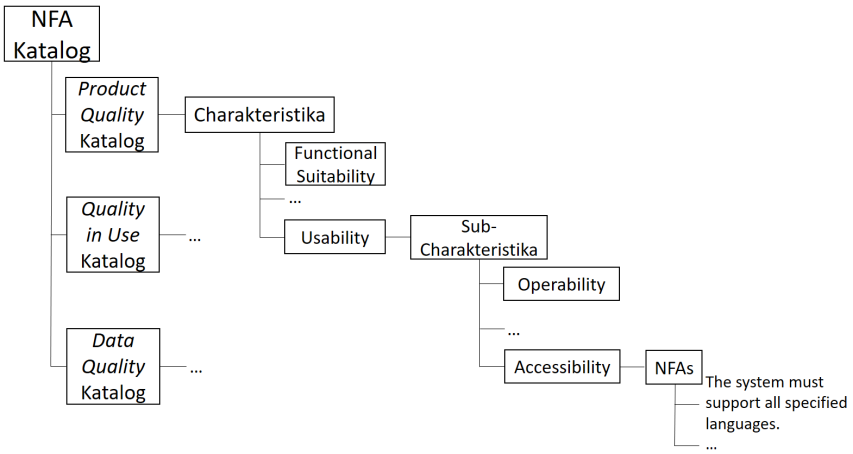


Abb. 1: Struktur des NFA-Katalogs entsprechend der ISO-Normenreihe 25000

ersten Fassung generischer NFAs dienen die genannten Messgrößen. Aus jeder Messgröße wird mindestens eine NFA abgeleitet, welche die Anforderung bzgl. der Charakteristik quantifiziert beschreibt. Die so gewonnenen NFAs werden entsprechend in die Struktur des NFA-Katalogs eingeordnet. In der somit erstellten Version des Katalogs sind 191 generische

NFAs enthalten. In der ersten Fassung des Katalogs sind die Begriffe der englischen ISO-Normen übernommen und zur Wahrung der Konsistenz die NFAs ebenfalls auf Englisch formuliert. Für den Einsatz im Bundesministerium ist eine deutsche Übersetzung geplant, um eine Einbindung der Fachexpertise zu ermöglichen.

Jede generische NFA im Katalog bezieht sich auf ein Subjekt wie System, Service oder Database. Außerdem kann mithilfe des Verbs die Priorität definiert werden (can, shall, must). Eine Beispiel-NFA aus dem Katalog lautet: „The system must support all specified languages“ (Product Quality - Usability - Accessibility); wobei sowohl Subjekt als auch Verb durch den Anwender kontextspezifisch definiert werden, bspw.: „The user interface shall support all specified languages“. Weiterhin muss der Anwender in einigen quantifizierbaren NFAs konkrete Werte festlegen: „The system must have a test coverage of at least [X]%“ (Product Quality - Reliability - Maturity).

2.3 Erweiterbarkeit des Katalogs

Der initiale Katalog ist in zweierlei Hinsicht erweiterbar. Neue NFAs können in die bestehende Struktur eingeordnet werden, entsprechend der Funktionalität des Katalogs als Wissensspeicher. Dieser kontinuierliche Aufbau des Katalogs muss planvoll durchgeführt werden, so sollte vor dem Hinzufügen neuer NFAs eine Dublettenprüfung stattfinden und der Detailgrad im Katalog konstant gehalten werden. Außerdem kann die Struktur des Katalogs erweitert werden, indem neue Unterkataloge mit entsprechenden Charakteristika und Sub-Charakteristika definiert werden. Hierdurch lassen sich wiederkehrende, projektspezifische Kontexte abbilden. Bei der Modernisierung der Haushaltsverfahren ist dies für spezielle Reglementierungen relevant.

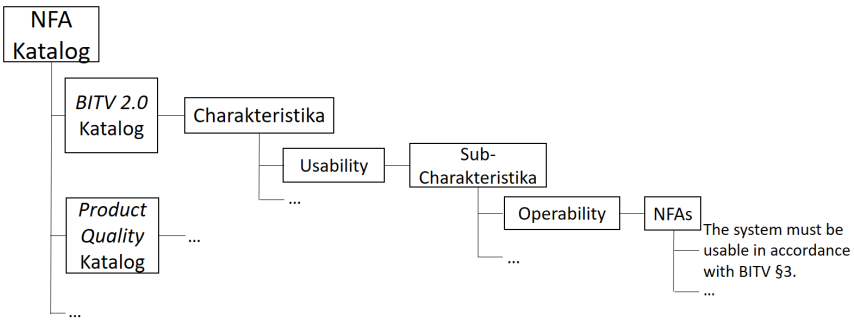


Abb. 2: Erweiterung des NFA-Katalogs um BITV 2.0

Die erste durchgeführte Erweiterung des Katalogs ist die Hinzunahme der Barrierefreie-Informationstechnik-Verordnung (BITV 2.0) [BIT11]. Die BITV 2.0 regelt die Barrierefreiheit, unter anderem für elektronische Dienstleistungen öffentlicher Stellen und elektronische Verwaltungsabläufe, in Deutschland. Aus der Analyse der BITV 2.0 gewonnene NFAs sind,

aufgrund des speziellen Detailgrads, in einem eigenen BITV 2.0 Katalog gesammelt. Abbildung 2 verdeutlicht die Erweiterung des NFA-Katalogs um einen speziellen Unterkatalog für BITV 2.0 NFAs. Einschließlich dieser Erweiterung umfasst die aktuell genutzte Version des Katalogs insgesamt 207 generische NFAs.

3 Modellbasierte Methode zur Anwendung des NFA-Katalogs

Der nächste Schritt ist die Abbildung des Katalogs in einem gegebenen Kontext, um konkrete NFAs für diesen Kontext zu erzeugen. In Anbetracht der geplanten kontinuierlichen Erweiterung des Katalogs ist ein Methode zur Filterung notwendig. Dabei wird der Gesamtsatz an NFAs im Katalog sinnvoll gefiltert und dem Nutzer zur Auswahl angeboten. Die Filterung berücksichtigt die Struktur des Katalogs und erlaubt eine grobe Vorauswahl nach relevanten Sub-Charakteristika, wodurch die Anzahl der zu betrachteten NFAs reduziert wird. Als Input für die Filterung des Katalogs können sowohl Architektur- als auch Prozessmodelle verwendet werden. Besonders bei der Automatisierung von Abläufen und der gemeinsamen Ausrichtung von Geschäftsprozessen und IT ist der Input einer Prozessbeschreibung relevant. Die Beschreibung und Darstellung der Filterung erfolgt mithilfe von Softgoal-Modellierung.

3.1 Softgoal-Modellierung

Softgoal Interdependency Graphen (SIG) dienen zur Modellierung von Softgoals, also Zielen welche nicht eindeutig erfüllbar sind. SIG zeigen Softgoals mit ihren gegenseitigen

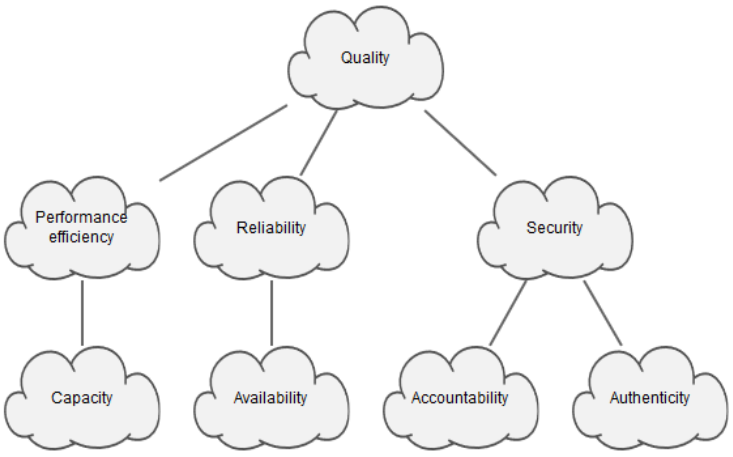


Abb. 3: Beispiel eines Softgoal-Baums

Einflüssen an, bspw. ob ein Softgoal andere Softgoals behindert oder unterstützt [CL09]. Bei der Anwendung des NFA-Katalogs werden Softgoals zur Modellierung der Filterkriterien verwendet. Dafür wird eine Baumstruktur aus Softgoals gebildet. Dieser Softgoal-Baum bildet auf seiner untersten Ebene die Sub-Charakteristika ab, welche im Zuge einer methodischen Vorgehensweise als relevant identifiziert werden. Die Sub-Charakteristika sind die Filterkriterien für den Katalog. Auf der obersten Ebene des Softgoal-Baums steht das Softgoal "Quality", welchem die ISO-Charakteristika entsprechend ihrer Struktur untergeordnet sind. Abbildung 3 zeigt ein Beispiel einer solchen Baumstruktur. Basierend auf dem finalen Softgoal-Baum können NFAs aus dem dadurch gefilterten Katalog ausgewählt werden. Die Modellierung anhand des Softgoal-Baums erlaubt es, die Einflüsse der aus

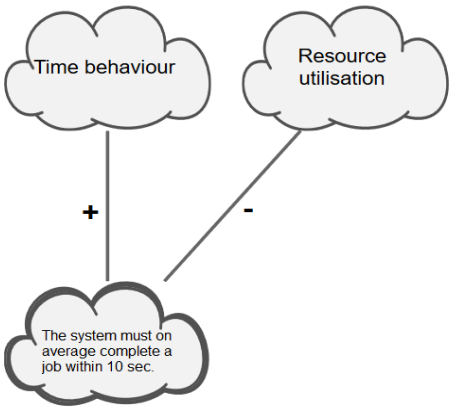


Abb. 4: Einflüsse einer NFA auf die Softgoals

dem Katalog gewählten NFAs auf die Softgoals darzustellen. Somit können potenziell negative Einflüsse zwischen Elementen des Softgoal-Baums modelliert werden, falls die Erfüllung eines ISO Sub-Charakteristikums ein anderes negativ beeinflusst. Dies ist in Abbildung 4 beispielhaft dargestellt, welche zwei Softgoals auf der untersten Ebene der Baumstruktur mit einer aus dem Katalog ausgewählten NFA zeigt. Diese gegenseitigen Einflüsse werden aktuell manuell modelliert, geplant ist allerdings diese zukünftig im NFA-Katalog zu hinterlegen.

3.2 BPMN-NFA Methode

Nachfolgend wird die BPMN-NFA Methode zur Erhebung nicht-funktionaler Anforderungen unter Verwendung von Prozessmodellen vorgestellt. Geläufig wird zum Modellieren von Geschäftsprozessen die „Business Process Model and Notation“ (BPMN) der Object Management Group (OMG) verwendet [OM11]. Unter anderem die REMO Methode von Vieira et al. benutzt BPMN-Diagramme als Input zur Anforderungserhebung [Vi12b]. NFAs werden in REMO, mithilfe von Brainstorming, basierend auf Aktivitäten und Message-Events erhoben [Vi12a]. Die hier vorgestellte Methode benutzt hingegen im

BPMN-Diagramm vorhandene Elemente und Muster zum Aufbau des Softgoal-Baums und für die anschließende Filterung des NFA-Katalogs. Abbildung 5 stellt den Ablauf der Methode dar.

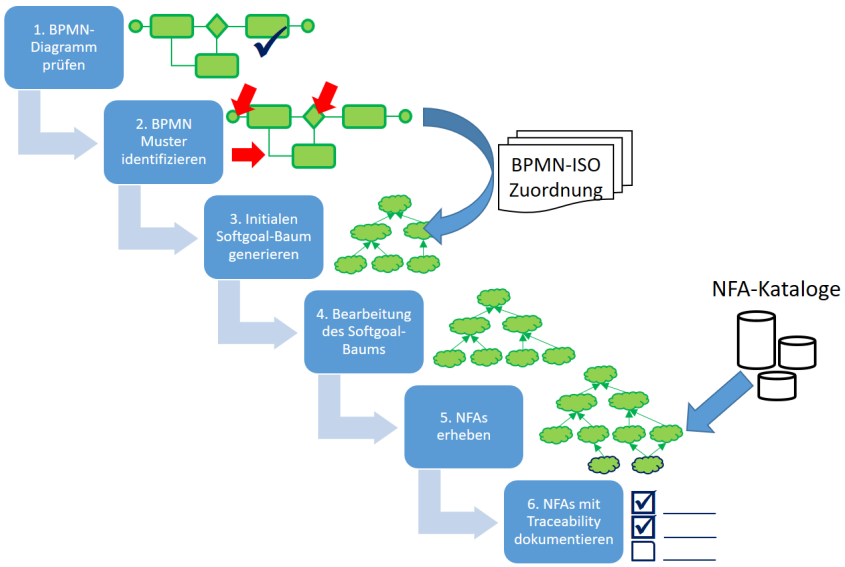


Abb. 5: Vorgehensweise der BPMN-NFA Methode

Zusammengefasst besteht die Methode aus den sechs aufeinanderfolgenden Arbeitsschritten:

1. Prüfung des BPMN-Diagramms
2. Identifikation vorhandener BPMN-Elemente/-Muster
3. Aufstellen des initialen Softgoal-Baums
4. Bearbeitung des Softgoal-Baums
5. Formulierung der NFAs
6. Dokumentation der NFAs

Grundvoraussetzung ist ein inhaltlich korrektes sowie aktuelles BPMN-Diagramm, was im ersten Arbeitsschritt überprüft wird. Daraufhin werden im Diagramm vordefinierte BPMN-Muster und -Elemente identifiziert und mit Stereotypen markiert. Jedem Muster/Element sind ausgewählte ISO Sub-Charakteristika zugeordnet. Diese allgemeine Zuordnung ist ein Grundbestandteil der BPMN-NFA Methode und für die erfolgreiche Durchführung der NFA-Erhebung von besonderer Bedeutung. Entsprechend wird die Zuordnung von BPMN zu ISO im nachfolgenden Unterkapitel detailliert erläutert. Mithilfe der fest definierten

Zuordnung wird automatisiert ein initialer Softgoal-Baum generiert. Bei der Verwendung einer Werkzeugumsetzung der Methode ist der dritte Arbeitsschritt somit ohne manuellen Aufwand verbunden.

Nach der automatisierten Generierung wird der Softgoal-Baum durch einen Fachexperten unter Nutzung der vorhandenen Kontextinformationen bearbeitet. Hierbei können einzelne Softgoals hinzugefügt oder entfernt werden. Die manuelle Bearbeitung ist Arbeitsschritt 4 der Methode, entsprechend des Ablaufs in Abbildung 5. Die automatische Generierung des initialen Softgoal-Baums begründet sich in allgemeinen Annahmen und ist somit notwendigerweise nicht für jeden Projektkontext vollständig übertragbar. Der manuelle Bearbeitungsschritt ermöglicht eine Umsetzung kontextspezifischer Aspekte. Es werden einerseits im initialen Softgoal-Baum vorhandene Elemente begründet verworfen, falls diese im konkreten Anwendungsfall keine Relevanz besitzen. Andererseits werden manuell Softgoals hinzugefügt, deren Notwendigkeit sich bspw. aus weiterführenden Projektinformationen ergeben kann, welche nicht konkret in BPMN modelliert sind.

Daraufhin wird mithilfe des Softgoal-Baums der NFA-Katalog nach potenziell relevanten NFAs gefiltert. Die finale Auswahl der NFAs aus dem Katalog erfolgt unter Anwendung der vorhandenen Kontextinformationen, also dem System- und Expertenwissen, welches zum Prozess vorliegt. Abschließend müssen die aus dem Katalog gewählten NFAs dokumentiert werden. Aufgrund des modellbasierten Ansatzes liegen hierfür ausführliche Informationen zur Nachverfolgbarkeit vor.

3.3 BPMN Muster

Die Generierung des initialen Softgoal-Baums basiert auf der Identifizierung von vordefinierten Mustern im Input. Jedem spezifizierten Muster sind potentiell relevante NFA-Kategorien zugeordnet. Wird während der Analyse ein bestimmtes Muster im Input gefunden, werden die entsprechenden Kategorien in den Softgoal-Baum eingefügt. Die Workflow Pattern nach

Workflow Muster	**Bedeutung in BPMN**
Parallel Split	Aufteilung des Sequence Flows in mehrere parallele Flows
Synchronisation	Zusammenführen paralleler Sequence Flows zu einem einzelnen Flow
Exclusive Choice	Auswahl einer Verzweigung im Sequence Flow aus mehreren Optionen
Multiple Choice	Selektive Aufteilung des Sequence Flows in parallele Flows
Discriminator	Zusammenführen mehrerer Sequence Flows, wobei der zuerst abgeschlossene Flow den Prozess weiterführt und nachfolgende ignoriert werden
Arbitrary Cycles	Sich zyklisch wiederholende Aktivitäten mit Verzweigungsmöglichkeiten
Multiple Instances	Mehrfache, gleichzeitige Ausführung einer Aktivität
Deferred Choice	Auswahl einer Verzweigung basierend auf Informationen zur Laufzeit
Cancel Activity	Abbruch einer Aktivität bei Eintritt eines bestimmten Umstands
Cancel Case	Abbruch eines kompletten Prozesses bei Eintritt eines bestimmten Umstands

Tab. 1: Beschreibung der Workflow Pattern

van der Aalst sind eine Sammlung von Mustern, welche in Ablaufdiagrammen vorkommen können [Aa03]. In Tabelle 1 sind die nachfolgend verwendeten Workflow Muster beschrieben. Wohed et al. analysierten, inwieweit die Workflow Pattern in BPMN abbildbar sind [Wo06].

Unter Anbetracht dieser Analyse werden die Muster, sowie die einzelnen BPMN-Elemente, auf ihre Relevanz für die Erhebung von NFAs untersucht. Die dabei erstellte Zuordnung basiert auf den ISO-Messgrößen der Qualitätsmodelle. In den ISO-Normen sind für jedes einzelne Sub-Charakteristika Messgrößen definiert. Zur Generierung einer Zuordnung wird

Workflow Muster	**ISO Sub-Charakteristika**
Parallel Split	Capacity; Co-existence; Modularity
Synchronisation	Resource utilisation; Interoperability
Exclusive Choice	Maturity; Modularity
Multiple Choice	Capacity; Co-existence; Maturity; Modularity
Discriminator	Resource utilisation; Interoperability; Fault tolerance
Arbitrary Cycles	Resource utilisation; Maturity
Multiple Instances	Time behaviour; Resource utilisation; Capacity
Deferred Choice	Time behaviour; Maturity; Modularity
Cancel Activity	User error protection; Maturity; Recoverability
Cancel Case	Maturity; Recoverability; Accountability; Analysability

Tab. 2: Relevanz von Product Quality Sub-Charakteristika für Workflow Pattern

für alle Workflow Pattern untersucht, ob durch eine Implementierung des Musters theoretisch Einfluss auf die jeweilige Messgröße ausgeübt wird. So konnten für einige Workflow Pattern jeweils zwei bis vier potentiell relevante ISO Sub-Charakteristika identifiziert werden.

Diese Zuordnung wurde in mehreren Anwendungsbeispielen überprüft und verfeinert. Für jeden Anwendungsfall wurde die BPMN-NFA Methode durchgeführt. Falls bei der Anwendung in Arbeitsschritt 4 weitgehende manuelle Änderungen notwendig waren, so war der initial generierte Softgoal-Baum potentiell nicht ausreichend. Die notwendigen Änderungen der verschiedenen Anwendungsfälle wurden zusammengetragen und verglichen, wodurch Rückschlüsse auf die Qualität der initialen Generierung entstanden. Hierdurch konnten notwendige Anpassungen an der Zuordnung identifiziert werden, wodurch diese iterativ verbessert wurde. In Tabelle 2 ist die Zuordnung für das Qualitätsmodell der *Product Quality* aufgeführt.

Außerdem werden bei der Generierung des initialen Softgoal-Baums die im BPMN-Diagramm vorhandenen Elemente betrachtet. zur Muehlen und Recker kommen in ihrer Analyse des BPMN-Gebrauchs zu der Erkenntnis, dass nur 20% der BPMN-Elemente regelmäßig in Diagrammen verwendet werden [zMR08]. Entsprechend wird für eine Auswahl relevanter BPMN-Elemente ebenfalls eine Zuordnung zu den ISO Sub-Charakteristika vorgenommen. Dabei wird das gleiche Vorgehen wie bei den Workflow Pattern verwendet. Die Relevanz eines Sub-Charakteristika ist mithilfe der ISO-Messgrößen abgeschätzt worden und jedem BPMN-Element wurden zwei bis fünf Sub-Charakteristika zugeordnet. Die

BPMN-Element	ISO Sub-Charakteristika
Start-Event	Capacity; Availability; Accountability; Authenticity
Error Event Definition	Fault tolerance; Recoverability; Integrity; Accountability; Analysability
Message Event Definition & Receive Task	Capacity; Interoperability; Availability; Confidentiality
User Task	Time behaviour; Appropriateness recognizability; Learnability; Operability; User error protection; User interface aesthetics; Accessibility
Service Task, Script Task & Business Rule Task	Time behaviour; Resource utilisation; Maturity; Availability
Sub Process	Interoperability; Availability; Integrity; Modularity
Call Activity	Time behaviour; Co-existence; Interoperability; Availability; Modularity
Message Flow	Resource utilisation; Interoperability; Confidentiality; Integrity
Data Object	Resource utilisation; Confidentiality; Integrity
Data Store	Resource utilisation; Interoperability; Availability; Confidentiality; Integrity
Lane (Akteure)	Capacity; Co-existence; Confidentiality; Authenticity
Sequence Flow mit Lane Wechsel	Interoperability; Availability; Confidentiality

Tab. 3: Relevanz von Product Quality Sub-Charakteristika für BPMN-Elemente

Ausnahme bildet hierbei das Element *User Task*, welchem aufgrund der unterschiedlichen Usability-Aspekte insgesamt sieben Sub-Charakteristika zugeordnet sind. Die Zuordnung wurde ebenfalls anhand der Anwendungsbeispiele überprüft und ist in Tabelle 3 aufgelistet.

3.4 Werkzeugunterstützung

Die BPMN-NFA Methode lässt sich grundsätzlich mit gängigen Modellierungswerkzeugen umsetzen. Viele BPMN- und UML-basierte Werkzeuge bieten die Möglichkeit zur Erweiterung der Modellierungskonzepte. So lassen sich Softgoal-Bäume bspw. auch mithilfe eines UML-Profils darstellen. Diese Nutzung eines UML-Profils ist allerdings im Anwendungskontext als weniger intuitiv empfunden worden als ein spezialisiertes Werkzeug.

Wir haben daher die beschriebene BPMN-NFA Methode in einer Werkzeugkette basierend auf den Bestandteilen ModelBus [Fra19] und ModelICE [Ing19] realisiert. Die Werkzeugkette ermöglicht eine Bearbeitung sämtlicher Modelle im Webbrowser und über Adaptoren sind gängige Modellierungswerkzeuge und -bibliotheken (wie bspw. Enterprise Architect, Eclipse Modelling, Papyrus, Diagram.js) integriert und anwendbar. Im Zuge des modellbasierten Vorgehens sind Informationen zur Nachverfolgbarkeit (Traceability) besonders zu berücksichtigten. Eine Nachverfolgbarkeit muss über Modellgrenzen hin-

weg erfolgen, weil Abhängigkeiten zwischen den Prozessbeschreibungen (in BPMN), den Architekturbeschreibungen (vorwiegend in SysML) und den Softgoal-Bäumen bestehen.

