GESELLSCHAFT
FÜR INFORMATIK

Fachausschuss
Management der Anwendungsentwicklung und -wartung (WI-MAW)
im FB Wirtschaftsinformatik

Jahrgang 27 Heft 1
ISSN 1610-5753

März 2021

Schwerpunktthema: Datengetriebene Anwendungen und Innovationstreiber im Projektmanagement zukunftsfähiger Organisationen
Martin Engstler, Masud Fazal-Baqaie (Hrsg.)

Inhalt

45. WI-MAW-Rundbrief

Inhaltsverzeichnis

Fachbeiträge

Berichte

Organisation

Vorwort

Martin Engstler[1], Masud Fazal-Baqaie[2]

[1] Hochschule der Medien, Nobelstr. 10, 70569 Stuttgart, engstler@hdm-stuttgart.de

[2] NEXT Data Service AG, Alt-Moabit 104, 10559 Berlin, masud@next-data-service.com

Im Herbst 2020 planten die Fachgruppen Projektmanagement (WI-PM) und Vorgehensmodelle (WI-VM) die bereits siebte gemeinsame GI-Fachtagung „Projektmanagement- und Vorgehensmodelle“, die PVM2020. Unter dem spannenden Leitthema „Datengetriebene Anwendungen und Innovationstreiber im Projektmanagement zukunftsfähiger Organisationen“ wurde dann im Frühjahr 2020 ein Call for Paper gestartet. Bereits zum Zeitpunkt der Veröffentlichung dieses Call for Paper war den organisierenden Fachgruppen bewusst, dass die Entscheidung zur Durchführung im bewährten Format einer Präsenztagung zunächst offengehalten werden musste. Die Möglichkeit einer Absage aufgrund von Corona-Beschränkungen bestand und wurde auch offen kommuniziert und hatte sich schließlich bewahrheitet. Umso mehr freute uns die positive Resonanz auf den Call for Paper zur PVM2020. Die hierbei eingereichten Beiträge in den Kategorien „Wissenschaftlicher Beitrag“ bzw. „Future-Track-Beitrag“ durchliefen den regulären Reviewprozess. Somit lag es nahe, den AutorInnen der angenommenen Beiträge ein angemessenes Publikationsformat für ihren Fachbeitrag über das hier vorliegende Sonderheft des Rundbriefs anzubieten. Über das Interesse und die Bereitschaft einiger AutorInnen zur Publikation der für die PVM2020 geplanten Tagungsbeiträge in diesem Sonderheft haben wir uns sehr gefreut, einen herzlichen Dank dafür. Ergänzend haben wir im Sinne der Förderung wissenschaftlichen Nachwuchses zwei weitere Beiträge mit aufgenommen, die ebenfalls einen strukturierten Reviewprozess im Herbst 2020 durchliefen. Auch diesen AutorInnen möchten wir an dieser Stelle herzlich danken.

Wir hoffen die kommende PVM wieder wie geplant durchführen zu können und haben sie daher auf das erste Halbjahr 2022 verschoben. Wir freuen uns bis dahin, sechs spannende Beiträge in diesem Sonderheft vorstellen zu können und wünschen allen eine erkenntnisreiche und anregende Lektüre.

Stuttgart und Berlin

Martin Engstler und Masud Fazal-Baqaie

Entrepreneurial Software Engineering: Towards a Hybrid Development Method for Early-Stage Startups

Daniel Brunner[1], Jürgen Münch[2], Marco Kuhrmann[1]

[1]University of Passau, Innstr. 33, 94032 Passau, Germany;

danielbrunner2007@web.de, kuhrmann@acm.org

[2]Reutlingen University, Danziger Str. 6, 71034 Böblingen, Germany,

j.muench@computer.org

Abstract: A considerable share of innovative software-intensive products is developed by startups. However, product development in an early-stage startup is not a sequential process. A business idea is usually based on a number of assumptions. The riskiest assumptions need to be tested. Depending on the test results, a product strategy may change several times. This raises the question of how to create sufficiently stable software using engineering principles despite a dynamic product strategy that is subject to many uncertainties. Hybrid development methods that combine agile aspects with classical engineering methods seem to be a good choice in such a start-up context. This paper proposes a lightweight hybrid development method that provides early-stage startups with a framework to support the development of single-feature minimum viable products. The method was derived from a start-up company's founding case and evaluated in expert interviews. The proposed method is intended to provide a basis for discussion between practitioners and scientists with the aim of better understanding the application of software engineering principles in software start-ups.

Keywords: Software Startup, Hybrid Method, Guideline, Method Proposal, Lean Startup, Entrepreneurial Software Engineering

1 Introduction

In the last decade, a number of disruptive software startups emerged [Gu15], and many of these startups grew and revolutionized the market, e.g., Flixbus, Uber, and AirBnB. Furthermore, in several fields, former startups have become market leaders, e.g., Amazon, Google, and Facebook. Startups contribute significantly to the world economy, and, through the accelerating digitalization, their impact will further grow.

An important area for startups is the mobile software market. Due to the increased use of smartphones and tablets in web-based environments, many new product ideas are developed and realized as *Apps*. Since there are rich and mature development frameworks available, the development of apps is a straightforward and fast approach to make an innovative idea reality, which improves the level of "attractiveness" of startups. Creativity is in the spotlight; there are no large hierarchically organized teams, and the main thing that matters is the product. However, even though there are easy-to-use tools available, "easy" is a relative term. Still, software development is a challenging business and developing high-quality software requires the skilled professionals.

Startup Challenges. Wassermann [Wa16] stated that software development in startups is often more "hacking" than a systematically implemented practice. Together with the high degrees of technological uncertainty and a way to fast – and often not reflected – implementation of "agile" methods, many startups fail. For instance, Giardono et al. [Gi+14] report that more than 60% of the startups fail within their first five years. Software development, notably sustainable software development, requires some organization. Once the product is in the market, increasingly complex processes need to be installed, e.g., for feature development, change management, bug fixing, and general innovation activities – not to forget the organization framework including, such as acquisition or sales.

Besides, a major challenge is that the business idea is based on many risky assumptions, e.g., is the problem to be solved an actual problem in the target customer segment and, moreover, is it worth solving? If one of the assumptions does not hold, the startup's business model is at stake. Therefore, risky assumptions must be tested as early as possible to adjust the product strategy accordingly. Quite often, a mockup or a software prototype, a so-called *Minimal Viable Product* (MVP), is used to test assumptions in the problem domain. At this point, software engineering principles are often neglected, since understanding the problem domain is more important than developing a sustainable software product. Yet, the software engineering principles become important once the product has been launched [Wa16].

This is where *Entrepreneurial Software Engineering* enters the stage. Entrepreneurial Software Engineering is concerned with the following question: *How to efficiently and effectively develop software-intensive systems as part of an entrepreneurial or innovation process?* A key concern in this regard is to ensure from the very beginning on that a process is used that allows for developing a product that is robust from the software engineering point of view. That is, for example, just in the development of the initial product (parts), the questions for the product's architecture must be asked, and if it is possible at all to design a solid architecture without knowing exactly what the final product will look like and which features it will have? Eventually, it is a matter of trade-off decisions, since there are different goals in the different phases of a startup or innovation process, ranging from testing ideas early over testing solution alternatives to ensuring high quality of the final product in the market. Entrepreneurial Software Engineering aims at appropriately supporting the different phases, at creating synergies between the development activities in these phases, and at minimizing waste.

Contribution and Outline. This paper proposes a lightweight hybrid method tailored for use in early-stage startups that are focusing on *single-feature MVPs*[1]. The method was inspired by the needs becoming obvious in the *Zippr* startup, by the principles of lean software development, and the concepts described in experiment-driven software development and continuous experimentation. The method was created using data about combined method and practice use, and the method was evaluated in expert interviews. The method provides an initial framework to be modified and extended by practitioners in their various contexts.

The remainder of the paper is organized as follows: Section 2 provides an overview of the background and related work. Section 3 introduces the proposed method in detail and provides details on the method's evaluation. The paper is concluded in Section 4.

2 Background and Related Work

There are several definitions of the term "startup" [BD12, Ri11, Su00]. For instance, Blank et al. [BD12] define a startup as a "temporary organization designed to search for a repeatable and scalable business model", and Ries [Ri11] defines a startup as "a human institution designed to deliver a new product or service under conditions of extreme uncertainty". Uncertainty arises from a variety of causes [Gi+14]: startups often do not know their customers or the markets. Often, startups offer solutions to problems that many customers were not even aware of [LM16]. Another issue is the rapid evolution of startups. Once the business model is identified, scaling the startup fast becomes the main focus. Other than established companies, startups can (usually) adopt quickly to external influences. However, a lack of resources is often found that hinders grow, e.g., limited human and physical resources [Gi+14]. Also, young venture team members are usually inexperienced and have too little (software) engineering skills. Startups usually aim to deliver their products to customers as fast as possible to establish themselves in

[1] This paper is based on the Bachelor Thesis "A Hybrid Development Method for Early-Stage Start-Ups" authored by Daniel Brunner at the University of Passau. The paper provides the proposed development method in an improved straightforward way. Further details, especially the initial evaluation of the proposed method, due to page limitations, has to be taken from the Bachelor Thesis.

the rapidly changing and progressing markets, which are under heavy competition. Reducing speed of the product delivery might lead to competitors releasing comparable solutions. To speed up the product delivery, many startups focus on one single product, whereas established companies offer a variety of different products in several horizontal and vertical markets.

Lean Startup. The *Lean Startup* concept was introduced by Ries [Ri11] in 2011. Lean Startup is based on customer-focused and agile software development, and implements Lean management techniques to guide the creation of new businesses and products. A key component is the set of *lean principles*, e.g., avoid waste to reduce the effort required for the product development. The principles can be implemented by combining business-driven hypothesis, experimentation, and iterative product releases. Especially iterative development requires deep knowledge of the needs of early customers – also often referred to as early adopters. Putting the customer into the spotlight can lead to reduced market risks, e.g., to avoid launching products that do not provide value.

Minimal Viable Products. The most relevant concept of the Lean Startup Principles in the context of this paper is the *Build-Measure-Learn* cycle (BML). The BML-cycle aims at turning assumptions about the product or its customers into knowledge that helps deliver valuable solutions. The BML-cycle helps increase the speed of the product development through fast feedback loops. In the *Build*-phase, testable assumptions – hypotheses – are posed that usually reflect the startup's vision of the product and business strategy. A *Minimum Viable Product* (MVP) is built, which is a "version of a new product, which allows a team to collect the maximum amount of validated learning(s) about customers with the least effort" [Ri11]. There are various types of MVPs, which can be used depending on the case or vision [DA16], e.g., landing pages, mockups, and single-feature MVPs.

Related Work. In the last decade, research interest in the field of startups increased. However, literature on software development activities in startups is scarce. Few literature reviews have been conducted on different startup-related topics. For instance, Paternoster et al. [Pa+14] conducted a systematic mapping study on software development practices. They found that startups tend to select and use development practices opportunistically. Klotins et al. [KUG15] identified development practices startups use and mapped findings to the *Software Engineering Body of Knowledge* (SWEBOK) knowledge areas. They found 11 out of 15 knowledge areas are covered by research. Zettel et al. [Ze+01] were among the first that developed a lightweight software development process for startup companies (the so-called LIPE process). They focused on the development of e-business applications and their process is widely based on Extreme Programming. The aim of this process is to exploit the long-term benefits from the application of software engineering methods and principles, and to use some of them from the very start. Besides, research did not provide guidelines for software development for any stage. Yet, Berg et al. [Be+18] found a shift in the research focus of software startup research. While most research is conducted within the SWBOK knowledge areas, increasing interest could be found in the process and management areas. Recently, Tegegne et al. [TSA19] studied the methods and practices startups use for their software development, and Giardino et al. [Gi+14] examined practices that are commonly applied in startups. Both studies showed that startups tend to use agile and Lean Startup methodologies to keep flexibility and to be able to adopt fast to the market and customer needs. They tend to use lightweight, often informal and customized methods [TSA19], which are often focused on iterative and incremental coding – Wasserman [Wa16] describes these methods as "low-ceremony processes". However, strictly following a method is uncommon in software startups due to limited resources and time pressure. Startups often select practices considered relevant from known methods and adapt these to the individual needs, which is referred to as a *hybrid method* [Ku+18]. Nevertheless, startups must also be prepared for the future, since without adequate processes, failure in the long term is likely [Cr02, GWA14, Wa16]. Startups have to balance the fast validation of the business model and provision of a high-quality product, which is often referred to as "Developers Dilemma"

[TSS16]. Having a solid framework in place, which can be evolved over time, reduces the risk of inadequate processes.

In Lean Startups, experimentation is used to validate product and business assumptions to seek a valuable product, and continuous value delivery is key. *Continuous experimentation* helps identify the features to build and deploy. Fagerholm et al. [Fa+14, Fa+17] introduced building blocks continuous experimentation and the "Stairway to Heaven" model describes an evolution path companies can take to be equipped for continuous experimentation [OAB12]. For conducting continuous experimentation within a company, Fagerholm et al. [Fa+17] introduced the *RIGHT* model, and Olsson and Bosch [OB14] propose the *HYPEX* model. Both models aim to reduce opinion-based decision-making and foster informed decision-making grounded in empirical evidence, e.g., obtained in (feature) experiments.

3 A Hybrid Development Method for Early-Stage Startups

This section presents the hybrid development method for early-stage startups. It starts with presenting the core requirements of the model, before the model as such is presented.

3.1 Requirements

As outlined before, software startups develop software under special conditions, notably in interdisciplinary teams in which the software development expertise is not necessarily strong. Therefore, such team require methodological and technical support. Form this assumption, we derive the following core requirements:

R1. The development method must be lightweight.
R2. The development method must allow for later scaling (once the product matures).
R3. The development method must include the continuous experimentation paradigm.
R4. The development method must support the fast and cost-effective MVP development.
R5. The development method must ensure that technical debt is avoided.

These core requirements are straightforward: the development method must be lightweight to provide an easy start for creative teams, but it must also allow for scaling. Once the product is in the market, further processes need to be installed, e.g., evolution processes and further business processes that need to be aligned with the software development. In order to define new requirements and features, and to identify those that potentially generate the most value, continuous experimentation should be included. Among other things, continuous experimentation should also be integrated with the cost-effective development of MVPs. Finally, related to R2, the development method should ensure that the software developed fulfills basic quality requirements. The goal is to avoid as many "quick hacks" as possible that need to be costly fixed later.

The development method, which is presented in the following, addresses all these requirements. Regarding the support for the cost-effective MVP-development, we refine the requirements. The development method shall specifically support the development of so-called *single-feature MVPs*, which are minimum versions of products that allow for gaining validated learnings by experimentation. A single-feature MVP can be considered a "prototype", which only implements the most important function that is required to make informed decisions about the value of a specific function. In more general terms, an MVP can also be seen as the minimum effort required to rapidly learn what the customer wants and needs. MVPs should be developed fast, to establish fast feedback cycles and, hence, foster innovation cycles.

3.2 Overview of the Development Method

Figure 1 provides the birds' eyes perspective on the development method. The development method consists of three basic stages:

1. Feature Requirements Generation and Evaluation (FRGE)
2. Design and User Experience Evaluation (DUXE)
3. Pre-Launch Testing (PLT)

These three stages describe the basic workflow. All three stages must be implemented in a sequence on a per-feature basis. The actual workflow is characterized by a number of artifacts that serve as input and output of the respective stages and that, furthermore, are linked with the specific development methods.

The starting point of the workflow is the *validated problem*. A validated problem can be an issue that has been evaluated and named relevant to customers. To validate a problem, Bosch et al. [Bo+13] suggest asking the following questions: *What is the problem? Who has the problem? Is the problem big enough to make a business out of it?*

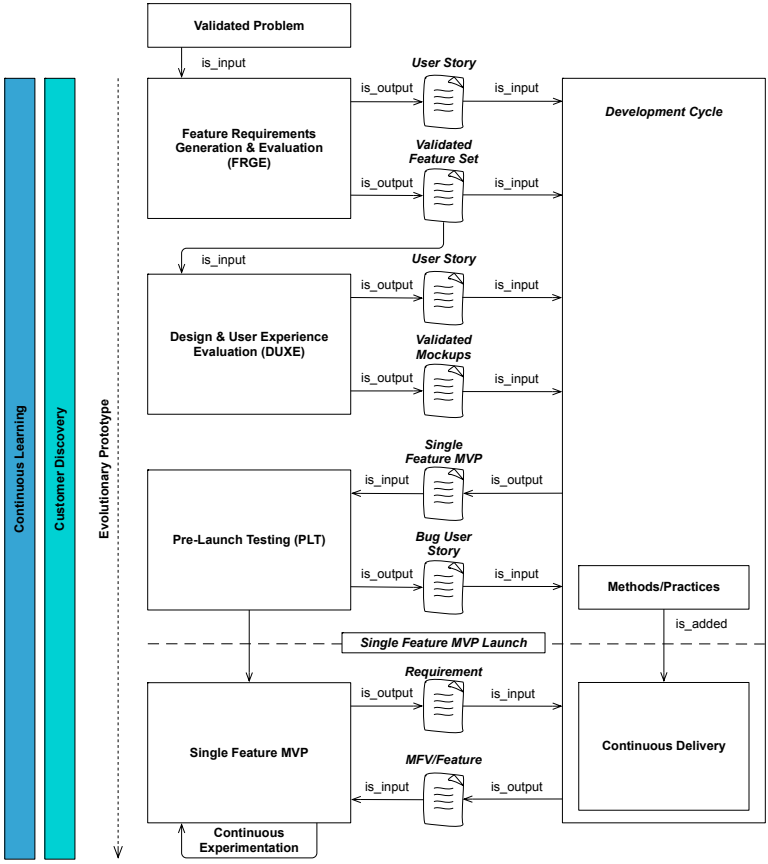

Figure 1 Overview of the hybrid development model for early-stage startups

Once the problem has been validated, the first stage – FRGE – is entered in which the initial feature set is compiled and the single-feature MVP is designed. Since it is the overall goal of the proposed development method, to help early-stage startups develop software fast, right in this stage, first user stories are issued to start the development activities. The second stage – DUXE – aims at ensuring the user experience (UX). It is crucial to start the UX evaluation as early as possible, e.g., with early adopters, to ensure that the features are properly implemented to provide a satisfying experience for the user and usability of the software. To refine the initial requirements and to align them as optimal as possible with the users' goals, experimentation is used. The final step is represented by the stage PLT in which the MVP is tested. Depending on the MVP, the tests range from automated unit test to complex test setups, e.g., installation, mobile usability, and energy consumption.

Key to the entire model is the availability of a continuous software engineering environment, which allows for continuous integration and delivery. Figure 1 also shows that two complementing activities are conducted along the MVP development: continuous learning and customer discovery. In the customer discovery, special emphasis must be put on early adopters (that at best can become "evangelists"). These persons are used to evaluate problems and the respective solution approaches.

3.3 Stage 1: Feature Requirements Generation and Evaluation

The goal of this stage is the collection and validation of the initial features of the software. Especially when aiming for a single-feature MVP, the number of features to be included is very limited and, therefore, the "right" features to be implemented need to be identified. This is especially crucial, since startups – as many other software-producing companies – tend to start coding early.

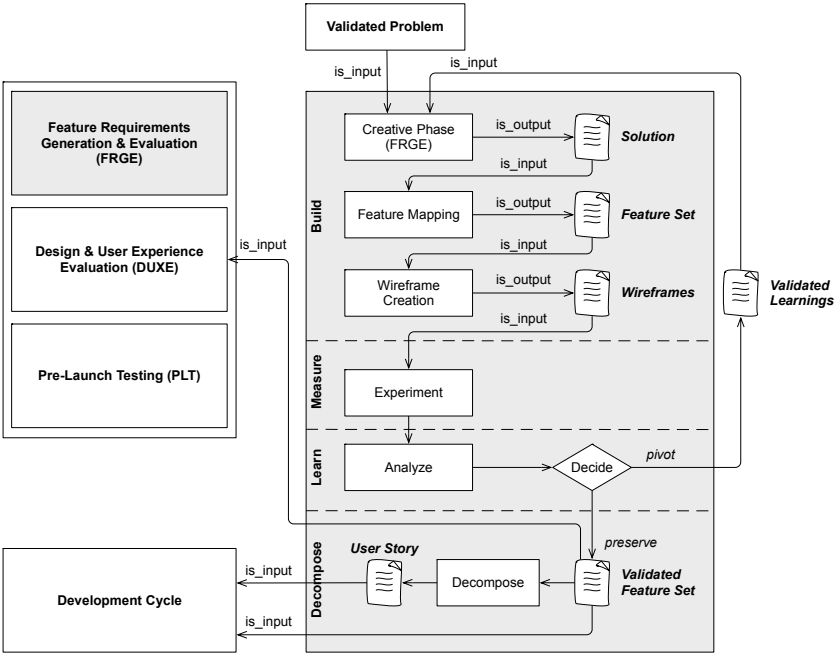

Figure 2 Refinement of the first stage FRGE

To avoid "working for trash" or to consume too many resources, the key requirements need to be identified and designed into the MVP properly. For this, the FRGE-stage is organized according to the Build-Measure-Learn cycle (Figure 2). In this stage, creativity techniques are used to develop an initial solution idea, which is structured using the feature mapping approach. To provide means for an early evaluation, wireframes should be created, which serve as input for the experimentation activities. Validated learnings from the analyses help improve further cycles. In case the feature refinement saturated and the maturity is considered high enough, the validated feature set is decomposed into user stories, which are handed over to the actual software development.

To support the decision-making, i.e., to leave the Build-Measure-Learn cycle, Bosh et al. [Bo+13] suggest two questions: *Is the current set of features sufficient to solve the customer's problem? Are customers willing to test the MVP?*

3.4 Stage 2: Design and User Experience Evaluation

Good user experience is key. Hence, the proposed development method includes a dedicated UX phase as a key element, which is shown in Figure 3.