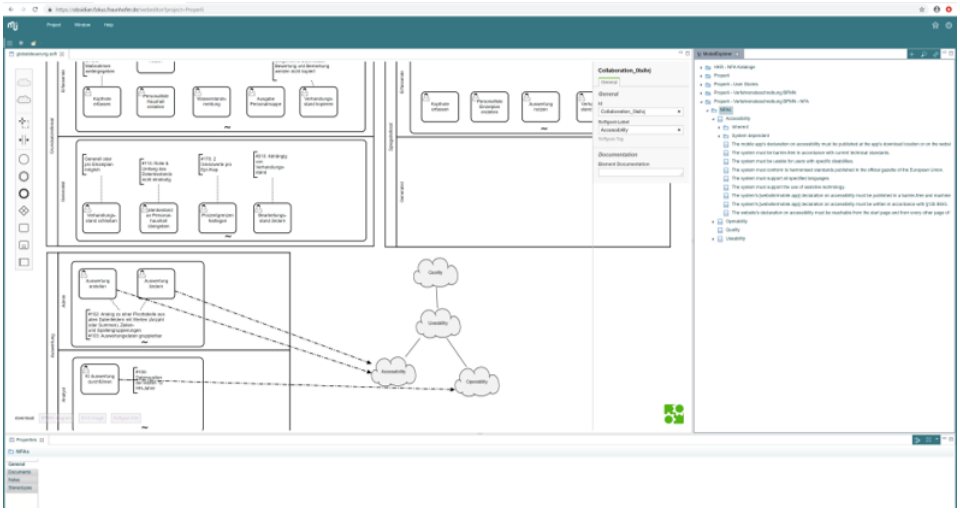


Abb. 6: Bearbeitung von BPMN und Softgoals mit ModelICE

Für die Umsetzung der Methode haben wir die Beschreibung der Prozesse und deren Beziehung zu Softgoal-Bäumen mittels einer Erweiterung von bpmn-js realisiert und in das Werkzeug ModelICE integriert. In Abbildung 6 ist die Umsetzung dargestellt. Zusätzlich zu der Darstellung der Softgoal-Bäume haben wir eine Erweiterung für ModelICE entwickelt, welche den NFA-Katalog mithilfe der Softgoals nach konkreten NFAs filtert. Die Verwendung von ModelICE erlaubt es, diese Funktionalität als Dienst in der Werkzeugkette zur Verfügung zu stellen und so bspw. im Enterprise Architect zu nutzen.

4 Anwendung der BPMN-NFA Methode

Die entwickelte BPMN-NFA Methode wird im Kontext der Modernisierung der Haushaltsverfahren des Bundes zur Erhebung von NFAs eingesetzt. Hierfür wird die beschriebene Werkzeugkette genutzt, um die Erkennung vorhandener BPMN-Elemente und -Muster im Diagramm zu unterstützen und teilautomatisiert durchzuführen. Nachfolgend ist die Anwendung der BPMN-NFA Methode auf den Beispielfall ProPerLi beschrieben. Es werden BPMN-Modelle des zu implementierenden Prozesses und die im vorherigen Kapitel beschriebene Zuordnung von BPMN zu den *Product Quality* Sub-Charakteristika genutzt.

4.1 ProPerLi Anwendungsfall

Mit ProPerLi (Produktion Personallisten) soll ein Softwaresystem zur Unterstützung der Aufstellung des Personalhaushalts des Bundes (Planstellen der Beamtinnen und Beamten, Stellen der Tarifbeschäftigten) geschaffen werden. Ziel ist es, ein durchgehendes, IT-gestütztes Verfahren zu entwickeln, dass den gesamten Prozess umfassen soll. Es sind die Prozessabläufe von der Erfassung aller Personalforderungen über die Beurteilung dieser Forderungen, die Darstellung von Verhandlungsergebnissen innerhalb der Bundesregierung und von Änderungen aus dem parlamentarischen Verfahren bis zur Produktion der Personallisten und zur Übergabe des Datenbestandes in eine bestehende Datenbank umzusetzen. Der Prozess wird aktuell manuell, unterstützt von Office-Software, durchgeführt und ist von Medienbrüchen geprägt und äußerst prüfungsintensiv, weil keine automatisierte Fehler- und Plausibilitätsprüfung existiert. Die Abläufe des Verfahrens liegen strukturiert als BPMN-Diagramme vor.

Mithilfe der modellierten BPMN-Diagramme werden die NFAs für ProPerLi erhoben. Die einzelnen Ablaufschritte der BPMN-NFA Methode sind in Abbildung 5 dargelegt. Die Erhebung der NFAs erfolgt durch Nutzung der NFA-Kataloge. Insgesamt wurden mithilfe der BPMN-NFA Methode 29 prozessbezogene NFAs aus dem *Product Quality* Katalog für ProPerLi erhoben, welche von einem Prozessexperten als korrekt eingestuft wurden. Dafür wurde ein Softgoal-Baum mit insgesamt 18 Sub-Charakteristika genutzt. Einige NFAs sind beispielhaft in Tabelle 4 aufgelistet.

ID	Sub-Charakteristika	NFA
1	Capacity	The system must allow at least 5 users to use the system simultaneously.
14	Accountability	The system must audit all user access to the system or data.
18	Availability	The database must be available for 95% of the scheduled system operational time.
22	Operability	The system must support all appropriate input devices.
29	Analyseability	The system must implement all specified diagnosis functions.

Tab. 4: Beispiel-NFAs für ProPerLi aus dem *Product Quality* Katalog

4.2 Bewertung der Methode

Die vorgestellte BPMN-NFA Methode und allgemein die Verwendung eines NFA-Katalogs zur Anforderungserhebung werden nachfolgend bewertet. Als Vergleich dienen die Ergebnisse eines Vorläuferprojekts von ProPerLi, in welchem NFAs für den Prozess der Personallisten durch eine externe Beratergesellschaft erhoben wurden. Hierbei erstellte die Beratergesellschaft eine NFA-Sammlung basierend auf bestehenden, unstrukturiert erhobenen Originalanforderungen im Kassen- und Rechnungswesen des Bundes. Diese

NFA-Sammlung war für die Anwendung im Kontext der Modernisierung der Haushaltsverfahren des Bundes vorbereitet und nach den Kategorien der ISO-Norm 9126 strukturiert. Die NFA-Sammlung der Beratergesellschaft wird nachfolgend als Vergleichs-NFAs bezeichnet.

In dieser Arbeit wurde ein strukturiert aufgebauter NFA-Katalog präsentiert, basierend auf den Messgrößen der ISO-Normenreihe 250xx. Der Katalog schließt inhaltliche Lücken, welche bei den Vergleichs-NFAs bestanden. Weiterhin erfolgte die Erstellung des NFA-Katalogs mithilfe eines geringeren Ressourcenaufwands als bei den Vergleichs-NFAs. Somit erfüllt der NFA-Katalog die Anforderungen, welche seitens der Prozessbeteiligten an eine NFA-Sammlung gestellt und durch die externe Beratergesellschaft nicht ausreichend abgedeckt wurden.

Die strukturierte Anwendung, mittels Softgoal-Modellierung basierend auf BPMN, ermöglichte im ProPerLi-Prozess eine für alle Stakeholder nachvollziehbare Erhebung von NFAs. Durch die Strukturierung des Vorgehens ist eine Abdeckung des Problemraums, konkret der Prozessbeschreibungen, sichergestellt. Außerdem ermöglicht die Implementierung der Methode eine Teilautomatisierung der Anwendung. Neben der erweiterten Abdeckung des Problemraums stellt die Methode mithilfe der BPMN-Modellierung ein Verständnis der Prozessabläufe bei allen Stakeholdern sicher.

Schlussendlich soll die vorgestellte Methode quantitativ bewertet werden. Die Vergleichs-NFAs umfassen 19 als allgemeingültig definierte NFAs. Ausgehend von der Kategorisierung nach ISO-Norm 9126 bietet sich ein Vergleich mit dem *Product Quality* Unterkatalog an. Dieser beinhaltet insgesamt 91 generische NFAs auf einer detaillierteren Abstraktionsstufe, wovon mithilfe der BPMN-NFA Methode 29 für den ProPerLi-Prozess erhoben wurden. Im Zuge eines inhaltlichen Vergleichs ist erkennbar, dass die Anforderungen des *Product Quality*-Katalogs konkreter sind, als die Vergleichs-NFAs. Insgesamt 17 der 19 Vergleichs-NFAs werden durch die 29 erhobenen ProPerLi-NFAs abgedeckt und durch diese detaillierter beschrieben. Die beiden nicht abgedeckten Vergleichs-NFAs behandeln die Anonymisierung von Testdaten und das Wartungskonzept. Diese beiden Aspekte werden von der Prozesssicht nicht abgedeckt, entsprechend könnte die Erhebung um bspw. eine Architekturanalyse erweitert werden, um den Systemkontext auf weitere fehlende NFAs zu überprüfen. Ein Rückvergleich unter Betrachtung der Kategorien der ISO-Norm 9126 ergibt, dass sieben BPMN-NFAs in den Vergleichs-NFAs überhaupt nicht und fünf weitere nicht ausreichend abgebildet sind. Somit sind 24% der ermittelten BPMN-NFAs komplett neu für den Projektkontext erhoben und insgesamt 41% stellen eine Erweiterung der vormals vorhandenen NFAs dar.

5 Zusammenfassung

Mit Hilfe des erweiterbaren Katalogs der Qualitätskriterien und dem Vorgehen zum Aufbau des Softgoal-Baums lassen sich für Architektur- und Prozessmodelle konkrete NFAs ableiten. In diesem Beitrag wurde die Anwendung der Methode für Prozessmodelle

dargestellt und untersucht. Aufgrund des systematischen Vorgehens wird ein umfassender und zugleich passgenauer Satz von NFAs erstellt. Das von uns implementierte webbasierte Werkzeug zu Erstellung von BPMN und Softgoal-Modellen ermöglicht ein interaktives teamübergreifendes Arbeiten, was insbesondere in den Schritten der Überarbeitung der Modelle mithilfe von Experten sehr hilfreich ist.

In der bisherigen Anwendung der Methode konnten vielversprechende Ergebnisse erreicht werden. Insbesondere, wenn mit der Erhebung von nicht-funktionalen Anforderungen komplett neu begonnen wird, lässt sich relativ leichtgewichtig ein umfassender Entwurf von NFAs für die verschiedenen Teile eines Systems erzeugen, um so frühzeitig in die Diskussion mit den Stakeholdern zu kommen. Durch die strukturierte Zuordnung der erzeugten NFAs zu den im Katalog benutzten ISO-Charakteristiken erleichtert sich auch die weitere Verarbeitung der Anforderungen in Ausschreibungen bzw. dann in der technischen Umsetzung.

5.1 Related Work

Die Wiederverwendung bekannter Anforderungen in neuen Softwareprojekten kann die Effizienz der Anforderungserhebung erhöhen. Toval et al. schlagen hierfür die SIREN Methode vor. SIREN verwendet ein spiralförmiges Vorgehen, bei dem Anforderungskataloge genutzt werden, bspw. ein Katalog für Security Profile [To08]. Das NFR Framework stellt Kataloge in Form von Softgoal Interdependency Graphen (SIG) zur Verfügung [CL09]. Cysneiros liefert weitere Kataloge, ebenfalls in SIG abgebildet, und das i*-System zur systematischen Nutzung von Katalogen [Cy07]. In der Literatur genannt sind unter anderem SIG für Usability [CWK05], Performance [Yu03] und Invisibility [Ma18]. Der in diesem Artikel beschriebene NFA-Katalog zeichnet sich durch eine Strukturierung basierend auf ISO-Normen und eine unkomplizierte Erweiterbarkeit aus.

Eine strukturierte Erhebung von NFAs ermöglicht bspw. die SENoR Methode, welche NFA-Vorlagen basierend auf den Charakteristika der ISO-Norm 25010 nutzt. Dieser Katalog an Vorlagen wird während der Methode in Brainstorming-Sitzungen verwendet, um die Stakeholder bei der NFA-Erhebung zu unterstützen [KN14]. Das PABRE-System liefert eine Sammlung an Anforderungsmustern für einige der ISO-Norm 25010 Charakteristika [PQF14]. Die BPMN-NFA Methode erhebt strukturiert NFAs unter Verwendung des vorgestellten NFA-Katalogs. Ein gleichsam strukturiertes Vorgehen ist auch für Architekturmodelle möglich. Mithilfe teilautomatisierter Arbeitsschritte wird manueller Aufwand abgenommen. Diese Automatisierung kann im Rahmen der Werkzeugunterstützung noch erweitert werden.

5.2 Ausblick

Die ersten Ergebnissen der bisherigen Anwendungen der Methode im Zuge der Modernisierung der Haushaltsverfahren und speziell im Submodul ProPerLi (Produktion Personallisten) liegen inzwischen vor. Die dabei gewonnenen Erkenntnisse werden zur Erweiterung des Katalogs und zur Verbesserung der Methode genutzt. Aktuell werden die Muster teilweise manuell markiert, daher ist die weitgehende Automatisierung der Mustererkennung durch verbesserte Werkzeugunterstützung unser nächster Schritt. Außerdem soll die Methode zukünftig in unterschiedlichen Anwendungsbereichen eingesetzt werden, wodurch die aktuell benutzten Muster überprüft und weiter verfeinert werden können. Ergänzend zur Anwendung im Submodul ProPerLi wird die Methode zukünftig auch bei der Modernisierung weiterer Haushaltsverfahren des Bundes eingesetzt.

Literaturverzeichnis

[Aa03] van der Aalst, W.M.P.; ter Hofstede, A.H.M.; Kiepuszewski, B.; Barros, A.P.: Workflow Patterns. Distributed and Parallel Databases, S. 5–51, 2003.

[BIT11] Barrierefreie-Informationstechnik-Verordnung - BITV 2.0, 09 2011.

[CL09] Chung, L.; Leite, Julio: On Non-Functional Requirements in Software Engineering. S. 363–379, 01 2009.

[CWK05] Cysneiros, Luiz; Werneck, Vera; Kushniruk, Andre: Reusable knowledge for satisficing usability requirements. S. 463– 464, 01 2005.

[Cy07] Cysneiros, Luiz Marcio: Evaluating the Effectiveness of Using Catalogues to Elicit Non-Functional Requirements. 2007.

[Fra19] Fraunhofer FOKUS: ModelBus, https://www.modelbus.org, 2019. Online, zuletzt aufgerufen: 13.10.2019.

[Ing19] Ingrano Solutions: ModelICE, https://www.modelice.io, 2019. Online, zuletzt aufgerufen: 13.10.2019.

[IS08] ISO: ISO/IEC 25012:2008 Data quality model. International Organization for Standardization, 2008.

[IS11] ISO: ISO/IEC 25010:2011 System and software quality models. International Organization for Standardization, 2011.

[IS14] ISO: ISO/IEC 25000:2014 Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — Guide to SQuaRE. International Organization for Standardization, 2014.

[IS15] ISO: ISO/IEC 25024:2015 Measurement of data quality. International Organization for Standardization, 2015.

[IS16a] ISO: ISO/IEC 25022:2016 Measurement of quality in use. International Organization for Standardization, 2016.

[IS16b] ISO: ISO/IEC 25023:2016 Measurement of system and software product quality. International Organization for Standardization, 2016.

[KN14] Kopczynska, Sylwia; Nawrocki, Jerzy R.: Using non-functional requirements templates for elicitation: A case study. 2014 IEEE 4th International Workshop on Requirements Patterns (RePa), S. 47–54, 2014.

[Ma18] Maia, Rainara; Andrade, Rossana; Oliveira, Káthia; Kolski, Christophe: Catalog of Invisibility Requirements for UbiComp and IoT Applications. 08 2018.

[OM11] OMG: Business Process Model and Notation Specification Version 2.0. Object Management Group, 2011.

[PQF14] Palomares, Cristina; Quer, Carme; Franch, Xavier: Requirements Reuse with the PABRE Framework. Requirements Engineering Magazine, 01 2014.

[To08] Toval, Ambrosio; Moros Valle, Begoña; Nicolás, Joaquín; Lasheras, Joaquín: Eight key issues for an effective reuse-based requirements process. Computer Systems Science and Engineering, 23:373–385, 11 2008.

[Vi12a] Vieira, Sergio; Viana, Davi; Nascimento, Rogerio; Conte, Tayana: Evaluating a technique for requirements extraction from business process diagrams through empirical studies. S. 1–10, 10 2012.

[Vi12b] Vieira, Sergio; Viana, Davi; Nascimento, Rogerio; Conte, Tayana: Using Empirical Studies to evaluate the REMO Requirement Elicitation Technique. 01 2012.

[Wo06] Wohed, Petia; Aalst, Wil M. P.; Dumas, Marlon; Ter, Arthur; Russell, Nick: On the Suitability of BPMN for Business Process Modelling. 10 2006.

[Yu03] Yu, Yijun; Mylopoulos, John; Yu, Eric; Leite, Julio; Linda, Leite; Liu, Lin: Software Refactoring Guided By Multiple Soft-Goals. 12 2003.

[zMR08] zur Muehlen, Michael; Recker, Jan: How Much Language Is Enough? Theoretical and Practical Use of the Business Process Modeling Notation. 03 2008.

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 141

Athos: An Extensible DSL for Model Driven Traffic and Transport Simulation

Benjamin Hoffmann[1] Neil Urquhart[2] Kevin Chalmers[3] Michael Guckert[4]

Abstract: Multi-agent systems may be considered appropriate tools for simulating complex systems such as those based around traffic and transportation networks. Modelling traffic participants as agents can reveal relevant patterns of traffic flow. Upsurging traffic in urban areas increases the relevance of such simulations and the insight they provide into reducing congestion and pollution. Developing multi-agent traffic simulations is a challenging task even for professional software developers. In contrast, domain experts need tools that can be quickly adapted to new questions emerging in their research without potentially error-prone communication with software developers. There is a need for simulation tools that are intuitive to domain experts yet flexible and adaptable by software developers as required. A model driven approach with an extensible domain specific language delivers an answer for both of these opposing requirements. The modeller is relieved from implementing time consuming programming details and can focus on the application itself. We present the domain specific language Athos that allows to create simulations of traffic and transport related problems declaratively. The models are platform independent and executable code can be generated for appropriate multi-agent platforms. The language is flexible and can be easily extended by exploiting the structure of the problem domain itself. In this paper, we present Athos and focus on how it can be extended by arbitrary traffic and routing algorithms through an annotation-based extension mechanism.

Keywords: Domain Specific Language; Traffic Simulation; Multi Agent System;

1 Introduction

Human cognition alone can no longer comprehend the growing complexity observed in many real world systems. Instead, such systems must be analysed using experimentation and simulations that reveal some of the system's internal mechanisms. Transport networks and the associated traffic flows in urban areas are examples of such complex systems. Increasing traffic, congestion, and a rising number of last-mile deliveries call for new ideas that must be assessed before implementation because real-world testing would be too expensive. Tools that allow realistic simulation of alternative traffic scenarios offer insight into the feasibility and effects of the options.

[1] Kompetenzzentrum für Informationstechnologie – Technische Hochschule Mittelhessen benjamin.hoffmann@mnd.thm.de
[2] School of Computing – Edinburgh Napier University N.Urquhart@napier.ac.uk
[3] Department of Media, Culture and Language – University of Roehampton Kevin.Chalmers@roehampton.ac.uk
[4] Kompetenzzentrum für Informationstechnologie – Technische Hochschule Mittelhessen michael.guckert@mnd.thm.de

Rapidly changing requirements and the potential variety of alternative scenarios require software solutions that can be easily adapted and extended. When using general-purpose languages (GPLs), traffic domain experts must rely on software engineers to modify the system. Communication of experts of different domains is a known source of misunderstanding and may lead to weak models and error-prone systems [DC12]. The typically low-level abstraction in GPLs prevents reuse of larger building blocks so that implementations have to start from scratch each time a new problem has to be solved. Conversely, using a proprietary simulation platform often does not provide sufficient flexibility to deliver answers to specific questions and may create undesirable dependencies.

Model-driven approaches offer potential to overcome this dilemma (see [Ho18a]). Domain-Specific Languages (DSLs) can overcome the problems of platform dependency and miscommunication. Due to their higher abstraction level, and a notation specific to the underlying domain, they more expressiveness [va00, do12]. With our DSL, named Athos, domain experts can specify traffic simulations and related optimisation problems declaratively without lower-level programming concerns. Athos models follow a multi-agent paradigm and are accessible to humans and can be transformed into executable programs for given target platforms (e.g., NetLogo[5] or Repast Simphony[6]).

As a language for modelling simulations and optimisation problems, Athos targets domain experts as language users. Algorithms are considered to be part of the supporting infrastructure that runs the simulations. Therefore they are not implemented in the Athos language itself. This approach keeps models as computationally-independent as possible. Nevertheless, Athos has an elaborate mechanism that allows developers access and integrate external algorithms into the development and runtime environment. This is an important feature, as, in dynamic simulations of traffic scenarios, optimisation problems have to be solved to measure relevant indicators of the subject of investigation. For example, when simulating a multi-vehicle delivery problem, we must first solve the underlying routing problem and then test against dynamic traffic patterns (see [Ho18b, Ho19b]). According to the philosophy of Athos, implementing basic algorithms is not within scope and must be provided externally through problem solvers that must integrate transparently. This way, Athos is an interface between domain experts and algorithm developers from academia. Domain experts do not only put academic algorithms to real-world application, but also define requirements and discover potential for improvement. Algorithm scientists possess the expertise required for the creation of efficient algorithms that fulfill the defined requirements.

The rest of this paper is organised as follows: Section 2 introduces Athos, points out the need for an extension mechanism, and outlines how such a mechanism was implemented for Athos. Section 3 provides an example that illustrates how to extend Athos by integration of an external algorithm. In Section 4, other languages in the domain of agent-based traffic modelling are discussed. Finally, section 5 concludes the paper and identifies important future work.

[5] https://ccl.northwestern.edu/netlogo/
[6] https://repast.github.io/

2 Athos

Athos is a DSL for the specification of traffic and transport simulations that comprise vehicle routing problems (VRPs). Written with the Xtext framework[7], it features a full set of tools for developing models (e.g. a textual editor, model checking, and a generator). Our approach is platform independent, though we currently use NetLogo as our main target platform. This section introduces the reader to the language before it discusses the challenges of the domain. These challenges are then shown to be the catalyst that led to the development of the extension mechanism presented in the final part of this section.

2.1 A Domain-Specific Language for Vehicle Routing Problems

Athos is a declarative language designed for domain experts with little to no programming experience. In a typical traffic and transport scenario, vehicles, agents (the terms are used interchangeably in this paper) aim to perform given tasks or solve predefined problems, such as following a predefined route or delivering goods from depots to a list of targets. Athos is designed for tasks ranging from simple routing problems – e.g., the travelling salesman problem (TSP) or the vehicle-routing problem with time-windows (VRPTW) – to models of complex urban scenarios that respect the dynamics of traffic flow. In Athos, agents mutually influence each other. Congestion effects that occur due to parts of the road network being frequented by too many agents can thus be observed and examined. Due to congestion, agents may reconsider their original plan and adapt their behaviour according to the present situation; e.g. by avoiding congestion. Athos can thus be used to specify scenarios in which emergent phenomena may occur and new insights into the flow of traffic gained.

```
1  model VRPTW_Example
2   world xmin 0 xmax 75 ymin 0 ymax 75
3   products product soap weight 1.0
4   agentTypes
5   agentType staticDelivery congestionFactor 60.0 maxWeight 200.0
6      behaviour awt awaitTour when finished do die;
7      behaviour die vanish;
8   functions
9   durationFunction normal  length default
10  complete network
11  nodes
12  node n1 (35.0, 35.0)
13  ...
14  node n51 (47.00, 47.00)
15 edges
```

List. 1: General structure of an Athos model

Listing 1 demonstrates the structure of an Athos program. It defines a network, agent behaviour and details of the delivery problem. The network or graph defines the roads (edges) that vehicles can traverse. A complete graph is generated and the Euclidean distance

[7] https://www.eclipse.org/Xtext/

is used as length. Alternatively, a graph can be specified with individual attributes for its edges. Listing 2 demonstrates how sources and demands can be specified using nodes of the network. The keyword `ea` indicates that an evolutionary algorithm is to be used to compute the tours for the vehicles. Additional parameters for the algorithm are provided. For each demand node, quantities, time windows, and service time are defined.

Implementing such simulations with a GPL and using problem solver libraries leads to code in which the original problem is difficult to manipulate. By contrast, Athos is a declarative language that focuses on the specification of the actual problem rather than its solution. The solution is determined by the behaviour of the agents. In Athos, behaviour specifications are encapsulated in Agent-Behaviour Blocks (ABBs) that are associated with appropriate algorithms (see Listing 2). These algorithms provide solutions for the respective problem which are finally re-translated into observable agent behaviour.

```
1 sources
2     n1 isDepot soap sprouts (staticDelivery)   agentsStart route
3      ea (n2, n3, n4, ..., n48, n49, n50, n51) popSize 30
4 demands
5      n1 hasDemand soap absQuantity 0.00 earliestTime  0
6         latestTime 230 serviceTime 0
7      ...
8      n51 hasDemand  soap absQuantity 13.00 earliestTime 124
9         latestTime 134 serviceTime 10
```

List. 2: Sources and demand specification in Athos

Athos models are processed by a generator that creates code for an appropriate target platform. In order to create this code, the generator applies a set of transformations to the model and generates code that can be executed on the target platform. Currently, the generator features a complete set of transformations for the NetLogo platform. NetLogo is a DSL for multi-agent programming that is supported by a suitable simulation environment [TW04]. Though NetLogo specialises on the *technical domain* of multi-agent simulations, it is not tailored towards a specific *application domain*. Therefore, a language like Athos, which is specifically designed for the domain of traffic and transport simulation, further facilitates the creation of appropriate models. Since Athos models are platform independent, the creation of additional transformation sets designed for other target platforms is straightforward and has already been described in our previous work [Ho18a].

2.2 Modelling Vehicle Routing Problems

Since Athos is tailored towards the domain of traffic simulation and transport optimisation, the language must offer means to concisely formulate VRPs. Thus, the meta model of the language provides elements to define various types of routing problems (see [Ho18a, Ho18b, Ho19a]). Surveying the literature reveals that there is a range of problems that can be subsumed under the general term VRP.

From a mathematical point of view, the process of adding additional requirements to a problem is referred to as the *generalisation* of the problem. For example, a TSP can be generalised to a problem that defines time windows for each node that must be visited. The TSP may then be considered as a special case of the more general problem, in which the time windows are maximised. From the perspective of modelling this relation can be inverted as the TSP with time windows owns potentially more attributes and each TSP with time windows certainly is a TSP. From a syntactical object-oriented perspective, adding attributes and behaviour specialises a more general concept. In our hierarchy we consider a TSP with time windows as a *specialisation* of TSP, i.e., TSP is the more general problem.

The family of VRPs can be arranged in a taxonomy. At the top, there is the most general problem that offers the fewest definable attributes. The deeper the problem is located within this hierarchy, the more variables are required by the problem type. The most general problem type is the TSP (see Dantzig et al. [DFJ54] for its original formulation). The TSP can be modelled on a complete graph with n *vertices* or *customers* $c_i (i = 1, \ldots, n)$ are connected by *edges* (or arcs, depending on whether the problem is *symmetric* or *asymmetric*, respectively). Beginning at a given vertex, referred to as the *base* or *depot*, we find a minimal-cost path of nodes (costs occur upon usage of an edge/arc that connects the two respective nodes) that starts and ends at the depot and visits every other vertex exactly once.

By adding the requirement that the touring *vehicle* must return to the depot after m nodes have been visited, we get the *Clover Leaf Problem (CLP)* [DFJ54].The TSP can be considered a special instance of the CLP in which $m \geq n - 1$ holds. Nevertheless, we see the CLP as a more specific problem as the TSP and classify it as a successor of the TSP in our taxonomy.

In the *Capacitated Vehicle Routing Problem (CVRP)* [BTV10], each vertex is assigned a predefined *demand* d_i $(i = 1, \ldots, n)$, while the vehicle has a limited *capacity* q. Since the vehicle may not perform partial deliveries, it is assumed that none of the demands exceed the capacity of the vehicle.

The *Heterogeneous Vehicle Routing Problem (HVRP)* [BBV08] accepts different vehicle types with different capacities, unlike the one vehicle type allowed in the CVRP. This problem can be further extended by introducing:

- *fixed costs* for each vehicle type that occur upon deployment of a vehicle of the respective type (the problem is then referred to as the HVRP with Fixed-costs (HVRPF)).
- vehicle-type dependent routing costs, i.e., the costs that occur for travelling a given edge depend on the type of the respective vehicle (this is known as the HVRP with vehicle-Depending routing costs (HVRPD))

The HVRP is a special case in which the fixed costs are set to zero and the costs that occur are the same for all vehicle types (c.f. [BBV08]).

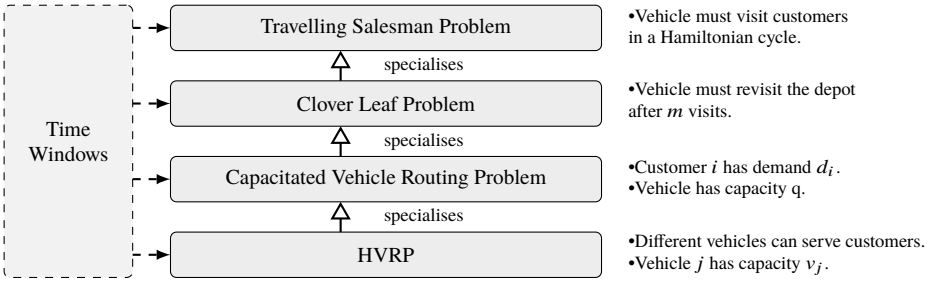


Fig. 1: Excerpt of Vehicle Routing Problem hierarchy

All the problem types defined above can further be extended by assigning time-windows to each customer. A time-window is a time-interval in which a visit must begin. If the vehicle arrives early, it is required to wait until the customer is ready (start of the interval). Conversely a vehicle that arrives after the end of the time window will be unable to service that customer. The depot is assigned a time-window which defines the time by which all vehicles have to return. Figure 1 illustrates the described hierarchy with the *generalises* relationships and the possible adoption of time windows.

This problem hierarchy represents a small excerpt of the taxonomy of VRP types found in the literature. Algorithms for solving such problems have been published and, as most of the problems are NP-hard, for larger problem instances meta-heuristics are applied to find feasible solutions of sufficient quality in a reasonable time. Algorithms and heuristics have elaborate sets of parameters controlling the operation of the computations for which appropriate values have to be provided. Athos transparently integrates such algorithmic solutions for problems underlying the simulations with an open mechanism for extended problems and algorithms. The language design must therefor address three major issues:

1. It must allow for an integration of various algorithmic approaches (e.g., ant colony-based or evolutionary algorithms) for each specific type of problem.
2. It must act as an interface that allows users to pass parameters together with their respective values to the chosen algorithms.
3. The language must be designed in a way so to support modelling the entire VRP hierarchy as far as reasonably possible.

2.3 Extensibility of the Language

Athos features a built-in extension mechanism that allows *algorithm developers* to integrate their implementations into the Athos environment. There is no need for a *language developer*

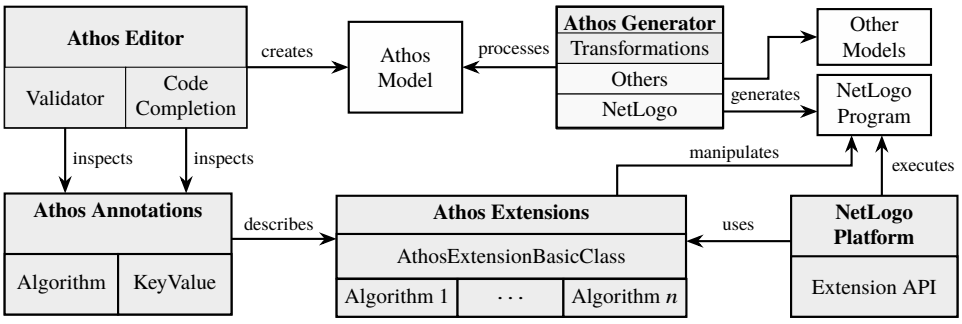


Fig. 2: Architecture of the Athos Development Environment

to change definition of the DSL; i.e., its abstract or concrete syntax, or its static semantics. There is also no need for a language developer to adapt the transformations to the respective target platform. Instead, all an algorithm provider needs to do is to create a Java class that extends a given base class, using two Java annotations provided by the Athos framework. The domain expert will then be able to use the newly integrated algorithm almost as if it was natively supported by the environment.

Figure 2 illustrates the general architecture of the DSL and its development framework. In the described scenario, NetLogo is used as a target platform. However, this approach works in principle with any Java-based multi-agent-simulation platform. Inside the Athos editor, an Athos model is developed and then further processed by the generator. The generator uses a subset of its transformations in order to generate a program that can be run on the NetLogo multi-agent simulation platform. The generated NetLogo program requires the provision of a special Athos extension package referred to as the *generic optimisation algorithm library*. Note that different target platforms might require to adapt this library to the respective target platform. The package encapsulates all routing-related algorithms that can be used inside the NetLogo program. In order to integrate an algorithm into the DSL framework, algorithm providers have to extend the base class `ExtendedOptimisationBehaviourImplementation` and use two distinct annotations provided by Athos. These annotations provide information to the extension algorithms. Moreover, the Athos Editor can use the information provided with these annotations to check certain constraints and provide code completion proposals. It is also important to note that an algorithm in the extension package can also manipulate the specifications of the NetLogo program. In the case study in Section 3, this will be used to assign routes of nodes to the vehicles.

Figure 3 provides an overview of the annotations and their respective attributes required for the integration of an algorithm into the Athos framework. The first of these two annotations is `@Algorithm` that marks a class as an algorithm that will be accessible in an Athos model. It features a `name` attribute that is used to assign a name to the algorithm for invocation inside an Athos model. The second annotation is `KeyTypeSet`. It is used to

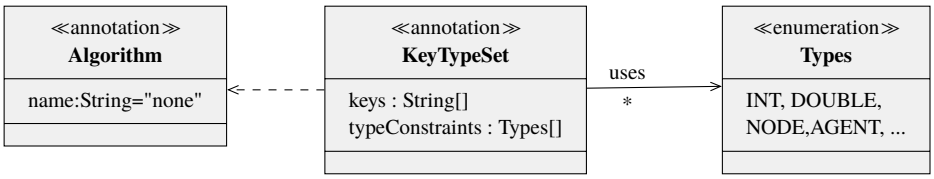


Fig. 3: Annotations Used to Integrate Additional Algorithms into Athos

Type	Explanation
`INT`,`DOUBLE`, `STRING`	Primitive types must meet the lexical requirements of the respective type.
`AGENT`	Parameter must refer to an existing agent type declared in the agents section of the model.
`NODE`	Parameter must refer to an existing node declared in the network.
`START_POS.`	Parameter must refer to a node in the network. This node should also sprout (create) the correct number of vehicles.
`LINK`	Parameter must refer to an edge declared in the network.

Tab. 1: Type System Applied in the Athos Extension Annotations

provide additional information that cannot be modelled with the native language elements provided by Athos. As an example, consider the implementation of a new heuristic that features a probabilistic parameter. It can also be used to model vehicle routing problems for which Athos does not natively feature the required modelling elements. The `KeyTypeSet` annotation has a `String` and a `Type` array as its attributes. Algorithm developers can assign a name to each parameter required by their algorithms and at the same time ensure type-safety by providing a type for each parameter. The parameter names and their types simply have to be at corresponding positions inside the respective array, i.e., the parameter with the name provided at `keys[0]` is of type `typeConstraints[0]` and so on. Parameters can have the primitive types string, integer, double. They can also have the complex types node, `START_POSITION`, or `AGENT`. Each of these types can also be defined as a one- or two-dimensional array. Table 1 gives an overview of the allowed parameter types together with a brief explanation of their semantics (an example will also be discussed in the case study in Section 3).

```
1 @Algorithm (name="EAForVRPTW")
2 @KeyTypeSet(
3     keys= { "startCity", "customers", "vehicleCapacity",
4             "populationSize","probabilityForGreedyCreation"},
5     typeConstraints={Types.NODE, Types.NODE_ARRAY, Types.INTEGER,
6         Types.INTEGER, Types.DOUBLE}
7 )
8 public class EvolutionaryAlgorithmForVRPTW
9     extends ExtendedOptimisationBehaviourImplementation {
```

List. 3: Annotation of an Algorithm for VRPTW

In [Ho19a], we describe the implementation of an evolutionary algorithm for the VRPTW based on that presented by Ombuki et al. [Om06]. Since this algorithm was used in the process of bootstrapping Athos, it is natively supported by the language. However, Athos is also open for any other algorithm that solves VRPTWs. Listing 3 provides an example that shows how the information required by Ombuki's algorithm could have been provided using Athos' extension mechanism. The `@Algorithm` annotation in line 1 marks the class as an algorithm that can be invoked from an Athos model with the name `EAForVRPTW`. This evolutionary algorithm requires some additional parameters (some omitted for brevity): The first parameter, `startcity`, is required so that the algorithm knows the location of the depot. The `customers` node array represents the customers of the problem – the nodes that have to be visited. The integer parameter `vehicleCapacity` sets the maximum capacity of the homogeneous fleet of vehicles assumed in the algorithm. The `populationSize` parameter of type integer is a common parameter for evolutionary algorithms that determines the number of chromosomes per generation. The parameter `propabilityForGreedyCreation` of type double is more special to the chosen algorithm. In the initialisation phase of the algorithm, it determines the probability for a new chromosome to be created by the application of a greedy heuristic. With the complementary probability the customers represented by the chromosome are simply ordered randomly.

```
1 sources
2  n1 isDepot soap sprouts (delivery1) agentsStart extended EAForVRPTW
3     startCity (n1)
4     customers (n2, n3, n5, n9, n10, n12, n15, n22)
5     vehicleCapacity 200
6     populationSize 20
7     probabilityForGreedyCreation 0.7
8     at 2
```

List. 4: Example Model With Behaviour Specification

The annotated algorithms in the package are inspected by the Athos framework. The validator uses the information provided via the annotations to determine new keywords that are valid when defining depots in the network. Listing 4 defines node `n1` as a depot from which a homogeneous fleet of vehicles with a capacity of 200 start their tour. In addition to the problem-related parameter values, the model also specifies parameter values that affect the execution of the algorithm. In this example, the population size is set to 20 and the probability that a chromosome is created with a greedy approach is set to 70 per cent.

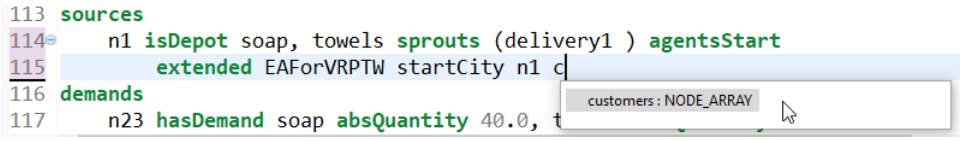


Fig. 4: Proposal Provision in the Athos Editor

Additionally, the proposal provider inspects the annotations in order to provide valid proposals. This is especially important given that the targeted users of the language are

domain experts rather than professional software engineers. In Figure 4, it is illustrated how the Athos IDE provides support in the invocation of the `EAForVRPTW` algorithm. After the specification of the name of the algorithm, the Athos framework inspects the associated `KeyValue`-Annotation in order to help the user specify the parameters in the correct order and with the correct types. In the illustrated example, the framework can deduce from the position of the cursor and the previous information, that the next parameter to be defined must be the `customers` parameter. A corresponding proposal is thus provided to the user who runs little risk of specifying the parameters incorrectly. The framework can also check that the provided information matches the types expected by the algorithm.

3 Example Use Case

In this section, we will provide a detailed examination on how to use the extension mechanism of Athos in order to integrate and apply new routing algorithms. We will show that this does not only allow integration of the most efficient algorithms, but also to extend the language and its infrastructure in a way that new problem types can be modelled and solved.

3.1 The HVRPD and an Instance Model

In section 2.2, the HVRPD was defined as a generalisation of the TSP from a mathematical point of view and as a specialisation of the TSP from an object-oriented point of view. In this example use case, we will model an even more specialised variant of the HVRPD. Here, there is not only one but two different products that are to be delivered. Hence, for each customer, two different demands have to considered, and for each vehicle type of the heterogeneous fleet two different capacities have to be modelled. Moreover, in this variant of the VRP, the vehicles do not start their tour from a central depot but from different nodes of the network. However, for the sake of modelling, the vehicles are regarded as being coordinated by one depot inside the network, even though they are not parked there.

Listing 5 shows how the HVRPD with multiple products and non-central vehicle starting locations can be modelled with Athos. The first two lines set the formal name of the model and the boundaries of the environment. Line 3 defines the product types that are to be delivered. In line 4, the required agent attributes are declared. As each vehicle has routing costs and a maximum quantity for each product, there are three declared attributes. Lines 5 to 15 define the agents that form the heterogeneous vehicle fleet. In this example, the fleet consists of three different agent types. The declaration of the first vehicle type spans from line 6 to 11. Lines 7 to 9 assign an actual value to each of the declared agent attributes. The behaviour exhibited by the first vehicle type is modelled in lines 10 and 11. The modelled behaviour has the vehicle waiting at its starting location (in a parking position) until it is assigned a route by the coordinating depot. After the route was performed, the agent is to disappear from the simulation.

```
1  model id001
2  world xmin 0 xmax 40 ymin 0 ymax 22
3  products product soap, product towels
4  agentAttributes soapCapacity, towelsCapacity, costPerDistance
5  agentTypes
6      agentType delivery1 congestionFactor 0.0
7          attr soapCapacity  120.0
8          attr towelsCapacity 240.0
9          attr costPerDistance  1.0
10         behaviour awt awaitTourExternal when finished do die;
11         behaviour die vanish;
12     agentType delivery2 congestionFactor 0.0
13         // omitted for reasons of brevity
14     agentType delivery3 congestionFactor 0.0
15         // omitted for reasons of brevity
16
17         // network specification omitted for reasons of brevity
18
19 sources
20     n1 isDepot soap, towels sprouts (delivery1 )
21     agentsStart extended JSpritHVRPSolver
22         customers (n1, n2, n3, n4) demands ("soap", "towels")
23         vehicleTypes ("delivery1", "delivery2", "delivery3")
24         capacities ("soapCapacity", "towelsCapacity")
25         costsPerDistance (1.0, 1.1, 1.2) instancesPerType (2.0, 1.0, 1.0)
26         startLocations ((n1,n1), (n2), (n4)) at 2
27     n1 sprouts quantity 2 (delivery1) at 1
28     n2 sprouts quantity 1 (delivery2) at 1
29     n4 sprouts quantity 1  (delivery3) at 1
30 demands
31     n23 hasDemand soap absQuantity 40.0, towels absQuantity 50.0
32     // additional demands omitted
```

List. 5: General Structure of an Athos Model

The declaration of the other types is done analogously and was omitted for reasons of brevity. The same goes for the specification of the network which was already explained in section 2.1. Line 19 starts the definition of sources – nodes inside the network where vehicles enter the simulation. Line 20 marks node `n1` as a depot. The `agentsStart` keyword followed by the `extended` keyword allows the invocation of an algorithm that possesses the aforementioned extension annotations required by Athos.

Lines 21 to 26 represent the code that invokes an externally provided algorithm. Line 21 specifies that the `JSpritHVRPSover` provided in the generic optimisation algorithm library is to be invoked. Line 22 specifies that `n1, n2, n3, n4` are the `customers` that have to be supplied with `soap` and `towels`. The actual demands of the customers are specified beginning in line 31. However, it would also be possible to use the extension mechanism to specify the demands of each customer[8]. Line 23 specifies the vehicle types that comprise the heterogeneous fleet controlled by the depot. Line 24 is required so that the algorithm can infer how to query for the actual capacities of a vehicle type[9]. In line 25, the

[8] This could actually be achieved in more than one way. It is possible to define two arrays, one containing the soap and the other the towel demand for each customer. Alternatively, a two-dimensional array could be specified that has both demands for each customer inside a tuple.

[9] Not every agent attribute is necessarily a capacity attribute. In lines 7 to 9 only two of the defined attributes are actually capacity attributes. To allow the algorithm to infer this fact, this information has to be provided

distance-dependent routing costs for each vehicle are defined, e.g. one distance unit travelled by a vehicle of type `delivery1` adds 1.0 cost units while the same distance adds 1.2 cost units when a vehicle of type `delivery2` is used. In the same way, the number of actual instances of each vehicle type is defined in this line. Finally, in line 26 the starting positions of the vehicles inside the network are defined. The two instances of type `delivery1` both start at node `n1`. The instance of `delivery2` starts at node `n2`, while the vehicle of type `delivery3` begins its tour at node `n4`. The keyword `at` defines at what point in time (which is measured in ticks) the coordinating depot has the vehicles start their tour.

It is important to note, that in this scenario the `startLocations` parameter is used to inform the algorithm on where the vehicles are to start their tour, i.e., at what node they wait for an order from the coordinating depot. However, the creation of these vehicles is initialised by the code provided in lines 27 to 28. The nodes defined as start locations in line 26 have to correspond to the actual start locations defined in lines 27 to 28. To enable the validator to check the model for this correspondence, the `startLocations` parameter was assigned the type `Types.START_POSITION_ARRAY_ARRAY`. This way, the validator automatically checks that each node declared as a starting location creates a sufficient number of vehicles prior to the execution of the algorithm. However, if the constraint fails, the validator only issues a message instead of an error. The reason for this is that the algorithm creator can perform the creation of the cars inside the algorithm.