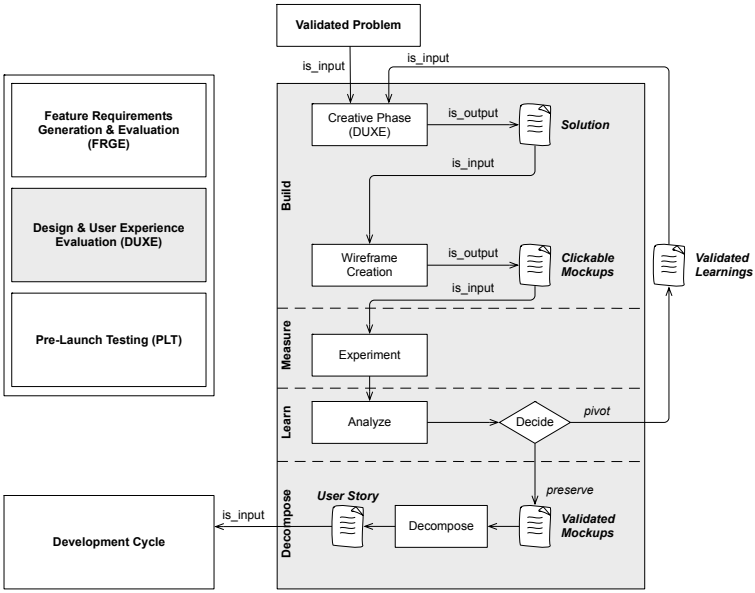

Figure 3 Refinement of the second stage DUXE

Similar to the first stage, the DUXE-stage starts with creativity activities. However, there is no explicit step in which new features are defined. Instead, the solution proposal from the creativity activities results in wireframe designs, which are evaluated. The rest of this stage is organized in the same way as the FRGE-stage. A slight, yet important difference compared to the FRGE-stage is the development of software-based prototypes (Mockups). While a simple, even a paper-based, approach for the wireframing is sufficient for the FRGE-stage, in the DUXE-stage, we recommend using clickable prototypes, since these provide a better means for early adopters and evaluators to test the MVP. A clickable prototype thus helps shortening the feedback cycles and to obtain feedback of better quality, as evaluators use an object that is closer to the desired product, e.g., general look & feel and so forth. Also, the mockups are subject to the experiments

in the Build-Measure-Learn cycle. To support the decision-making, the following two questions should be asked: *Is the user experience sufficient for early adopters? Does the customer fully understand the use of the product?*

Once the results of the analysis stabilized, the feature set together with the prototyped wireframes are handed over to the development. For this, new user stories are created, which are implemented and gradually integrated with the other system components.

3.5 Stage 3: Pre-Launch Testing

Once software is deployed, it remains in the IT-ecosystem. If the quality of the released software is insufficient, quality problems and various risks will occur during the system's lifetime. Therefore, the third essential part of the proposed development method is *quality assurance*.

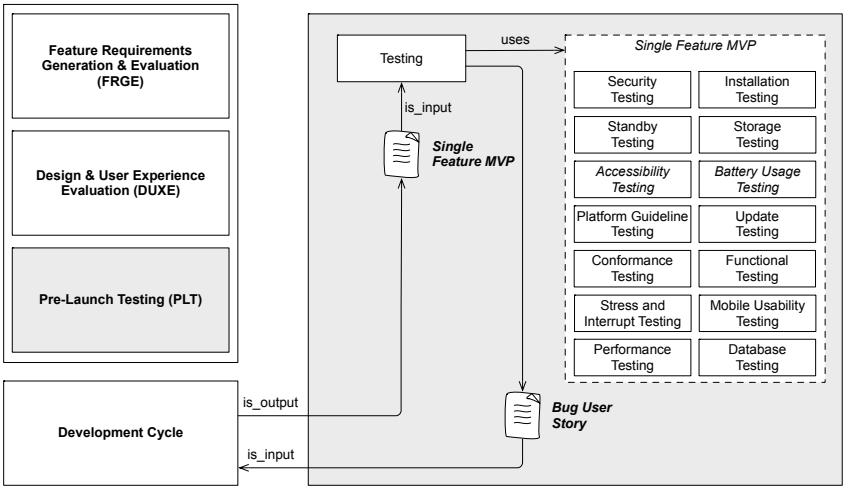

Figure 4 Refinement of the third stage PLT (optional test activities are italic)

Figure 4 shows the refinement of the PLT-stage in which the testing-related activities are located. Once a single-feature MVP has reached a state in which it can be potentially released, it is put into the testing stage. Depending on the actual kind of the product, different testing methods are required, e.g., battery usage testing for smartphone apps or mobile usability testing for progressive Web apps. However, it is important to mention that the test environment must be designed such that a continuous testing and integration is possible. Since all the stages shown above are designed as incremental-iterative processes, (automated) tests should be run as soon as user stories have been implemented. Therefore, a continuous software engineering environment is advantageous, since it is possible to select and implement a user story and, once the user story is implemented, to commit the new code to a repository, which automatically triggers a build agent. However, this also requires a specific setup of development methods and their alignment, which is presented in the following section.

3.6 Detailed Workflow in the Development Cycle

Since the selection of the development methods and practices severely impacts the organizational effort of a development project, but also lays the foundation for a sustainable product development, the selection and combination of relevant methods and practices must be done with care.

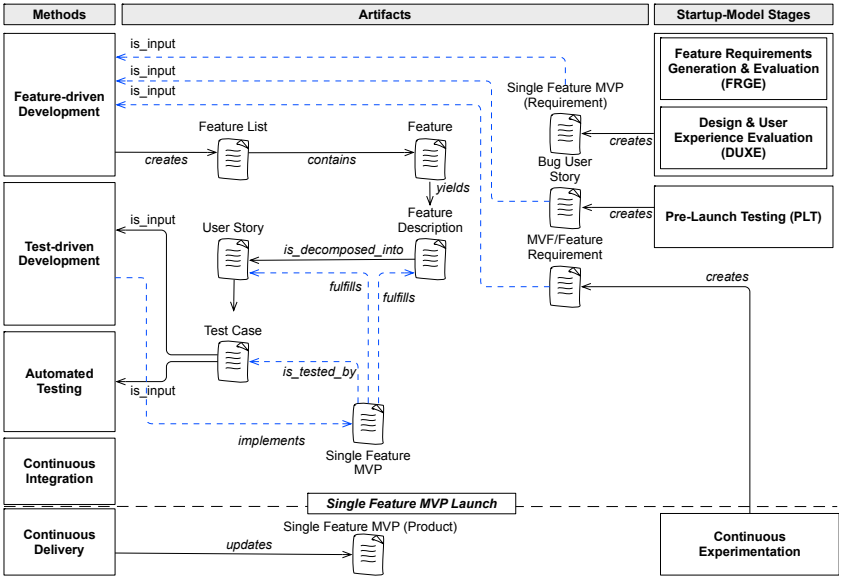

Figure 5 Overview of the startup-model stages, the artifact and the flow of the artifacts in relation to the key development methods

Figure 5 provides an integrated perspective of the development cycle, the artifacts created in the different stages and the combination of development methods and practices utilized in the *Zippr* startup. As Figure 5 shows, the selection of the development methods was driven by their potential for automation. The base method selected is *Feature-driven Development*, since this method already provides a basic idea of the development and organization of feature hierarchies and a per-feature design and development cycle. The actual coding activities are organized using the *Test-driven Development* approach, complemented with automated unit testing. The technical backbone is provided by a continuous integration and delivery infrastructure. Besides the artifacts introduced so far, the continuous experimentation, which is implemented after the single-feature MVP launch, adds the new artifact *Minimal Viable Feature* (MVF), which is created from a hypothesis formulated within the continuous experimentation process.

3.7 Evaluation

The development method presented above is the result from an evaluation performed in an expert review. In total, three experts have been interviewed personally or via Skype. The experts were provided with the initial version of the method alongside with a list of 19 questions (three for the top-level model description, three per stage, and four for the development cycle). The list of questions included questions for the model's consistency, the occurrence of redundancies, the suitability and the completeness of the model.

As a result of the evaluation, the stages FRGE and DUXE have been renamed and their internal structure has been improved (25 comments in total). Furthermore, two more general methods and five extra stage-method assignments have been suggested by the reviewers. In general, the outcome of the expert interview was that the model builds a solid basis for further use and evaluation in practice.

4 Conclusion and Future Work

In this paper, we proposed a lightweight hybrid development method for early-stage startups. The purpose of the method proposal is to provide early-stage startups that potentially lack software design and development expertise with a slim and easy-to-use guideline for organizing the software development activities. Special emphasis was put on the integration of the creativity techniques used to develop and refine ideas to lay the foundation for the software development. This includes procedures to create ideas, to develop feature bundles, wireframes and clickable prototypes and, eventually, to link all these activities to a lightweight method that supports agile and lean software development. Since the method presented is the initial step, a complementing evaluation of the method is subject to future work. This include the critical review of the method and their improvement based on practical learnings. Furthermore, variants of this methods will be developed to provide a more generic framework that supports broader applicability.

References

[Gu15] Guttentag, D.: Airbnb: disruptive innovation and the rise of an informal tourism accommodation sector. In: Current issues in Tourism 18.12 (2015), pp. 1192–1217.

[Wa16] Wasserman, A. I.: Low ceremony processes for short lifecycle projects. In: Managing Software Process Evolution. Springer, 2016, pp. 1–13.

[BD12] Blank, S.; Dorf, B.: The startup owner's manual: The step-by-step guide for building a great company. BookBaby, 2012.

[Gi+14] Giardino C. et al.: What do we know about software development in startups? In: IEEE software 31.5 (2014), pp. 28–32.

[LM16] Lindgren, E.; Münch, J.: Raising the odds of success: the current state of experimentation in product development. In: Information and Software Technology 77 (2016), pp. 80–91.

[Ri11] Ries, E.: The lean startup: How today's entrepreneurs use continuous innovation to create radically successful businesses. Crown Books, 2011.

[Su00] Sutton, S. M.: The role of process in software start-up. In: IEEE Software 17.4 (2000), pp. 33–39.

[DA16] Duc, A. N.; Abrahamsson, P.: Minimum viable product or multiple facet product? The Role of MVP in software startups. In: International Conference on Agile Software Development. Springer. 2016, pp. 118–130.

[Be+18] Berg V. et al.: Software startup engineering: A systematic mapping study. In: Journal of Systems and Software 144 (2018), pp. 255–274.

[Cr02] Crowne, M.: Why software product startups fail and what to do about it. Evolution of software product development in startup companies. In: IEEE International Engineering Management Conference. Vol. 1. IEEE. 2002, pp. 338–343.

[Fa+14] Fagerholm F. et al.: Building blocks for continuous experimentation. In: Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering. ACM. 2014, pp. 26–35.

[Fa+17] Fagerholm F. et al.: The RIGHT model for continuous experimentation. In: Journal of Systems and Software 123 (2017), pp. 292–305.

[GWA14] Giardino, C.; Wang, X.; Abrahamsson, P.: Why early-stage software startups fail: a behavioral framework. In: International Conference of Software Business. Springer. 2014, pp. 27–41.

[KUG15] Klotins, E.; Unterkalmsteiner, M.; Gorschek, T.: Software engineering knowledge areas in startup companies: a mapping study. In: International Conference of Software Business.

Springer. 2015, pp. 245–257.

[Ku+18] Kuhrmann M. et al.: Hybrid Software Development Approaches in Practice: A European Perspective. In: IEEE Software (2018).

[OAB12] Olsson, H. H.; Alahyari, H.; Bosch, J.: Climbing the 'Stairway to Heaven' – A Multiple-Case Study Exploring Barriers in the Transition from Agile Development towards Continuous Deployment of Software. In: 38th Euromicro Conference on Software Engineering and Advanced Applications. IEEE. 2012, pp. 392– 399.

[OB14] Olsson H. H.; Bosch, J.: The HYPEX model: from opinions to data- driven software development. In: Continuous Software Engineering. Springer, 2014, pp. 155–164.

[Pa+14] Paternoster N. et al.: Software development in startup companies: A systematic mapping study. In: Information and Software Technology 56.10 (2014), pp. 1200–1218.

[TSA19] Tegegne, E. W.; Seppänen, S.; Ahmad, M. O.: Software development methodologies and practices in start-ups. In: IET Software 13.6 (2019), pp. 497–509.

[TSS16] Terho, H.; Suonsyrjä, S.; Systä, K.: The Developers Dilemma: Perfect Product Development or Fast Business Validation? In: International Conference on Product-Focused Software Process Improvement. Springer. 2016, pp. 571–579.

[Bo+13] Bosch, J. et al.: The early stage software startup development model: a framework for operationalizing lean principles in software startups. In: International Conference on Lean Enterprise Software and Systems. Springer. 2013, pp. 1–15.

[Ze+01] Zettel, J., Maurer, F., Münch, J. Wong, L.: LIPE: A Lightweight Process for E-Business Startup Companies Based on Extreme Programming, In: Proceedings of the International Conference Product Focused Software Process Improvement, Springer, 2001.

Optimal IT Project Selection – Quantification of Critical Scoring Criteria

Christin Karrenbauer and Michael H. Breitner

Leibniz Universität Hannover, Institut für Wirtschaftsinformatik

{karrenbauer | breitner}@iwi.uni-hannover.de

Abstract: The management of IT project portfolios is challenging because of IT projects' complexity, dynamics, unknowns, and uncertainties. IT projects account for a large IT budget proportion and significantly influence value contribution, strategic development, goal achievements, and competitive advantages. Many IT projects still fail, exceed time and resources, and do not reach their planned goals because of wrong decisions, unsatisfactory evaluation, and missing selection criteria. Thus, a continuous IT project scoring and selection is crucial to enable an optimal portfolio composition. We conduct a systematic literature review and 14 semi-structured qualitative expert interviews to develop a uniform and holistic scoring approach. Our findings show that IT projects' urgency, strategy, efficiency, risk, and complexity are critical IT project scoring criteria. Our scoring approach increases objectivity and quality in evaluating planned and running IT projects and allows more convincing and transparent decisions.

1 Introduction

Expenditures in Information Technology (IT) projects rapidly increased worldwide, and the amount is expected to rise further [Ga20]. The IT of a company or an organization is of high importance and a critical success factor and thus influences long-term performances [CS13, Ma19]. It is crucial to permanently select and manage the "right" IT projects to build an optimal IT project portfolio, achieve goals, create value, be innovative, and stay competitive because IT projects account for a large proportion of IT budgets. So the decision nowadays is not whether or not to invest in IT projects but to identify those that together contribute most to the operational and strategic goals [CS13]. Thereby, all IT project proposals share and compete for the same scarce resources and are carried out under the same management [AG99, LRS20, PMI13]. In this context, we define IT project portfolio management (ITPPM) as a continuous and dynamic process in which IT project proposals are collected and together with ongoing IT projects (re-)scored, (re-)prioritized, (re-)selected, and (re-)scheduled considering different constraints, interdependencies, resource limitations, and stakeholder interests [CEK99, Ke11, ML07, PMT15].

IT projects are challenging to manage because of their cross-functionality, dynamics, non-routine, temporary, and complex nature with resulting unknowns and uncertainties. A selection of IT projects is further connected with many difficulties, as both qualitative and quantitative factors must be considered [ANJ10]. Various IT projects fail and do not reach their planned goals because of wrong evaluation and selection criteria and decisions [VT16]. Instead, decisions often are based on subjective influences, personal perceptions, and experiences as there often exist no uniform evaluation and selection criteria and methods. There is the need to accurately define IT project evaluation criteria as those decisively influence the scoring and selection, enable transparent decisions, and support decision-makers [BR00]. We therefore develop a uniform and holistic scoring approach with relevant selection criteria especially for IT projects. It differs from already existing ones as it does not only rely on IT related theoretical foundations, but also takes practical expert knowledge from various IT project managers of several industries into account. We further introduce an exemplary scoring scale besides critical scoring criteria. The holistic scoring approach increases objectivity and quality in scoring, the evaluation of both IT project proposals and running IT projects, and enables uniform scores and criteria. First, we provide the theoretical background for an ITPPM cycle and IT project selection and scheduling

methods. Next, we describe our research design and how we conducted our literature review and expert interviews to identify critical IT project scoring criteria. Afterward, we present our findings, develop a scoring model with an exemplary scale, and shortly introduce an optimization model to support the ITPPM decision process. Finally, we discuss our results, their implications, and outline conclusions.

2 Theoretical Background

Each ITPPM cycle is possible to divide into different stages. In a first pre-screening stage, promising IT projects from various departments are identified as well as "must" IT projects defined. After that, each IT project proposal is evaluated individually according to pre-set criteria in the stages of individual IT project analysis and screening. Also, first IT project proposals that do not fulfill knockout criteria are rejected. Next, in the stage of optimal IT portfolio selection, the most valuable IT projects that suit the selection criteria best are selected and scheduled within the planning periods. There is the possibility to readjust the IT project portfolio in the last stage, in case of internal or external changes [AG99]. When following these stages, IT projects are more likely to gain their goals and be successful, as they prevent an IT project's overload [BU12]. However, this process is challenging because of dynamics, complexities, uncertainties, unknowns, synergies, and contradicting criteria [MMM19, Zh17].
Project portfolio selection and scheduling (PPSS) is a mainstream in the project portfolio management literature. It is about whether or not to implement project groups or single projects to build a project portfolio to obtain goals [MSS17]. PPSS aims to quantify the "right" projects taking into account various interdependencies, limitations, and constraints and schedules them within a distinct planning horizon [CN13, ZHT19]. There exist different financial and non-financial PPSS approaches and methods. In general, these include single criterion analysis, scoring models, and optimization models. For the latter, it is possible to differentiate between integer linear programming, multi-criteria selection, fuzzy programming, mixed-integer linear programming, multi-objective programming, non-linear programming, and stochastic programming. Mohagheghi et al. [MMM19] and Zhang et al. [Zh17] summarize and give an overview of such approaches. However, there is the need to properly define selection criteria as they profoundly influence the scoring and selection results, irrespective of the method applied.

3 Research Method

In a first step, we performed a systematic literature review to get an overview of relevant IT-PPM, scoring, and selection literature [BR15, TP15, WW02]. We searched for combinations of IT, project portfolio management, selection criteria, and scoring in the journals included in the Association for Information Systems Senior Scholars Basket of Eight, Project Management Journal, International Journal of Project Management, and Science Direct. We looked manually through the articles' references for additional relevant literature after we identified the key literature. We then used Google Scholar for a forward and author search to find further articles that cite the key articles and for a similarity search.
In a second step, we conducted explorative expert interviews to gain insights into practical IT project portfolio evaluation and selection and to identify common criteria used in practice. It enables a comparison with criteria used in literature to identify similar ones or potential differences. We conducted 14 semi-structured and guideline-oriented interviews between May and June 2019. Table 1 gives an overview of the experts, the industries, and their positions.

Interview	Industry	Position
INT.1	Insurance	Strategic IT project portfolio steering

INT.2	Paper factory	Head of IT Application Department and Head of IT Business Department
INT.3	Consulting	Member of the IT project management team/ IT manager
INT.4	Automotive	IT project manager
INT.5	Automotive	Member of the IT project management team
INT.6	Banking	Member of the IT project management team
INT.7	Paper factory	Leader and member in several IT Projects
INT.8	Consulting	Member in several IT projects
INT.9	Energy supplier	Member of the IT project management team
INT.10	Software manufacturer	Global HR IT project portfolio planning
INT.11	Insurance	Senior IT project manager
INT.12	Insurance	IT project portfolio manager
INT.13	Energy supplier	IT project portfolio manager
INT.14	Consulting	Consultant in several IT projects

Table 1: Overview of the Experts Interviewed

To ensure generalizability, the interviewed experts work in various sectors and positions with different tasks, but all have in common their strong involvement in the ITPPM decision process. One expert works in software development, four experts in the banking and insurance sector, and two experts each work in the paper industry, in the automotive industry, in consulting companies, and in energy supplier companies. We carried out a pretest before we conducted the interviews to validate our interview guideline [JW10]. Next, we sent the interview guide and an explanation about the interview's setting and purpose to each expert. Dependent on the expert's location, four interviews took place in person and the remaining ten by telephone. The interview lasted on average 60 minutes, and we documented them by transcripts. We analyzed the transcripts using an inductive qualitative content analysis according to Mayring [Ma14] supported by the software MAXQDA 18. By paraphrasing the experts' answers, we analyzed the interviews. During this process, we summarized the encoding units into a concise, language-unified form and deleted non-relevant phrases. Based on similar paraphrases, we defined generalized categories and assigned anchor rules. We then assigned relevant paraphrases of each interview to them. This resulted in a detailed overview of all expert statements regarding different categories, which then in a next step allowed a structured analysis of the interviews' insights.

4 Results

4.1 Literature Review

Based on our literature review, we were able to identify often considered IT project evaluation and selection criteria. We categorized them on the basis of the most important literature concerning this topic into seven categories: complexity, efficiency, interdependencies, resource limitations, risk, strategy, and urgency, see Table 2. Where possible, we divided the categories again. However, some literature only named or referred to evaluation criteria without neither a proper definition of the criteria nor their characteristics or examples. We included these in the category "General" in the table. All listed (sub-)criteria have been mentioned multiple times in literature; single mentioned ones were omitted. Criteria are ordered alphabetically and do not reflect the importance.

An IT project's complexity is one extracted criteria from literature. It involves evaluations of how many changes occur with an IT project proposal's implementation and describes the resulting degree of changes. Another evaluation criterion is an IT project's efficiency. It mainly encompasses cost and/or benefit analyses, growth rates, and economic returns, like for example Net Present Value and Return on Investment. Underlying interdependencies between single IT projects are also important to consider. Time interdependencies occur if one or more IT projects

have to be finished before another one can start, as they are required as an input. Mutual exclusiveness exits if an inclusion of one or several IT projects directly leads to the exclusion of other IT projects. Synergies reflect an IT project's either positive or negative influence on other IT projects. Also important to consider are existing resource constraints. These include a restricted availability of monetary and human resources. An IT project's risk evaluation is a further widely used criterion in literature, however, often without a specification what exactly needs to be regarded. Still, there is a general agreement that it is crucial to balance the risk within the whole portfolio to avoid the inclusion of too many risky IT projects that could weaken results. Often used evaluation criteria are the existing experience of project team members and a risk's probability of occurrence with its resulting consequences. The listed strategy criterion includes analyses of whether an IT project's implementation influences a company's or an organization's competitive advantages and whether it suits the strategic alignment and supports goals. The often named urgency criterion involves on the one hand an IT project's implementation need to replace existing systems to keep the daily business running and on the other to conform with regulatory requirements.
Despite our identified critical criteria, an identification and categorization of all crucial evaluation and selection criteria is almost impossible because criteria are different between companies and organizations and rely on their environment [MMM19].