3.2 Implementation of the Algorithm

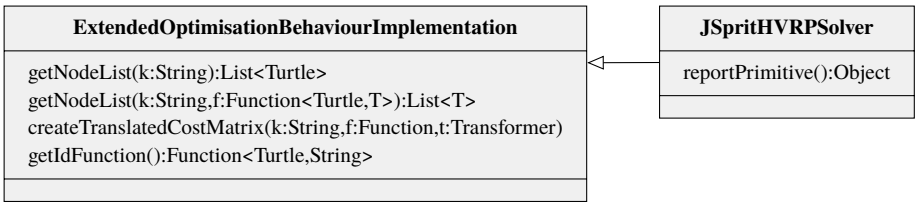


Fig. 5: Convenience Methods Provided by the Extension Framework to Extension Algorithms

The annotation of the algorithm functions as discussed in section 2.3 and exemplified in Listing 3. An example implementation is provided online[10]. As is depicted in Figure 5, classes that implement an algorithm must implement the `reportPrimitive()`-Method. The obtainment of the required extension parameter values is facilitated by convenience methods (e.g. `getNodeList()`) provided by the base class. Though the extension mechanism works for all Java-based simulation platforms, in this example the extensions for the NetLogo platform are presented. Since the algorithms that solve the problems are independent of the NetLogo platform, there must be a conversion mechanism, that allows a mapping from NetLogo nodes (known within NetLogo as `Turtles`) to the corresponding node data structures used in the algorithm (e.g. `Location` that can be identified by a

[10] `https://athos.mnd.thm.de/public/JSpritHVRPSolver.txt`

`String` id). For this, a default mapping `Function` is provided by a base class that can be changed by overwriting the `getIdFunction` method. The base class also provides facilitated access to the agents of the simulation together with their data structures. This way, the algorithm can easily obtain and manipulate data from entities of the simulation. This mechanism is used to equip each agent with a tour from the solution of the algorithm.

4 Related Work

Literature on DSLs specifically designed for the domain of transportation and traffic scenarios incorporating multi-agent environments and VRPs is scarce. Research has been published that examines general DSLs with a domain in the context of multi-agent systems. This section compares Athos with DSL-like approaches as well as with simulation platforms.

Steil et al. [St11] present a model for the expression, execution, evaluation and engagement of routing plans for patrols (e.g. police patrols) which they call the 4Es model. A DSL named *Turn* is used to describe algorithms that allow an agent to successively determine the next node on a route in a given graph network. Turn lets the user specify algorithms for the target selection as a composition of *set reduce functions (SRFs).* An SRF takes a set of possible target nodes and reduces this set according to certain criteria. Turn is used to chain and configure an arbitrary number of SRFs. An SRF chain is executed until only one node is left which then becomes the next target node. If the subset comprises multiple nodes, one of these nodes is chosen randomly. Algorithms written in Turn are executed by the *PatrolSim* environment which also evaluates routes produced by the algorithms according to a predefined set of metrics. A geographic information system (GIS) engages users through provision of patrol routes upon request.

Similar to Athos, the 4Es model can be used to model specific routing problems. The creation of satisfactory patrol routes is a problem related to the VRPTW. Though Athos and Turn can both be used to direct agents through a network of nodes, they differ in the way target nodes are specified. While Turn is mainly a composition of rules that successively reduce a set of potential target nodes, Athos allows an explicit specification of which node (or set of nodes) to visit next or requires that a set of nodes must be visited in an optimal order according to a given cost function. Another difference is the way behaviour changes are defined in both languages. In Turn, events that lead to a change of behaviour are transparently “hidden” inside the SRF and controlled by changing the SRF. Whenever an SRF returns an empty set of nodes it is skipped and replaced by another SRF which leads to a change of behaviour. In Athos, it is possible to explicitly state when a change of behaviour has to occur.

While in the 4Es model the cost to travel between any two nodes is defined as the minimum number of vertices between these nodes (i.e., the *minimum hop-count*), Athos allows the definition of arbitrary cost functions on the edges of the network. Turn does not reflect dynamic changes in the network, but uses the same distances throughout the entire simulation.

Athos cost functions can contain factors that dynamically change depending on the current traffic situation and allow for richer scenarios.

The GAMA framework [Gr13] supports the creation of spatialised, multi-scale agent-based models. It features a DSL named GAML with which agent-based simulation experiments can be designed. Hence, it seeks to raise the abstraction and allow for easy parameterisation and flexible visualisation of the models. GAML allows description of arbitrary details of the model. Multiple levels of abstraction can be used inside a simulation. So it is possible to look at a coarse-granular model for a road network with multiple detailed models that represent the inside of buildings adjacent to the roads. GAML can be considered a language that increases the abstraction level of models compared to those created in other simulation platforms like Repast Simphony or Netlogo. Since the GAMA framework does not focus on one specific application domain, models written in GAML can get complex. Data types and the definition of aspects to visualise agents in the simulation are hard to understand for domain experts with only limited experience in software engineering.

MATSim [HNA16] is a multi-agent microsimulation system based on a principle of co-evolution, in which agents are equipped with a set of plans which they can try and evaluate. In each cycle a plan is selected, applied and evaluated. With a given probability, agents modify different dimensions of their plans. For example, agents can vary the time they leave a given location, choose a different route or switch to a different mode of transport. Each agent seeks to optimise its individual outcome. Macijewski and Nagel present a MATSim-based approach to evaluate algorithms for the DVRP [MN12]. Their work intends to plan demand-responsive transport services (DRT severcies) using the MATSim framework. Using the MATSIm framework again requires deep programming and does not have a declarative perspective which we find more suitable for domain experts.

All of the approaches mentioned above can be used to model real-world VRPs and simulations. However, none of them can be used to generate traffic simulations from a pure specification in a DSL without considerable additional effort. This effort would be required for the development of either an appropriate simulation platform that processes the specification or a generator which would transform the specification to an executable traffic simulation. In this aspect, Athos is a consequent implementation of an instrument with which models of dynamic routing problems can be formulated and solved thus making use of the capabilities of the agent-based philosophy.

5 Conclusion and Future Work

We have demonstrated how Athos was opened for integration of additional VRP algorithms by the introduction of an annotation-based extension mechanism. We have also shown how this extension mechanism can be used to control arbitrary parameters of additional algorithms and how it allows for modelling additional problem properties not natively supported by the DSL without changing core elements of the DSL. Moreover, we discussed

how this extension mechanism can feature constraint checks. We see two perspectives that have to be further elaborated next. One is the flexibility of the language for both the language developer and the domain expert and the other is a systematic evaluation of the language.

Flexibility of Language The generic nature in which problem solvers can be accessed in the generated code allows for future extension of the system. In section 2.3 we presented two important roles: the *domain-expert* uses the language for their research into traffic and transportation, the *algorithm developer* can extend the language. Domain-experts identify specific requirements and formulate them so that software engineers can then implement appropriate Athos extensions. Such individual changes do not affect the kernel of the language and must not be integrated into the language itself.

Evaluation of the Language Initial experiments and interviews with domain experts have shown that Athos code is easily comprehensible for a traffic expert even without training. Writing models is obviously more difficult and requires at least a minimal amount of training. Future work will evaluate the language in field experiments and results will be reflected back to the language and its design in order to further improve its usability. Besides Challenger's multi agent systems specific SEA_ML framework [CKT15] for evaluating DSLs we plan to also blend the Cognitive Dimension of Notations framework (see [BG03]) and additional approaches (e.g., [RdBZ17]) into our evaluation methodology.

References

[BBV08] Baldacci, Roberto; Battarra, Maria; Vigo, Daniele: Routing a Heterogeneous Fleet of Vehicles. In (Golden, Bruce; Raghavan, S.; Wasil, Edward, eds): The Vehicle Routing Problem: Latest Advances and New Challenges, volume 43 of Operations Research/Computer Science Interfaces, pp. 3–27. Springer US, Boston, MA, 2008.

[BG03] Blackwell, Alan; Green, Thomas: Notational systems–the cognitive dimensions of notations framework. HCI Models, Theories, and Frameworks: Toward an Interdisciplinary Science. Morgan Kaufmann, 2003.

[BTV10] Baldacci, Roberto; Toth, Paolo; Vigo, Daniele: Exact algorithms for routing problems under vehicle capacity constraints. Annals of Operations Research, 175(1):213–245, 2010.

[CKT15] Challenger, Moharram; Kardas, Geylani; Tekinerdogan, Bedir: A systematic approach to evaluating domain-specific modeling language environments for multi-agent systems. Software Quality Journal, pp. 1–41, 2015.

[DC12] Dalal, Sandeep; Chhillar, Rajender Singh: Case Studies of Most Common and Severe Types of Software System Failure. International Journal of Advanced Research in Computer Science and Software Engineering, 2(8):341–347, 2012.

[DFJ54] Dantzig, George; Fulkerson, Ray; Johnson, Selmer: Solution of a large-scale traveling-salesman problem. Journal of the operations research society of America, 2(4):393–410, 1954.

[do12] do Nascimento, Leandro Marques; Viana, Daniel Leite; Am Neto, Paulo Silveira; Martins, Dhiego A. O.; Garcia, Vinicius Cardoso; Meira, Silvio R. L.: A systematic mapping study on domain-specific languages. In: Proceedings of the 7th International Conference on Software Engineering Advances (ICSEA'12). pp. 179–187, 2012.

[Gr13] Grignard, Arnaud; Taillandier, Patrick; Gaudou, Benoit; Vo, Duc An; Huynh, Nghi Quang; Drogoul, Alexis: GAMA 1.6: Advancing the art of complex agent-based modeling and simulation. In: International Conference on Principles and Practice of Multi-Agent Systems. pp. 117–131, 2013.

[HNA16] Horni, A.; Nagel, K.; Axhausen, K.W.: The Multi-Agent Transport Simulation MATSim. London: Ubiquity Press., 2016. DOI: http://dx.doi.org/10.5334/baw.

[Ho18a] Hoffmann, Benjamin; Chalmers, Kevin; Urquhart, Neil; Farrenkopf, Thomas; Guckert, Michael: Towards Reducing Complexity of Multi-agent Simulations by Applying Model-Driven Techniques. In: Advances in Practical Applications of Agents, Multi-Agent Systems, and Complexity: The PAAMS Collection - 16th International Conference, PAAMS 2018, Toledo, Spain, June 20-22, 2018, Proceedings. pp. 187–199, 2018.

[Ho18b] Hoffmann, Benjamin; Guckert, Michael; Farrenkopf, Thomas; Chalmers, Kevin; Urquhart, Neil: A Domain-Specific Language For Routing Problems. In: ECMS 2018 Proceedings edited by Nolle L. et.al. pp. 262–268, 05252018.

[Ho19a] Hoffmann, Benjamin; Chalmers, Kevin; Urquhart, Neil; Guckert, Michael: Athos - A Model Driven Approach to Describe and Solve Optimisation Problems: An Application to the Vehicle Routing Problem with Time Windows. In: Proceedings of the 4th ACM International Workshop on Real World Domain Specific Languages. RWDSL '19, ACM, New York, NY, USA, pp. 3:1–3:10, 2019.

[Ho19b] Hoffmann, Benjamin; Guckert, Michael; Chalmers, Kevin; Urquhart, Neil: Simulating Dynamic Vehicle Routing Problems With Athos. In: ECMS 2019 Proceedings edited by Mauro Iacono, Francesco Palmieri, Marco Gribaudo, Massimo Ficco. ECMS, pp. 296–302, 06142019.

[MN12] Maciejewski, Michał; Nagel, Kai: Towards Multi-Agent Simulation of the Dynamic Vehicle Routing Problem in MATSim. In (Wyrzykowski, Roman; Dongarra, Jack; Karczewski, Konrad; Waśniewski, Jerzy, eds): Parallel Processing and Applied Mathematics. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 551–560, 2012.

[Om06] Ombuki, Beatrice M.: Multi-Objective Genetic Algorithms for Vehicle Routing Problem with Time Windows. Applied Intelligence, 24(1):17–30, 2006.

[RdBZ17] Rodrigues, Ildevana Poltronieri; de Borba Campos, Márcia; Zorzo, Avelino F.: Usability Evaluation of Domain-Specific Languages: a Systematic Literature Review. In: International Conference on Human-Computer Interaction. pp. 522–534, 2017.

[St11] Steil, Dana A.; Pate, Jeremy R.; Kraft, Nicholas A.; Smith, Randy K.; Dixon, Brandon; Ding, Li; Parrish, Allen: Patrol Routing Expression, Execution, Evaluation, and Engagement. IEEE Transactions on Intelligent Transportation Systems, 12(1):58–72, 2011.

[TW04] Tisue, Seth; Wilensky, Uri: NetLogo: Design and implementation of a multi-agent modeling environment. In: Proceedings of agent. volume 2004, pp. 7–9, 2004.

[va00] van Deursen, Arie; Klint, Paul; Visser, Joost et al.: Domain-specific languages: An annotated bibliography. Sigplan Notices, 35(6):26–36, 2000.

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 157

Modeling Low-Level Network Configurations for Analysis, Simulation and Testing

Marcel Schuster[1] Markus Germeier[2] Frank Hilken[3] Martin Gogolla[4] Karsten Sohr[5]

Abstract: In this paper, we present an approach to specifying a network configuration model based on the ISO/OSI reference model. The network configuration model is capable of modeling the lower layers of networks enabling several use cases: (a) analyze existing network configurations, e.g., to find configuration errors and identify which components contribute to the error; (b) simulate changes made to the configuration and predict their consequences; (c) serve as documentation for the network; and (d) visualize the network in order to better understand it and to make clear its structure to everyday users. These analysis techniques were applied to essential parts of a professional network center.

Keywords: UML and OCL model; Network configuration model; Network configuration analysis; Network configuration documentation.

1 Introduction

The administration of large-scale computer networks is tedious and error-prone. Since network documentation is often difficult to establish and maintain for administrators, a supporting methodology to systematically model and test a computer network is desirable. In particular, this applies to the lower layers of the well-known *Open Systems Interconnection* (OSI) model [IT94] as these layers deal with technical details where often administration errors occur.

Here, we address this problem with the help of UML and OCL modeling and the validation of UML models. We first define a UML/OCL model, which represents the frame for expressing the concepts of the two lowest layers of the OSI model. This allows us to represent network concepts, such as "network component", "interface", "interface link", and "Virtual LAN" (VLAN), by using UML class diagrams and complementing OCL constraints. The concrete description of a computer network is then expressed in form of a UML object diagram.

After defining our language for representing computer networks based on UML, we use the UML-based Specification Environment (USE) [GBR07, GHD17] for different validation tasks. First, we automatically read network descriptions for a real-world network (a university

[1] University of Bremen, Database Systems Group, Bremen, Germany, maschu@informatik.uni-bremen.de
[2] Center for Networks (ZFN), Bremen, Germany, zfn-leitung@uni-bremen.de
[3] University of Bremen, Database Systems Group, Bremen, Germany, fhilken@informatik.uni-bremen.de
[4] University of Bremen, Database Systems Group, Bremen, Germany, gogolla@informatik.uni-bremen.de
[5] Center for Computing and Communication Technologies (TZI), Bremen, Germany, sohr@tzi.de

housing center) and transform them into a corresponding UML object diagram. This diagram is subject to the validation tasks. If certain OCL constraints are violated, this gives an administrator hints that her network descriptions might be inconsistent. The ability to query the system state helps to identify the cause for the inconsistencies. For example, we found a concrete administration error in the network configuration. After reporting, this error was immediately eliminated by the administration of the housing center.

Our main contribution lies in the practical application of UML modeling and model validation in the area of network administration. In particular, the model concentrates on the lower layers 2 and below, which are seldom covered in other approaches. Based on this UML model we test concrete network configurations, which are represented as UML object diagrams. We then demonstrate the effectiveness of our approach with a real-world computer network at our university.

The rest of the paper is structured as follows. After presenting the technical background we explain our network configuration model for computer networks in Sect. 3. Section 4 describes the validation tasks in detail and Section 5 discusses related work. A discussion of the experience with our approach concludes the paper.

2 Background

2.1 Communication in Computer Networks

The OSI model is a communication model that describes the communication between two systems in seven layers [IT94], among them a *physical layer* and a *data link layer*. The data link layer provides mechanisms for detecting and correcting so-called frame errors. The "Ethernet" is a widely distributed and commonly used technology for LAN network communications [TW11]. A frequently used standard in corporate networks is IEEE 802.1Q [IE14a] which specifies virtual LANs (VLAN). VLANs provide the possibility for a logical separation of physical infrastructure for performance and security reasons. The implementation requires VLAN-aware network components and the main idea is to assign network interfaces to a VLAN by configuring a number. The number is called "VLAN-ID" and represents the membership of an interface to a certain VLAN. Figure 1 shows a switch interconnecting four hosts. Interface 1 and 2 belong to VLAN 10 and interface 3 and 4 to VLAN 20. Due to the VLAN memberships of the interfaces, only hosts A and B can communicate with each other on layer 2 (analogously hosts C and D).

In practice one can differentiate between "access interfaces", which can be member of exactly one VLAN, and "trunk interfaces", which can be member of multiple VLANs [TW11]. Every incoming frame to an interface is classified and assigned to a certain VLAN. Another commonly used Ethernet standard in corporate networks is IEEE 802.1AX [IE14b], which introduces the aggregation of physical links between network components. Parallel point-to-point connections between two components can be aggregated to one logical interconnection

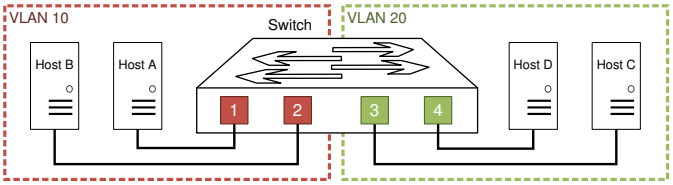


Fig. 1: Two VLANs on the ports of a switch.

called *Link Aggregation Group* (LAG) resulting in higher bandwidth and higher resilience due to redundancy. The link aggregation groups are not limited to interconnect only two network components. The according standard defines a so-called *Distributed Resilient Network Interconnect* (DRNI), which allows the aggregation of multi-homed connections resulting in a multi-chassis LAG (MC-LAG). For the operation of the individual components of the DRNI, an *Intra-Portal Link* (IPL) is mandatory, which is used for synchronization of status/configuration data.

These network techniques are some examples of commonly used concepts in industrial networks. Each one raises the complexity of network configurations in terms of sheer size and keeping track of the network structure in general. A modeling and analysis approach supporting network administration is desirable.

2.2 University of Bremen Housing Center

The housing center of the University of Bremen is the central data center which consolidates the IT infrastructure of all research groups. It is divided into two fire compartments, which have a total capacity of 4,000 rack-mounted servers. It is connected with a 80 Gbit/s bandwidth to the campus network.

The network design of the housing center follows the common data center network design consisting of three layers: *core layer*, *distribution layer* and *access layer*. The network components of the *core layer* build the heart of the network. They have high requirements regarding performance and resilience as a failure can lead to a complete outage of the housing center. The components of the *distribution layer* connect the core components to the access layer. The network components of the *access layer* are located in the two fire compartments of the housing center and provide the network access of the housed servers. The housing center uses components from Cisco Systems.

Despite not having many network components in the setup of the housing center, the individual configurations are complex and hard to maintain. This, for example, includes multiple VLANs for the separation of network traffic, LAG and MC-LAG setups for increasing the resilience of interconnections and diverse virtual routing instances (VRF) containing IP addresses and IP routes for an independent package routing. The configuration

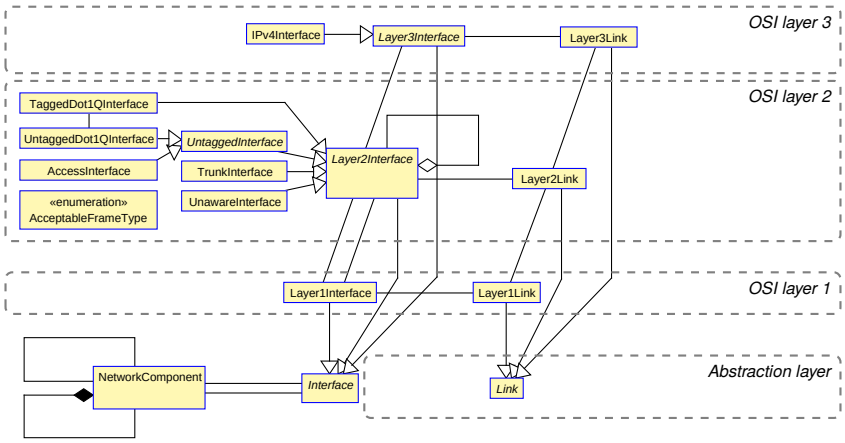


Fig. 2: Class diagram of the network configuration model.

files of all network components currently sum up to 32,500 lines for a proper interface configuration.

3 Network Configuration Model in UML/OCL

3.1 Major Challenges

Network administrators must address different challenges. One general challenge is the bottom-up configuration as networks are not configured as a whole. Each network component is configured individually and network administrators must make sure that the overall configuration is functional and secure so that no unwanted communication is possible. To track down errors, system tools like `ping` or `traceroute` are often used, which can help one roughly locate errors, but success is not guaranteed. In addition, sophisticated knowledge of the network is needed w.r.t the network components and wiring with their configurations consisting of VLANs, LAGs, IP addresses, subnets, IP routes, and firewalls.

Another challenge is the overview of logical topologies that evolve from the physical topology with its given configurations. With the technologies provided by Ethernet (e.g. VLAN and LAG) different network components can have different views on the network topology. Some components might not be reachable due to some VLAN restrictions and physical links might be aggregated to single logical LAGs. It is hard to keep track of these different network views. A documentation of the network is indispensable, yet difficult to create and maintain.

A further challenge are the differences between the theory of network standards and the practice. Companies often use a terminology for their products, which may be different

from standards; sometimes they even behave different than proposed. This makes it hard for administrators to understand the products and the comparison between different manufacturers. As an example, the access and trunk interfaces for the use of VLANs are very common but neither defined nor mentioned in IEEE 802.1Q. The standard allows a very flexible handling of VLANs at interfaces, which may not be useful or secure. This might be the reason why the simplified interface concepts are enforced in practice.

3.2 Network Configuration Model

To address the aforementioned challenges in network administration, a representation of static low-level configuration data of network components was established in UML and OCL. This model can help network administrators to analyze, simulate, document and visualize the details of a computer network. The model itself aligns to the OSI reference model and commonly used Ethernet standards like VLANs and link aggregation, which results in a homogeneous terminology and abstracts proprietary implementations. The advantage of focusing on static configuration data and neglect state data makes the model protocol independent, which leads to a wider applicability of the model. For the usage of the model, it makes no difference which proprietary equipment the network is built with and which implementations of common algorithms are used.

The basic idea of the network configuration model is that computer networks consist of `network components`, `interfaces` and `links`:

- **Network component:** A network component is an abstract representation of a physical or virtual device, which can be integrated into a computer network. It may represent something like a virtual machine, rack-mounted server, switch or router which is capable of dealing with VLANs and link aggregation. Network components do not belong to any layer.
- **Interface:** A network component has interfaces (or ports), which connect them to other network components. Interfaces can be physical or virtual as well and contain a layer-specific network address like a MAC address on layer 2 or an IP address on layer 3. Interfaces can be assigned to different layers on which they operate.
- **Link:** Interfaces of network components are interconnected by physical or virtual links. A link represents a bidirectional connection between exactly two interfaces. Two over a link interconnected interfaces are called *opposites* or *corresponding*.

Figure 2 shows an overview of the class diagram of the network configuration model, which consists of 16 classes, 14 associations, 48 invariants and 21 query operations. Unlike the OSI reference model, this model starts at the bottom with an *abstract layer* represented by the abstract classes `Interface` and `Link`, which combine common concepts from all layers. The specializations of these classes can be assigned specific layers, which are based on the OSI reference model. This results in layer-specific interfaces like the `Layer1Interface` with

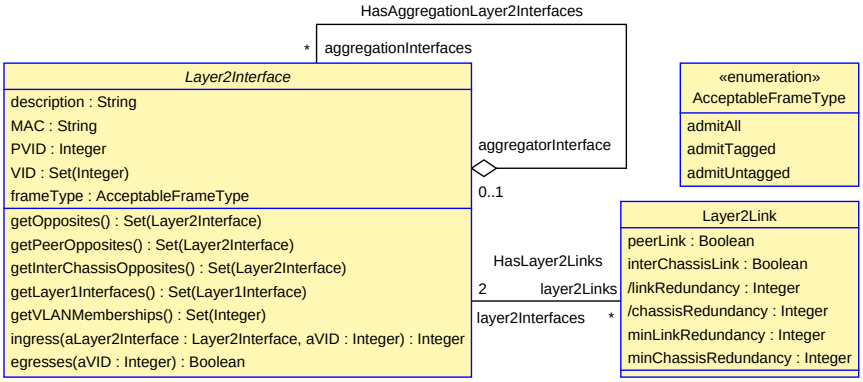


Fig. 3: Abstract class `Layer2Interface` with the class `Layer2Link` of the second layer.

a corresponding `Layer1Link`. The model represents the lower layers of the OSI reference model. In particular the focus is on the first two layers, which are fully implemented. Additionally, a rudimentary implementation of the third layer exists only focusing on the necessary parts for this project.

The first layer of the model correlates to the first layer of the OSI reference model. Due to the strong relationship of the first layer and the physical medium, both can be seen as one in context of this UML model. Because of this simplification, objects of the class `Layer1Link` can be seen as a physical medium like a copper twisted-pair cable. Analogously, objects of `Layer1Interface` correlate to physical cable connectors.

In contrast to the straightforward layer 1, the second layer of the UML model is much more complex and correlates to the data link layer of the OSI reference model (from Fig. 3). The class `Layer2Interface` represents a logical (or virtual) interface capable of dealing with technologies like VLAN and link aggregation so that objects of this class can be seen as Ethernet interfaces. Similarly, objects of the class `Layer2Link` are logical (or virtual) links interconnecting Ethernet interfaces. Due to the virtual characteristic, multiple layer-2 links can be associated with an Ethernet interface depending on the number of VLAN memberships.

Beyond a description and a MAC as a layer-specific network address, the class `Layer2Interface` defines three more attributes: `PVID`, `VID` and `frameType` (see Fig. 3). With the help of these attributes, it is possible to represent a VLAN configuration compatible to the standard IEEE 802.1Q. Different from the standard, the VLAN-IDs are assigned to the interfaces to define the membership and not the other way around (as it is done in practice). To help network administrators with an easier reference to the practice, some specializations of layer-2 interfaces were implemented like `AccessInterface` or `TrunkInterface`. These specializations determine a certain allocation of the aforementioned attributes for an easier mapping. An `AccessInterface`, for example, is member of one VLAN and is able to receive

and send frames without a VLAN-tag. This results in a valid assignment of the `PVID` and an empty `VID` set with the default `frameType` configuration. The other specialization classes can be mapped to a specific attribute setting analogously.

To represent link aggregations as specified in standard IEEE 802.1AX a structural solution was chosen. With the help of the reflexive aggregation `HasAggregationLayer2Interfaces`, it is possible to aggregate layer-2 interfaces. Based on the IEEE standard, the aggregate of the whole-part relationship is called "aggregator interface", whereas the component is called "aggregation interface". An aggregator interface can be associated with multiple network components to represent a multi-chassis LAG (like a DRNI). The presence of a LAG or MC-LAG accommodates strong structural requirements. For example, aggregator interfaces can only be interconnected to other aggregator interfaces to ensure the increased resilience of the logical link. In case of MC-LAGs, "Peer Links" (like an IPL) are required between the associated network components to enable synchronization of status information, which are modeled via attributes on layer-2 links (see Fig. 3).

Two derived attributes were established in the class `Layer2Link` to visualize the resilience of the logical link in the context of link aggregation:

- `/linkRedundancy` computes on how many layer-1 links this logical link depends. Any value greater than 1 represents link redundancy where every but one physical link can fail without affecting the availability of the logical link.
- `/chassisRedundancy` calculates the chassis redundancy, which is relevant when using MC-LAGs, whose interfaces are distributed across multiple network components. Any value greater than 1 represents redundancy so that network components on each side of the logical link can fail without breaking the connection.

The UML model also contains multiple derived associations. They are useful for the inspection of large system states as they help visualizing implicit information. Figure 4 shows a simple system state as an object diagram. This system state consists of two network components named `NC-A` and `NC-B`, which are physically interconnected. The resulting connection is represented by a layer-1 link, which is associated to two layer-1 interfaces. The existence of this layer-1 connection leads to the existence of the derived association called `D_Layer1Connection` in the context of network components. This association connects two network components when they are actually interconnected on the first layer. A modeler can then hide objects of the first layer without losing information.

Furthermore, the system state in Fig. 4 contains two interconnected access interfaces, which are member of the VLAN 1. They are based on the existing layer-1 objects and represent a logical connection between the two network components. The derived association `D_DependsOnLayer1Link` associates objects of class `Layer2Link` to objects of class `Layer1Link` to represent on which physical link the logical link is based on. The existence of

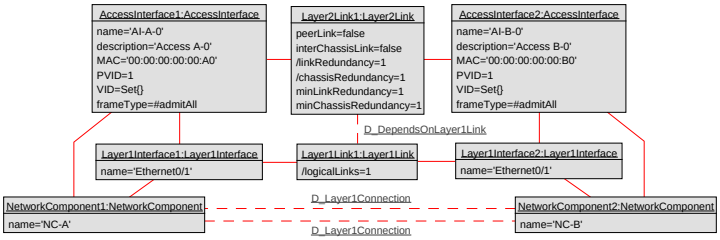


Fig. 4: Valid state: components *NC-A* and *NC-B* interconnected on different layers.

a derived layer-1 link is mandatory and can be inferred by the associations of the neighbored objects.

4 Analysis of the Housing Center

4.1 Basic Analysis Process

The first step of the housing center analysis is to have a valid representation as a USE system state. Due to the infrastructure complexity, the focus lies on the described core network components with two servers modeled as an example. Nevertheless, the complexity of the pursued system state is high resulting from the complex configuration of individual network components.

To avoid to go through the network configurations and manually construct the USE system state, a Java parser was implemented for an automatic transformation. Usually the Cisco network components are configured using specific configuration commands via an SSH interface. The overall configuration can be displayed and exported using particular commands. Besides this configuration, the parser expects an export of the Cisco Discovery Protocol (CDP) information for each network component. The CDP is a proprietary implementation of the Link Layer Discovery Protocol (LLDP), which helps to identify neighbored network components. This information helps to infer the physical cabling among the considered network components and create the layer-specific interconnections.

Fig. 5: Processing the configuration files and input for USE.

An overview of the parsing process is shown in Fig. 5. In our case, the parser processes 14 files of configuration commands and CDP information. In total, they consist of about 32,500 lines. The parsing process itself only takes less than 1 second and outputs a so-called "SOIL file" containing OCL-like commands to build a USE system state. The SOIL file contains about 6,500 lines of statements for object and link creation and setting attribute

values. With this SOIL file and the class diagram, USE establishes the system state of the network.

Figure 6 shows the object and link count of the fully loaded system state of the housing center in USE. The object diagram consists of 13 network components, 712 (14+288+68+268+50+2+22) interfaces with 180 `Link` objects of different layers in addition to 1,760 regular and 857 derived links. About one third of the total links are derived showing their importance as these links are implicitly derived from various constraints. At this point, the system state satisfies only 45 of the 48 invariants. A detailed analysis of the three failed invariants is discussed in the next section.

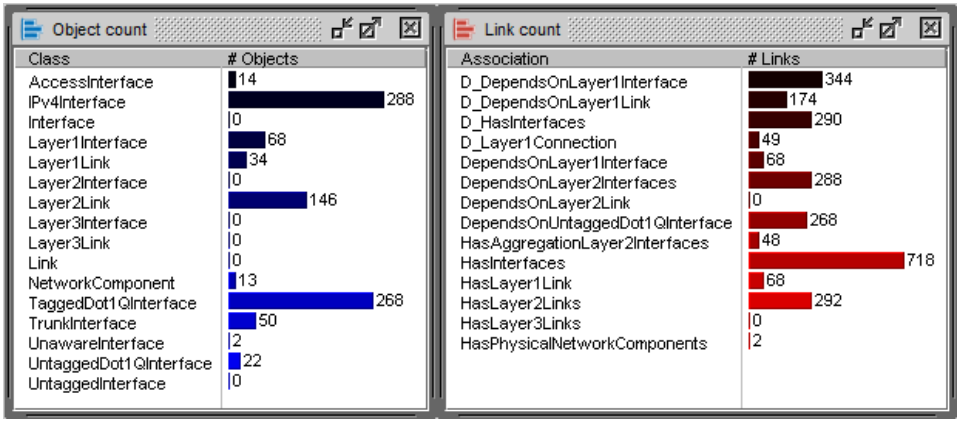


Fig. 6: Object and link count of the housing center system state in USE.

A network administrator now has many options to use the system state for her needs. It is possible to visualize the system state in USE to obtain a rough overview of the individual objects. Due to the quantity of objects and links, it is not advisable to display everything at once. USE provides features to show specific objects by name, type or OCL query and discover their neighbored objects by path length. Using this technique, a network administrator can display, for example, only interfaces and links of a certain layer providing a layer-specific investigation of the system state. The embedded derived associations help to keep track of the overall context, even if not every object is visible.

A network administrator can also query information with OCL. She can, for example, query specific data from a network component to obtain a tabular configuration overview. Even Cisco-specific analysis commands, like `show vlan brief` or `show interface trunk`, can be rebuilt in OCL. Using its condensed output, she can compare and verify the system state with the real world network component or the other way around. More powerful OCL queries are possible, which include configuration data from different network components, e.g., showing physical interfaces not posessing a link, every LAG or MC-LAG defined in the network, or virtual links not meeting the required resilience parameters (indicating broken physical links).

In practice, an administrator would have to compare the distributed configurations manually. Additionally she needs specific knowledge about the infrastructure itself combined with a sophisticated understanding of the influence of different configurations on the overall functionality of the network.

4.2 Incidents Found

The built USE system state does not satisfy every defined invariant. Nevertheless, the UML multiplicities together with the structure checking invariants are fulfilled, which states that the structure is valid and that the Cisco parser successfully parsed the configuration files. The three failing, configuration checking invariants can be examined further using the `check` command on the USE shell.

One of these invariants handles the uniqueness property of MAC addresses, whose transformation is not yet implemented in the parser. The second invariant detected orphaned "layer-2 subinterfaces", which do not have a corresponding interface. This could lead to miscommunication between network components, because the sent frames will not be processed accordingly. The third invariant will be the focus of further analysis, exemplifying the analysis process. A network administrator can get detailed information about the evaluation using the command `check -d`. The additional parameter leads to a detailed output, in which one can see which objects are violating the invariant:

```
use> check -d
     checking invariant 8: FAILED.
     Instances of Layer2Interface violating the invariant:
       Set{TrunkInterface23, TrunkInterface26}
     checked 48 invariants in 3.059s, 3 failures.
```

The focus is now on the failed `invariant 8`. This invariant checks if two corresponding interfaces are able to receive frames of certain VLANs the respective other interface is member of. The output of the command shows that the two objects `TrunkInterface23`, `TrunkInterface26` are violating the invariant. Object `TrunkInterface23` will be further used to demonstrate the options of the UML model in combination with USE.

Using a query, first the corresponding opposite interface of `TrunkInterface23` is found. For this specific case the query operation `getOpposited()` is implemented, which returns all corresponding interfaces. In this case, there is only one corresponding interface `TrunkInterface4`:

```
use> ?TrunkInterface23.getOpposites() -> Set{TrunkInterface4}
```

Next, we compare the VLAN memberships of the determined interfaces, as this was the cause of the invariant violation. To retrieve the VLAN memberships of an interface, the query operation `getVLANMemberships()` is used. Querying the difference of the VLAN memberships of the two interfaces shows the range of VLANs that are defined on one

interface and not on the other. Here, the object `TrunkInterface23` is member of ten more VLANs (1200–1209) than the object `TrunkInterface4`, which causes the invariant to fail.

```
use> ?(TrunkInterface23.getVLANMemberships() -
        TrunkInterface4.getVLANMemberships())->asSet()
     Set{1200,1201,1202,1203,1204,1205,1206,1207,1208,1209}
```

Now that the source of the violation is known, the network context can be explored further in order to identify the components with faulty configurations. Using a simple OCL query returning a tuple, the interface names and names of the associated network components are detected:

```
use> ?Set{TrunkInterface23,TrunkInterface4}->collect(i |
        Tuple{NM=i.name, NC=i.networkComponents.id().name})
     Bag{Tuple{NM='port-channel10',
               NC=Bag{'nexus-5500-a-1','nexus-5500-a-2'}},
         Tuple{NM='port-channel10',
               NC=Bag{'nexus-7000-a','nexus-7000-b'}}}
```

The output shows that both examined interfaces are named *port-channel10*. It also states that both interfaces are part of an MC-LAG, as they both are associated to two network components. Using the hide-and-show feature of USE, it is possible to visualize only the small examined context of the network as shown in Fig. 7. This confirms that the two interfaces are part of an MC-LAG. Overall, the analysis shows that the invariant violation was caused by an interface that interconnects the distribution layer with the access layer over an MC-LAG.

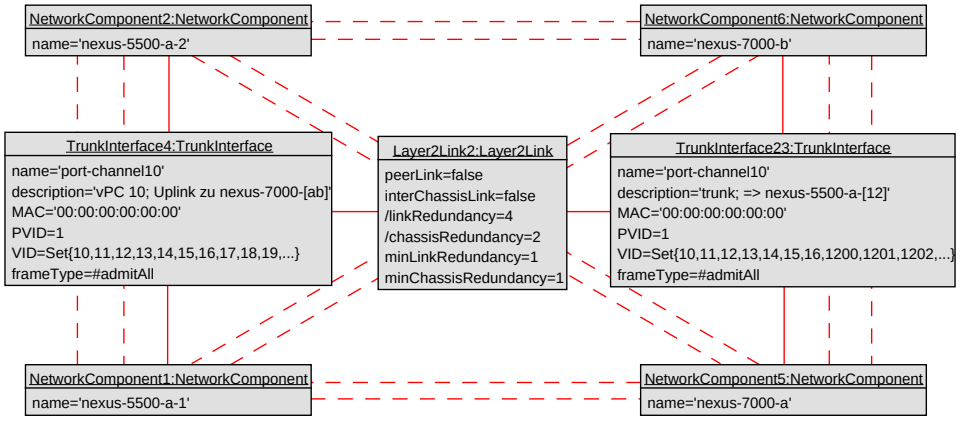


Fig. 7: Context of the found configuration error in the network components.

The impact of the discovered configuration error is a limited reachability of the network components. The `TrunkInterface23` is able to send frames belonging to VLAN 1200 to its connected interface `TrunkInterface4`. As the receiving interface of the frames is not a member of VLAN 1200, it will refuse and drop frames, resulting in a one-directional

miscommunication for VLANs 1200..1209. It is recommended to match the VLAN memberships of linked interfaces.

In practice this kind of error is hard to detect. The error will only become noticed when a certain reachability from one network component to another is not available. The task of error tracing can be very complex here due to the special conditions of the miscommunication, as the reachability for one VLAN might be present but fails for another. The search for misconfiguration can be simplified using our network configuration model. The defined invariants allow one to check the basic configuration and show the objects violating these constraints. With the use of a few query commands an administrator can extract and visualize the errors. An administrator can also create own queries for a quick inspection of the resilience attributes of certain links or any other data of interest.

4.3 Results and Discussion

When reporting the detected incidents to the administrator, the feedback was very positive. Ignoring the MAC addresses, in case of the first incident, it turned out that during the time of the acquisition of the configuration, an update has been pushed to some network components while still being scheduled for others. This led to a known inconsistency in the configuration that was later corrected. Nonetheless, this incident represents a configuration error in the extracted data and was identified as such by the model. As for the second incident, the misconfigured VLAN ranges have indeed not been recognized before and can be corrected with the detailed information extracted during the analysis.

Overall, the results show the suitability of the model to find errors of various kinds in the configuration of networks that would otherwise be hard to find in the distributed files containing configurations per network component. Additionally, the options to query the network configuration allows to locate causes of incidents directly. Furthermore, the model has demonstrated its suitability for industrial networks: in terms of the technologies that are used in such environments, which include proprietary equipment, and also in terms of performance.

5 Related Work

The “Interconnected Asset Ontology” (IO) [Bi12, BS14b, BS14a] resembles our work in its underlying goals. IO follows an ontology-based approach to the representation and inference of computer-network layers. It uses SPARQL as a query language to obtain information from computer networks. In particular, IO can represent VLANs, LAGs, and MC-LAGs as well. The topological model of our work can, hence, be seen as an UML/OCL-based alternative to IO, however, with a focus on the lower OSI-layers. An advantage of using UML is the fact that we can visualize network system states and exploit the USE model validator’s capability to complete system states.

Other works deal with the static analysis of computer networks. Xie et al. present an approach to determining reachability in IP-based networks [Xi05]. Their model is based on graphs and determines reachability between two network components by tracing all possible "headers" from a source to a target. Kazemian et al. refine this approach by also providing a technique for the detection of cycles and slices in network topologies [KVM12]. For this purpose, they use static configurations of network components, which are represented in a geometric model. This model also focuses on IP-based networks, but headers do not refer to a specific OSI layer. A similar approach is proposed by Al-Shaer et al., who model computer networks as finite state machines and follow a model-checking approach to analyze the model w.r.t. reachability and consistency [Al09].

Our approach resembles the aforementioned works in several ways. All techniques deal with the representation and analysis of static data of a computer network, however, with a focus on the IP layer of a computer network (including firewalls). In contrast, our work allows an administrator to consider the lower OSI-layers, which are not topic of the other works. In particular, we now support concepts such as VLAN and link aggregation while at the same time adopting ideas from the analysis of IP-based networks, such as reachability, cycle and slice analysis.

Our work has also connections to approaches for network configuration and administration based on formal semantics or formal tools and on proving-like techniques. A general formal configuration language is discussed in [AH16], and the application to real scenarios is demonstrated. In [Ma15] a tool based on SAT solving is used for network and access control list configuration. The configurations are checked through various standard and complex service access queries. [FM10] introduce a formal approach allowing to formally and optimally configure a network so that a given security policy is respected and by taking into account the quality of services. Special emphasis is put on the consideration of firewalls. The approach in [Ro06] presents a logic-based model that is suitable for describing networks and intrusions. The approach is implemented in Prolog and allows to analyze important static properties of networks. The work in [Fo06] proposes an abstract model for network configuration that is intended to help to understand fundamental underlying issues in self-configuration. When individual network components that are securely configured are connected together in an apparently secure manner, configuration cascades resulting in a misconfigured network are detected. Within the area of mobile-ad-hoc-networks (MANETS) the proposal from [So05] describes a formal approach to modeling and reasoning about auto-configuration protocols to support the detection of malicious nodes. Global security requirement for a network are defined that characterize the good behavior of individual nodes to assure a global property.

Work on configuring networks has also been done in prover-like contexts. [Ha15] focuses on the behavior of individual switches, and demonstrates that even simple configuration rule updates can result in inconsistent packet switching. The paper demonstrates that consistent configuration updates require guarantees of strong switch-level atomicity from both hardware and software layers of switches. [DKC15] presents the tool topoS which

automatically synthesizes low-level network configurations from high-level security goals. The tool topoS is formally verified with Isabelle/HOL, which prevents implementation errors. None of the above approaches employs standardized modeling languages like UML and OCL to describe networks.

6 Lessons Learned and Conclusion

We have performed a full analysis of the static configurations on a real-life network and were able to show the effectiveness of our approach with the example of a professional network. In particular, with the model and the USE tool, two incidents in the extracted configuration snapshot were uncovered. The process was fully automatic in that the existing configuration files were processed by a parser to create the system state of the model on which USE evaluates the model invariants and gives detailed information about the results. In case of a failed invariant, the cause is reported and can be used as an entry point for further analysis steps. The analysis of errors works on the model level, which gives network administrators an abstract view and control over the state of the network, e.g. connections of network components can be interactively followed using navigation in OCL. For this purpose, the model contains several query operations that perform commonly used tasks, such as following virtual links.

Our approach models network configurations from the bottom, starting with network components and physical cables from OSI layer 1. Third-party tools offered by network companies only allow one to inspect network configurations from the third layer or above. Therefore, configuration errors in lower layers cannot be detected using these tools and less convenient methods have to be used to analyze the lower layers, e.g. the low-level options `ping` and `traceroute`.

Aside from static analysis, representing the network as a UML system state allows for modifications that directly affect the whole network. These modifications can be analyzed before they are implemented whether they are free of errors and have the desired effect on the model, e.g. regarding reachability.

OCL and its formal basis allow for verbally formulated requirements of a network to be precisely represented using the network configuration model. Thus custom requirements can be automatically checked on a system state or be implemented as invariants, to be used in the future. These requirements can also be used to create a new network. The generation of the links and configurations can then be generated by a model completion tool like USE. Additional parameters can be defined using higher layers of the model.

The documentation options of the network configuration model are useful at all development stages. Before installing a network, the network requirements can be checked and during network maintenance, changes can be simulated. With the parser for the Cisco configurations, the system state can be generated from existing networks at any time reducing initial costs.

More parsers can be implemented to support more technologies and brands. These parsers also allow for continuous integration during maintenance and for continuous documentation.

Being able to query the model, visualize the system state and test setups before they are physically installed are only some of the general features for UML/OCL models. Also, besides the USE tool, a multitude of tools exists to analyze yet more properties of general UML/OCL models, like the configuration model. Certainly, it also would have been possible to use a DBMS and define integrity rules, but employing USE allows us to automatically generate a series of test cases rather than defining only single cases.

In conclusion, we believe that the presented methodology helps network administrators. Specifically, administration of large-scale computer networks often becomes tedious and error-prone, but the abstractions and gained overview enable a better understanding of networks—existing and new ones—and validation detect faults in the configurations. Finally, we do not claim that the network configuration model presented in this paper is fully exhaustive. Further investigations can lead to more improvements, but the experience so far supports the claim that it is worthwhile to invest into model-based automatic validation techniques for networks and that these techniques pose a benefit for all networks.

Bibliography

[AH16] Anderson, P.; Herry, H.: A Formal Semantics for the SmartFrog Configuration Language. J. Network Syst. Manage., 24(2):309–345, 2016.

[Al09] Al-Shaer, E.; Marrero, W.; El-Atawy, A.; Elbadawi, K.: Network Configuration in A Box: Towards End-to-End Verification of Network Reachability and Security. In: ICNP. IEEE Computer Society, pp. 123–132, 2009.

[Bi12] Birkholz, H.; Sieverdingbeck, I.; Sohr, K.; Bormann, Carsten: IO: An Interconnected Asset Ontology in Support of Risk Management Processes. In: Seventh International Conference on Availability, Reliability and Security, ARES 2012. IEEE Computer Society, pp. 534–541, 2012.

[BS14a] Birkholz, H.; Sieverdingbeck, I.: Improving root cause failure analysis in virtual networks via the interconnected-asset ontology. In: Proceedings of the Conference on Principles, Systems and Applications of IP Telecommunications, IPTComm 2014, Chicago, Illinois, USA, October 1-2, 2014. ACM, pp. 2:1–2:8, 2014.