Criterion	Sub-Criteria	Source
Complexity	Degree of complexity	[JK99, SC19]
	General	[CZL09, CN13, CS13]
	Number of changes	[CN13, Se16, SS02]
Efficiency	Cost and/or benefit analysis	[AG99, ANJ10, Ba92, BS04, Ch02, CN13, CS13, JK99, LL08, SK95, SS02, Wi92]
	Economic returns	[AG99, CN13, IL02, JK99, LL08, Re05, RO08, Wi92, Zh17]
	Growth rate	[JK99, SS02]
Interdependencies	General	[CN13, SC19, SK95]
	Mutual exclusiveness	[AG99, CN13, Re05]
	Synergies	[CN13, CS13]
	Time-dependencies	[AG99, BS04, Re05]
Resource Limitations	General	[ANJ10, SK95]
	Human Resources	[AG99, Ch02, Ch20, CN13, JK99, Re05]
	Monetary	[AG99, CN13, Re05, RO08, SC19, Wi92]
Risk	Available experience	[Ch20, CZL09, CN13, Re05]
	General	[AG09, ANJ10, CS13, IL02, LL08, Re05, SK95, Zh17]
	Probability of occurrence and consequences	[AG99, Ba92, CZL09, CN13]
Strategy	Competitive advantages	[AG99, Ch02, IL02, JK99, SS02, Wi92]
	Increase in market share	[LL08, IL02]
	Strategic alignment/goals	[AG99, ANJ10, Ba92, Ch02, CN13, JK99, Re05, SC19, SS02,Wi92, Zh17]
Urgency	Need for daily business	[JK99, Wi92]
	Need for renewal	[Ba92, JK99]
	Regulatory requirements	[JK99, RO08]

Table 2: Overview of Commonly Used IT Project Evaluation Criteria in Literature

4.2 Expert Interviews

According to the experts, an IT project's contribution to a company's or an organization's strategy is often considered when evaluating IT project proposals [e.g. INT.2, INT.9, INT.11]. For example, according to INT.12 "single IT projects need to be oriented on the organization's strategy and the derived IT strategy." One expert even says that strategic orientation is one of

the most important factor to consider when evaluating IT project proposals [INT.14]. Also important in this context is, for example, the "increase in the internal or external customer benefit or loyalty" [INT.10]. However, the experts also admit that a strategic quantification often is connected with lots of difficulties, and values are not always quantifiable [INT.3].
Further, the experts agree that "regulatory requirements and need for renewal need to be considered" [INT.1], so an IT project's operational urgency [e.g., INT.11, INT.13]. Here it is essential to analyze the effects and consequences that are to be expected through a non-compliance with regulatory requirements. It also includes an IT project's contribution to modernize, replace, or supplement existing systems and their influence on the daily business [INT.1, INT.2, INT.4, INT.7]. "IT projects that are necessary for the persistence of the organization are highly prioritized" [INT.14]. According to INT.2, "many IT projects result from liabilities, like expiring maintenance contracts, new safety, or network requirements". In some companies, such IT projects even represent a significant part of the implemented IT projects [INT.2, INT. 6]. They are often considered as mandatory that definitely need to be part of the portfolio because of their importance [INT.3, INT.5].
Another identified important evaluation criterion is human resources [e.g., INT.2, INT.8]. It encompasses an IT project's evaluation regarding the requirement of internal or external human resources for an execution. As the availability is restricted, it is a limiting factor for prioritization [e.g., INT.11, INT.13, INT.14]. Resulting, the availability of human resources has a significant influence on a prioritization process. An evaluation is based on the "effort in terms of human resources and time" [INT.8]. However, in most IT projects, there is a need for specialized competencies and experts, which are widely required but only scarce in their availabilities. These often represent a bottleneck and hugely influence decisions [INT.2, INT.8, INT.12].
An IT project's risk is a further important evaluation criterion mentioned by the experts [e.g., INT.2, INT.4, INT.6]. Here, an IT project is evaluated regarding the risk that corresponds with carrying out an IT project. "An evaluation about what can go wrong in terms of risk needs to be made" [INT.4]. This includes the analysis of potential occurring risks and the remaining consequences and also impacts of a non-performance of single IT projects [INT.3, INT.6]. However, not only an IT project's risk itself is evaluated, but also its influence on other IT projects. "Evaluations about other IT projects being positively or negatively influenced by a certain IT project are made" [INT.8].
Mostly all experts agree that an IT project's evaluation should also consider interdependencies between different IT projects [e.g., INT.3, INT.8, INT.12]. Dependent on the company's or organization's size, "they cannot be considered for the whole organization but only for specific parts" [INT.5]. However, the execution differs a lot. Some experts say they have structured methods like, for example, strategy roadmaps to analyze existing interdependencies [INT.3, INT.7, INT.9]. Others include interdependencies in their business cases to mainly prevent redundancies and the execution of similar IT projects [INT.10, INT.12]. In contrast, others consider interdependencies, but not in a structure way [INT.6]; they, for example, rely on portfolio managers who "know them and take them into account when prioritizing IT projects" [INT.11]. Independent of the method applied, the identification of interdependencies reduces an IT project portfolio's overall complexity and makes it easier to handle [INT.5, INT.8]. They also partly determine the order of the IT projects to be executed in case of chained IT projects by identifying the need for one IT project being finished before another one can start [INT.7, INT.9].
Efficiency evaluation is another frequently quoted criteria by the experts [e.g., INT.2, INT.5, INT.10]. An IT project is evaluated regarding its costs in relation to its benefits [INT.1, INT.13]. Experts agree that costs are necessary to consider and are a limiting factor because of restricted budgets [e.g., INT.6, INT.8, INT.9]. "It is important to consider if the required budget is available" [INT.4]. They also agree on the importance of benefits evaluation which are direct results of an IT project's execution [INT.1, INT.7, INT.8, INT.14]. However, the experts are divided on issues to be evaluated. For some experts, only a potential cost reduction as a direct impact of an IT project's execution is considered [INT.6, INT.7]. One expert expands this by

"maintenance requirements because at some point maintenance is not possible anymore" [INT.1]. Others consider the "value for internal and external customers" [INT.10] or the "growth through a better market position or diversification of business" [INT.3] as critical.
Some experts explained that specific IT projects are not evaluated regarding objective criteria, but rather are chosen due to decisions of the organization's executives. Meaning, the organization's politics influence the evaluation process, too, even though these decisions are not always based on a rational level [INT.14]. "Sometimes, the power of having a high position in the organization is used to gain resources for an IT project" [INT.6]. The same happens in relatively small family-run organizations. Here "the family has a huge impact on which IT projects are performed" [INT.2].

5 Model Development and Applicability Check

Our literature review and expert interviews show many similar evaluation criteria. To reconcile both points of view, we define such (sub-)criteria as critical mentioned in literature and/or in the expert interview and include them into our scoring model, expect interdependencies, organizational politics, and resource limitations. It summarizes the most common and relevant criteria into one score and enables a categorization of the following criteria: urgency, strategy, efficiency, risk, and complexity. These five criteria with its sub-criteria set the basis for our generic scoring approach. Their application enables a more objective, transparent, and manageable evaluation of underlying IT project proposals, resulting in more reliable evaluation decisions. Equation 1 shows the general mechanism of our scoring model.

$$a_i = \sum_{c=1}^{C} \left(\frac{\left(\sum_{n=1}^{N_c} v_{c_{n,i}}\right)}{\#N_c} w_c \right) \qquad \text{(Equation 1)}$$

Thereby, $v_{c_{n,i}}$ denotes the values for each sub-criteria with $c \in \{1, \dots, C\}$ being the criteria, e.g., urgency or complexity, and $n \in \{1, \dots, N_c\}$ their sub-criteria, e.g., need for modernization or degree of deviation from daily business. $\#N_c$ represents the number of occurring sub-criteria for an underlying criterion. Parameter w_c denotes a criterion's weight, i.e., the importance of criterion c with $\sum_{c=1}^{C} w_c = 1$. All weights w_c and values $v_{c_{n,i}}$ have to be defined for all (sub-)criteria. An individual IT project's score a_i for an IT project i is then calculated by the sum of products of the sub-criteria's mean score with the weight of the corresponding criterion. We limited the application of an outer weighted sum to prevent more manual effort. Depending on an organization, one can set the inner sum as a weighted sum, too. In this case, for each sub-criterion n of criterion c a weight must be defined. Again, the sum of all weights for all sub-criteria n of one criteria c being equal to one.
In a next step, we further used the findings of our literature review and expert interviews to develop an exemplary scoring table and scale, see Table 3. It defines which specification a (sub-)criteria must have to get a particular score. For this, we used a mix of verbal and numerical scales. Each IT project proposal and running IT project must be evaluated according to each sub-criterion and gets a score suitable to the description. The higher the total score is, the higher is the IT project's importance and value.

(Sub-)Criteria		Score 1	Score 2	Score 3	Score 4	Score 5
Complexity	# of involved business departments	> 13	10-13	7-9	4-6	< 4
	Degree of deviation from daily business	significant changes	considerable changes	isolated changes	minor changes	no changes

Efficiency	Investment recovery (periods)	> 20	16-20	11-15	5-10	< 4
	Long-term cost savings	no effects	barely noticeable	noticeable	considera-ble	highly significant
	Impact on the growth rate	no effects	barely noticeable	noticeable	considera-ble	highly significant
Risk	Probability of occur-rence	> 15%	11-15%	6-10%	2-5%	< 2%
	Potential damage (share of revenue)	> 12%	10-12%	7-9%	3-6%	< 3%
	# of similar past IT pro-jects of IT project leader	none	1-2	3-4	5-6	> 6
	Positive impact on other IT projects	none	1-3%	4-6%	7-10%	> 10%
	Negative impact on other IT projects	> 10%	7-10%	4-6%	1-3%	none
Strategy	Competitive advantage	none	barely noticeable	noticeable	considera-ble	highly significant
	Business goals support	none	barely noticeable	noticeable	considera-ble	highly significant
	Increase in market share	no effects	barely noticeable	noticeable	considera-ble	highly significant
Urgency	Non-compliance with regulatory requirements	non existing	short-term disruptions	considerable disruptions	legal conse-quences	sanctions
	Need to keep daily business running	no need	only few pro-cesses	several pro-cesses	many processes	core pro-cesses
	Need for modernization	within next 6+ years	within next 5 years	within next 4 years	within next 3 years	within next 2 years

Table 3: Illustrative ITPPM Scoring Table

Because each IT project proposal is scored according to the same (sub-)criteria and scale, it allows a comparison between different proposals. However, a "one fits all" scale is impossible because of a company's or an organization's heterogeneity. Instead, an adjustment with specific verbal or numerical criteria and sub-criteria is necessary. However, scoring results rather should serve as an input for further prioritization, selection, and scheduling decisions than used as the only method for these processes to overcome its disadvantages [LK01]. The scale further takes no resource limitations, interdependencies, and organization's politics into account. These are important to consider separately in additional restrictions and do not directly influence the importance of an IT project. Regarding the resource constraints, it is especially important to prevent resource overload, a situation in which selected IT projects require more resources than there are available. Different kinds of interdependencies must be identified and considered when selecting and scheduling IT projects. However, they do not indicate the importance of an IT project, but rather prevent redundancies and influence the temporal order of IT project implementations. As some selection decisions rely on organizational politics irrespective of an IT project's importance, we further excluded this criterion from our scoring approach.

Based on our findings, we developed a mathematical optimization model as a basis for a decision support tool for ITPPM decisions and used the scoring results as an input. An application of our tool allows to optimally select and schedule different IT project proposals for an optimal portfolio composition. It considers different constraints, interdependencies, and resource limitations extracted from the expert interviews and literature. It aims to maximize a company's or an organization's total value contribution by adding up the individual IT project scores. We here further integrated several restrictions for resource limitations, interdependencies, and organizational politics, which we excluded from our scoring approach. Thereby, the tool does not only ensure that IT projects can only start once during the time horizon. It also prevents resource

exceedances of limited resources, for example, budget [e.g., SC19, INT.4, INT.5, INT.8] and human resources [e.g., Ch20, INT.2, INT.11, INT.12] throughout any given planning period. Mandatory IT projects cannot be excluded from the IT project portfolio independent of their score, resulting because of e.g., law regulations, strategic considerations, or other reasons [cf. criterion of organization's politics, e.g., INT.2, INT.6, INT.14]. Once an IT project is defined as such, the tool ensures its inclusion into the portfolio. The tool also considers the mutual exclusiveness between IT projects [e.g., CN13, INT.10, INT.12 - criterion of interdependencies]. Temporal interdependencies between IT projects imply that certain IT projects have to be finished (predecessor IT projects) before others can start (successor IT project) [e.g., RE05, INT.7, INT.9]. Once a successor IT project is selected into the portfolio, the tool ensures that all related predecessor IT projects are selected, too. Here it is important that the starting period of the successor IT project is sometime after the predecessor IT project's. It allows a chronology determination of chained IT projects. Thereby, it is possible to include one or several predecessor IT projects without its successor IT project(s), but not vice versa.

We provide an applicability check with a generic IT project portfolio selection problem, including 25 IT project proposals and eight planning periods, for example, two years to evaluate and demonstrate our tool's feasibility. We use our scoring model, see Equation 1, and the criteria and scales in Table 3 to evaluate each IT project proposal. Based on the importance of the criteria to support the organization's goal achievements, the criteria strategy and urgency both get a weight of 0.3, risk a weight of 0.2, and the remaining efficiency and complexity a weight of 0.1 each. We consider a restricted availability and requirement of three resources: general external resources (consultant, soft- & hardware costs), general internal resources (project team), and internal domain-specific resources (key employees). Periodical resource availabilities are given as input and requirements are set by decision-makers. We assume two mandatory IT projects and two pairs of mutually exclusive ones. One large IT project is divided into four smaller one-periodical IT projects that need to be finished chronologically (temporal interdependency). Figure. 1 illustrates the optimization results with selected IT projects and their schedules in a Gantt chart. In the underlying case, 17 IT projects are selected into the portfolio and scheduled within the entire planning horizon with a portfolio value of 53.61. For comparison, without interdependencies, limitations, and constraints, the maximal portfolio value contribution is 75.71.

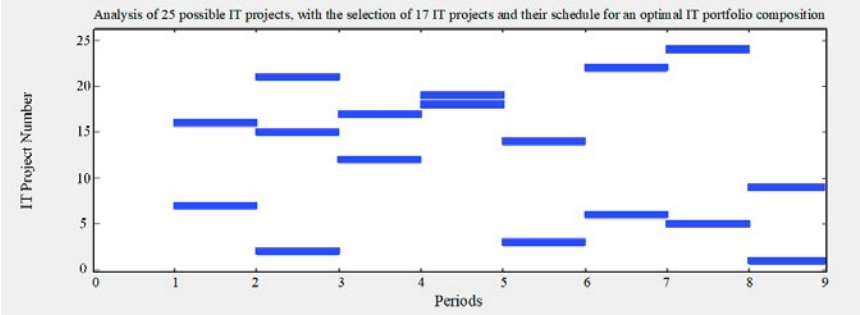

Figure 1: Optimization Results for the Generic Portfolio

6 Discussion and Implications

The increasing importance of IT projects for a company's or organization's long-term performances also increases the importance of selecting the "right" IT projects into a portfolio. This results in a need to properly define IT project evaluation and selection criteria to support and

enable transparent decisions. In a first step, we therefore conducted a literature review to identify often used selection criteria in ITPPM literature before we conducted 14 expert interviews to get more practical insights into this topic. With our findings, we were able to identify eight critical selection criteria. Most of them serve as an input for our scoring model and exemplary scale (complexity, efficiency, risk, strategy, and urgency) and the remaining for our optimization tool (interdependencies, resource constraints, and organization's politics). An application of our scoring model allows a uniform scoring of each IT project proposal. It serves as an input for the optimization tool and thus supports decision-makers in the optimal IT project portfolio selection and scheduling while considering temporal interdependencies and the existence of mutual exclusive IT project proposals. It also allows to define "must" IT projects and the determination of periodical resource requirements and availabilities.

During the expert interview analysis, we noticed that irrespective of the sector the experts work in, many named critical scoring criteria were the same. Only sub-criteria and characteristics were different for some criteria. We therefore concluded that general scoring criteria are very similar to each other irrespective of the sector. Only attributes of the individual sub-criteria differ. The identified criteria in literature and named by the experts were also very similar. There seems to be a common general understanding and agreement in both literature and practice on critical scoring criteria. The different sub-criteria result from the heterogeneity in size, the company's or organization's sector, and underlying regulations. So a "one fits all" model is not possible to develop. Instead, companies and organizations must define specific sub-criteria, scales, and weights. Our scoring model serves as an example that can be individually adapted by companies and organizations.

An application of our scoring model allows to evaluate all IT project proposals uniformly. It reduces decision-makers' subjective influences on final portfolio compositions and results are more transparent and reliable. However, it is crucial to precisely define scales and weights for each criterion, as they have a significant influence on the final portfolio composition. Because each IT project proposal must be scored according to each (sub-)criteria, their number is critical to prevent a high manual effort. It is necessary to consider between having too many criteria which lead to a high manual effort as well as having enough criteria to be able to evaluate each IT project proposal adequately. Our scoring model then serves as an input for our optimization tool. An application simplifies and automates the overall strategic evaluation, selection, and scheduling of IT projects for an IT portfolio composition and thus improves the quality and efficiency of optimization results.

7 Conclusions

Knowledge about IT project evaluation and selection is of high importance and inevitable for a company's or an organization's goal achievement, value contribution, and competitiveness. With our literature review and expert interviews we were able to quantify an IT project's complexity, efficiency, risk, strategy, and urgency as critical scoring criteria. They serve as a basis for our holistic scoring approach. It allows to evaluate IT project proposals uniformly and increases objectivity and quality. Our scoring model further serves as a basis for our optimization tool which considers various interdependencies, limitations, and constraints. It recommends which IT projects to select and schedules them within the planning horizon while simplifying and automating the overall strategic IT project evaluation, selection, and scheduling decisions.

References

[AG99] Archer, N. P.; Ghasemzadeh, F.: An Integrated Framework for Project Portfolio Selection. International Journal of Project Management 17 (4), pp. 207-216, 1999.

[AG09] Abid, F.; Guermazi, D.: Multi-Stage IT Project Evaluation: The Flexibility Value Obtained by Implementing and Resolving Berk, Green and Naik (2004) Model. Journal of Computational and Applied Mathematics 233 (1), pp. 73-82, 2009.

[ANJ10] Asosheh, A.; Nalchigar, S.; Jamporazmey, M.: Information Technology Project Evaluation: An Integrated Data Envelopment Analysis and Balanced Scorecard Approach. Expert Systems with Applications 37 (8), pp. 5931-5938, 2010.

[Ba92] Bacon, C. J.: The Use of Decision Criteria in Selecting Information Systems/Technology Investments. MIS Quarterly 16 (3), pp. 335-353, 1992.

[BR00] Bannister, F.; Remenyi, D.: 2000. Acts of Faith: Instinct, Value and IT Investment Decisions. Journal of Information Technology 15 (3), pp. 231-241, 2000.

[Br15] Brocke, vom, J., et al.: Standing on the Shoulders of Giants: Challenges and Recommendations of Literature Search in Information Systems Research. Communications of the Association for Information Systems 37 (1), pp. 205-224, 2015.

[BS04] Bardhan, I.; Sougstad, R.: Prioritizing a Portfolio of Information Technology Investment Projects. Journal of Management Information Systems 21 (2), pp. 30-60, 2004.

[BU12] Buchwald, A.; Urbach, N.: Exploring the Role of Un-Enacted Projects in IT Project Portfolio Management. In Proceedings of the 33th International Conference on Information Systems. Orlando, FL, USA, December 16-19, 2012.

[CEK99] Cooper, R. G.; Edgett, S. J.; Kleinschmidt, E. J.: New Product Portfolio Management: Practices and Performance. Journal of Product Innovation Management 16 (4), pp. 333-351, 1999.

[Ch02] Chen, C.-T: A Decision Model for Information System Project Selection. Engineering Management Conference (2), pp. 585-589, 2002.

[Ch20] Chen, R. et al.: A Competence-Time-Quality Scheduling Model of Multi Skilled Staff for IT Project Portfolio. Computers & Industrial Engineering, 139:106183, 2020.

[CN13] Chiang, R. I.; Nunez, M. A.: Strategic Alignment and Value Maximization for IT Project Portfolios. Information Technology and Management 14 (2), pp. 143-157, 2013.

[CS13] Cho, W.; Shaw, M. J.: Portfolio Selection Model for Enhancing Information Technology Synergy. IEEE Transactions on Engineering Management 60 (4), pp. 739-749, 2013.

[CZL09] Chen, T.; Zhang, J.; Lai, K.-K.: An Integrated Real Options Evaluating Model for Information Technology Projects under Multiple Risks. International Journal of Project Management 27 (8), pp. 776-786, 2009.

[Ga20] Gartner: Gartner Says Global IT Spending to Reach $3.9 Trillion in 2020. www.gartner.com/en/newsroom/press-releases/2020-01-15-gartner-says-global-it-spendingto-reach-3point9-trillion-in-2020, last accessed: June 14, 2020.

[IL02] Irani, Z.; Love, P. E. D.: Developing a Frame of Reference for Ex-Ante IT/IS Investment Evaluation. European Journal of Information Systems 11 (1), pp. 74-82, 2002.

[JK99] Jiang, J. J.; Klein, G.: Information System Project-Selection Criteria Variations within Strategic Classes. IEEE Transactions on Engineering Management 46 (2), pp. 171-176, 1999.

[JW10] Johnston, A. C.; Warkentin, M.: Fear Appeals and Information Security Behaviors: An Empirical Study. MIS Quarterly 34 (3), pp. 549-566, 2010.