[BS14b] Birkholz, H.; Sieverdingbeck, I.: Supporting Security Automation for Multi-chassis Link Aggregation Groups via the Interconnected-Asset Ontology. In: Ninth International Conference on Availability, Reliability and Security, ARES 2014. IEEE Computer Society, pp. 126–133, 2014.

[DKC15] Diekmann, C.; Korsten, A.; Carle, G.: Demonstrating topoS: Theorem-prover-based synthesis of secure network configurations. In (Tortonesi, M.; Schönwälder, J.; Madeira, E. R. Mauro; Schmitt, C.; Serrat, J., eds): 11th International Conference on Network and Service Management, CNSM 2015, Barcelona, Spain, November 9-13, 2015. IEEE Computer Society, pp. 366–371, 2015.

[FM10] Fall, M.N.; Mejri, M.: Network Security: Formal and Optimized Configuration. In (Fujita, H., ed.): New Trends in Software Methodologies, Tools and Techniques - Proceedings of the 9th SoMeT_10, September 29 - October 1, 2010, Yokohama City, Japan. volume 217 of Frontiers in Artificial Intelligence and Applications. IOS Press, pp. 229–246, 2010.

[Fo06] Foley, S.N.; Fitzgerald, W.M.; Bistarelli, S.; O'Sullivan, B.; Foghlú, M.: Principles of Secure Network Configuration: Towards a Formal Basis for Self-configuration. In: Autonomic Principles of IP Operations and Management, 6th IEEE International Workshop on IP Operations and Management, IPOM 2006, Dublin, Ireland, Proceedings. pp. 168–180, 2006.

[GBR07] Gogolla, M.; Büttner, F.; Richters, M.: USE: A UML-based specification environment for validating UML and OCL. Sci. Comput. Program., 69(1-3):27–34, 2007.

[GHD17] Gogolla, M.; Hilken, F.; Doan, K.-H.: Achieving Model Quality through Model Validation, Verification and Exploration. Journal on Computer Languages, Systems and Structures, Elsevier, NL, 2017. Online 2017-12-02.

[Ha15] Han, J.; Mundkur, P.; Rotsos, C.; Antichi, G.; Dave, N.H.; Moore, A. W.; Neumann, P. G.: Blueswitch: Enabling Provably Consistent Configuration of Network Switches. In (Brebner, G.J.; Bachmutsky, A.; Das, C., eds): Proceedings of the Eleventh ACM/IEEE Symposium on Architectures for networking and communications systems, ANCS 2015, Oakland, CA, USA, May 7-8, 2015. IEEE Computer Society, pp. 17–27, 2015.

[IE14a] IEEE Computer Society: . IEEE Standard for Local and Metropolitan Area Networks: Bridges and Bridged Networks, 11 2014.

[IE14b] IEEE Computer Society: . IEEE Standard for Local and Metropolitan Area Networks: Link Aggregation, 12 2014.

[IT94] ITU-T – International Telecommunication Union: . X.200: Information technology – Open Systems Interconnection – Basic Reference Model: The basic model, 7 1994.

[KVM12] Kazemian, P.; Varghese, G.; McKeown, N.: Header Space Analysis: Static Checking for Networks. In: NSDI. USENIX Association, pp. 113–126, 2012.

[Ma15] Maity, S.; Bera, P.; Ghosh, S. K.; Al-Shaer, E.: Formal integrated network security analysis tool: formal query-based network security configuration analysis. IET Networks, 4(2):137–147, 2015.

[Ro06] Rolando, M.; Rossi, M.; Sanarico, N.; Mandrioli, D.: A formal approach to sensor placement and configuration in a network intrusion detection system. In (Bruschi, D.; Win, B. De; Monga, M., eds): Proceedings of the 2006 international workshop on Software engineering for secure systems, SESS 2006, Shanghai, China, May 20-21, 2006. ACM, pp. 65–71, 2006.

[So05] Song, T.; Ko, C.; Tseng, C. H.; Balasubramanyam, P.; Chaudhary, A.; Levitt, K. N.: Formal Reasoning About a Specification-Based Intrusion Detection for Dynamic Auto-configuration Protocols in Ad Hoc Networks. In (Dimitrakos, T.; Martinelli, F.; Ryan, P. Y. A.; Schneider, S. A., eds): Formal Aspects in Security and Trust, Third International Workshop, FAST 2005, Newcastle upon Tyne, UK, July 18-19, 2005, Revised Selected Papers. volume 3866 of LNCS. Springer, pp. 16–33, 2005.

[TW11] Tanenbaum, A.-S.; Wetherall, D.: Computer Networks. Pearson Prentice Hall, 5 edition, 2011.

[Xi05] Xie, G.G.; Zhan, J.; Maltz, D.A.; Zhang, H.; Greenberg, A.G.; Hjálmtýsson, G.; Rexford, J.: On static reachability analysis of IP networks. In: INFOCOM. IEEE, pp. 2170–2183, 2005.

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 173

Modellierung, Verifikation und Synthese von validen Planungszuständen für Fernsehausstrahlungen

Imke Drave[1], Timo Henrich[2], Katrin Hölldobler[1]
Oliver Kautz[1], Judith Michael[1], Bernhard Rumpe[1]

Abstract: Für die Gestaltung von audiovisuellen Medienangeboten wie Fernsehprogramm und Video-On-Demand Angeboten müssen Lizenzverträge abgeschlossen werden. Aus diesen Verträgen ergeben sich Vorgaben, die bei der Planung der Angebote eingehalten werden müssen. Mit wachsender Anzahl an Verträgen und Formulierungen innerhalb der Verträge und ohne eine Möglichkeit diese einheitlich bzw. formal zu beschreiben, besteht das Risiko von Fehlinterpretationen und daraus resultierenden Fehlplanungen, welche zu Nachverhandlungen oder Vertragsstrafen führen können. Diese Arbeit beschreibt eine Domänenspezifische Sprache (DSL), die die Modellierung solcher Restriktionen durch Endanwenderinnen ohne tieferliegende Informatikkenntnisse ermöglicht. Darüber hinaus wurde die Sprache genutzt um eine Verifikation von Planungen sowie die Berechnung von erlaubten Verplanungszeiträumen zu automatisieren. Der praktische Einsatz dieser DSL in der Programmplanung hat gezeigt, dass Missverständnisse im Bezug auf die Bedeutung einer Restriktion minimiert, die Ermittlung erlaubter Verplanungszeiträume automatisiert und somit das Risiko von Fehlplanungen deutlich reduziert werden konnte.

Keywords: Domänenspezifische Sprache; Programmplanung; Verifikation; Vertragliche Restriktionen

1 Einleitung

Motivation und Problemstellung. Die Planung von Fernsehausstrahlungen ist an eine hohe Anzahl vertraglicher Regularien gebunden, wie beispielsweise die Anzahl und/oder Zeiträume, zu denen Wiederholungen gespielt oder bestimmte Staffeln gestartet, bzw. auf welchen Sendern Filme gezeigt werden dürfen. Diese Regelungen sind durchwegs *komplex und spezifisch*. Eine hohe Anzahl an Verträgen führt zudem zu einer großen Restriktionsdichte, mit der die Sendeplanung zu kämpfen hat. Ein Verstoß gegen diese Regelungen, beispielsweise durch Fehlinterpretationen und daraus folgender Fehlplanung, kann zu Nachverhandlungen oder Vertragsstrafen führen.

Die sich aus den Verträgen ergebenden Restriktionen wurden zur formalen Beschreibung von Restriktionen *mündlich* oder in Form von *informellen Kommentaren* an die Programmplaner weitergegeben. Infolgedessen gab es *keine einheitliche Notation für die Restriktionen*, wodurch gleiche Restriktionen durch verschiedene Kommentare beschrieben werden aber auch der gleiche Kommentar von verschiedenen Personen unterschiedlich interpretiert

[1] Software Engineering, RWTH Aachen University, Aachen, Germany
[2] CBC Cologne Broadcasting Center GmbH, Cologne, Germany

werden konnte. Ohne eine formale Beschreibung war es zudem nicht möglich die *Einhaltung der Restriktionen automatisiert zu überprüfen* oder *potenzielle Verplanungszeiträume zu berechnen*. All dies erfolgte vor Entwicklung der im Folgenden erläuterten Domänenspezifische Sprache (DSL) und deren Infrastruktur *manuell* durch die Programmplaner, wodurch Fehlplanungen möglich waren. Fehlplanungen können eine Neuverhandlung der Lizenzverträge erfordern oder sogar zu Vertragsstrafen führen. Hinzu kommt, dass die *Komplexität* der Restriktionen mit einer wachsenden Anzahl an zusätzlichen audiovisuellen Angebotsmöglichkeiten stetig steigt. Hieraus hat sich zudem der Bedarf für eine formale Beschreibung mit festgelegter Semantik ergeben. Eine solche Beschreibung bietet die Möglichkeit zur automatisierten Überprüfung der Einhaltung der vertraglichen Restriktionen und Berechnung von Zeitfenstern, in denen eine Verwertung stattfinden darf.

Forschungsfragen. Die zuvor beschriebenen Probleme werfen die folgenden Forschungsfragen auf, die in diesem Papier beantwortet werden:

- Welche Anforderungen bestehen an eine DSL, die für die Beschreibung von Vertraglichen Restriktionen verwendet werden kann?
- Welche Syntax ist für diese DSL notwendig, um die Restriktionen durch Endanwenderinnen ohne tieferliegende Informatikkenntnisse beschreiben lassen zu können?
- Welche zusätzlichen Funktionalitäten sind notwendig, um die Programmplanung bei ihrer Planungstätigkeit zu unterstützen?

Hauptbeitrag dieser Arbeit. Diese Arbeit beschreibt eine DSL, die die Modellierung solcher Restriktionen durch Endanwenderinnen ohne tieferliegende Informatikkenntnisse ermöglicht. Neben den iterativ identifizierten *Anforderungen* an solch eine Sprache behandelt dieses Papier einen *kleinen Teil der wichtigsten Sprachelemente* der Restriktions-DSL sowie umgesetzte Funktionalitäten, die den *Arbeitsaufwand der Beteiligten reduziert*: Man kann beispielsweise die *Einhaltung* der Restriktionen in der Planungsphase *automatisiert überprüfen* oder während des Planungsprozesses *Vorschläge für Zeitfenster* erhalten, in denen eine Verwertung stattfinden darf, d.h. Zeitfenster, die keine Restriktionen verletzen. Der praktische Einsatz dieser DSL in der Programmplanung hat gezeigt, dass Missverständnisse in Bezug auf die Bedeutung einer Restriktion minimiert, die Ermittlung erlaubter Verplanungszeiträume automatisiert und somit das Risiko von Fehlplanungen deutlich reduziert werden konnte.

Aufbau. Der nächste Abschnitt geht genauer auf die Anwendungsdomäne ein und beschreibt die wichtigsten Begrifflichkeiten aus dem Bereich der Fernsehausstrahlung und Fernsehprogrammplanung. Abschnitt 3 fasst die wichtigsten Anforderungen an die entwickelte DSL zusammen und beschreibt daraus resultierende Designentscheidungen. Eine Teilmenge der wichtigsten Elemente der DSL wird in Abschnitt 4, durch Beispiele untermalt, erklärt. Abschnitt 5 geht genauer auf die Unterstützung des Planungsprozesses durch Verifikation von Planungszuständen sowie die Synthese der Verplanungsmöglichkeiten, d.h. den Vorschlag möglicher Zeitfenster, die keine Restriktionen verletzen, ein. Abschnitt 6 beschreibt verwandte Arbeiten aus diesem Themengebiet und zeigt den State-of-the-Art

dieses Forschungsbereichs auf. Der letzte Abschnitt fasst die Arbeit zusammen und bietet einen Ausblick auf mögliche Erweiterungen.

2 Begrifflichkeiten und Problemstellungen in der Fernsehdomäne

In diesem Abschnitt werden die Begrifflichkeiten aus dem Bereich der Fernsehausstrahlung und Fernsehprogrammplanung sowie die Fragestellungen, die zur Entwicklung einer DSL geführt haben, erläutert. Neben dem klassischen, über Fernsehgeräte abzurufendem Fernsehprogramm, hat inzwischen auch der Abruf von audiovisuellen Inhalten zu beliebigen Zeitpunkten (im Folgenden Video-On-Demand, kurz VOD) an Bedeutung gewonnen. Im Bezug auf audiovisuelle Medienangebote unterscheidet man zwischen dem linearen und dem nicht-linearen Bereich [Ha09]. Der lineare Bereich umfasst die klassische Ausstrahlung über Fernsehgeräte. Hierbei werden die verschiedenen Angebote wie Filme und Episoden zeitlich nacheinander zu vordefinierten Zeiten angeboten. Der nicht-lineare Bereich hingegen umfasst das On-Demand Angebot. Hierbei stehen audiovisuelle Angebote zeitgleich zum Abruf zur Verfügung.

Um ein audiovisuelles Medienangebot auszugestalten ist sowohl für den linearen als auch für den nicht-linearen Bereich der Erwerb von Lizenzen notwendig. Vertraglich werden hierbei die Rahmenbedingungen bzw. Restriktionen für die audiovisuellen Angebote festgelegt. Dies umfasst insbesondere 1) den Lizenzzeitraum, 2) die Anzahl der Ausstrahlungen (Runs) sowie Wiederholungen (Reruns), 3) die Anzahl und Länge der VOD Bereitstellungsmöglichkeiten sowie 4) die erlaubten Sender bzw. VOD Plattformen. Ausstrahlungen bzw. VOD Bereitstellungen dürfen nur innerhalb des Lizenzzeitraums stattfinden. Der Lizenzzeitraum definiert also den frühestmöglichen Start sowie das späteste erlaubte Ende einer Ausstrahlung bzw. Bereitstellung.

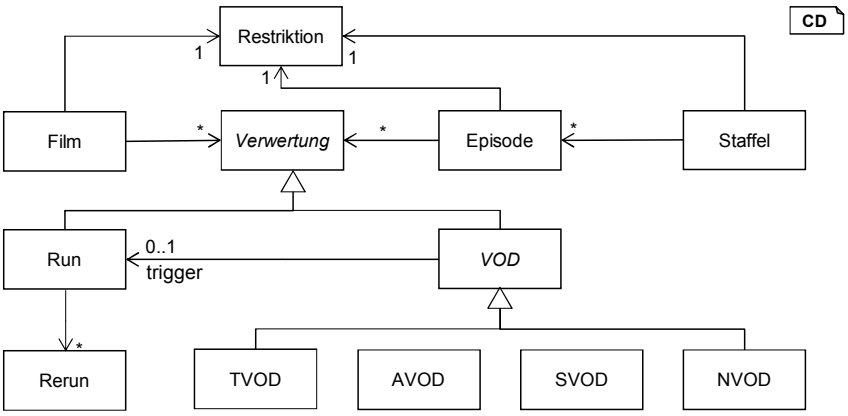


Abb. 1: Modell der Fernsehdomäne mit Fokus auf Verwertungen und deren Restriktionen

Sowohl die Ausstrahlungen im linearen Bereich als auch die Bereitstellung als VOD im nicht-linearen Bereich wird unter dem Begriff der Verwertung zusammengefasst. Eine Verwertung im nicht-linearen Bereich kann zudem abhängig von einer Verwertung im linearen Bereich sein. Ein typisches Beispiel hierfür ist die 7 tägige Bereitstellung als VOD im Anschluss an eine Ausstrahlung im linearen Bereich. In diesem Fall spricht man davon, dass die lineare Verwertung die nicht-lineare triggert. Bei VOD Verwertungen kann zwischen verschiedenen Finanzierungsvarianten unterschieden werden. So kann der Abruf gekauft werden (Transactional-VOD, kurz TVOD), durch Werbung (Advertising-VOD, kurz AVOD) oder durch ein Abonnement (Subscription-VOD, kurz SVOD) finanziert sein. Ebenfalls dem VOD Bereich zugerechnet wird Near-VOD (kurz NVOD). Hierbei wird die Ausstrahlung zeitversetzt zu festgesetzten Zeitpunkten angeboten, beispielsweise jede Viertelstunde. Der Zeitpunkt kann hierbei nicht komplett frei durch den Endverbraucher gewählt werden. Verwertungen gehören entweder zu Episoden, die wiederum zu einer Staffel gehören oder zu Sendungen, die nicht in Staffeln organisiert sind wie beispielsweise Filme oder Nachrichtensendungen (im Folgenden vereinfacht als Film bezeichnet). Jede Verplanung muss konform zu den im Lizenzvertrag festgelegten Bedingungen erfolgen. Entsprechend kann jedem Film, jeder Episode und sogar jeder Staffel eine Restriktion zugeordnet werden, die beschreibt welchen Einschränkungen die Verwertung genügen muss. Existiert sowohl für die Staffel als auch für die Episode eine Restriktion, so müssen beide Bedingungen erfüllt sein. Abbildung 1 visualisiert die zuvor erläuterten Begrifflichkeiten und Beziehungen in Form eines Klassendiagramms.

Für die Überprüfung, ob alle vertraglichen Bedingungen eingehalten werden, sowie für die Planung einer weiteren Verwertung, sind sowohl die *vergangenen Verwertungen* als auch der *aktuelle Planungsstand* relevant.

3 Anforderungen an die DSL und ihre Entwicklungsmethodik

Ziel der DSL ist die formale Beschreibung von vertraglichen Restriktionen. Zur Erreichung dieses Ziels wurde eine agile, an Scrum [Gl16] angelehnte Vorgehensweise gewählt. Die Entwicklung erfolgte in 2-3-wöchigen Sprints. In enger Zusammenarbeit mit dem Kunden konnte nach jedem Sprint das entstandene Inkrement des Produkts besprochen sowie neue bzw. veränderte Anforderungen identifiziert werden. Die Entwicklung der DSL erfolgte auch entlang der Domäne inkrementell. Zunächst wurden Modellierungselemente für die Formulierung von Restriktionen im linearen Bereich entwickelt. Hierzu wurden sukzessive die verschiedenen Möglichkeiten Ausstrahlungen bzw. Bereitstellungen einzuschränken identifiziert und die DSL hierfür umgesetzt. Anschließend wurde die DSL im weiteren Verlauf der Entwicklung auf den nicht-linearen Bereich ausgeweitet.

Neben der Anforderung, dass die in den aktuellen Verträgen vorhandenen Einschränkungen mithilfe der DSL ausgedrückt werden können, wurden zusammenfassend die folgenden weiteren Anforderungen an die DSL identifiziert.

1. Da die Modellierer nicht zwangsläufig einen technischen Hintergrund haben, sollen
   a) mithilfe der DSL Restriktionen angelehnt an die natürliche Sprache formuliert werden können,
   b) einfache Strukturen komplexer Logik vorgezogen werden und
   c) Formatierung und Formulierungen bzw. Prioritäten einer Klammerung vorgezogen werden.
2. Es wird grundsätzlich in deutscher Sprache formuliert, teils gemischt mit englischen Fachbegriffen.
3. Die DSL soll die Grundannahme treffen, alles nicht explizit Verbotene, erlaubt sei.
4. Für die DSL vorgegeben und nicht durch diese veränderlich sind der Lizenzzeitraum, die erlaubten Sender bzw. Plattformen (im Folgenden unter dem Begriff *Nutzer* zusammengefasst) sowie die insgesamt maximal erlaubte Anzahl an Verwertungen und Wiederholungen.
5. Neben der reinen Formulierung von Restriktionen soll
   a) eine Verifikation der aktuellen Planung anhand der Restriktionen sowie
   b) eine Synthese aller möglichen Zeitfenster, in denen die Planung einer Verwertung möglich ist, realisiert werden.
6. Kompaktheit von Restriktionen soll durch die Definition von Syntactic Sugar und Defaults erreicht werden.

Durch die Anforderungen 1a und 1b war die Designentscheidung klar, dass weder eine mathematische noch eine OCL-basierte Notation verwendet wird. Stattdessen werden logische Operatoren wie Konjunktion, Disjunktion und Negation durch natürliche Sprache formuliert (vgl. Abschnitt 4).

Durch Anforderung 4 ergibt sich der Rahmen, innerhalb dessen Restriktionen die Verwertungen weiter beschränken können. Beispiele hierfür sind gesperrte Zeiträume innerhalb des Lizenzzeitraums, nur bestimmte Anzahlen (kleiner als die maximale Anzahl) erlaubter Verwertungen innerhalb bestimmter Zeiträume (z.B. "nur 3 Verwertungen im Januar") oder Zeitrahmen, innerhalb derer Reruns stattfinden können (z.B. "72 Stunden nach dem Run").

4 Syntax der Sprache

Die Modelle der DSL heißen *Restriktionen*. Eine Restriktion besteht aus mehreren Alternativen, die mehrere Regeln beinhalten können. Die Restriktion kann als logische Disjunktion von konjugierten Regeln aufgefasst werden. Die Regeln sind syntaktisch in mehrere Arten klassifizierbar. Im Folgenden werden der allgemeine syntaktische Aufbau von Restriktionen, sowie die einzelnen Regelarten beschrieben.

```
1 Alternative 1:
2 Der 1. & 4. Run innerhalb von 01.01.2018 bis 30.06.2018 turnus immer von 20:00 bis
3 22:00 auf Nutzer XYZ
4 Alternative 2:
5 Alle Run innerhalb von 01.01.2018 bis 30.06.2018 turnus ohne auf Nutzer XYZ
```

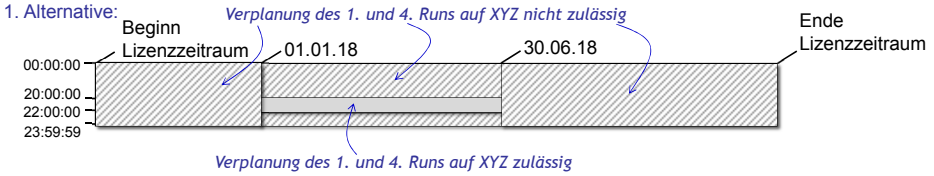


Abb. 2: Bezug im `Was` einer Regel.

```
1 Kein Run innerhalb von 30.06.2018 bis 31.12.2018 turnus ohne auf Nutzer XYZ
2 UND
3 Maximal 1 Run innerhalb von 01.01.2018 bis 31.12.2019 turnus jährlich von Januar
4 Beginn bis Februar Ende auf Nutzer XYZ
```

Abb. 3: Kunjunktion von Regeln und genereller Aufbau von Restriktionen.

4.1 Allgemeiner Aufbau von Restriktionen

Wie in Abbildung 2 dargestellt, werden Disjunktionen über `Alternativen` beschrieben. Diese werden über das Schlüsselwort `Alternative` eingeleitet und können nummeriert werden. Konjunktionen werden durch das Schlüsselwort `UND`, das zwischen zwei Regeln gesetzt wird, gekennzeichnet. Ein Beispiel hierfür ist in Abbildung 3 zu sehen. Jede Regel aus den vier Teilen *Was*, *Wann*, *Zyklus* und *Wo*.

Was. Innerhalb des *Was* wird angegeben, welche Art der Verwertung mit der Restriktion eingeschränkt wird und ob eine bestimmte Verwertung oder die Anzahl der Verwertungen restringiert sind. Essentiell für den Was-Teil einer Regel ist die Art der restringierten Verwertung. Die Verwertungsart kann auf einen Spezifikator (`Der` oder `Alle`), oder einen Quantifikator (`Maximal` oder `Kein`) folgen.