[Ke11] Kester, L. et al.: Exploring Portfolio Decision-Making Processes. Journal of Product Innovation Management 28 (5), pp. 641-661, 2011.

[LK01] Lee, J. W.; Kim, S. H.: An Integrated Approach for Interdependent Information System Project Selection. International Journal of Project Management 19 (2), pp. 111-118, 2001.

[LL08] Liang, C.; Li, Q.: Enterprise Information System Project Selection with Regard to BOCR. International Journal of Project Management 26 (8), pp. 810-820, 2008.

[LRS20] Linhart, A.; Röglinger, M.; Stelzl, K.: A Project Portfolio Management Approach to Tackling the Exploration/Exploitation Tradeoff. Business & Information Systems Engineering 62 (2), pp. 103-119, 2020.

[Ma14] Mayring, P.: Qualitative content analysis: theoretical foundation, basic procedures and software solution. Open Access Repository, Klagenfurt, 2014 (available at https://www.ssoar.info/ssoar/handle/document/39517, last accessed: June 14, 2020).

[Ma19] Maruping, L. M. et al.: A Risk Mitigation Framework for Information Technology Projects: A Cultural Contingency Perspective. Journal of Management Information Systems 36 (1), pp. 120-157, 2019.

[ML07] Martinsuo, M.; Lehtonen, P.: Role of Single-Project Management in Achieving Portfolio Management Efficiency. International Journal of Project Management 25 (1), pp. 56-65, 2007.

[MMM19] Mohagheghi, V.; Mousavi, M. S.; Mojtahedi, M.: Project Portfolio Selection Problems: Two Decades Review from 1999 to 2019. Journal of Intelligent & Fuzzy Systems 17 (4), pp. 1-15, 2019.

[MSS17] Meredith, J. R.; Samuel, M. J.; Shafer, M. S.: Project Management: A Managerial Approach. 10th Edition, Wiley, New York, 2017.

[PMI13] Project Management Institute: The Standard for Portfolio Management. 3rd Edition, Project Management Institute, Newtown Square, PA, 2013.

[PMT15] Pellegrinelli, S.; Murray-Webster, R.; Turner, N.: Facilitating Organizational Ambidexterity Through the Complementary Use of Projects and Programs. International Journal of Project Management 33 (1), pp. 153-164, 2015.

[Re05] Reyck, B. D. et al.: The Impact of Project Portfolio Management on Information Technology Projects. International Journal of Project Management 23 (7), pp. 524-537, 2005.

[RO08] Rosacker, K. M.; Olson, D. L.: An Empirical Assessment of IT Project Selection and Evaluation Methods in State Government. Project Management Journal 39 (1), pp. 49-58, 2008.

[SC19] Sweetman, R.; Conboy K.: Finding the Edge of Chaos: A Complex Adaptive Systems Approach to Information Systems Project Portfolio Management. In Proceedings of the 27th European Conference on Information Systems, Uppsala, Sweden, June 8-14, 2019.

[Se16] Setterstrom, A. J.: IT Project Manager Decision-Making Authority: An Empirical Examination of Antecedents and Consequences. In Proceedings of the 37th International Conference on Information Systems, Dublin, Ireland, December 11-14, 2016.

[SK95] Santhanam, R.; Kyparisis, G. J.: A Multiple Criteria Decision Model for Information System Project Selection. Computers & Operations Research 22 (8), pp. 807-818, 1995.

[SS02] Shang, S.; Seddon, P. B.: Assessing and Managing the Benefits of Enterprise Systems: The Business Manager's Perspective. Information Systems Journal 12 (4), pp. 271-299, 2002.

[SS03] Serafeimidis, V.; Smithson, S.: Information Systems Evaluation as an Organizational Institution - Experience from a Case Study. Information Systems Journal 13 (3), pp. 251-274, 2003.

[TP15] Templier, M.; Paré, G.: A Framework for Guiding and Evaluating Literature Reviews. Communications of the Association for Information Systems 37 (1), pp. 112-137, 2015.

[VT16] Varajão, J.; Trigo, A.: Evaluation of IS Project Success in InfSysMakers: An Exploratory Case Study. In Proceedings of the 37th International Conference on Information Systems. Dublin, Ireland, December 11-14, 2016.

[Wi92] Willcocks, L.: Evaluating Information Technology Investments: Research Findings and Reappraisal. Information Systems Journal 2 (4), pp. 243-268, 1992.

[WW02] Webster, J.; Watson, R. T.: Analyzing the Past to Prepare for the Future: Writing a Literature Review. MIS Quarterly 26 (2), pp. xiii-xxiii, 2002.

[Zh17] Zhang, X. et al.: The Tradeoffs Between Portfolios in Multi-Attribute Project Portfolio Selection. In Proceedings of the 11th Annual IEEE International Systems Conference, Montreal, Canada, April 24-27, 2017.

[ZHT19] Zhang, X.; Hipel K. W., Tan, Y.: Project Portfolio Selection and Scheduling Under a Fuzzy Environment. Memetic Computing 11 (4), pp. 391-406, 2019.

Anforderungen strukturiert mit Schablonen dokumentieren in PARIS

Oliver Linssen

ifid - Institut für IT-Management & Digitalisierung
FOM Hochschule für Oekonomie & Management gemeinnützige GmbH, Essen

oliver.linssen@fom.de

Abstract: Die präzise Spezifikation und Dokumentation von Anforderungen an Produkte und (Dienst-) Leistungen ist von zunehmender Bedeutung in der Softwareentwicklung, wie auch im Projektmanagement. Mit Hilfe der Schablonen aus PARIS (PAtterns for RequIrements Specification) können Anforderungen in standardisierter Form dokumentiert werden. Durch diese standardisierte Form wird die Qualität von Anforderungsdokumenten, Lastenheften, Leistungsbeschreibungen positiv beeinflusst. Dieser Aufsatz stellt Sprachschablonen für Stakeholderanforderungen und Qualitätsanforderungen vor. Außerdem wird gezeigt, wie mit Begründungen die Aussagenkraft und Nachvollziehbarkeit von Anforderungen erhöht wird. Es wird weiterhin eine Möglichkeit gezeigt, Anforderungen an komplexe, zusammengesetzte System zu formulieren.

Keywords: Requirements Engineering, Anforderungsanalyse, Requirements Specification, Requirements Template, Requirements Boilerplate, Anforderungsschablone, Sprachschablone, Satzschablone, Requirements Pattern, Anforderungsmuster.

1 Einleitung

1.1 Problemstellung

Nach Ansicht des Verfassers gewinnt durch die zunehmende Arbeitsteilung in den Wertschöpfungsprozessen der Unternehmen die präzise Spezifikation und Dokumentation von Anforderungen an Produkte und (Dienst-)Leistungen eine immer größere Bedeutung. Was früher informell in Emails, Telefonaten oder im Sozialbereich der Unternehmen („Kaffeeküche“) abgesprochen wurde, muss heute möglichst präzise festgelegt werden, um Friktionen und technische, wie auch organisatorische Schnittstellenprobleme zu vermeiden oder wenigstens zu reduzieren. Die massiv zunehmende Komplexität der Werkzeugketten, die zunehmende Anzahl der eingesetzten IT-Lösungen und die immer weitergehende Integration der betrieblichen Prozesse (Stichwort „DevOps“ [1]) erhöhen hier weiterhin den Druck, durch konstruktive Maßnahmen [2, p. 65] Probleme eher zu vermeiden als Fehler zu beheben. Datengetriebene Anwendungen, die häufig aus sehr vielen Einzelanwendungen bestehen [3], erfordern nach Ansicht des Verfassers eine präzise Dokumentation der Anforderungen, um die Risiken von Fehlschlägen zu reduzieren. Die präzise Spezifikation und Dokumentation von Anforderungen an Produkte und (Dienst-)Leistungen ist die Aufgabe der Disziplin *Requirements Engineering* [4], einem Teilgebiet des *Software Engineerings* [5]. Auch im *Projektmanagement* [6, p. 4] und im *Business Engineering* [7] ist die Bedeutung präziser Projektziele, Produktanforderungen oder Leistungsbeschreibungen unbestritten.

1.2 Ziel des Beitrags

Ziel dieses Aufsatzes ist es, Verbesserungen für die Dokumentation von Anforderungen mit Sprachschablonen (*requirements templates*, *requirements boilerplates*) vorzustellen. Formuliert werden konkrete Vorschläge, die in der Mustersprache PARIS (PAtterns for RequIrements Specification) enthalten sind. PARIS versteht sich als umfassender Ansatz, Anforderungen in einer einheitlichen Form mit Hilfe einer reduzierten bzw. kontrollierten Sprache (*reduced natural language* oder *controlled natural language*) [8] zu dokumentieren.

1.3 Aufbau des Beitrags

In Abschnitt 2 werden die Aufgaben des Requirements Engineerings skizziert. Dabei wird insbesondere auf die Anforderungsdokumentation eingegangen. Man kann in diesem Bereich zwei unterschiedliche Ansätze unterscheiden, die modellbasierte Dokumentation und die Dokumentation mit natürlicher Sprache. In Abschnitt 3 werden Ansätze vorgestellt, in denen mit Hilfe von Sprachschablonen oder Textbausteinen Anforderungen dokumentiert werden. Abschnitt 4 bietet einen Überblick über den Ansatz PARIS (PAtterns for RequIrements Specification). Der Abschnitt 5 stellt Lösungen aus PARIS vor, die aus dem agilen Ansatz (Abschnitt 5.1 und 5.2) und aus der Notwendigkeit hergeleitet wurden, Anforderungen an Teilsysteme, Komponenten usw. (Abschnitt 5.3) zu dokumentieren. In Abschnitt 5.4 wird eine Schablone für Qualitätsanforderungen vorgestellt. Abschnitt 6 beinhaltet ein Fazit und einen Ausblick. Der Anhang in Abschnitt 7 beinhaltet eine tabellarische Übersicht mit anderen Ansätzen, die Sprachschablonen oder Textbausteine verwenden. Auf diese Tabelle wird in Abschnitt 3 eingegangen.

2 Requirements Engineering

Dieser Abschnitt erklärt, wie im Requirements Engineerings mit Hilfe von Anforderungsschablonen das Problem der natürlichsprachigen Anforderungsdokumentation gelöst werden soll.

2.1 Aufgabe des Requirements Engineering

Requirements Engineering befasst sich mit dem Erfassen, Dokumentieren, Prüfen, Abstimmen und Verwalten von Anforderungen [9]. Im deutschen sind auch die Begriffe Anforderungsanalyse und Anforderungsmanagement üblich, wodurch allerdings die Konnotation einer Ingenieursdisziplin verlorengeht, die im englischen Begriff enthalten ist. Das Requirements Engineering ist ein Teilgebiet des Software Engineering [10]. In zunehmendem Maße beschäftigt sich aber auch das Projektmanagement mit dieser Thematik [6]. Das Requirements Engineering sieht sich einer Reihe von Herausforderungen gegenüber [11], wie zum Beispiel dem Problem, die „richtigen“ oder „relevanten“ Anforderungen zu ermitteln [12]. Ein anderes Problemfeld ist, dass Anforderungen häufig zu vage dokumentiert sind [11].

2.2 Probleme der Anforderungsdokumentation

Im Rahmen des Requirements Engineerings entsteht eine wohlgeordnete Menge von Anforderungen, die der Ausgangspunkt für die weiteren Schritte in einem Projekt sind. Für diese Dokumentation existiert eine Vielzahl von Begriffen. Exemplarisch genannt seien Lastenheft, Leistungsbeschreibung, Anforderungsdefinition, Requirements Specification, Zieldefinition. Unvollständige, unpräzise und widersprüchliche Anforderungen sind ein Dauerproblem, welches in beträchtlichem Maße zum Scheitern von Projekten führt [13, p. 35 ff.], [11]. Im Requirements Engineering werden eine Vielzahl von Techniken verwendet, um diesem Problem zu begegnen [13]–[16]. Hier lassen sich grundsätzlich zwei Arten der Anforderungsdokumentation unterscheiden, die jedoch nicht komplementär, sondern einander ergänzend verwendet werden. Modellbasierte Ansätze verwenden grafische Darstellungen oder mathematische Modelle, um die Anforderungen zu dokumentieren. Letztere bezeichnet man als formale Spezifikation. Die Anforderungen werden in diesem Fall mit Hilfe von Aussagen- und Prädikatenlogik dokumentiert [17]. Diese Ansätze zeichnen sich durch eine Reihe von Vorteilen aus, wovon Präzision nur einer ist. Grafische Darstellungen zu verwenden entspricht der guten Praxis aller Ingenieurswissenschaften, exemplarisch sei hier Hochbau [18], Maschinenbau [19] oder Elektrotechnik [20] genannt. In der IT und insbesondere im Software Engineering werden eine Vielzahl von grafischen Darstellungen verwendet. Hier wird exemplarisch auf die UML [21] oder

BPMN [22] verwiesen. Ein umfassender Überblick über entsprechende Methoden und Notationen findet sich in [16].

Man kann beobachten, dass in sehr vielen Fällen Anforderungen mit natürlicher Sprache dokumentiert werden. Dies ist zunächst überraschend, weil natürliche Sprache offensichtlich unpräzise, mehrdeutig, überaus komplex und interpretationsfähig ist [14, p. 212 ff.]. Die Ursachen sind nicht Thema dieser Veröffentlichung, aber Pragmatismus, Zeitnot und fehlende Ausbildung spielen nach Ansicht des Verfassers eine wichtige Rolle. Eine Möglichkeit, um die oben genannten Probleme der Dokumentation von Anforderungen mit Freitext bzw. ungebundener Sprache zu reduzieren, ist die Verwendung einer reduzierten bzw. kontrollierten Sprache (*reduced natural language* oder *controlled natural language*) [8]. Kontrollierte Sprachen verwenden Regeln, um die Grammatik und den Wortschatz natürlicher Sprachen einzuschränken. Im Requirements Engineering wird auf diese Weise der Aufbau der Anforderungen normiert. Dabei werden *Anforderungsschablonen* (*requirements templates*, *requirements boilerplates*) verwendet, um die Struktur einer vollständigen einzelnen Anforderung festzulegen [9]. Für einzelne Abschnitte in Anforderungen (zum Beispiel Bedingungen) werden *Textbausteine* definiert, die Vorgaben für die Formulierung beinhalten.

3 Verwandte Arbeiten

Die Idee, mit Hilfe einer reduzierten Sprache Anforderungen zu dokumentieren, ist nicht neu. Da in der wissenschaftlichen Literatur keine Übersicht über solche Ansätze existiert, wird mit der Tabelle 5 (im Anhang) der aktuelle Stand der Forschung in Form einer tabellarischen Übersicht dokumentiert. Die Beiträge wurden seit dem Beginn der Entwicklung von PARIS im Jahr 2013 im Rahmen von mehreren Literaturrecherchen unter anderem in IEEE Xplore Digital Library, ACM Digital Library, arXiv, BASE, DBLP und Google Scholar gesammelt, wobei unter anderem die Suchbegriffe „Requirements Template“, „Requirements Boilerplate“ und „Requirements Pattern“ verwendet wurden. Die Liste wurde durch eine Rückwärtssuche vervollständigt, weil bereits bekannte Veröffentlichungen bei einer Schlagwortsuche nicht in der Treffermenge enthalten waren. Einige Quellen waren schon bekannt ([23], [24], [9], [25], [26]) oder wurden im Rahmen von anderen Arbeiten (zum Beispiel [27], [28]) identifiziert. Berücksichtigt wurden Veröffentlichungen, die die Formulierung einzelner Anforderungen behandeln. Die Veröffentlichungen, die sich mit der Struktur des gesamten Anforderungsdokuments beschäftigen, wurden nicht berücksichtigt, wenn sie nicht außerdem die Formulierung einzelner Anforderungen thematisierten. Neben wissenschaftlichen Ansätzen wurden außerdem Ansätze aus der nicht-wissenschaftlichen Literatur [24]–[26], [29]–[32] berücksichtigt.

Die erste Spalte („Jahr“) in Tabelle 5 beinhaltet das Jahr der Veröffentlichung. Hier muss beachtet werden, dass bei Monografien [9], [29], [30], [33] nicht das Jahr der Erstveröffentlichung verwendet wurde, sondern das der aktuellen Auflage. Die Tabelle gibt somit nur eingeschränkt wieder, in welcher zeitlichen Reihenfolge die Ansätze entstanden sind. Wenn der Ansatz einen Namen besitzt, ist dieser in der zweiten Spalte („Name“) aufgeführt. Die dritte Spalte („Umfang“) gibt die Anzahl der in der Veröffentlichung vorgestellten Schablonen oder Textbausteine an. Hier wurden die Schablonen oder Bausteine in den Veröffentlichungen gezählt. Wenn in Veröffentlichungen die genannten Schablonen als Beispiele bezeichnet wurden und aus der Veröffentlichung der Gesamtumfang nicht zu ermitteln war, wurde U (*unbekannt*) angegeben. Die vierte Spalte („Typ“) gibt an, ob in der Veröffentlichung Schablonen (S) oder Textbausteine (B) definiert werden. Die fünfte Spalte („Inhalt“) gibt an, für welche Aufgaben Schablonen oder Bausteine formuliert werden. Die Zahlen in Klammern geben an, wie viele Varianten definiert werden. Die sechste und letzte Spalte („Quelle(n)“) verweist auf die Quelle(n), in der der Ansatz veröffentlicht wurde. Der zugehörige Quellenbeleg findet sich im Literaturverzeichnis. Zwei Ansätze fallen aus dem Rahmen. Gilb definiert in PLANGUAGE [23], [34] keine Sprachschablonen, sondern eine komplette Spezifikationssprache, insbesondere für nichtfunktionale

Anforderungen. Der Requirement Pattern Catalog [31] besteht nicht aus Sprachschablonen für einzelne Anforderungen, sondern schlägt den Aufbau ganzer Textabschnitte vor.

Eine detaillierte Untersuchung der unterschiedlichen Ansätze wird aktuell durchgeführt. Diese Untersuchung wird zum Teil dadurch erschwert, dass nicht in allen Veröffentlichungen die Schablonen vollständig publiziert sind (zum Beispiel [28], [35], [36]), sondern teilweise nur einige Beispiele genannt werden. Gut zu erkennen ist aber, dass die Mehrheit der Ansätze den Aufbau von funktionalen Anforderungen festlegt, also Anforderungen an das Verhalten von Systemen. Dies ist evident, weil diese Anforderungsart den größten Anteil der Anforderungen ausmacht. Auch in der ISO 29148 werden Schablonen für funktionale Anforderungen formuliert [37]. Klar erkennbar ist die Zunahme wissenschaftlicher Arbeiten zu diesem Thema ab 2009/2010. Die Ursachen hierfür sind noch zu untersuchen. Unter Praktikern ist im deutschsprachigen Raum insbesondere das Werk von Rupp et.al. [29] sehr bekannt. Die Schablone für User Storys [24], die im agilen Bereich der De-facto-Ansatz ist, kann als ein früher Ansatz betrachtet werden, den Aufbau von Anforderungen zu normieren.

Die Fülle der vorliegenden Arbeiten zeigt, dass die Verwendung einer reduzierten bzw. kontrollierten Sprache in Form von Anforderungsschablonen ein etablierter Ansatz ist, Anforderungen zu dokumentieren. Die Satzschablone für funktionale Anforderungen des IREB (International Requirements Engineering Board) hat sich als ein De-facto-Standard im Requirements Engineering etabliert [9], weil sie Inhalt einer weitverbreiteten Zertifizierung für das Requirements Engineering ist [38]. Diese Schablone ist unkompliziert, schnell zu lernen, weit verbreitet und hat sich in vielen Situationen als ausreichend bewährt. Sie wurde als Ausgangspunkt für die Entwicklung der ESFA (Erweiterte Schablone für Funktionale Anforderungen) in PARIS verwendet. Ihr Aufbau ist im Detail in [39] beschrieben.

4 Überblick über PARIS

PARIS (PAtterns for RequIrements Specification) besteht aus einer Reihe von Sprachschablonen, um Anforderungen in standardisierter Form zu dokumentieren. Durch diese standardisierte Form wird die Qualität von Anforderungsdokumenten, Lasten- und Pflichtenheften oder Leistungsbeschreibungen positiv beeinflusst. Die Historie und die Entwurfsziele von PARIS sind nicht Thema dieser Veröffentlichung, sondern in [39] dokumentiert. In Tabelle 1 sind die aktuell vorhandenen Schablonen zusammengefasst.

Schablone	**Kurze Erläuterung**
Funktionale Anforderung	Anforderung an die Funktionalität eines Systems oder eines Teils eines Systems. Wird als ESFA [39] bezeichnet.
Technische Anforderung	Technische Vorgabe an die Funktionalität eines Systems oder eines Teils eines Systems.
Qualitätsanforderung	Anforderung an Qualitätsmerkmale, wie zum Beispiel Geschwindigkeit, Benutzerfreundlichkeit, Robustheit. Wird als ESQUA (Erweiterte Schablone für Qualitäts-Anforderungen) bezeichnet und in Abschnitt 5.4 behandelt.
Dienstleistung	Anforderung an Tätigkeiten von Personen, Personengruppen, Körperschaften, Organisationen oder Institutionen.
Stakeholderanforderung	Anforderung von Stakeholdern (zum Beispiel Kunden, Benutzer, Gesetzgeber, Behörden).

Schablone	Kurze Erläuterung
Eigenschaft	Anforderung an die Beschaffenheit zum Beispiel eines Systems oder eines Teils eines Systems.
Technische Eigenschaft	Technische Vorgabe an die Beschaffenheit zum Beispiel eines Systems oder eines Teils eines Systems.
Abnahmekriterium	Kriterium, unter welchen Bedingungen eine Anforderung als erfüllt gilt.
Ziel	Festlegung über einen zu erreichenden Zustand.
Kontext	Angabe über die Umgebung bzw. das Umfeld, welches zum Beispiel für ein System oder ein Teil eines Systems relevant ist.
Glossareintrag	Definition eines Begriffs.