Wann. Der *Wann*-Teil einer Regel gibt den Zeitraum an, der durch die Regel restringiert wird und wird durch das Schlüsselwort `innerhalb von` eingeleitet. Zum Beispiel wird in der ersten in Abbildung 2 dargestellten Regel der Zeitraum vom ersten Januar 2018 bis zum dreißigsten Juni 2018 restringiert.

Zyklus. Mithilfe des Zyklus kann der angegebene Zeitraum zyklisch unterteilt werden. Mit der Angabe von Zyklen können Regeln in kompakter Form formuliert werden, die für mehrere äquidistant verteilte Zeitintervalle gelten sollen. Ein Beispiel hierfür ist ein Run, der auf einem bestimmten Nutzer nur zur Prime-Time, also zwischen 20:00 und 22:00 Uhr, ausgestrahlt werden darf. Alternative 1 in Abbildung 2 bildet diese Restriktion ab. Das einleitende Schlüsselwort ist in diesem Fall `turnus`. In diesem Beispiel müssen der erste Run

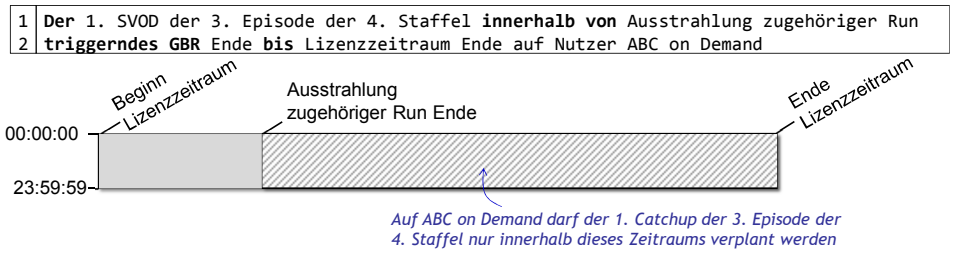


Abb. 4: Eine Restriktion mit einer Referenz auf die Verplanung eines triggernden Runs.

und der vierte Run in der ersten Jahreshälfte von 2018 an einem beliebigen Tag zwischen 20:00 und 22:00 Uhr verplant werden, falls sie auf dem Nutzer `XYZ` verplant werden. Eine Verplanung am 02.01.2018 um 18:00 Uhr auf dem Nutzer `XYZ` wäre beispielsweise nicht zulässig. Das Schlüsselwort `immer` schränkt den zulässigen Verplanungszeitraum auf die nachfolgend angegebene relative Zeitspanne ein (im Beispiel 20:00 bis 22:00 Uhr). Die relative Zeitspanne kann auch durch Daten, Wochentage oder Monate definiert werden. Ein Beispiel für letzteres ist die zweite Regel in Abbildung 3 (Zeile 3). Hier wird ein `jährlicher`-Zyklus angegeben, was bedeutet, dass die Regel jedes Jahr auf den angegebenen relativen Zeitraum (Anfang Januar bis Ende Februar) anzuwenden ist. Weitere mögliche Zyklen sind `wöchentlich`, `monatlich` oder `täglich`. Der Zyklus `ohne` bedeutet, dass die Regel den gesamten `Wann`-Zeitraum restringiert (vgl. Beispiel in Abbildung 2, Zeile 5).

Wo. Das *Wo* einer Regel beschreibt die Nutzer, auf denen Verplanungen durch die Regel eingeschränkt werden. Verplanungen auf Nutzern, die nicht durch den Wo-Teil einer Regel beschrieben werden, sind durch die Regel nicht restringiert. Über die Phrase `auf Nutzer` wird das Konstrukt eingeleitet. Anschließend müssen einzelne Nutzer oder Nutzergruppen beschrieben werden.

Trigger und zugehörig. Wie in Abschnitt 2 beschrieben, werden nicht-lineare Verwertungen häufig durch Ausstrahlungen im linearen Bereich getriggert. Die Verplanungen einer nicht-linearen Verwertung erfolgt dann in der Regel relativ zur Verplanung der zugehörigen triggernden linearen Verwertung. In Abbildung 4 ist ein Beispiel gezeigt, das das Ausstrahlungsende des triggernden Runs eines SVODs referenziert. Die Regel spezifiziert, dass das referenzierte SVOD nicht vor dem Ende der Ausstrahlung des zugehörigen triggernden Runs auf dem angegebenen Nutzer verplant werden darf. Zusätzlich kann die im *Was* referenzierte Verwertung durch Angabe eines Bezugs auf Runs, Episoden oder Staffeln weiter spezifiziert werden (vgl. Abbildung 4, `der 3. Episode der 4. Staffel`).

4.2 Arten von Regeln

Spezifizierende Regeln. Eine Regel gilt als *spezifizierend*, wenn der Was-Teil einer Regel einen Spezifikator beinhaltet. Ein Spezifikator wird über die Schlüsselwörter `Alle`, `Der`

```
1 SVOD zu maximal 2 Episoden gleichzeitig innerhalb von 01.01.2018 bis 30.06.2018 auf
2 Nutzer ABC on Demand
```

```
1 SVOD zu maximal 2 Runs gleichzeitig innerhalb von 01.01.2018 bis 30.06.2018 auf
2 unterschiedlichem Nutzer
```

Abb. 5: Beispiele einer Gleichzeitige-Verplanung und einer Gleichzeitige-Nutzer Regel.

oder `Die` eingeleitet. Ein Spezifikator kann eine durch `&`-getrennte Liste (vgl. Alternative 1 in Abbildung 2) oder eine Spanne (zum Beispiel `2.-6.`) beinhalten. Weiterhin können mithilfe des Schlüsselwortes `Alle` alle Verwertungen, für die die Restriktion hinterlegt ist, restringiert werden. Beispiele für spezifizierende Regeln sind in Abbildung 2 sowie in Abbildung 4 dargestellt. Es gilt jeweils, dass die Verplanungen aller nicht referenzierten Verwertungen nicht restringiert sind.

Quantifizierende Regeln. Quantifizierende Regeln zeichnen sich durch einen *Quantifikator* zu Beginn des Was-Teils aus. Es gibt zwei solcher Quantifikatoren, `maximal` und `kein`. Nach dem Schlüsselwort `maximal` muss immer eine positive ganze Zahl stehen, wobei `Maximal 0` äquivalent zum Schlüsselwort `Kein` ist. Beispiele für quantifizierende Regeln sind in Abbildung 3 dargestellt.

Gleichzeitige-Verplanung Regeln. *Gleichzeitige-Verplanung Regeln* restringieren die gleichzeitige Verplanung von Verwertungen. Im oberen Teil von Abbildung 5 ist ein Beispiel einer solchen Regel angegeben. Sie besagt, dass auf dem Nutzer `ABC on Demand` innerhalb des genannten Zeitraumes nur maximal zwei verschiedene Episoden gleichzeitig verplant werden dürfen. Auf allen anderen Nutzern, beziehungsweise außerhalb des angegebenen Zeitraums, ist die Verwertung der SVODs nicht restringiert.

Gleichzeitige-Nutzer Regeln. *Gleichzeitige-Nutzer Regeln* restringieren die gleichzeitige Verwertung auf unterschiedlichen Nutzern. Im unteren Teil von Abbildung 5 werden SVODs auf unterschiedlichen Nutzern restringiert. Ist diese Regel hinterlegt, dann dürfen auf unterschiedlichen Nutzern im angegebenen Zeitraum gleichzeitig maximal zwei SVODs verplant werden. Auf diesen beliebigen zwei Nutzern dürfen jedoch beliebig viele Verwertungen verplant werden.

5 Interpretationen von Restriktionen

Dieser Abschnitt behandelt die Verifikation von Planungszuständen und die Synthese von möglichen Verwertungsfenstern zur Unterstützung des Verplanungsvorgangs.

Um die Begriffe Verifikation und Synthese genauer zu erläutern, benötigt man das Konzept des *Planungszustands*, welches auf Verplanungstypen, Zeitfenstern und Mengen von Nutzern basiert. Diese Begriffe werden in Unterabschnitt 5.1 eingeführt. Zur Synthese werden elementare Operationen auf Zeitfenstermengen benötigt, welche in Unterabschnitt 5.2

Verwertung	Verplanungstyp	Zeitfenster	Nutzer
1	SVOD	[01.01.2019 00:00:00 - 28.02.2019 23:59:59]	Sender1
2	AVOD	[01.02.2019 00:00:00 - 31.03.2019 23:59:59]	Sender2
3	SVOD	[01.01.2019 00:00:00 - 31.12.2019 23:59:59]	Sender1

Abb. 6: Beispiel für einen Verplanungszustand in tabellarischer Darstellung

präsentiert werden. Anschließend beschreibt Unterabschnitt 5.3 den Verifikations- und den Synthesealgorithmus.

5.1 Verplanungstypen, Zeitfenster, Nutzer, Verplanungszustände

Die Menge der möglichen *Verplanungstypen* ist gleich der Menge der Verwertungstypen Run, SVOD, AVOD, TVOD, NVOD, Rerun. Verplanungstypen werden genutzt, um zu spezifizieren, mit welchem Typen eine Verwertung verplant wird.

Ein Zeitpunkt ist ein bis auf Sekundenebene voll-spezifiziertes Datum. Ein *Zeitfenster* besteht aus einem Startzeitpunkt und einem Endzeitpunkt, wobei der Startzeitpunkt vor dem Endzeitpunkt liegt. Ein Zeitfenster repräsentiert die Menge aller Zeitpunkte zwischen (und inklusive) Startzeitpunkt und des Endzeitpunkt. Beispielsweise beschreibt [01.01.2019 00:00:00 - 31.12.2019 23:59:59] das Zeitfenster, das am 01.01.2019 um Mitternacht beginnt und in der letzten Sekunde des Jahres 2019 endet. Zeitfenster werden genutzt, um zu spezifizieren, in welchem Zeitraum eine Verwertung verplant wird.

Die Menge der *Nutzer* beinhaltet alle möglichen Sender für Fernsehausstrahlungen. Ein Nutzer kann verwendet werden, um zu spezifizieren, auf welchem Sender eine Verwertung verplant wird.

Ein *Planungszustand* über einer endlichen Menge von Verwertungen ist eine Zuordnung, die jeder Verwertung der Menge einen Verplanungstypen, ein Zeitfenster und einen Nutzer zuordnet. Abbildung 6 zeigt ein Beispiel für einen Verplanungszustand in tabellarischer Darstellung. So ist zum Beispiel die Verwertung 3 als SVOD im Zeitfenster [01.01.2019 00:00:00 - 31.12.2019 23:59:59] auf dem Nutzer Sender1 verplant.

5.2 Elementare Operationen auf Zeitfenstermengen

Der Synthesealgorithmus basiert auf den elementaren Operationen des Schnitts, der Vereinigung und der Differenz von Mengen paarweise disjunkter Zeitfenster. Eine Menge von Zeitfenstern (im Folgenden auch Zeitfenstermenge genannt) repräsentiert die Vereinigung der Mengen der Zeitpunkte, die durch die Zeitfenster der Menge repräsentiert werden. Zwei Zeitfenster werden *disjunkt* genannt, wenn jeder Zeitpunkt, der durch das eine Zeitfenster

repräsentiert wird, ungleich zu jedem Zeitpunkt ist, der durch das andere Zeitfenster repräsentiert wird. Die Darstellung einer Menge von Zeitpunkten durch die kleinste Menge von paarweise disjunkten Zeitfenstern, die genau die Menge der Zeitpunkte repräsentiert, ist eine kanonische Darstellungsform.

Die *Vereinigung* zweier Zeitfenstermengen ist die kleinste Menge paarweise disjunkter Zeitfenster, die alle Zeitpunkte repräsentiert, die von mindestens einer der beiden Zeitfenstermengen repräsentiert werden. Der *Schnitt* zweier Zeitfenstermengen ist analog definiert. Die *Differenz* von einer Zeitfenstermenge zu einer anderen Zeitfenstermenge ist die kleinste Menge paarweise disjunkter Zeitfenster, die genau die Menge der Zeitpunkte der ersten Menge repräsentiert, die nicht durch die zweite Menge repräsentiert werden.

5.3 Verifikation und Synthese

Die *Verifikation* eines Planungszustands bezüglich einer Restriktion ist die Überprüfung, ob der Planungszustand die in der Restriktion spezifizierten Einschränkungen erfüllt.

Restriktionen beschränken die möglichen Verplanungen von Verwertungen. Jede Restriktion besteht aus einer Menge von Alternativen. Jede Alternative besteht aus einer Menge von Regeln. Ein Verplanungszustand erfüllt eine Restriktion genau dann, wenn der Verplanungszustand mindestens eine der Alternativen erfüllt. Ein Verplanungszustand erfüllt eine Alternative genau dann, wenn der Verplanungszustand jede Regel der Alternative erfüllt. Um zu überprüfen, ob ein Verplanungszustand eine Restriktion erfüllt, muss also geprüft werden, ob der Verplanungszustand jede Regel mindestens einer Alternative erfüllt.

Die *Synthese* der Verplanungsmöglichkeiten für eine Verwertung bezüglich eines Planungszustands, eines Nutzers und einer Restriktion ist die Berechnung der kleinsten Zeitfenstermenge, die alle Zeitpunkte repräsentiert in der die Verwertung auf dem Nutzer verplant werden könnte, sodass der entstehende Verplanungszustand die Restriktion erfüllt. Jedes Zeitfenster in der berechneten Zeitfenstermenge ist durch den für die Verwertung hinterlegten Lizenzzeitraum beschränkt.

In der Synthese wird für jede Regel eine Zeitfenstermenge berechnet, die alle Zeitpunkte repräsentiert, zu denen eine Verplanung der gegebenen Verwertung auf dem gegebenen Nutzer bezüglich des gegebenen Verplanungszustands und der Regel möglich ist. Die Zeitfenstermenge für eine Alternative ergibt sich aus dem Schnitt der Zeitfenstermengen der Regeln der Alternative. Die Zeitfenstermenge für eine Restriktion ergibt sich aus der Vereinigung der Zeitfenstermengen der Alternativen der Restriktion. Vor der Synthese für eine Regel wird zunächst geprüft, ob der Verplanungszustand die Regel erfüllt. Falls dies nicht der Fall ist, dann kann keine Verplanung der zu verplanenden Verwertung einen validen Planungszustand ergeben. Somit wird in diesem Fall die leere Menge zurückgegeben. Bei der Beschreibung der Syntheseprozeduren vernachlässigen wir die Beschreibung der

```
1 Maximal 3 Run innerhalb von 01.01.2019 bis 31.12.2019 auf Nutzer Sender1, Sender2
```

Abb. 7: Eine quantifizierende Regel.

Verifikationen, die vorab durchgeführt werden. Wir beschränken die Beschreibungen somit auf den Fall, dass der gegebene Planungszustand die Regel erfüllt.

5.3.1 Verifikation und Synthese für quantifizierende Regeln

Jede quantifizierende Regel beschränkt die maximale Anzahl A an Verplanungen von Verwertungen aus einer Verwertungsmenge V in einer Zeitfenstermenge Z auf einer Menge von Nutzern N.

Beispielsweise sagt die in Abbildung 7 dargestellte Regel aus, dass maximal drei Runs innerhalb des Zeitfensters vom 01.01.2019 bis zum 31.12.2019 auf den Nutzern `Sender1` oder `Sender2` verplant werden dürfen.

Zur Verifikation, ob ein Verplanungszustand die Regel erfüllt, wird

- unter den in der Regel referenzierten Verwertungen aus der Menge V,
- die in einem der Zeitfenster aus der Zeitfenstermenge Z verplant sind,
- die Anzahl der Verwertungen gezählt, die
- auf einem Nutzer aus N verplant sind
- und geprüft, ob die gezählte Anzahl kleiner oder gleich der Maximalanzahl A ist.

Zur Synthese wird zunächst geprüft, ob die gegebene Verwertung in der Menge der referenzierten Verwertungen V enthalten ist und ob der gegebene Nutzer in der Menge der Nutzer N enthalten ist. Ist dies nicht der Fall, dann restringiert die Regel die Verplanung der gegebenen Verwertung auf dem gegebenen Nutzer nicht. Somit wird der Lizenzzeitraum der Verwertung zurückgegeben. Ansonsten wird die Anzahl der verplanten Verwertungen aus der Menge V gezählt, die in einem Zeitfenster aus der Menge Z auf einem Nutzer aus der Menge N verplant wurden. Ist diese Anzahl nicht kleiner als die Maximalanzahl A, so darf die gegebene Verwertung nicht mehr in den Zeitfenstern der Menge Z auf dem angegebenen Nutzer verplant werden, da die Regel ansonsten nicht mehr erfüllt wäre. In diesem Fall gibt die Synthese die Differenz vom Lizenzzeitraum der gegebenen Verwertung zu der Zeitfenstermenge Z zurück. Ansonsten gibt die Synthese den Lizenzzeitraum zurück.

5.3.2 Verifikation und Synthese für spezifizierende Regeln

Jede spezifizierende Regel beschränkt die Verplanung von Verwertungen aus einer Verwertungsmenge V auf einer Zeitfenstermenge Z für eine Menge von Nutzern N. Die

```
1 Die 2., 3. Verwertung innerhalb von 01.01.2019 bis 31.12.2019 auf Nutzer Sender1, Sender2
```

Abb. 8: Eine spezifizierende Regel.

Verplanungsmöglichkeiten auf den Nutzern, die nicht in der Menge N enthalten sind, werden nicht eingeschränkt.

Beispielsweise sagt die in Abbildung 8 dargestellte Regel aus, dass für die zweite und für die dritte Verwertung gilt, dass sie innerhalb des Jahres 2019 verplant werden müssen, falls sie auf `Sender1` oder `Sender2` verplant werden. Die Verplanung auf anderen Nutzern ist uneingeschränkt.

Zur Verifikation, ob ein Verplanungszustand die Regel erfüllt, wird

- für jede in der Regel referenzierten Verwertung aus der Menge V überprüft,
- ob sie innerhalb eines der Zeitfenster aus der Menge Z verplant ist, falls
- sie auf einem Nutzer aus der Menge N verplant ist.

Zur Synthese wird zunächst geprüft, ob die gegebene Verwertung in der Menge der referenzierten Verwertungen V enthalten ist und ob der gegebene Nutzer in der Menge der Nutzer N enthalten ist. Ist dies nicht der Fall, dann restringiert die Regel die Verplanung der gegebenen Verwertung auf dem gegebenen Nutzer nicht. Somit wird der Lizenzzeitraum der Verwertung zurückgegeben. Ansonsten wird die Zeitfenstermenge Z zurückgegeben.

5.3.3 Verifikation und Synthese für gleichzeitige-Verplanung Regeln

Jede gleichzeitige-Verplanung Regel beschränkt die maximale Anzahl A an Verplanungen von Verwertungen aus einer Verwertungsmenge V, die innerhalb der Zeitfenster aus einer Zeitfenstermenge Z auf den Nutzern aus einer Nutzermenge N gleichzeitig verplant sein dürfen.

Zum Beispiel sagt die in Abbildung 9 abgebildete Regel aus, dass auf den Nutzern `Sender1` und `Sender2` keine SVOD innerhalb des Jahres 2019 gleichzeitig verplant sein dürfen.

Zur Verifikation, ob ein Verplanungszustand die Regel erfüllt, wird

- für die in der Regel referenzierten Verwertungen aus der Menge V,
- die in einem der Zeitfenster aus der Menge Z und
- auf einem Nutzer aus der Menge N verplant sind,
- die größte Anzahl an Verwertungen bestimmt, dessen Planungszeiträume sich paarweise überschneiden
- und geprüft, ob die gezählte Anzahl kleiner oder gleich der Maximalanzahl A ist.

```
1 SVOD zu maximal 0 Runs pro Episode gleichzeitig
2                                    innerhalb von 01.01.2019 bis 31.12.2019
3                                    auf Nutzer Sender1, Sender2
```

Abb. 9: Eine gleichzeitige-Verplanung Regel.

Zur Synthese wird zunächst geprüft, ob die gegebene Verwertung in der Menge der referenzierten Verwertungen V enthalten ist und ob der gegebene Nutzer in der Menge der Nutzer N enthalten ist. Ist dies nicht der Fall, dann restringiert die Regel die Verplanung der gegebenen Verwertung auf dem gegebenen Nutzer nicht. Somit wird der Lizenzzeitraum der Verwertung zurückgegeben. Ansonsten wird für jede Teilmenge der Größe A der Verwertungen aus V, die in einem der Zeitfenster aus Z auf einem Nutzer aus N verplant sind, der Schnitt der Planungszeiträume der Verwertungen der Teilmenge gebildet. Falls der Schnitt der Planungszeiträume der Verwertungen einer Teilmenge leer ist, dann gibt es kein Zeitfenster an dem die A Verwertungen der Teilmenge gleichzeitig verplant sind. Ansonsten ist der Schnitt ein Zeitfenster, an dem alle A Verwertungen der Teilmenge gleichzeitig verplant sind. Eine Verplanung der angefragten Verwertung auf dem angefragten Nutzer ist in diesem Zeitfenster daher nicht mehr zulässig. Es wird das Resultat zurückgegeben, das entsteht, wenn man die wie oben berechneten Schnitte sukzessive vom Lizenzzeitraum der gegebenen Verwertung subtrahiert.

5.3.4 Verifikation und Synthese für gleichzeitige-Nutzer Regeln

Jede gleichzeitige-Nutzer Regel beschränkt die maximale Anzahl A von Verplanung von Verwertungen aus einer Verwertungsmenge V, die innerhalb der Zeitfenster aus einer Zeitfenstermenge Z gleichzeitig auf unterschiedlichen Nutzern verplant werden dürfen.

Beispielsweise sagt die in Abbildung 10 dargestellte Regel aus, dass maximal eine Verwertung innerhalb des Jahres 2019 gleichzeitig auf unterschiedlichen Nutzern verplant sein darf. Sind also zwei Verwertungen innerhalb des Jahres 2019 gleichzeitig verplant, dann müssen diese auf demselben Nutzer verplant sein.

Zur Verifikation, ob ein Verplanungszustand die Regel erfüllt, wird

- für die in der Regel referenzierten Verwertungen aus der Menge V,
- die in einem der Zeitfenster aus der Menge Z verplant sind,
- die größte Anzahl an Verwertungen bestimmt, dessen Planungszeiträume sich paarweise überschneiden und Nutzer sich paarweise unterscheiden
- und geprüft, ob die gezählte Anzahl kleiner oder gleich der Maximalanzahl A ist.

Zur Synthese wird zunächst geprüft, ob die gegebene Verwertung in der Menge der referenzierten Verwertungen V enthalten ist. Ist dies nicht der Fall, dann restringiert die Regel die Verplanung der gegebenen Verwertung nicht. Somit wird der Lizenzzeitraum

```
1 SVOD zu maximal 1 Runs pro Episode gleichzeitig
2                                   innerhalb von 01.01.2019 bis 31.12.2019
3                                   auf unterschiedlichen Nutzern
```

Abb. 10: Eine gleichzeitige-Nutzer Regel.

der Verwertung zurückgegeben. Ansonsten wird für jede Teilmenge der Größe A der Verwertungen aus V, die in einem der Zeiträume aus Z paarweise auf verschiedenen Nutzern verplant sind, der Schnitt der Planungszeiträume der Verwertungen der Teilmenge gebildet. Falls der Schnitt einer Teilmenge leer ist, dann gibt es kein Zeitfenster an dem die A Verwertungen der Teilmenge gleichzeitig verplant sind. Ansonsten ist der Schnitt ein Zeitfenster, an dem alle A Verwertungen der Teilmenge gleichzeitig verplant sind. Eine Verplanung der angefragten Verwertung auf dem angefragten Nutzer in diesem Zeitfenster ist dann nicht mehr zulässig, falls alle Verwertungen der Teilmenge nicht auf dem angefragten Nutzer verplant sind. Es wird das Resultat zurückgegeben, das entsteht, wenn man die wie oben berechneten Schnitte sukzessive vom Lizenzzeitraum subtrahiert.