Tabelle 1: Schablonen in PARIS

PARIS konzentriert sich auf Schablonen für die einzelnen Elemente von Anforderungsspezifikationen. Aus diesem Grund beinhaltet PARIS keine Konzepte, zum Beispiel für die Gesamtstruktur einer Anforderungsspezifikation (bzw. Lasten- oder Pflichtenheft), die Priorisierung von Anforderungen oder die Referenzierung von Anforderungen untereinander oder mit den anderen Artefakten, die im Rahmen der Systementwicklung entstehen. Für die Gesamtstruktur des Anforderungsdokuments kann auf etablierte Ansätze wie zum Beispiel das V-Modell XT [40] oder auf den ISO-Standard 29148 [37] zurückgegriffen werden.

5 Lösungen aus PARIS

In diesem Abschnitt werden exemplarisch einige Lösungen aus PARIS vorgestellt, die aus konkreten Problemen von Anwendern, Seminarteilnehmern und in Abschlussarbeiten entstanden sind.

5.1 Stakeholderanforderungen

Mit der Schablone für funktionale Anforderungen ESFA werden primär Anforderungen an ein System oder eines Teils eines Systems dokumentiert. Das bedeutet, dass die Funktionalität beschrieben wird, die ein System Anwendern oder über eine Schnittstelle bereitstellt. Diese Schablonen sind nicht geeignet, um zum Beispiel die Anforderungen von Stakeholdern zu erfassen, die keine Benutzer eines Systems sind. Hierunter fallen beispielsweise Vorgaben des Gesetzgebers oder regulatorische Anforderungen. Hierunter fallen aber zum Beispiel auch die Wünsche von Anwohnern, dass ein Bauprojekt nicht zu einer übermäßigen Lärmbelastung für sie führt. Dies ist insbesondere für das Projektmanagement wichtig, welches sich naturgemäß mit allen Stakeholdern beschäftigt [6].

In der agilen Welt wird mit den User Storys [24] eine andere Sicht favorisiert. Hier steht nicht die Funktionalität eines Systems im Fokus, sondern die Absicht eines Stakeholders. Ein wesentlicher Unterschied besteht darin, dass mit User Storys Anforderungen aller Stakeholder dokumentiert werden können, unabhängig, ob sie ein Benutzer eines Systems sind oder nicht. Diese Idee wurde in PARIS übernommen. Die Schablone für Stakeholderanforderungen ergänzt in PARIS die Schablone für funktionale Anforderungen. Stakeholderanforderung bedeutet in PARIS eine Anforderung eines Stakeholders (zum Beispiel Gesetzgeber, Behörden, regulierende Einrichtungen, Anwohner, Shareholder) oder eines abstrakten Ursprungs (zum Beispiel „der Markt“, „öffentliche Meinung“), durch den ein Wunsch oder eine Vorgabe dokumentiert wird. Die Schablone existiert in zwei Varianten:

Schablone mit Bedingung:

<Bedingung> ("FORDERT" | "FORDERN") (<Stakeholder> |<Herkunft, Ursprung, Quelle, Ort der Forderung>) [Spezifikation der Anforderungsquelle] ", DASS" <Inhalt der Forderung> <Modalität> [Begründung] "." (01)

Schablone ohne Bedingung:

(<Stakeholder> |<Herkunft, Ursprung, Quelle, Ort der Forderung>) ("FORDERT" | "FORDERN") [Spezifikation der Anforderungsquelle] ", DASS" <Inhalt der Forderung> <Modalität> [Begründung] "." (02)

Anforderungsschablonen werden in PARIS mit vereinfachten Backus-Naur-Formen [41] definiert. Spitze Klammern („<“ und „>“) bedeuten, dass das Element ausgefüllt werden muss. Für optionale Elemente werden eckige Klammern („[“ und „]“) verwendet. Runde Klammern („(“ und „)“) fassen Alternativen zusammen, wobei die Alternativen mit einem senkrechten Strich („|“) voneinander getrennt werden. Eines der Elemente muss ausgewählt werden. Wörter in Anführungsstrichen sind Terminalsymbole. Die Farben wurden auf Wunsch von Seminarteilnehmern als zusätzliche Hervorhebung eingeführt, um die Lesbarkeit der Schablonen zu erhöhen. Die Bedeutung der einzelnen Elemente der Schablone ist in Tabelle 2 erläutert.

Element	**Erläuterung**
Bedingung	Eine beliebig komplexe Bedingung, die zum Beispiel als Vorbedingung im Sinne von Bedingung → Inhalt der Forderung verwendet wird.
FORDERT, oder FORDERN, … DASS	In PARIS vordefinierte Schlüsselwörter. Die Schlüsselwörter werden ohne Großschreibung in die Anforderungen übernommen.
Stakeholder	Bezeichnung eines Stakeholders.
Herkunft, Quelle, Ort der Forderung	Wird verwendet, wenn die Anforderung einen abstrakten Ursprung hat, wie zum Beispiel „das Renommee“, „der Markt“, „die öffentliche Meinung“.
Spezifikation der Anforderungsquelle	Präzisierende Angabe, wo die Anforderung herstammt, wie zum Beispiel die Angabe eines konkreten Gesetzes.
Inhalt der Forderung	Vorgabe oder Wunsch des Stakeholders oder eines abstrakten Ursprungs.
Modalität	Angabe einer Modalität. In der Regel ein Modalverb.
Begründung	Eine Begründung. Die Angabe ist optional.

Tabelle 2: Elemente der Schablone für Stakeholderanforderungen

Hierzu drei fiktive Beispiele:

Der Gesetzgeber fordert in § 238 (Buchführungspflicht) des HGB, dass jeder Kaufmann Bücher führen muss. (a)

Das BSI fordert, dass Unternehmen ein Sicherheitsmanagementsystem einrichten müssen. (b)

Das IT-Management fordert, dass nach der Umsetzung eines Change Requests automatisierte Regressionstests ausgeführt werden müssen. (c)

Stakeholderanforderungen sind häufig allgemeiner als funktionale Anforderungen, weil hier noch nicht festgelegt ist, ob und welches (IT-)System die Anforderung erfüllt. Aus Stakeholderanforderungen können im weiteren Verlauf zum Beispiel Systemanforderungen entstehen.

Aus der Anforderung des Gesetzgebers in Beispiel (a) können Anforderungen an eine IT-Lösung zur Finanzbuchhaltung entstehen. Prinzipiell denkbar ist aber auch, dass aus einer Stakeholderanforderung zum Beispiel eine Anforderung an einen Dienstleister entsteht. In der Praxis konnte beobachtet werden, wie wichtig es ist, dass auch die Stakeholderanforderungen erfasst werden, weil sonst unter Umständen die Information verlorengeht, warum eine Systemanforderung existiert.

Für die unterschiedlichen Schablonentypen in PARIS wurden Frageschemata definiert, um die Anforderungen strukturiert abzufragen. Damit soll die strukturierte Ermittlung von Anforderungen gefördert werden. In Tabelle 3 ist zu sehen, wie die Stakeholderanforderung durch ein Frageschema „befüllt" werden kann.

Frage	**Element der Schablone**
Unter welcher Bedingung wird etwas getan?	<Bedingung>
Fordert Wer?	("FORDERT" | "FORDERN") (<Stakeholder> |<Herkunft, Ursprung, Quelle, Ort der Forderung>)
Wo?	[Spezifikation der Anforderungsquelle]
Was?	", DASS" <Inhalt der Forderung>
Mit welcher Verbindlichkeit wird es gefordert?	<Modalität>
Warum wird es gefordert?	[Begründung]

Tabelle 3: Frageschema für Stakeholderanforderungen.

5.2 Begründungen

Selbst wenn durch die Anforderungen dokumentiert ist, *was* gefordert ist, dann geht häufig die Information über den Grund verloren, *warum* es gefordert wurde. Es führt dazu, dass die Anforderungen im Zeitverlauf wiederholt hinterfragt und mehrfach diskutiert werden. Eine Auswertung der untersuchten Quellen (siehe Tabelle 1) ergab, dass in Anforderungen in der Regel keine Begründungen dokumentiert werden. Auch hier wird in User Storys ein anderer Weg beschritten. In einer wohldefinierten User Story ist eine Begründung als dritter variabler Teil enthalten:

As a <type of user>, I want <some goal> so that <some reason>. ([42], S. 239. Unterstreichung vom Verfasser ergänzt.)

Hierzu findet sich folgendes Beispiel:

As a vice president of marketing, I want to select the time frame to use when reviewing the performance of past advertising campaigns so that I can identify profitable ones. [42], S. 247.

Diese Begründungen verbessern erheblich die spätere Nachvollziehbarkeit und das Verständnis der Anforderungen. Aus diesem Grund beinhalten alle PARIS-Schablonen die Möglichkeit, eine Begründung zu dokumentieren. Damit kann das obige Beispiel zur Buchführungspflicht wie folgt erweitert werden:

Der Gesetzgeber fordert in § 238 (Buchführungspflicht), dass jeder Kaufmann Bücher führen muss, weil sich sonst kein sachverständiger Dritter innerhalb angemessener Zeit einen Überblick über die Geschäftsvorfälle und über die Lage des Unternehmens verschaffen kann.

Das folgende Beispiel stellt eine funktionale Anforderung mit einer Begründung dar:

Am letzten Tag des Monats muss das System Libri dem Administrator ermöglichen, eine Auswertung der Verbrauchsdaten zu erzeugen, weil die Verbrauchsdaten die Grundlage für die Rechnungserstellung sind.

Dabei wurde die Schablone für funktionale Anforderungen aus [39] verwendet:

<Bedingung> <Modalität> <System> <Benutzer> "ERMÖGLICHEN," [Objektbeschreibung] <Prozessbeschreibung> [Begründung] "." (03)

Begründungen können auf vielfältige Art und Weise verwendet werden, zum Beispiel als Verweis auf den geschäftlichen Nutzen, ein Gesetz, einen Anforderungssteller, einen Stakeholder, ein Dokument, ein Produkt eines Mitbewerbers, eine Zielgruppe, ein System- oder Projektziel oder eine Managemententscheidung. Sie haben sich als guter Fortschritt für die Verständlichkeit von Anforderungen erwiesen. Sie sind in den Schablonen als optionales Element definiert, weil Anforderungen nicht immer einer Begründung bedürfen oder die Begründung trivial ist. Es ist von geringem Wert, zu begründen, warum zum Beispiel eine Addition korrekt ausgeführt werden muss, oder warum in einem Kundenverwaltungssystem Kundendaten angelegt werden müssen.

5.3 Anforderungen an Teile eines Systems

Im Requirements Engineering werden Anforderungen an ein „System“ formuliert, also die Außensicht des Benutzers. Dies war in der Vergangenheit sinnvoll, weil es nicht die Aufgabe des Anforderungsstellers (zum Beispiel Kunde, Anwender, Auftraggeber) war, die Anforderungen einzelnen Komponenten oder Bauteilen des Systems zuzuordnen, die er in der Regel noch nicht definiert waren. Microservices verändern in erheblichem Maße die IT-Landschaften [43], was auch auf das Requirements Engineering einwirkt. Auch wenn keine strenge Definition des Begriffs Microservice existiert, besteht Einigkeit darüber, dass durch Microservices große, komplexe Anwendungen in eine Vielzahl von kleinen („atomaren“) Diensten zerlegt werden [44]. In einer Microservices-Architektur existiert im klassischen Sinn kein „System“, sondern eine flexible, sich möglicherweise ständig verändernde Ansammlung von Diensten (den Microservices), die gemeinsam die notwendige Funktionalität bereitstellen. Der Systembegriff, der in der Praxis häufig mit Anwendung gleichgesetzt wird, verliert hier erheblich an Bedeutung. In bestimmten Situationen benötigt man die Möglichkeit, auf jeder Betrachtungsebene Anforderungen formulieren zu können, auch wenn dies in der Regel nicht von Kunden oder Anwendern erfolgen wird, sondern eher von entsprechend qualifizierten Fachleuten. Auch für das Projektmanagement ist dies von Bedeutung, wenn Anforderungen zum Beispiel an die Lieferanten unterschiedlicher Teile eines Gesamtsystems spezifiziert werden müssen.

PARIS besitzt hierfür ein flexibles Konzept, Anforderungen an Systeme, Subsysteme, Komponenten, Module, Services, Objekte oder selbstdefinierte Bezüge zu richten. Das Element System aus Schablone (03) ist in PARIS wie folgt definiert:

System = ([Artikel] [Ergänzung] <Bezug> [Ergänzung] || [Artikel] [Ergänzung] <Eigenname> [Ergänzung]); (04)

Die Bezeichnung System aus (04) setzt sich aus einem Bezug und/oder einem Eigennamen (sowie einem Artikel und Ergänzungen (Ergänzung), auf die hier aber nicht eingegangen wird) zusammen. Der Bezug in (05) definiert eine Reihe von vorgegebenen Begriffen, lässt aber auch den Raum, eigene Bezüge zu definieren:

Bezug = ("SYSTEM" | "SUBSYSTEM" | "BAUGRUPPE" |"KOMPONENTE" | "MODUL" |"OBJEKT" |"SERVICE" | <Eigener Bezug>); (05)

Damit können sehr flexibel Anforderungen auf unterschiedlichsten Ebenen formuliert werden. Tabelle 4 beinhaltet einige fiktive Beispiele in tabellarischer Form, um die Zuordnung zu erleichtern.

[Artikel]	<Bezug>	<Eigenname>
das	System	
das	System	Libri
die	Komponente	
die	Komponente	RepGen
der	Service	Kunde

Tabelle 4 Beispiele zu Bezügen und Namen

Auf diese Weise kann eine funktionale Anforderung an einen Service Kunde formuliert werden:

Der Service Kunde muss ermöglichen, den Namen des Kunden zu erhalten.

Verwendet wird dabei eine Variante der Schablone ESFA für funktionale Anforderungen, mit deren Hilfe Anforderungen an Schnittstellen dokumentiert werden [39]:

<System> <Modalität> "ERMÖGLICHEN," [Objektbeschreibung] <Prozessbeschreibung>"." (06)

Nur in der Monografie von Ebert [33, p. 107] ist eine ähnliche Lösung erkennbar. Die Schablone für funktionale Anforderungen aus [39] kann auch verwendet werden, um zu dokumentieren, wie eine Anforderung an ein Gesamtsystem intern in Anforderungen an einzelne Subsysteme, Komponenten oder Services aufgeteilt wird.

5.4 Qualitätsanforderungen

Stakeholder wünschen eine kostengünstige Lösung, eine benutzerfreundliche Anwendung, einen schnellen Drucker, eine hochwertige Ausstattung (zum Beispiel bei einem Hotelzimmer), einen ordentlichen Handwerker oder eine leistungsfähige Kantine. Hierdurch wird durch den Stakeholder eine bestimmte Qualität gewünscht. Nach [45, p. 20] ist eine Qualitätsanforderung „*eine Anforderung, die sich auf einen Qualitätsaspekt bezieht, der nicht durch funktionale Anforderungen abgedeckt ist.*“ Dass ein Drucker druckt, ist die funktionale Anforderung. Dass der dabei außerdem schnell sein soll, ist die Qualitätsanforderung, wobei Geschwindigkeit der Qualitätsaspekt ist. Das Problem in der Praxis ist nicht die Qualitätsanforderung, sondern die fehlende Klarheit, was genau gefordert ist, bzw. wann es erreicht ist. Diese Herausforderung gilt es sowohl im Requirements Engineering, als auch im Projektmanagement zu bewältigen. Wenn Starke [46, p. 37] schreibt, Qualität sei nur indirekt messbar und man könne nur einzelne Eigenschaften messen, dann ist dem zuzustimmen. Verwendet man die übliche Terminologie des Qualitätsmanagements [47], wird die Qualität durch (Qualitäts-)merkmale bestimmbar. Die Qualität einer Sache setzt sich nach dem FCM-Modell von McCall [48] aus mehreren Qualitätsmerkmalen (engl. *Factor*) zusammen, die weiterhin in Qualitätsteilmerkmale (engl. *Criteria*) zerlegt werden können. Für die Teilmerkmale werden Indikatoren (engl. *Metrics*) benötigt, um sie mess- bzw. bewertbar zu machen. Die Definition aus [15, p. 462] verallgemeinernd, wird mit dem Maß eine Skala und eine Methode definiert, mit der ein Wert für diese Skala bestimmt werden kann.

Eine erste Anforderungsschablone, die diesen Zusammenhang abgebildet hat, wurde 2014 entwickelt, anschließend getestet und ab 2016 in Seminaren unterrichtet. Die Schablone basierte in weiten Teilen auf Schlüsselwörtern, die aus PLANGUAGE [23] entnommen wurden. Die Zerlegung in Qualitätsmerkmale und Qualitätsteilmerkmale wurde aber von Anwendern als mühsam und zeitaufwändig bezeichnet. Die Dokumentation der Zerlegung in einer Schablone ist eigentlich auch unnötig, weil man hier (zumindest für Software) auf etablierte Qualitätsmodelle zurückgreifen kann [49]. Die aktuelle Version wurde 2020 entwickelt und im Rahmen einer Master Thesis in einer Bundesbehörde evaluiert und dort von den Beteiligten positiv bewertet. Sie vereinfacht das FCM-Modell zum Beispiel in der Form, dass keine Unterscheidung

zwischen Qualitätsmerkmal und Qualitätsteilmerkmal existiert. Die Schablone wird als ESQUA (Erweiterte Schablone für QualitätsAnforderungen) bezeichnet und besteht aus folgenden Elementen:

Qualitätsanforderung = <Qualitätsmerkmal> 1{Qualitätskriterium}m; (07)

Eine Qualitätsanforderung (07) besteht aus einem Qualitätsmerkmal und einem oder mehreren Qualitätskriterien. Mit (08) wird das Qualitätsmerkmal aus (07) mit einer Bedingung definiert:

Qualitätsmerkmal = <Bedingung> <Modalität> (<System>|<Stakeholder>) "DAS QUALITÄTSMERKMAL BESITZEN" <Qualitative Eigenschaft> [Begründung]"."; (08)

Bei einem Qualitätsmerkmal ohne Bedingung fällt die <Bedingung> in (08) weg und <Modalität> tauscht mit (<System>|<Stakeholder>) die Position. Das Qualitätsmerkmal definiert, welche Art von Qualität gefordert wird. Um den Lernaufwand zu reduzieren, folgt der Aufbau der Schablonen in PARIS einem einheitlichen Grundaufbau. Darum ähnelt das Qualitätsmerkmal dem Aufbau einer funktionalen Anforderung, ersetzt aber die Prozessbeschreibung (siehe (03)) durch eine geforderte Qualitative Eigenschaft. Im Qualitätsmerkmal ist der Grundaufbau der Schablonen mit „*Bedingung – Hauptteil – Begründung*“ erkennbar, wobei der Hauptteil (siehe (08)) aus den Elementen <Modalität> (<System>|<Stakeholder>) "DAS QUALITÄTSMERKMAL BESITZEN" <Qualitative Eigenschaft> besteht. Der Aufbau eines Qualitätskriteriums aus (07) ist wie folgt definiert:

Qualitätskriterium = <Indikator> <Messverfahren> <Maßeinheit> <Ziel>; (09)

Ein Qualitätskriterium setzt sich zusammen aus dem Indikator, einem Messverfahren, einer Maßeinheit und einem Ziel.

Indikator = "QUALITÄTSINDIKATOR:" <Maß, welches zur Bewertung der Qualität verwendet wird>; (10)

Der Indikator erklärt das Maß, welches zur Bestimmung des Qualitätsmerkmals verwendet werden soll.

Messverfahren = "MESSVERFAHREN:" <Art der Messung>; (11)

Das Messverfahren erklärt, wie gemessen werden soll.

Masseinheit = "MASSEINHEIT:" <Maßeinheit, die im Messverfahren verwendet wird>; (12)

Die Maßeinheit legt fest, welche Maßeinheit verwendet wird.

Ziel = "ZIEL:" <Welche/n Wert/e der Qualitätsindikator erreichen muss>; (13)

Das Element Ziel legt fest, welcher Wert oder welche Werte erreicht werden müssen.

Dieser Aufbau erzwingt, dass in der Qualitätsanforderung durch ein Messverfahren präzise festgelegt wird, welche Erwartung mit dem Qualitätsmerkmal verbunden ist. Werden die Ziele in den Qualitätskriterien erfüllt, gilt die Qualitätsanforderung als erreicht. Hierzu soll folgendes fiktive Beispiel betrachtet werden, welches bewusst einfach gehalten ist:

Die Kantine muss das Qualitätsmerkmal besitzen, leistungsfähig zu sein, weil die Verpflegung der Mitarbeiter einen nachweisbaren Einfluss auf die Produktivität der Arbeitsergebnisse am Nachmittag besitzt.

Qualitätsindikator: Zeit, die man an der Essensausgabe wartet.
Messverfahren: Zufällige Zeitmessung zwischen 12:00 und 13:30 an der Ausgabe für Hauptgerichte. Die Zeitmessung beginnt, wenn sich der Kunde an der Schlange der Hauptausgabe anstellt. Die Zeitmessung endet, wenn der Zahlvorgang an der Kasse abgeschlossen ist.

Maßeinheit: Sekunden
Ziel: Voll erfüllt: x <= 360. Nicht erfüllt: x > 360.

Qualitätsindikator: Anzahl der angebotenen Hauptgerichte.
Messverfahren: Zählen der angebotenen Hauptgerichte zwischen 11:30 und 14:00. Der Zeitpunkt der Zählung ist zufällig.
Maßeinheit: Natürliche Zahl N.
Ziel: Voll erfüllt: x >= 3. Ausreichend erfüllt: x >= 2. Nicht erfüllt: x < 2.

Andere Qualitätsmerkmale („Auswahl", „Zufriedenheit") führen natürlich zu völlig anderen Qualitätsindikatoren. Der höhere Aufwand, eine Qualitätsanforderung – bestehend aus Qualitätsmerkmalen und Qualitätskriterien – zu formulieren, ist deutlich erkennbar. Aber der Grundaufbau ist einfach genug, um im Rahmen der Anforderungsanalyse auch von Nicht-Technikern verstanden und angewendet zu werden. In [50, p. 42] und in [33] finden sich ähnliche Vorschläge in tabellarischer Form, wobei dort die Qualitätsanforderung nur aus einem Qualitätskriterium besteht. Diesen Vorschlägen fehlt außerdem die Möglichkeit, Modalität, Bedingung, Begründung und das System zu spezifizieren, wie hier in (08) geschehen.