6 Verwandte Arbeiten

Es existiert eine Reihe von wissenschaftlichen Arbeiten, die *Gesetzestexte mit Hilfe von DSLs* für unterschiedliche Zwecke darstellen. Insbesondere die Themen Steuergesetze und Datenschutz-Grundverordnung (DSGVO) sind hier weiter verbreitet. [En01] beschreibt eine Methode zur systematischen Übersetzung von (neuen) Rechtsvorschriften in die Prozesse der *Dutch Tax and Customs Administration* (DTCA in Dutch: Belastingdienst) unter Verwendung einer erweiterten UML/OCL-Sprachvariante. [Ag18] nutzt eine Erweiterung der Open Digital Rights Language (ODRL) [Gr18] zur Darstellung digitaler Rechte sowie gesetzlicher Verpflichtungen aus unterschiedlichen Rechtsvorschriften. Die Spezifikation von Datenschutzerklärungen wird in [Ca19] durch eine DSL ermöglicht, die Statements aus Excel bzw. natürlichsprachlichem Text extrahiert und dann in tabellarischer, graphischer und textueller Form darstellen kann. Auch die Darstellung von Gesetzestexten durch Ontologien ist ein stark erforschter Bereich [El16; Mo10].

Im Zusammenhang mit dieser Arbeit ist jedoch insbesondere die Definition von *Vertragsbestandteilen* von Interesse. Auch hier gibt es Arbeiten insbesondere zu intelligenten Verträgen im Themengebiet Blockchain [FN16; MMN17], die bereits stark mit natürlicher Sprache arbeiten. [JR18] ermöglicht die Modellierung von vertraglichen RAM- Anforderungen für die Generierung von Berechnungsmodellen im Zusammenhang mit den Ausschreibungsunterlagen für die Lieferung oder Wartung großer und komplexer technischer Systeme mittels einer DSL, die stark an die natürliche Sprache angelehnt ist. [Hv11] hingegen hat eine Vertrags-Spezifikationssprache entwickelt, die stark mit mathematischen Konstrukten arbeitet und Code-nahe ist. [GM05] beschäftigt sich mit der Verletzung von Vertragsbedingungen ebenso unter Verwendung einer formalen Vertragsspezifikationssprache. Nach unseren bisherigen Recherchen gibt es jedoch keine DSL, die sich mit dem Themengebiet

Fernsehausstrahlung und -programmplanung beschäftigt hat und durch Endanwenderinnen ohne tieferliegende Informatikkenntnisse verwendet werden kann.

7 Zusammenfassung

In diesem Papier haben wir eine DSL zur Spezifikation von vertraglichen Restriktionen im Bereich der linearen und nicht-linearen audiovisuellen Medienangebote vorgestellt. Die DSL wurde zu drei verschiedenen Zwecken entwickelt: 1. Formale Beschreibung der Bedingungen die sich aus Lizenzverträgen für audiovisuelle Medien ergeben, 2. Verifikation der Planung von Verwertungen im linearen und nicht-linearen Bereich im Hinblick auf die Erfüllung der vertraglich festgelegten Bedingungen und 3. Synthese von Zeitfenstern für zu verplanende Verwertungen basierend auf den spezifizierten Restriktionen und dem aktuellen Planungsstand bzw. der Verwertungshistorie. Die DSL befindet sich beim Kunden im produktiven Einsatz und erleichtert die Arbeitsaufgaben sowohl für die Beteiligten der Abteilung, die die Einschränkungen aus den Vertragstexten entnimmt, als auch für die Programmplaner. Die DSL hat es ermöglicht, dass Einschränkungen nun einheitlich formuliert werden können sowie die Überprüfung der Einhaltung automatisiert erfolgen kann. Darüber hinaus wurde zur Unterstützung der Programmplanung die Synthese der Verplanungsmöglichkeiten realisiert, mithilfe derer sich die Programmplaner mögliche Verplanungszeiträume berechnen lassen können.

Durch den iterativen, agilen Entwicklungsprozess konnte die DSL bereits in sehr frühen Stadien dem Kunden zur Verfügung gestellt, vom Kunden evaluiert und Kundenfeedback in die weitere DSL-Entwicklung einbezogen werden. Die frühe Mitgestaltung durch den Kunden hat dazu geführt, dass die Sprache gut für den gewünschten Anwendungszweck geeignet ist. Dem gegenüber konnte festgestellt werden, dass bei einer sehr frühen Nutzung erster Versionen der DSL, Änderungen von bereits entwickelter Funktionalität mit höheren Kosten und höherem Widerstand verbunden sind.

Mithilfe der DSL lassen sich bereits viele Varianten an Restriktionen, wie sie in den Lizenzverträgen von audiovisuellen Medien auftreten, formulieren. Ein Aspekt, der bislang nicht unterstützt wird, ist die Planung von Werbeeinblendungen, die eigenen Restriktionen unterliegen. Die hier in ihren wichtigsten Zügen vorgestellte DSL bietet einen guten Ausgangspunkt um diese Form der Restriktionen ebenfalls in einer DSL abzubilden.

Literatur

[Ag18] Agarwal, S.; Steyskal, S.; Antunovic, F.; Kirrane, S.: Legislative Compliance Assessment: Framework, Model and GDPR Instantiation. In (Medina, M.; Mitrakas, A.; Rannenberg, K.; Schweighofer, E.; Tsouroulas, N., Hrsg.): Privacy Technologies and Policy. Springer International Publishing, S. 131–149, 2018, ISBN: 978-3-030-02547-2.

[Ca19] Caramujo, J.; Rodrigues da Silva, A.; Monfared, S.; Ribeiro, A.; Calado, P.; Breaux, T.: RSL-IL4Privacy: a domain-specific language for the rigorous specification of privacy policies. Requirements Engineering 24/1, S. 1–26, 2019.

[El16] El Ghosh, M.; Naja, H.; Abdulrab, H.; Khalil, M.: Towards a Middle-out Approach for Building Legal Domain Reference Ontology. International Journal of Knowledge Engineering 2/3, S. 109–114, 2016.

[En01] van Engers, T. M.; Gerrits, R.; Boekenoogen, M.; Glassée, E.; Kordelaar, P.: POWER: Using UML/OCL for Modeling Legislation - an Application Report. In: Proceedings of the 8th International Conference on Artificial Intelligence and Law. ICAIL '01, ACM, S. 157–167, 2001, ISBN: 1-58113-368-5.

[FN16] Frantz, C. K.; Nowostawski, M.: From Institutions to Code: Towards Automated Generation of Smart Contracts. In: IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS*W). S. 210–215, 2016.

[Gl16] Gloger, B.: Scrum: Produkte zuverlässig und schnell entwickeln. Carl Hanser Verlag GmbH Co KG, 2016.

[GM05] Governatori, G.; Milosevic, Z.: Dealing with contract violations: formalism and domain specific language. In: Ninth IEEE International EDOC Enterprise Computing Conference (EDOC'05). S. 46–57, 2005.

[Gr18] Group, W. O. C.: ODRL Information Model 2.2, 2018, URL: https://www.w3.org/TR/odrl-model/.

[Ha09] Hasebrink, U.: Lineares und nicht-lineares Fernsehen aus der Zuschauerperspektive: Spezifika, Abgrenzungen und Übergänge. Hans-Bredow-Insitut für Medienforschung an der Universität Hamburg, 2009.

[Hv11] Hvitved, T.: Contract Formalisation and Modular Implementation of Domain-Specific Languages, PhD Thesis, University of Copenhagen, 2011.

[JR18] Joanni, A.; Ratiu, D.: Modeling and Valuation of Contractual RAM Requirements Using Domain-Specific Languages. In: 2018 Annual Reliability and Maintainability Symposium (RAMS). S. 1–6, 2018.

[MMN17] Magazzeni, D.; McBurney, P.; Nash, W.: Validation and Verification of Smart Contracts: A Research Agenda. Computer 50/9, S. 50–57, 2017.

[Mo10] Mommers, L.: Ontologies in the Legal Domain. In (Poli, R.; Seibt, J., Hrsg.): Theory and Applications of Ontology: Philosophical Perspectives. Springer Netherlands, Dordrecht, S. 265–276, 2010, ISBN: 978-90-481-8845-1.

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 189

Komposition Domänenspezifischer Sprachen unter Nutzung der MontiCore Language Workbench, am Beispiel SysML 2

Katrin Hölldobler[1], Nico Jansen[1], Bernhard Rumpe[1], Andreas Wortmann[1]

Abstract: MontiCore ist eine Language Workbench zum Design und zur Realisierung von textuellen domänenspezifischen Sprachen (DSLs). MontiCore ermöglicht die Erforschung von modellbasierten Softwareentwicklungsmethoden durch eine Vielfalt von DSLs und Sprachkomponenten. Darüber hinaus wird MontiCore erfolgreich in akademischen wie auch industriellen Projekten in verschiedenen Domänen wie der Automobilbranche, im Bereich Energiemanagement und der Robotik eingesetzt. In diesem Tutorial erklären wir die kompositionale Entwicklung von DSLs mit MontiCore am Beispiel SysML 2.

Keywords: Modellgetriebene Softwareentwicklung, Language Workbench, domänenspezifische Sprachen, Komposition, Modularität

1 Gegenstand des Tutorials

Die modellbasierte Softwareentwicklung (MBSE) verwendet Modelle, um die Komplexität der zu entwickelnden Systeme zu reduzieren. Zur Entwicklung dieser Modellen werden General-Purpose Modellierungssprachen wie beispielsweise die UML oder domänenspezifische Sprachen (DSLs) verwendet. DSLs bieten hierbei den Vorteil, dass sie sich besser für die jeweilige Domäne eignen, allerdings auch spezifisch für diese entwickelt werden müssen. MontiCore [HR17] ist eine Language Workbench, die die Entwicklung entsprechender, textueller DSLs und Modellierungssprachen erleichtert. Hierzu bietet MontiCore ein EBNF-artiges Grammatikformat sowie Möglichkeiten zur modularen Entwicklung und Komposition von DSLs und dazu passenden Tools und Infrastruktur [KRV10, HRW18]. Das Ziel der Entwicklung von MontiCore war und ist es eine leistungsstarke und effiziente Workbench zu erschaffen, die eine agile Entwicklung von DSLs zusammen mit dazu passender Infrastruktur für Analysen, Transformationen und Codegenerierung, ermöglicht. Wichtige Features von MontiCore sind zudem:

- Kombinierte Spezifikation von konkreter und abstrakter Syntax
- Anpassbare Generierung von Parser und abstraktem Syntaxbaum
- Generierung von Analyseinfrastruktur inklusive Visitoren
- Variable Grammatikverarbeitung via interpretierten Groovy Skripten
- Konfigurierbares Logging and Prozessreporting
- Freemarker Template Engine für leichtgewichtige Codegenerierung

[1] Software Engineering, RWTH Aachen University, Aachen, Germany

2 Ziele des Tutorials

In diesem Tutorial präsentieren wir die Language Workbench MontiCore. Das ca. zweistündige Tutorial teilt sich in einen Präsentations- und einen Hands On-Teil. Es wird erklärt wie DSLs und Modellierungssprachen mit MontiCore entwickelt werden können. Hierzu dienen Modellierungssprachen der SysML 2 [Ob] als Beispiel. Der Fokus des Tutorials liegt auf der Entwicklung neuer Sprachen durch Komposition und Erweiterung vorhandener Sprachen und Sprachkomponenten. Dieses Tutorial richtet sich an modellierungserfahrene Personen, die verstehen möchten, wie eine Language Workbench funktioniert, um modular neue Sprache aus vorhandenen Sprachkomponenten zu konstruieren. Idealerweise kennen die Teilnehmer bereits Modellierunssprachen wie die UML oder SysML 2 und die Programmiersprache Java. Das Tutorial umfasst:

- Definition von DSLs mit der Language Workbench MontiCore
- Die MontiCore Sprachbibliothek und deren Kombinationsmöglichkeiten
- Entwicklung von SysML 2 Modellierungssprachen durch Komposition und Erweiterung vorhandener Sprachen und Sprachkomponenten

Bernhard Rumpe leitet den Lehrstuhl für Software Engineering der RWTH Aachen University. Sein Hauptinteresse sind rigorose und praktische Software- und Systementwicklungsmethoden, die auf adäquaten Modellierungstechniken basieren. Dies umfasst sowohl agile Entwicklungsmethoden als auch Modellengineering basierend auf UML-artigen Notationen und domänenspezifischen Sprachen. Er hat einen Beitrag zu einer Vielzahl an Modellierungstechniken geleistet wie beispielsweise der UML Standardisierung. Er wendet Modellierungstechniken auf autonomes Fahren, Gehirnsimulation, BIM Energiemanagement, Produktionsautomatisierung und vieles mehr an. In seinen Projekten arbeitet er intensiv mit allen großen, deutschen Automobilherstellern, Energieunternehmen, Versicherungen und Banken, einem großen Flugzeughersteller, einem Raumfahrtunternehmen sowie innovativen Start-Ups der IT-Branche zusammen.

Literaturverzeichnis

[HR17] Hölldobler, Katrin; Rumpe, Bernhard: MontiCore 5 Language Workbench Edition 2017. Aachener Informatik-Berichte, Software Engineering, Band 32. Shaker Verlag, December 2017.

[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.

[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.

[Ob] Object Management Group: , SYSML V2. `http://www.omgsysml.org/SysML-2.htm`. [Online; accessed 18. November 2019].

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 191

Multi-level Modelling with the FMML^{x} and the XModeler^{ML}

Tutorial

Tony Clark[1], Ulrich Frank[2]

Abstract: Multilevel modelling is a new modelling paradigm that enables additional abstraction through an unlimited number of classification levels and thus contributes to reuse, integration and flexibility. The XModeler is a language engineering tool that allows for the common representation of (meta) models and code. The multi-level modelling and execution language FMML^{x} is implemented in the XModeler by extending its genuine meta model. This tutorial provides a comprehensive introduction into the foundations of multi-level modelling and its application. It will be shown how multi-level modelling enables relaxing fundamental conflicts of system design. During a practical exercise with the XModeler^{ML}, the participants will experience a new style of modelling that integrates language design with traditional ways of modelling.

Keywords: Conceptual Modelling; Model-Based Development and Maintenance; Self-Referential Enterprise Systems

1 Audience and Goals

The tutorial targets researchers and industrial practitioners who have an interest in how system modelling can better support the development life cycle from requirements through to implementation. The participants are expected to be familiar with object-oriented modelling and software construction. The tutorial is dedicated to three main goals. First, it is to raise the awareness of multi-level modelling in general and to foster a discussion with the participants about its prospects and related challenges. Second, it aims at an overview of the foundations of the language engineering and execution environment XModeler and the multi-level modelling language FMML^{x}. Third, the participants are to be enabled to develop and run multi-level models specified with the FMML^{x}.

2 Subject and Scope

Early ideas that relate to multi-level modelling go back to the programming language Smalltalk which includes a metaclass for every class. Later, Atkinson and Kühne coined

[1] Aston University, Birmingham tony.clark@aston.ac.uk
[2] Universität Duisburg-Essen ulrich.frank@uni-due.de

the term "multi-level modelling" and proposed it as an extension of the UML (which, unfortunately, did not happen to a satisfactory extent). In recent years an active community of researchers who work on various approaches to multi-level modelling has evolved. The annual workshop series "MULTI", which is part of the Models conference, is the pivotal platform of this community. The XModeler is an integrated (meta-) modelling and (meta-) programming environment that features a common representation of models and code. Its recursive meta-model XCore enables the creation of classes on multiple levels. The FMML^x extends XCore to support the representation of specific classification levels and intrinsic features. Using the FMML^x requires to go beyond the familiar paradigm of conceptual modelling that is characterized by a clear distinction of modelling language and model, as well as of model and code. With the FMML^x, they are all developed simultaneously. This opens up new perspectives in modelling and software development. At the same time, it demands for new modelling methods. An introductory case study will demonstrate the peculiarities and prospects of using the FMML^x. Presentations of the XModeler and the FMML^x will enable participants to understand core foundational aspects of multi-level modelling and programming. The final part of the tutorial comprises of a further case study. After analysing the requirements, the participants will learn how to use the XModeler^{ML} (a new version of the original XModeler that features the FMML^x) to develop a multi-level model and a corresponding application system. This will not only enable them to use a multi-level language engineering and modelling facility. They will also get an idea of a new generation of application systems, which are integrated with the models they are based on at run-time. Users of these systems are empowered to navigate the conceptual foundation of the software they use, and if needed, adapt it to changing requirements.

The XModeler is provided on the web pages of the LE4MM project (https://www.wi-inf.uni-duisburg-essen.de/LE4MM/). The project pages also include various resources such as screencasts, an extensive bibliography and publications. Since the software is constantly being further developed, the participants are advised to check the web pages for the latest version shortly before the tutorial.

3 Lead Academics

Tony Clark is Professor for Software Engineering at Aston University in Birmingham. He is one of the developers of XMF and the Xmodeler. Tony was co-founder and Technical Director of Xactium Ltd, a software modelling tools company. Tony has been involved in a number of commercial and industrial projects including contributing to the UML 2.0 standard, and consultancies with companies including British Aerospace, BT and CitiGroup.

Ulrich Frank holds the chair of Information Systems and Enterprise Modelling at the Institute of Computer Science and Business Information Systems at the University of Duisburg-Essen. His main research topic is enterprise modelling, i.e. the development and evaluation of modelling languages, methods and corresponding tools. Further areas of research include method engineering, models at run time and methods for IT management.

Dominik Bork, Dimitris Karagiannis, Heinrich C. Mayr (Hrsg.): Modellierung 2020,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2020 193

Models as Programs

A Tutorial

Bernhard Thalheim[1]

Abstract: Models are one of the main instruments for system development in computer science and engineering. So far models have been used for system description and system prescription, i.e. essentially as a blueprint for development. Models might however become programs for themselves. This approach allows to claim that models will become the kernel element of true fifth generation programming.

Keywords: models as programs; model-based programming; modelling

1 Towards True Fifth Generation Programming

Programming became more and more comfortable with development of third and fourth generation programming languages. Although the fifth generation project did not achieve its goals, the necessity for more comfortability is still challenging. This tutorial delineates the path towards *next generation programming* [JT19] based on *literate modelling* with model suites[2] that generalises model-driven development and conceptual-model programming [TSF19].

2 Tutorial Outline and Schedule

2.1 Models as next generation programs for everybody

Programming has become a technique for everybody, especially for non-computer scientists. Programs became an essential part of modern infrastructure. Programming is nowadays a socio-material practice in most disciplines of science and engineering. Solution development for real life complex systems becomes however an obstacle course due the huge variety of languages and frameworks used, due to impedance mismatches among libraries, due to conflicting environments, due to vanishing programming expert knowledge and culture,

[1] Department of Computer Science, Christian-Albrechts University of Kiel, 24098 Kiel, Germany, thalheim@is.informatik.uni-kiel.de

[2] A *model suite* consists of a coherent collection of explicitly associated models. A model in a model suite is used for different purposes such as communication, documentation, conceptualisation, construction, analysis, design, explanation, and modernisation. The model suite can be used *as a program of next generation* and will be mapped to programs in host languages of fourth or third generation. So, we claim that *models will become programs of next generation programming*.

due to novel and only partially understood paradigms such as componentisation and app programming, due to the inherent tremendous complexity, due to programming in the large and programming in the web, and due to legacy and integration problems. Our goal is very ambitious. This tutorial addresses the challenges by developing foundations and technologies for next generation programming and testing them in three central areas of computer science. Models will become programs of next generation programming. We developed and implemented literate modelling with model suites as generalisations of model-driven development and of conceptual-model programming.

2.2 Model suite description language

The model language consists of an associated bundle of languages for model elements that users may modify depending on their needs or simply reuse them as already established subcultures. These elements are different from ordinary programs because they are essentially declarative rather than imperative. Similar to UML stereotypes, MaP model class and model style languages are ready for application, can be extended, combined, and adapted. Users don't have to work on the details of the models as programs (MaP). The system takes over the integration and composition work as it deduces the consequences of the model.

2.3 Technologies, techniques, methodologies, and modelling moulds

The development of models in a model suite is based on a model library with models that can be used as an inherited or initial model for systematic composition of the model suite. The approach to model construction is canonised on the basis of methodologies and modelling moulds which systematically combine known and novel techniques and technologies for model development. Modelling moulds enable the modeller to reuse systematics and theories that have been successfully deployed in the past. They enable us to start with application space models, with deep models, with generic models, and with reference models without explicit reinvestigation of these models. The explicit agreement on a given mould eases, enables, and supports an economic development process.

2.4 Environment for an extension towards modelware as next generation literate modelling

Our approach aims at development of a general infrastructure for treatment of models as programs. This infrastructure includes also standardised solutions that can be reused in other applications. These solutions are based on application space models and deep models that are typically less volatile than normal models. Therefore, the library allows quick and well-based modelling by non-specialists which may concentrate on development of normal models instead of developing the entire holistic model. They may accept the library models as a basis and then use generic and reference models as a starting point for normal model development. The model suite is also transformed to programs in programming languages of third or fourth generations. This approach disentangles modellers from programming and allows them to concentrate on the model development. The model is then the product. The transformed program is of a higher quality and more liable.

2.5 Compiler-compiler approach to model realisation for models as programs

The modelling infra-structure is an essential element for realisation of model suites and for treatment of models as programs of next generation programming. The compiler-compiler approach is enhanced by the layered handling of models. This generation is the basis for language and platform independence of the models themselves. Compilation allows the integration of standards. Model suite libraries become then the kernel for modelware. Models become directly executable.

Literaturverzeichnis

[BT08] Börger, Egon; Thalheim, Bernhard: A method for verifiable and validatable business process modeling. In: Advances in Software Engineering, S. 59–115. Springer, 2008.

[JT19] Jaakkola, Hannu; Thalheim, Bernhard: Models as programs: The envisioned and principal key to true fifth generation programming. In: In Proc. 29'th EJC, S. 170. Lappeenranta, Finland, 2019.

[KT16] Kramer, Frank; Thalheim, Bernhard: Holistic Conceptual and Logical Database Structure Modeling with ADOxx. In: Domain-Specific Conceptual Modeling, S. 269–290. Springer, 2016.

[MT19] Molnár, András J.; Thalheim, Bernhard: Usage Models Mapped to Programs. In: New Trends in Databases and Information Systems. Springer International Publishing, Cham, S. 163–175, 2019.

[TSF19] Thalheim, Bernhard; Sotnikov, Alexander; Fiodorov, Igor: Models: The Main Tool of True Fifth Generation Programming. In: Selected Papers of the XXII International Conference Ënterprise Engineering and Knowledge Management (EEKM 2019), S. 161–170. 2019.