6 Fazit und Ausblick

Sprachschablonen stellen eine leistungsfähige Möglichkeit dar, die Anforderungsdokumentation zu verbessern. Dieser Aufsatz zeigt, wie in PARIS Stakeholderanforderungen dokumentiert werden. Weiterhin wurde gezeigt, wie die Aussagekraft von Anforderungen durch Begründungen erhöht wird und wie Anforderungen flexibel auf unterschiedlichen Ebenen formuliert werden können. Außerdem wurde eine Schablone vorgestellt, mit deren Hilfe Qualitätsanforderungen formuliert werden.

Die Entwicklung von PARIS wurde 2013 begonnen und wird aktuell fortgeführt. In [39] wird dargestellt, wie bei Bedarf weitere Schablonen entwickelt werden. PARIS wird in Industrieseminaren unterrichtet und im Rahmen von Abschlussarbeiten an der FOM[1] eingesetzt. Aktuelle Arbeiten befassen sich mit dem Einsatz der PARIS-Schablonen in Ausschreibungsdokumenten, Anforderungen an ein NLP-Tool zur Analyse natürlichsprachiger Anforderungen, Anforderungen im Projektportfoliomanagement und Anforderungen an die Skalierung von Microservices. Eine weitere Arbeit untersucht detailliert die Gemeinsamkeiten und Unterschiede der Ansätze, die auf Schablonen und Textbausteinen basieren (siehe Kapitel 3). Es werden weitere Forschungspartner zur Evaluierung und Weiterentwicklung der Schablonen gesucht. Geplant ist, die verfügbaren Schablonen auf Researchgate[2] als Teil des PARIS-Projekts zu veröffentlichen.

Anhang: Auf Schablonen und Textbausteinen basierende Ansätze im Requirements Engineering

Jahr	Name	Umfang	Typ	Inhalt	Quelle(n)
1989	Planguage	U	B	Methode und Vokabular insb. für Qualitätsanforderungen.	[23], [34]
1992	Natural Language Approach	8	B	Struktur bzw. Eigenschaft (2), Verhalten (4), Randbedingung (1), Sätze (2)	[51]
1994	–	5	B	Funktionale Anforderung, Fähigkeit eines Systems, Randbedingung, Funktionale Anforderung an ein Subsystem, Anforderung an Schnittstelle eines Subsystems	[52]
1995	–	1	S	Funktionale Anforderung	[53]
1999	L-Pattern, R-Pattern	3	S&B	Informationen, Funktionale Anforderung (Use Case), Nichtfunktionale Anforderung	[54]

[1] FOM Hochschule für Oekonomie & Management: https://www.fom.de.
[2] ResearchGate: https://www.researchgate.net.

Jahr	Name	Umfang	Typ	Inhalt	Quelle(n)
2003	–	38	B	Funktionale Anforderung (2), Ereignis (6), Reaktion (3), Berechnung (1), Bedingung (9), Beziehung (7), Ausnahme (5), Muster für besondere Aspekte (4), Nichtfunktionale Anforderung (1)	[55]
2004	User Story	1	S	Stakeholderanforderung	[24]
2006		28	B	Logische Aussagen (15), Allgemeine Anforderung (2), Ereignis & Bedingung (4), zeitliche Randbedingung (4), Nebensatz (3)	[56]
2007	Requirement Pattern Catalog	37	S	Grundsätzliche Aufgaben (6), Informationen (6), Entitäten (4), Benutzeranforderungen (3), Performance-Anforderungen (5), Modifizierbarkeit (6), Zugangskontrolle (5), Kaufmännische Anforderungen (2)	[31]
2009	–	1	S	Funktionale Anforderung	[57]
2009	–	U	S	Exemplarisch werden 3 Templates erwähnt: Benutzeranforderung, Benutzeranforderung unter Berücksichtigung von Ereignissen und Betriebszustand, Periodisch ausgeführte Systemfunktionalität	[58]
2009	RELAX	14	B	Modalität (2), Temporale Logik (6), Erfüllungsgrad (2), Unsicherheit (4)	[59]
2009 ff.	EARS	7	S	Grundsätzlicher Aufbau einer Anforderung, Ubiquitäre Anforderung, Ereignisgesteuerte Anforderung, Unerwünschtes Verhalten, Zustandsbasierte Anforderung, Optionale Anforderung, Komplexe (i.S.v. kombinierte) Anforderung	[60]–[63]
2010	SPES2020	5	S	Systemstruktur (Aufbau), Aufzählung von Funktionalität, Input einer Funktionalität, Zustandsveränderung von Systemen, Zeitliche Randbedingung. Erwähnt wird ein Gesamtumfang > 100 Muster.	[28]
2010	MBP	22	B	Textbausteine für Anforderungen an Mobilität	[64]
2011	Adv-EARS	6	S	Ubiquitäre Anforderung, Ereignisgesteuerte Anforderung, Unerwünschtes Verhalten, Zustandsbasierte Anforderung, Optionales Feature, Hybride Anforderung. Erweiterung von [60],[63].	[65]
2011	–	3	S	Funktionale Anforderung (3)	[37]
2012	–	31	B	Kern einer Anforderung (12), Bedingung (5), Randbedingung (14)	[66]
2012	Volere Requirements Specifications Template, Atomic Requirements Shell	2	S	Funktionale Anforderung, Nichtfunktionale Anforderung	[26], [25]
2013	Guided Natural Language (GNL)	U	U	Erwähnt werden 3 Arten von Boilerplates (Capability, Functional Requirement, Constraint) und 6 Arten von Mustern.	[36]
2014	–	4	S	Vertraulichkeit (4). Erwähnt wird, dass insgesamt 16 Schablonen existieren.	[35]
2014	SAREMAN	59	B	Systemstruktur (5), Zustand (8), Ereignis (18), Aktivität (10), Timing & Performanz (13), Mengen (2), Verschiedenes (3)	[67]
2014	–	7	S	Funktionale Anforderung mit/ohne Randbedingung (4), Benutzeranforderung (1), Eigenschaft von Prozessergebnissen (2)	[33]
2014	NL Requirements Boilerplates	39	S	Funktionale Anforderung (9), Performance-Anforderung (12), Qualitätsanforderung (8), Randbedingung (10)	[68]
2014	MASTeR	7	S&B	Funktionale Anforderung (3), Eigenschaft, Auszuführender Prozess, Randbedingung, Bedingung	[29]
2015	Parameterized Safety Requirements Templates	6	S	Anforderungen im Bereich Security	[69]
2015	–	4	S	Funktionale Anforderung, Randbedingung, Umweltfaktoren, Anforderung an Tauglichkeit	[70]
2015	RESA	6	B	Anforderung, Logische Aussage, Operator, Präposition, Baustein-Komposition, If-elseif-else	[71]
2015	LELIE	U	S	Beispiele werden genannt für Funktionale Anforderung (2) mit 7 Varianten (zum Beispiel zeitl. Restriktionen)	[72]
2015	Satzschablone	3	S	Funktionale Anforderung (3) (Identisch mit der MASTeR-Schablone in [29])	[9]
2015	–	1	S	Gesetzliche Anforderung	[27]
2017	–	10	S	Stakeholderanforderung, Funktionale Anforderung, Randbedingung bzgl. Zeit, Menge und Umgebungsvariablen (8)	[30]
2017	Context RDS	U	B	6 Textbausteine werden exemplarisch genannt	[32]
2019	–	1	S	Funktionale Anforderung (Erweiterung von [29])	[73]
2020	–	1	S	Funktionale Anforderung (Weiterentwicklung von [73])	[74]

Tabelle 5: Auf Schablonen und Textbausteinen basierende Ansätze im Requirements Engineering

Quellen

[1] G. Kim, J. Humble, P. Debois, and J. Willis, *Das DevOps-Handbuch: Teams, Tools und Infrastrukturen erfolgreich umgestalten*. Heidelberg: O'Reilly, 2017.

[2] D. W. Hoffmann, *Software-Qualität*, 2nd ed. Berlin, Heidelberg: Springer Vieweg, 2013.

[3] T. Franz, "Datengetriebene IT-Projekte im Wandel," *Heise Developer (online)*, 2015, [Online]. Available: https://m.heise.de/developer/artikel/Datengetriebene-IT-Projekte-im-Wandel-2734130.html.

[4] ISO/IEC/IEEE, "International Standard - Systems and software engineering--Vocabulary 24765-2017," *ISO/IEC/IEEE 24765: 2017 (E)*, 2017.

[5] M. Broy and M. Kuhrmann, *Einführung in die Softwaretechnik*. Springer Vieweg, 2021.

[6] GPM Deutsche Gesellschaft für Projektmanagement e.V., Ed., *Kompetenzbasiertes Projektmanagement (PM4): Handbuch für Praxis und Weiterbildung im Projektmanagement*. Nürnberg: GPM Deutsche Gesellschaft für Projektmanagement, 2019.

[7] International Institute of Business Analysis, Ed., *A Guide to the Business Analysis Body of Knowledge (BABOK Guide) Version 2.0*, 2nd ed. Toronto, Canada, 2009.

[8] T. Kuhn, "A Survey and Classification of Controlled Natural Languages," *Computational Linguistics*, vol. 40, no. 1, pp. 121–170, Mar. 2014, doi: 10.1162/COLI_a_00168.

[9] K. Pohl and C. Rupp, *Basiswissen Requirements Engineering: Aus- und Weiterbildung nach IREB-Standard zum Certified Professional for Requirements Engineering Foundation Level*, 4., überarb. Aufl. Heidelberg: dpunkt Verlag, 2015.

[10] P. Bourque and R. E. Fairley, Eds., *Guide to the Software Engineering Body of Knowledge, Version 3.0*. Washington DC: IEEE Computer Society, 2014.

[11] D. M. Fernández *et al.*, "Naming the pain in requirements engineering: Contemporary problems, causes, and effects in practice," *Empirical Software Engineering*, vol. 22, no. 5, pp. 2298–2338, Oct. 2017, doi: 10.1007/s10664-016-9451-7.

[12] B. Davey and K. R. Parker, "Requirements elicitation problems: a literature analysis," *Issues in Informing Science and Information Technology*, vol. 12, pp. 71–82, 2015.

[13] A. van Lamsweerde, *Requirements Engineering: From System Goals to UML Models to Software Specifications*. Chichester, West Sussex, England; New York: Wiley, 2009.

[14] A. M. Davis, *Software Requirements: Objects, Functions and States (Revised Edition)*, 2nd ed. Upper Saddle River, N.J: Prentice Hall PTR, 1993.

[15] H. Balzert, *Lehrbuch der Softwaretechnik: Softwaremanagement*, 2nd ed. Heidelberg: Spektrum Akademischer Verlag, 2008.

[16] H. A. Partsch, *Requirements-Engineering systematisch: Modellbildung für softwaregestützte Systeme*, 2nd ed. Berlin, Heidelberg: Springer, 2010.

[17] D. Bjørner and K. Havelund, "40 Years of Formal Methods - Some Obstacles and Some Possibilities?," in *FM 2014: Formal Methods - 19th International Symposium, Singapore, May 12-16, 2014. Proceedings*, 2014, vol. 8442, pp. 42–61, doi: 10.1007/978-3-319-06410-9_4.

[18] E. Neufert, M. Lohmann, P. Merkel, M. Brockhaus, and J. Kister, *Bauentwurfslehre: Grundlagen, Normen, Vorschriften*, 42nd ed. Wiesbaden: Springer Vieweg, 2018.

[19] W. Skolaut, *Maschinenbau: Ein Lehrbuch für das ganze Bachelor-Studium*, 2., akt. u. erw. Aufl. Berlin: Springer Vieweg, 2018.

[20] W. Plaßmann and D. Schulz, *Handbuch Elektrotechnik: Grundlagen und Anwendungen für Elektrotechniker*, 7., neu Bearb. Aufl. Wiesbaden: Springer Vieweg, 2016.

[21] Object Management Group (OMG), Ed., *Unified Modeling Language™ (UML®)*, Ver. 2.5. o. O.: Object Management Group (OMG), 2015.

[22] B. Rücker and J. Freund, *Praxishandbuch BPMN: Mit Einführung in DMN*, 6., akt. Aufl. München: Hanser Verlag, 2019.

[23] T. Gilb, *Competitive Engineering: A Handbook for Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage*. Oxford, UK: Butterworth-Heinemann, 2005.

[24] M. Cohn, *User Stories Applied: For Agile Software Development*. Boston, MA: Addison-Wesley, 2004.

[25] J. Robertson and S. Robertson, "Volere requirements specification template," *Atlantic System Guild www.systemguild. com*, 2012.

[26] S. Robertson and J. Robertson, *Mastering the Requirements Process: Getting Requirements*

Right, 3rd Ed. Upper Saddle River, N.J: Pearson Education, 2012.
[27] C. Johner, M. Hölzer-Klüpfel, and S. Wittorf, *Basiswissen Medizinische Software: Aus- und Weiterbildung zum Certified Professional for Medical Software*, 2., überarb. u. aktual. Aufl. Heidelberg: dpunkt Verlag, 2015.
[28] J. Holtmann, "Mit Satzmustern von textuellen Anforderungen zu Modellen," *OBJEKTspektrum*, no. 6, pp. 1–5, 2010.
[29] C. Rupp and SOPHISTen, *Requirements-Engineering und -Management: Aus der Praxis von klassisch bis agil*, 6., aktual. u. erw. Aufl. München: Hanser Verlag, 2014.
[30] J. Dick, E. Hull, and K. Jackson, *Requirements Engineering*, 4th ed. New York, NY: Springer, 2017.
[31] S. Withall, *Software Requirement Patterns*. Sebastopol, CA: Microsoft Press, 2007.
[32] J. Dick and J. Llorens, "Using statement-level templates to improve the quality of requirements. An Integrate white paper." Integrate Systems Engineering Ltd., 2017, [Online]. Available: http://integrate.co/downloads/White%20Paper%20-%20templating_2017.pdf.
[33] C. Ebert, *Systematisches Requirements Engineering: Anforderungen ermitteln, dokumentieren, analysieren und verwalten*, 5., überarb. Aufl. Heidelberg: dpunkt Verlag, 2014.
[34] T. Gilb, "A planning language (a PLanguage)," *ACM SIGAPL APL Quote Quad*, vol. 19, no. 4, pp. 169–177, 1989.
[35] M. Riaz, J. Slankas, J. King, and L. Williams, "Using templates to elicit implied security requirements from functional requirements - a controlled experiment," in *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement - ESEM '14*, Torino, Italy, 2014, pp. 1–10, doi: 10.1145/2652524.2652532.
[36] M. Ortel *et al.*, "Requirements Engineering," in *CESAR - Cost-efficient Methods and Processes for Safety-relevant Embedded Systems*, A. Rajan and T. Wahl, Eds. Vienna: Springer Vienna, 2013, pp. 69–143.
[37] ISO/IEC/IEEE, "International Standard 29148 - Systems and software engineering – Life cycle processes – Requirements engineering," *ISO/IEC/IEEE 29148:2018(E)*, pp. 1–104, Nov. 2018, doi: 10.1109/IEEESTD.2018.8559686.
[38] IREB International Requirements Engineering Board e.V., "IREB Certified Professional for Requirements Engineering. Foundation Level. Lehrplan Version 2.2.1 v. 24.7.2017," 2017. https://www.ireb.org/en/downloads/#syllabus-foundation-level (accessed Jan. 02, 2018).
[39] O. Linssen, "PARIS - Die Entwicklung einer Mustersprache zur Dokumentation von Anforderungen," *Rundbrief GI-Fachausschuß Management der Anwendungsentwicklung und -wartung*, vol. 26, no. 44, pp. 7–24, 2020.
[40] Verein zur Weiterentwicklung des V-Modell XT e.V. (Weit e.V.), *V-Modell XT. Das deutsche Referenzmodell für Systementwicklungsprojekte. Version: 2.3*. München, 2019.
[41] N. Wirth, "What can we do about the unnecessary diversity of notation for syntactic definitions?," *Communications of the ACM*, vol. 20, no. 11, pp. 822–823, 1977.
[42] M. Cohn, *Succeeding with Agile: Software Development Using Scrum*. Upper Saddle River, N.J, u.a.: Addison-Wesley Longman, 2009.
[43] P. Di Francesco, P. Lago, and I. Malavolta, "Migrating towards microservice architectures: an industrial survey," in *2018 IEEE international conference on software architecture (ICSA)*, 2018, pp. 29–2909.
[44] J. Lewis and M. Fowler, "Microservices - a definition of this new architectural term," *martinfowler.com*, Mar. 25, 2014. https://martinfowler.com/articles/microservices.html.
[45] M. Glinz, *A Glossary of Requirements Engineering Terminology. Version 2.0.0*. IREB, 2020.
[46] G. Starke, *Effektive Softwarearchitekturen: Ein praktischer Leitfaden*, 6., überarb. Aufl. München: Carl Hanser Verlag GmbH & Co. KG, 2014.
[47] W. Geiger and W. Kotte, *Handbuch Qualität: Grundlagen und Elemente des Qualitätsmanagements: Systeme - Perspektiven*, 5th ed. Wiesbaden: Vieweg+Teubner, 2007.
[48] J. A. McCall, P. K. Richards, and G. F. Walters, "Factors in software quality. volume i. concepts and definitions of software quality," General Electric Company, RADC-TR-77-369, Vol 1 of 3, 1977.
[49] ISO/IEC, *ISO/IEC 25010:2011-03, Software-Engineering - Qualitätskriterien und Bewertung von Softwareprodukten (SQuaRE) - Qualitätsmodell und Leitlinien*. 2011.
[50] M. Glinz, H. van Loehoud, S. Staal, and S. Bühne, "Handbuch für das CPRE Foundation Level nach dem IREB-Standard," International Requirements Engineering Board, Ver. 1.0.0, Nov.

2020.
[51] C. Rolland and C. Proix, “A natural language approach for requirements engineering,” in *International Conference on Advanced Information Systems Engineering*, 1992, pp. 257–277.
[52] I. Hooks, “Writing good Requirements,” *INCOSE International Symposium*, vol. 4, no. 1, pp. 1247–1253, 1994, doi: 10.1002/j.2334-5837.1994.tb01834.x.
[53] L. LaPlue, R. A. Garcia, and R. Rhodes, “A rigorous method for formal requirements definition,” *INCOSE International Symposium*, vol. 5, no. 1, pp. 429–434, Jul. 1995, doi: 10.1002/j.2334-5837.1995.tb01893.x.
[54] A. Durán Toro, B. Bernárdez Jiménez, A. Ruiz Cortés, and M. Toro Bonilla, “A requirements elicitation approach based in templates and patterns,” *Universidad de Sevilla. Departamento de Lenguajes y Sistemas Informáticos*, 1999.
[55] C. Denger, D. M. Berry, and E. Kamsties, “Higher quality requirements specifications through natural language patterns,” in *Proceedings 2003 Symposium on Security and Privacy*, Nov. 2003, pp. 80–90, doi: 10.1109/SWSTE.2003.1245428.
[56] S. F. Tjong, N. Hallam, and M. Hartley, “Improving the quality of natural language requirements specifications through natural language requirements patterns,” in *The Sixth IEEE International Conference on Computer and Information Technology (CIT’06)*, 2006, pp. 199–199.
[57] A. T. Bahill and F. F. Dean, “Discovering system requirements,” in *Handbook of Systems Engineering and Management. Andrew P. Sage, William B. Rouse (Eds.), Chapter 4*, 2nd ed., John Wiley & Sons, 2009, pp. 205–266.
[58] C. Busby-Earle and E. Mugisa, “Towards Writing Secure Software Requirements,” presented at the IASTED International Conference Software Engineering, Innsbruck, Austria, 2009.
[59] J. Whittle, P. Sawyer, N. Bencomo, B. H. Cheng, and J.-M. Bruel, “Relax: Incorporating uncertainty into the specification of self-adaptive systems,” in *2009 17th IEEE International Requirements Engineering Conference*, 2009, pp. 79–88.
[60] A. Mavin, P. Wilkinson, A. Harwood, and M. Novak, “Easy approach to requirements syntax (EARS),” in *2009 17th IEEE International Requirements Engineering Conference*, 2009, pp. 317–322, doi: https://doi.org/10.1109/RE.2009.9.
[61] A. Mavin, P. Wilksinson, S. Gregory, and E. Uusitalo, “Listens Learned (8 Lessons Learned Applying EARS),” in *2016 IEEE 24th International Requirements Engineering Conference (RE)*, Sep. 2016, pp. 276–282, doi: 10.1109/RE.2016.38.
[62] A. Mavin and P. Wilkinson, “Ten Years of EARS,” *IEEE Software*, vol. 36, no. 5, pp. 10–14, Sep. 2019, doi: 10.1109/MS.2019.2921164.
[63] A. Mavin and P. Wilkinson, “Big Ears (The Return of ‘Easy Approach to Requirements Engineering’),” in *2010 18th IEEE International Requirements Engineering Conference*, Sep. 2010, pp. 277–282, doi: 10.1109/RE.2010.39.
[64] S. Gopalakrishnan and G. Sindre, “A study on Mobile Requirements Elicitation by Boilerplate Requirements Specification Language,” *International Conference on Electronic Business ICEB 2010 Proceedings*, pp. 613–623, 2010.
[65] D. Majumdar, S. Sengupta, A. Kanjilal, and S. Bhattacharya, “Adv-EARS: A Formal Requirements Syntax for Derivation of Use Case Models,” in *Advances in Computing and Information Technology*, vol. 198, D. C. Wyld, M. Wozniak, N. Chaki, N. Meghanathan, and D. Nagamalai, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 40–48.
[66] V. Johannessen, “CESAR-text vs. boilerplates: What is more effcient - Requirements written as free text or using boilerplates (templates)?,” Master’s Thesis, Norwegian University of Science and Technology, Trondheim, Trondheim, 2012.
[67] T. Tommila and A. Pakonen, *Controlled natural language requirements in the design and analysis of safety critical I&C systems*. Nuclear Waste Management Fund, Finland RESEARCH REPORT VTT-R-01067-14., 2014.
[68] N. Ibrahim, W. M. N. W. Kadir, and S. Deris, “Documenting requirements specifications using natural language requirements boilerplates,” in *2014 8th. Malaysian Software Engineering Conference (MySEC)*, Sep. 2014, pp. 19–24, doi: 10.1109/MySec.2014.6985983.
[69] P. O. Antonino, M. Trapp, P. Barbosa, and L. Sousa, “The Parameterized Safety Requirements Templates,” in *2015 IEEE/ACM 8th International Symposium on Software and Systems Traceability*, Florence, Italy, May 2015, pp. 29–35, doi: 10.1109/SST.2015.12.
[70] R. S. Carson, “Implementing Structured Requirements to Improve Requirements Quality,” *INCOSE International Symposium*, vol. 25, no. 1, pp. 54–67, Oct. 2015, doi: 10.1002/j.2334-

5837.2015.00048.x.
[71] N. Mahmud, C. Seceleanu, and O. Ljungkrantz, “ReSA: An ontology-based requirement specification language tailored to automotive systems,” in *10th IEEE International Symposium on Industrial Embedded Systems (SIES)*, Siegen, Germany, Jun. 2015, pp. 1–10, doi: 10.1109/SIES.2015.7185035.
[72] J. Kang and P. Saint-Dizier, “An Approach to Improve the Language Quality of Requirements,” in *Language Production, Cognition, and the Lexicon*, N. Gala, R. Rapp, and G. Bel-Enguix, Eds. Cham: Springer International Publishing, 2015, pp. 399–418.
[73] R. Mazo and C. Jaramillo, “Hacia una nueva plantilla para la especificación de requisitos en lenguaje natural semi-estructurado,” presented at the XXII Conferencia Iberoamericana de Software Engineering, Hotel Memories Miramar, La Habana, Cuba, Apr. 2019.
[74] P. Vallejo, R. Mazo, C. Jaramillo, and J. M. Medina, “Towards a new template for the specification of requirements in semi-structured natural language,” *Journal of Software Engineering Research and Development*, vol. 8, p. 3 ff., Feb. 2020, doi: 10.5753/jserd.2020.473.

ProPhi: Eine neue Methode zur Auswahl einer passenden Projektmanagementphilosophie

Godwin Scholz[1] und Thomas Schuster[2]

[1] Landesbank Baden-Württemberg, Am Hauptbahnhof 2, 70173 Stuttgart,

Godwin.Scholz@LBBW.de

[2] Hochschule Pforzheim, Tiefenbronner Straße 65, 75175 Pforzheim,

thomas.schuster@hs-pforzheim.de

Abstract: Neue Projekte und Technologien erfordern stetig Anpassungen des Projektmanagements. Damit einhergehend haben sowohl die Anzahl und die Art der Managementmethoden Veränderungen erfahren. Noch vor einigen Jahren waren hauptsächlich klassische Vorgehensmodelle im Einsatz, heute stellen Unternehmen mehr und mehr auf agile Methoden um oder haben diesen Wandel bereits vollzogen. Seit einiger Zeit wird dies durch hybride Vorgehensweisen ergänzt, die mittlerweile die Möglichkeit bieten klassische und agile Philosophien zu vereinen. Dieser Artikel untersucht die Frage, unter welchen Bedingungen ein klassisches, ein agiles oder ein hybrides Vorgehen sinnvoll ist. Auf Basis verschiedener anerkannter Projektparameter führen wir eine Studie durch, um deren praktischen Einfluss zu erheben. Hieraus leiten wir eine innovative Auswahlmethode ab, mit der die Projektmanagementphilospohie bestimmt werden kann. Unsere Methode zielt darauf ab in der praktischen Anwendung sowohl einfach einsetz- als auch individualisierbar zu sein.

Keywords: Projektmanagement, Hybrides Projektmanagement, Agiles Projektmanagement, Klassisches Projektmanagement, Auswahlmethode, Projektmanagementphilosophie.

1 Einleitung

Seit einigen Jahren unterliegen die Prinzipien des Projektmanagements einem zunehmenden Wandel. Agile Vorgehensweisen werden in diesem Zuge immer häufiger eingesetzt. Neben agilen und klassischen Ansätzen etablieren sich seit einiger Zeit auch verschiedene hybride Projektphilosophien. Darüber hinaus gibt es philosophieübergreifend eine Vielzahl an Methoden und Vorgehensmodellen. Bekannte Beispiele im agilen Bereich sind Scrum, XP oder Kanban. Auch existieren eine Vielzahl klassischer Vorgehensweisen, beispielsweise PMI. Die Vielzahl deckt einerseits unterschiedliche Projektanforderungen ab, stellt die Verantwortlichen in Unternehmen im ersten Schritt allerdings auch vor die Herausforderung, welche Projektphilosophie für ein konkretes Projekt passend ist. Im zweiten Schritt muss dann eine weitere Auswahlentscheidung getroffen werden, um aus den ebenso zahlreichen Vorgehensmodellen und Methoden einer gewählten Philosophie passend auszuwählen. Dabei wird allgemein hin anerkannt, dass es weder auf Projekt- noch auf Unternehmensebene eine Einheitslösung für die gesamte Projektlandschaft gibt [Ku19a]. Zur Einrichtung eines Projekts gehört daher immer auch die Auswahl einer passenden Projektmanagementphilosophie sowie den dazugehörigen geeigneten Methoden. Hybride Ansätze stellen den neuesten Trend durch die Vermischung beider Philosophien mit dem Ziel das Beste aus beiden Welten zu vereinen [KR14]. Die Kombination agiler sowie klassischer Methoden erweist sich häufig als vorteilhaft [Sc19]. Viele Unternehmen und deren Projektverantwortliche sehen klassische und agile Philosophie als zweit getrennte Welten, die unvereinbar nebeneinander stehen [Ha13]. Im Jahr 2017 ergab eine Studie, dass nur 37 % der befragten Unternehmen hybride Projektvorgehensweisen in Betracht ziehen [KK17].

Es ist weiterhin davon auszugehen, dass die Wahl des Projektvorgehens vielfach willkürlich, stark vom Management beeinflusst oder nicht auf Fakten (Projektkontext) basiert getroffen

wird. Ferner ist eine bisweilen eine willkürliche Mischung aus agilen und traditionellen Komponenten zu beobachten. Diese Beobachtung ergibt sich aus einer Studie, in der untersucht wurde, ob Unternehmen bei der Entscheidung über die Wahl des passenden Vorgehensmodells systematisch bei der Entscheidungsfindung vorgehen. Dabei ergab sich auch, dass nur bei 41% der Befragten bei der Entscheidungsfindung methodisch vorgehen [Ko16]. Dieser Zustand ist unserer Ansicht nach fragwürdig und gleichzeitig die Motivation, den Unternehmen bessere Entscheidungsgrundlagen und -verfahren an die Hand zu geben. Ziel dieses Artikels ist es daher eine Methode zur Unterstützung der Entscheidungsfindung für eine fundierte Wahl zwischen hybriden, traditionellen und agilen Projektmanagementphilosophien bereitzustellen. Die zentrale Forschungsfrage lautet daher: „Wie kann eine optimale Projektphilosophie für ein spezifisches Projekt identifiziert werden?“

Der Artikel ist diesbezüglich wie folgt gegliedert: In Abschnitt 2 wird der aktuelle Stand der verwandten Arbeiten dargestellt. Dabei werden auch bereits bekannte Parameter betrachtet, welche bei der Auswahl einer Projektmanagementphilosophie unterstützen. In Abschnitt 3 wird eine Methode zur Unterstützung bei der Auswahl der passenden Projektmanagementphilosophie vorgestellt. In Abschnitt 4 wird eine Evaluierung vorgenommen, bei der die Methodik untersucht und diskutiert wird. Abschnitt 5 schließt mit einer Schlussfolgerung und Erkenntnissen für die weiteren Forschungsschritte.

2 Stand der Wissenschaft

Für das Verständnis des Begriffs hybrides Projektmanagement ist es wichtig, diesen gleich zu Beginn zu definieren. In der Literatur gibt es hierfür unterschiedliche Definitionen. Angermeier definiert hybrides Projektmanagement als Verknüpfung von mindestens zwei Managementsystemen für das Management eines Projekts [An16]. Allerdings wird in der Literatur unter dem Begriff häufig die Synthese von klassischen und agilen Methoden verstanden. Angermeier lässt dagegen auch rein klassische oder agile Mischungen als hybrid gelten. Sandhaus, Berg und Knott setzen in ihrem Verständnis den Schwerpunkt auf die sequenzielle Kombination des klassischen Phasenmodells mit agilen Methoden [SBK14]. Kurtz und Sauer sehen hybrides Projektmanagement dagegen als die Kombination agiler sowie klassischer Techniken [KS18]. Komus und Kuberg definieren einen hybriden Projektmanagement-Ansatz als Vermischung oder Kombination klassischer und agiler Methoden [KK17]. Timinger und Seel interpretieren hybrides Projektmanagement uneinheitlich. So fassen beide Autoren hybrides Projektmanagement als Bezeichnung für Vorgehensmodelle auf, welche klassische sowie agile Ansätze beinhalten und kombinieren [TS16]. Timinger spricht allerdings auch dann von hybridem Projektmanagement, wenn die verwendeten Kombinationen rein agil oder klassisch sind [Ti17]. Hüsselmann versucht die teilweise unterschiedlichen Akzentuierungen der Definitionen zu vereinen, indem er hybrides Projektmanagement als die Verwendung von unterschiedlichen Standards oder Vorgehensmodellen zusammenfasst [Hü18]. Diese Arbeit verwendet eine verschärfte Definition Hüsselmanns mit der Einschränkung, dass lediglich die Kombination klassischer sowie agiler Vorgehensweisen als hybrid gilt.

Konkrete Vorgehensmodelle und Projektmanagementmethoden der klassischen, agilen oder hybriden Philosophien sind nicht Bestandteil der weiteren Betrachtungen, da der Schwerpunkt dieses Artikels auf der Philosophieauswahl liegen soll. Ferner gibt es in der Softwareentwicklung verschiedene Ansätze, um Vorgehensweisen auf die jeweilige Entwicklungssituation anzupassen. Bekannte Methoden fokussieren die Anpassung konkreter Vorgehensmodelle (Tailoring) einige schränken dies auch auf die ausschließliche Anpassung von Methoden bestimmter Projektmanagementphilosophien ein [Ali00, Die16]. Zum einen steht hierbei weniger die Philosophie im Fokus, zum anderen fehlen auch in diesem Bereich derzeit noch vielfach einfach einzusetzende Werkzeuge, um Projektmanager bei der Auswahl zu unterstützen.

Auf Basis klarer Auswahlkriterien kann eine zielorientierte Entscheidung für Projekte getroffen werden. Ein Augenmerk liegt auf Parametern, die als Einflussfaktoren für das Projektmanagement gelten. Mehrere Autoren haben verschiedene Parameter definiert und erweitert. Basierend auf den von Boehm und Turner beschriebenen grundlegenden Parametern sowie verschiedenen Erweiterungen, wird im Folgenden ein entsprechender Parametersatz vorgestellt.

Komplexität des Projektgegenstandes (KO) [Šp14]: Komplexe Projekte sind von Unvorhersehbarkeit geprägt. Das Ergebnis oder die genauen Anforderungen sind offen, ebenso wie der Weg dorthin. Im Gegensatz dazu können komplizierte Projekte mit bekannten Methoden geleitet werden, und die Anforderungen sind klar definiert. Klassische Projektmanagementmethoden erlauben eine strukturierte Bearbeitung von Projektobjekten mit geringer Komplexität [Pa18]. Chaotische oder komplexe Projekte sind dagegen prädestiniert für agile Projektmanagementmethoden [Ku19b].

Projektteamgröße (GR): Agile Methoden eignen sich für kleinere Teams. Beispielsweise erfüllen agile Methoden ihren Zweck bei Teams von bis zu neun Personen, während sich hybride und klassische Methoden eher für größere Teams empfehlen [Ku19b].

Gefährdungspotenzial (GP): Für Projekte, deren Ergebnisse ein hohes Gefährdungspotenzial aufweisen, haben sich traditionelle Projektmanagementmethoden gut bewährt [BT09]. Agile Projektmanagementvorgehen wie Scrum sind für kritische Projekte nicht geeignet. Mögliche Sicherheitsrisiken können aufgrund einer weniger detaillierten Planung übersehen oder zu spät bemerkt werden [Pa18]. Eine klassische Vorgehensweise ist in einigen Fällen unerlässlich, z. B. beim Bau eines Kernkraftwerks. Der schrittweise Bau (Sicherheitseinrichtungen in späteren Inkrementen) oder ein nicht vorhandenes Risikomanagement ist in diesem Extrembeispiel leichtsinnig und gefährlich.

Stabilität der Anforderungen (SA): Stabile Projektanforderungen werden mit klassischen Methoden gut bewältigt. Volatile Anforderungen hingegen sind prädestiniert für iterative agile Methoden, da Änderungen regelmäßig eingearbeitet und verarbeitet werden können [Pa18]. Eine agile Vorgehensweise ist daher bei vorherrschender Dynamik, Unsicherheit und Instabilität der Anforderungen geeignet [BT09].

Qualifikationsgrad der Teammitglieder (QT): Ein agiles Projektvorgehen eignet sich insbesondere bei eher gut qualifizierten, intrinsisch motivierten Mitarbeitern (gemäß der X-Y-Theorie nach McGregor) [Br17, Mc73].

Kultur (KU): In hierarchisch organisierten Unternehmenskulturen sind es die klar definierten Rollen und Regeln der klassischen Methoden, die besonders geschätzt werden. In weniger streng organisierten Unternehmenskulturen lassen sich agile Methoden damit leichter integrieren [BT09, Pa18].

Teamverteilung (TV): Hier stellt sich die Frage, ob das Projektteam an einem Standort arbeitet [Br17, Šp14]. Es ist schwierig, im Falle einer physischen Trennung agile Methoden anzuwenden, daher sind klassische Methoden in diesen Fällen besser geeignet [Ku19b].

Ausrichtung der Organisation bei Projekten (AO) [Šp14]: Um eine agile oder klassische Implementierung zu ermöglichen, muss die bestehende Organisationsstruktur dies unterstützen. Wenn die Organisation rein klassisch oder agil ist, fehlt das Wissen und oft auch die Akzeptanz für andere Philosophien [Pa18].

Produktlebenszyklus (PL): Produkte mit kürzeren Produktentwicklungszeiten oder geringerem Wert eignen sich für die Anwendung agiler Methoden [Br17]. Eine hohe Modularität und Kapselung der Komponenten spricht für eine Entwicklung in Inkrementen und damit den Einsatz agiler Methoden [Br17].

Boehm und Turner entwickelten fünf grundsätzliche Parameter (KU, GP, GR, SA, QT) [BT09]. Diese wurden teilweise erweitert. Die Parameter können in die Kategorien projektspezifische

Parameter und Umgebungsparameter zugeordnet eingeordnet werden. Zum einen sind die Kultur, der Qualifikationsgrad der Teammitglieder, die Ausrichtung der Organisation bei Projekten und die Teamverteilung sogenannte Umgebungsparameter. Auf der anderen Seite werden unter dem Begriff projektspezifische Parameter die Stabilität der Anforderungen, das Gefährdungspotenzial, die Projektteamgröße, die Komplexität und der Produktlebenszyklus eingeordnet [Pa18]. Diese Kategorisierung wird in den folgenden Abschnitten von Bedeutung sein. Bislang wurden die Parameter nicht methodisch in Verbindung zueinander betrachtet und ausgewertet. Sie standen lediglich für sich als einzelne Indikatoren und gaben eine Tendenz, wie ein Projekt durchgeführt werden sollte. Allerdings können einzelne Parameter unterschiedliche Ausrichtungen haben. Wie ein Projekt durchgeführt werden sollte wenn die Hälfte der Parameter agil ausgeprägt sind, die anderen Parameter dagegen in eine klassische Richtung deuten bleibt dabei unbeantwortet. Ebenso offen bleibt die Relevanz der einzelnen Parameter.

3 Eigener Ansatz

Bislang stellen bekannte Ansätze entweder nur die Parameter zur Auswahl einer Projektmanagementphilosophie dar, ohne eine konkrete Auswahlmethode zu bieten oder setzen eine Ebene tiefer in der Anpassung von einzelnen Vorgehensmodellen an. Die alleinige Definition der relevanten Parameter reicht jedoch nicht, um eine Projektmanagementphilosophie eindeutig festzulegen. Der nachfolgend vorgestellte integrierte Ansatz soll genau diese Lücke schließen und durch die Verknüpfung und methodische Analyse der Parameter eine klare Auswahlentscheidung ermöglichen.

In konkreten Projektsituationen sind die Parameter bei der verknüpften Betrachtung außerdem unterschiedlich zu gewichten. Unsere Methode ermöglicht daher eine Analyse, die die Parameterkategorien und individuelle Parametergewichtungen integriert. Für die Gewichtungen definieren und evaluieren (siehe unten) wir Standardwerte. Die Methode besteht aus fünf Schritten, welche in Abbildung 1 im Überblick dargestellt sind und nachfolgend detailliert beschrieben werden. Im Ergebnis können Unternehmen methodisch und einfach die optimale Projektphilosophie zu identifizieren.

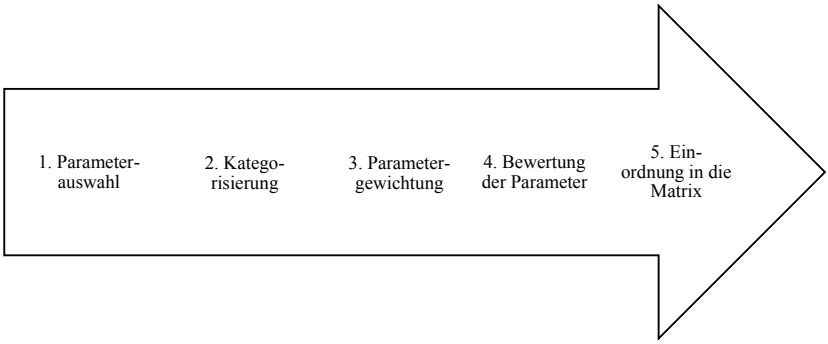

Abbildung 1: Prozess der vorgestellten Methode

Im Folgenden werden die einzelnen Schritte jeweils einzeln erläutert. Die Schritte 1-3 sind optional, da die Methode bereits vordefinierte, kategorisierte und gewichtete Parameter bereitstellt. Diese Vorkonfiguration basiert auf den von uns erkannten durchschnittlichen Projektanforderungen. Die Modifikation durch Anwender (Parameteranpassungen oder eigene Gewich-

tungen) sind ein wichtiger Faktor, um einen konkreten Projektkontext korrekt erfassen und bewerten zu können.

Schritt 1 (Parameterauswahl): Im ersten Schritt müssen Parameter definiert werden, welche eine Aussage darüber treffen, ob ein Projekt in agiler oder klassischer Philosophie durchgeführt werden sollte. Hierbei kommen die Parameter aus Abschnitt 2 zum Einsatz. Einige Parameter sind qualitativ und schwierig quantifizierbar, wie die Unternehmenskultur. Andere Parameter sind leicht quantifizierbar, wie die Größe eines Teams.

Schritt 2 (Kategorisierung): Die Parameter werden in projektspezifische Parameter und Umgebungsparameter unterschieden. Diese Einteilung stützt sich ebenfalls auf den in Abschnitt 2 vorgestellten Stand der Wissenschaft. Die Kategorien sollten von den Anwendern der Methode auch nicht manipuliert werden, da diese für Schritt 5 essenziell sind.

Schritt 3 (Parametergewichtung): In diesem Schritt werden die Parameter innerhalb ihrer Kategorie gewichtet. Für eine möglichst allgemeingültige Vorabkonfiguration der vorgestellten Parameter wurde eine Studie durchgeführt. Dabei wurde die Hypothese untersucht, dass nicht alle Parameter in gleicher Weise relevant sind. Die Durchführung der Studie wird in Abschnitt 4 ausführlich dargestellt. In Tabelle 1 sind die in der Studie ermittelten Gewichte für die einzelnen Parameter dargestellt. Es ist ersichtlich, dass diese durchaus unterschiedlich stark gewichtet sind. Im Falle der Verwendung anderer Parameter müssen alle Gewichtungen neu vorgenommen werden. Es wird daher empfohlen Anpassungen der Parameterkombinationen nur mit Augenmaß anzuwenden.

	Erläuterung	**Note**	**Gewichtung**
Umgebungsparameter	*Gewichteter Mittelwert (x-Koordinate)*		100 %
Kultur (KU)	1 bei alter (klassischer) Unternehmenskultur 6 bei junger (agiler) Unternehmenskultur		19 %
Qualifikationsgrad der Teammitglieder (QT)	1 Mitarbeiter mit starker Tendenz zur Theorie-X 6 Mitarbeiter mit starker Tendenz zur Theorie-Y		34 %
Ausrichtung der Organisation bei Projekten (AO)	1 Rein traditionelle Organisation 6 Rein agile Organisation		26 %
Teamverteilung (TV)	1 Dezentrale Teamverteilung 6 Zentralisierte Teams		21 %
Projektspezifische Parameter	*Gewichteter Mittelwert (y-Koordinate)*		100 %
Stabilität der Anforderungen (SA)	1 Stabile Projektanforderungen 6 Volatile Projektanforderungen		21 %
Gefährdungspotenzial (GP)	1 Hohes Gefährdungspotenzial 6 Unkritische Vorhaben		19 %

	Erläuterung	Note	Gewichtung
Projektteamgröße (GR)	1 Großes Vorhaben / große Teamgröße 6 Kleines Vorhaben / kleine Teamgröße		15 %
Komplexität des Projektgegenstandes (KO)	1 Projekt mit geringer Komplexität 6 Projekt mit hoher Komplexität		31 %
Produktlebenszyklus (PL)	1 Längere Produktentwicklungszeiten oder hoher Wert 6 Kürzere Produktentwicklungszeiten oder geringerer Wert		14 %

Tabelle 1: Kalkulationsschema zur Berechnung der Matrixkoordinaten

Schritt 4 (Bewertung der Parameter): In diesem Schritt müssen die Anwender der Methode die dargestellten Parameter bewerten. Wir empfehlen hierfür eine Bewertungsskala von 1-6 zu verwenden. Die Ausprägung 1 steht für eine völlig klassische Ausprägung, während eine Parameterausprägung von 6 für eine völlig agile Ausprägung steht. Teilweise kann es den Anwendern schwerfallen, die teilweise abstrakten Parameter zu bewerten. Es ist daher elementar sich mit den Parametern vor der Anwendung intensiv auseinanderzusetzen und das eigene Unternehmen und Projekt zu reflektieren. Als Hilfsmittel und um ein Gefühl für die Parameter zu entwickeln haben sich verschiedene Methoden bewährt. So ist bspw. bei der Bewertung des Parameters Komplexität das Cynefin-Framework oder die Stacey-Matrix hilfreich [Ku19b] [Sn00]. Nach Bewertung der Umweltparameter und projektspezifischen Parameter, kann für beide Kategorien der gewichtete Mittelwert berechnet werden.

Schritt 5 (Einordnung in die Matrix): Die beiden Mittelwerte aus Tabelle 1 werden als Koordinaten in der nachfolgenden vorgestellten Matrix (Abbildung 2) verwendet. Der mittlere gewichtete Umgebungsparameter dient dabei als x-Koordinate und der Mittelwert der projektspezifischen Parameter als y-Koordinate. Hinsichtlich der Koordinatenzuordnung erfolgt eine Orientierung an der Matrix der Boston Consulting Group (BCG). Bei der BCG-Matrix wurde als x-Achse der relative Marktanteil und als y-Achse das Marktwachstum festgelegt [He70]. Der Marktanteil ist als interne Variable zu sehen, das Marktwachstum als nicht beeinflussbare externe Variable. Übertragen auf die nun dargestellte Matrix bedeutet dies, dass die Projektvariablen (beeinflussen die y-Achse) von extern bestimmt und an sich nicht zu verändern sind. So kann ein bestimmtes Projekt prinzipiell für eine agile Vorgehensweise tauglich sein (häufiges Beispiel: Softwareentwicklung). Dagegen sind die gewichteten Umgebungsparameter auf der x-Achse vom Unternehmen abhängig und daher auch theoretisch veränderbar. Die 25-Felder-Auswahlmatrix teilt beide Koordinatenachsen in jeweils 5 Bereiche mit insgesamt 5 möglichen Ausprägungen. Die Ausprägungen basieren auf die Unterteilung Habermanns [Ha13]. Es wird dabei zwischen klassischen, agilen und hybriden Ausprägungen unterschieden. Dabei ist ein wichtiges Unterscheidungskriterium, ob ein einzelnes eingehendes System als grundsätzlich überlegen angesehen wird. Habermann vergleicht den Sachverhalt mit einem Hybridauto. Hier entscheidet zwar ein technischer Parameter, welche bestimmte Antriebsart (Benzin, Elektro, Beides) zum Einsatz kommt. Grundsätzlich gilt in diesem Zusammenhang aber die Direktive den Verbrennungsmotor möglichst selten zu nutzen [Ha13]. Eine hybride Projektvorgehensweise kann ebenfalls eine Tendenz zum Klassischen oder Agilen besitzen.

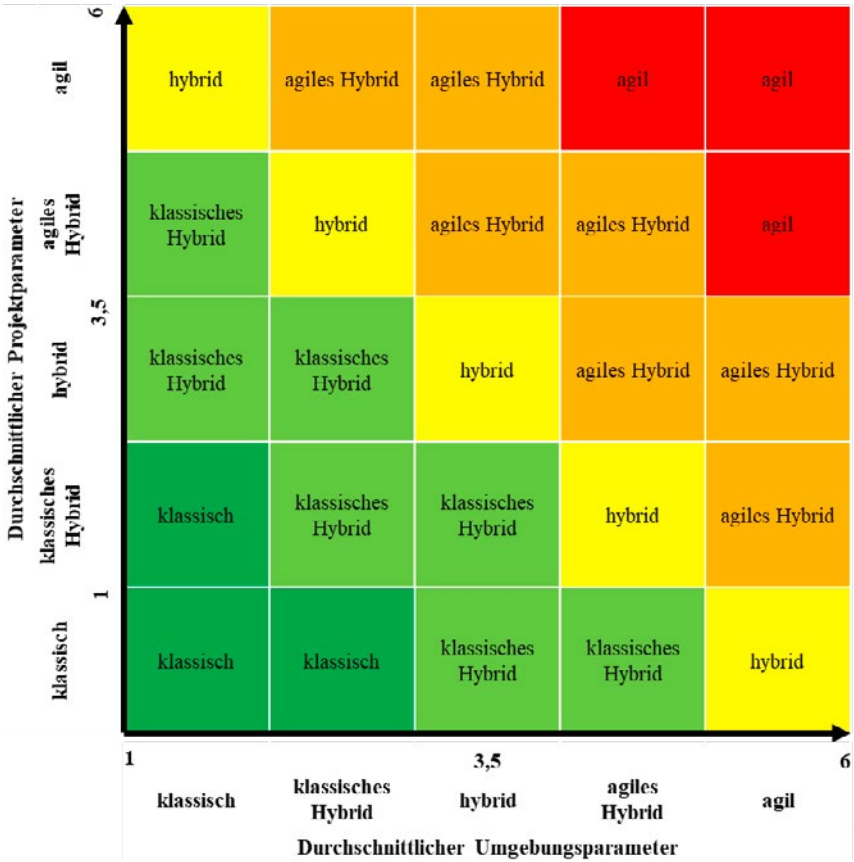

Abbildung 2: Matrix zur Auswahl der passenden Projektphilosophie

Mit dieser Methode wird es Unternehmen ermöglicht, eine passende Projektphilosophie auszuwählen. Es ist auch möglich einzelne Teilprojekte zu evaluieren, da die projektspezifischen Parameter in den Teilprojekten stark unterschiedlich ausgeprägt sein können. Sollten jeweils ein Teilprojekt klassisch und hybrid gesteuert werden, wird das Gesamtprojekt insgesamt gesehen hybrid gesteuert. Die Umgebungsparameter sollten sich allerdings nicht unterscheiden, da bspw. die Unternehmenskultur in Normalfall projektunabhängig ist. Dies bedeutet auch, dass unterschiedliche Projekte im gleichen Kontext nur hinsichtlich der projektspezifischen Parameter neu bewertet werden muss.

4 Evaluierung

In diesem Kapitel wird dargestellt, wie die Parametergewichtungen aus Tabelle 1, als essenzieller Bestandteil der vorgestellten Methode, ermittelt wurden. Die Gewichtungen sind ein wichtiger Bestandteil für die schnelle Anwendung der Methode. Darüber hinaus wurden die vorgestellten Parameter bisher nicht in Verbindung gesehen wurden und nicht methodisch nach Wichtigkeit für ein Projekt differenziert. Die Benennung ohne klare Handlungsanweisung wie

vorzugehen ist, wenn einer der Parameter agil ausgeprägt ist und zwei weitere klassisch ausgeprägt sind ist jedoch nicht hinreichend. Bisherige Untersuchungen legen teilweise sogar nahe, dass die Parameter gleichwertig seien. Im Rahmen der Studie wurde als Gegenhypothese formuliert, dass die Parameter unterschiedlich wichtig sind. Die Parametergewichtung wirkt später stark auf den gewichteten Mittelwert in der oben vorgestellten Methode, weshalb es wichtig ist hierzu eine möglichst genaue und allgemeine Einschätzung über die Gewichtungen zu gewinnen. Die Durchführung und Ergebnisse werden im Folgenden präsentiert und basieren auf den langjährigen Erfahrungen verschiedener Forscher, Projektmanager und Projektbeteiligter, die an der Studie teilgenommen haben. Die Gewichtung wurde auf Basis der Befragung so konstruiert, dass diese möglichst einfach und möglichst breit angewendet werden kann.

Die Feldphase der Befragung fand im Zeitraum vom 07. August 2019 bis 05. September 2019 statt. Gestreut wurde die Umfrage über projektmanagementspezifische Gruppen in sozialen Netzwerken wie XING und LinkedIn sowie direkt an verschiedene Unternehmen, Projektmitarbeiter und Wissenschaftler im Bereich des Projektmanagements. Die Befragten wurden gebeten die Parameter paarweise zu vergleichen. Für jede mögliche Parameterkombination innerhalb einer Kategorie mussten die Teilnehmer entscheiden, ob Parameter 1 (p_1) oder Parameter 2 (p_2) wichtiger ist. Es gab also nur 2 Entscheidungsmöglichkeiten. Dieses Vorgehen beruht auf der Methode des paarweisen Vergleichs mit gleichzeitiger Überführung im Hintergrund in eine Relevanzmatrix nach Gausemeier und Plass [GP14]. Wenn also wie in Tabelle 2 p_2 als wichtiger wie p_1 eingeschätzt wird, erhält der als wichtiger eingeschätzte Parameter im direkten Vergleich einen Punkt und der andere null Punkte. Die Methode führt wie in Tabelle 2 dargestellt im Ergebnis zu den benötigten Parametergewichtungen, indem die Relevanzsumme (Zeilensumme) pro Parameter gebildet wird und durch die Gesamtrelevanzsumme über alle Parameter in Relation gesetzt wird. Zudem können die verschiedenen Parameter schnell und systematisch verglichen werden, selbst wenn sie mitunter nicht immer leicht zu vergleichen sind [So15].

	p_1	**p_2**	**p...**	**p_n**	**Relevanzsumme**	**Gewichtung**
p_1		0	0	0	0	0 %
p_2	1		0	0	1	17 %
p...	1	1		0	2	33 %
p_n	1	1	1		3	50 %
0 ≙ Zeile ist unwichtiger als Spalte 1 ≙ Zeile ist wichtiger als Spalte						

Tabelle 2: Grundstruktur der Relevanzmatrix

An der Studie haben sich insgesamt 45 Personen beteiligt. Die Teilnehmerstruktur hinsichtlich der hauptsächlichen Projektfunktion der Befragten ist in Abbildung 3 dargestellt. Projektleiter / Projektmanager sind in der Umfrage stark repräsentiert. Dies ist für die Auswertung wichtig, da diese die projektrelevanten Entscheidungen treffen und Verantwortung für die Durchführung tragen. Es ist allerdings auffallend, dass überwiegend traditionelle Rollen vertreten sind und agile Funktionsträger wie Entwickler oder Scrum Master nur einen geringen Anteil der Befragten ausmachen. Darüber hinaus haben sich nur 45 Personen beteiligt, sodass wir anstreben die Untersuchung künftig zu wiederholen, um den Teilnehmerkreis zu erhöhen und die Konfidenz der Validität der ermittelten Gewichtung zu stärken.

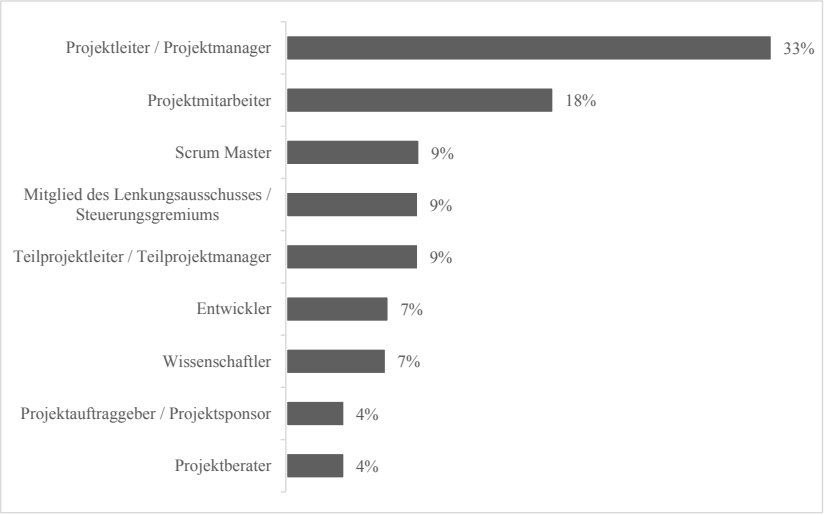

Abbildung 3: Teilnehmerstruktur hinsichtlich ihrer Projektfunktion

Die Verteilung der Projekterfahrung ist in Abbildung 4 dargestellt. Der Median liegt bei 15 Jahren Projekterfahrung. Daher ist zu vermuten, dass die Teilnehmer durchaus mit der Thematik der Agilität vertraut sind und diese mit der klassischen Projektdenkweise vergleichen können, da erstere in den letzten 20 Jahren zunehmend populärer wurde.

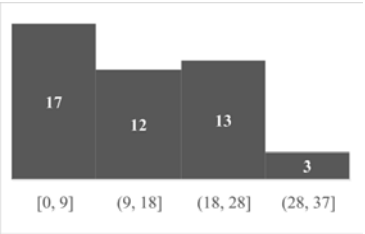

Abbildung 4: Histogramm zur Projekterfahrung der Teilnehmer in Jahren

Die Ergebnisse in Tabelle 3 zeigen, dass die verschiedenen Parameter als unterschiedlich relevant wahrgenommen werden. So ist bei den Umgebungsparametern der Qualifikationsgrad der Teammitglieder (QT) mit 34 % deutlich stärker gewichtet als die Parameter Kultur (KU) und Teamverteilung (TV). Bei einer Gleichgewichtung der Parameter würde die Gewichtung pro Parameter 25 % (100 %/Anzahl der Parameter der Kategorie) betragen. Ein ähnliches Bild ergibt sich bei den projektspezifischen Parametern. Die projektspezifischen Parameter wurden ebenfalls unterschiedlich gewichtet. So spielt der Produktlebenszyklus (PL) und die Projektteamgröße (GR) eine vergleichsweise untergeordnete Rolle. Dagegen ist die Komplexität des Projekts (KO) wesentlich höher gewichtet als bei einer Gleichgewichtung der Parameter von jeweils 20 %. Durch die Umfrage wurde die Hypothese bestätigt, dass nicht alle Parameter gleich gewichtet werden sollten. Die Parametergewichtungen können durch die Anwender der Methode übernommen werden, sofern keine eigenen Parameter ergänzt wurden. Die Studie

stellt eine wichtige Basis für das vorgestellte Berechnungsmodell (Tabelle 1) und basiert auf der Erfahrung der erfahrenen Studienteilnehmer aus Wissenschaft und Praxis.

Umgebungsparameter	KU	QT	AO	TV	Relevanzsumme	Gewichtung
KU		14	18	21	53	19 %
QT	31		28	33	92	34 %
AO	27	17		25	69	26 %
TV	24	12	20		56	21 %

Projektspezifische Parameter	SA	GP	GR	KO	PL	Relevanzsumme	Gewichtung
SA		28	25	13	29	95	21 %
GP	17		26	13	27	83	19 %
GR	20	19		8	22	69	15 %
KO	32	32	37		39	140	31 %
PL	16	18	23	6		63	14 %

Tabelle 3: Aggregierte Relevanzmatrizen

5 Zusammenfassung und Ausblick

Beeinflusst durch Veränderungen im Management von Softwareentwicklungsprojekten und das Aufkommen agiler Projektmethoden, wird Projektmanagement heute im Allgemeinen anders gedacht. Während agile Methoden immer mehr das herkömmliche Projektmanagement ablösen, zeigen beide Philosophien je nach Einsatzumgebung ihre Daseinsberechtigung. Darüber hinaus entstand das hybride Projektmanagement, um die Kluft zwischen beiden Philosophien zu überbrücken und die Vorteile von beiden Philosophien zu nutzen. In der Praxis wird die Wahl zwischen diesen drei Philosophien oft ohne einen fundierten Entscheidungsprozess getroffen. Um den Auswahlprozess zu verbessern fehlte bislang eine konkrete Handlungsanleitung für die Praxis.

Entgegen der gegenwärtigen Praxis ist es für eine optimale Auswahl wichtig, projektspezifische und Umgebungsparameter als Kriterien bereits im Entscheidungsprozess zu nutzen. Die von uns vorgestellte Methode bietet einen Auswahlprozess von Projektphilosophien durch den kombinierten Einsatz bekannter Parameter und ermöglicht eine neue individualisierbare Auswahl. Auf der Grundlage einer Studie konnte die Bedeutung der einzelnen Parameter differenziert werden. Als Nächsten Schritt sind weitere Untersuchungen geplant, so soll ein webbasiertes Instrument zur Unterstützung des gesamten Auswahlverfahrens entwickelt werden. Ergänzend soll die Gewichtung der Parameter künftig wiederholt untersucht werden, um die Signifikanz der Vorkonfiguration zu bestätigen und auch mögliche Änderungen (Concept Drift) zu erfassen. In Ergänzung hierzu planen wir eine Methode zur Anpassung der Parametergewichtungen an die individuelle Projektsituation umzusetzen, sodass Projektverantwortliche auch die Auswahl optimal anpassen können. Ein weiteres Forschungsvorhaben ist die Entwicklung von automatisiert generierten Projektvorgehensmodellen auf Basis der Parameterausprägungen. Dazu zählt auch die Eingruppierung bestehender Modelle in die Matrix der Auswahlmethode, sodass bekannte Vorgehensweisen direkt aus der Matrix abgelesen und verwendet werden können.

Literaturverzeichnis

[Ali00] Cockburn A. (2000): Selecting a Project's Methodology. Humans and Technology, IEEE SOFTWARE.

[An16] Angermeier G. (2017): Hybrides Projektmanagement. Projekt Magazin. https://www.projektmagazin.de/glossarterm/hybrides-projektmanagement, zuletzt geprüft am 10.04.2019.

[Br17] Brehm L.; Feldmüller D. und Rieke T. (2017): Konfiguration des hybriden Projektmanagements für die Entwicklung technischer, physischer Produkte. In Barton T. et al. (Hg.), Angewandte Forschung in der Wirtschaftsinformatik: Prozesse, Technologie, Anwendungen, Systeme und Management. 1. Aufl. Heide: mana-Buch. S. 30-39.

[BT09] Boehm B. und Turner R. (2009): Balancing Agility and Discipline: A Guide for the Perplexed. 7. Aufl. Boston: Addison-Wesley.

[Die16] Diebold P.; Zehler T.; Schmitt A. et al. (2016): Prozessverbesserung durch fragmentierte Anwendung von Scrum & Co. Projektmanagement und Vorgehensmodelle, Gesellschaft für Informatik e.V. S. 135-143.

[GP14] Gausemeier J. und Plass C. (2014): Zukunftsorientierte Unternehmensgestaltung: Strategien, Geschäftsprozesse und IT-Systeme für die Produktion von morgen. 2. Aufl. München: Carl Hanser.

[Ha13] Habermann F. (2013): Hybrides Projektmanagement – agile und klassische Vorgehensmodelle im Zusammenspiel. HMD – Praxis der Wirtschaftsinformatik (5). S. 93-102.

[He70] Henderson B. (1970): The Product Portfolio. http://image-src.bcg.com/Images/BCG_The_Product_Portfolio_tcm108-139921.pdf, zuletzt geprüft am 10.05.2020.

[Hü18] Hüsselmann C. (2018): #03 Was ist eigentlich Agilität?. und hybrid? Process and Project. zuletzt geprüft am 10.05.2019.

[KK17] Komus A. und Kuberg M. (2017): Status Quo Agile. Deutsche Gesellschaft für Projektmanagement. https://www.gpm-ipma.de/fileadmin/user_upload/GPM/Know-How/Studie_Status_Quo_Agile_2017.pdf, zuletzt geprüft am 10.05.2020.

[Ko16] Komus A. (2016): Studie „agiles PMO" - Studienbericht -. Process and Project. https://www.process-and-project.net/studien/rq-studie-agiles-pmo/, zuletzt geprüft am 10.05.2020.

[KR14] Klein T.P. und Reinhart G. (2014): Approaches for Integration of Agile Procedures into Mechatronic Engineering of Manufacturing Systems. In Zäh M. (Hg.), Enabling Manufacturing Competitiveness and Economic Sustainability. 1. Aufl. Cham: Springer. S. 225-230.

[KS18] Kurtz K. und Sauer J. (2018): Auswirkungen des Einsatzes hybrider Methoden auf die Projektsteuerung. In Mikusz M. et al. (Hg.), Projektmanagement und Vorgehensmodelle 2018. 1. Aufl. Bonn: Gesellschaft für Informatik. S. 73-83.

[Ku19a] Kuhrmann M. (2019): Reines agiles Vorgehen kein "Allheilmittel". projektMANAGEMENT aktuell (3). S. 6-15.

[Ku19b] Kuster J.; Bachmann C.; Huber E. et al. (2019): Handbuch Projektmanagement: agil - klassisch - hybrid. 4. Aufl. Wiesbaden: Springer Gabler.

[Mc73] McGregor D. (1973): Der Mensch im Unternehmen (The Human Side of Enterprise). 3. Aufl. Düsseldorf und Wien: Econ.

[Pa18] Paukner M.; Seel C. und Timinger H. (2018): Projektparameter für das Tailoring hybrider Projektmanagementvorgehensmodelle. In Barton T. et al. (Hg.), Angewandte Forschung in der Wirtschaftsinformatik. 1. Aufl. Heide: mana-Buch. S. 166-176.

[SBK14] Sandhaus G.; Berg B. und Knott P. (2014): Hybride Softwareentwicklung: Das Beste aus klassischen und agilen Methoden in einem Modell vereint. 1. Aufl. Wiesbaden: Springer Vieweg.

[Sc19] Schröder M.; Steinhorst U. und Winter M. (2019): Hybrides Projektmanagement – Einbindung agiler Arbeitsweisen im Rahmen der fortschreitenden Digitalisierung. In Schröder M. und Wegner K. (Hg.), Logistik im Wandel der Zeit – Von der Produktionssteuerung zu vernetzten Supply Chains. 1. Aufl. Wiesbaden: Springer Gabler. S. 829-844.

[Sn00] Snowden D. (2000): The Social Ecology of Knowledge Management. In Despres C. und Chauvel D. (Hg.), Knowledge horizons : the present and the promise of knowledge management. 1. Aufl. Woburn: Butterworth–Heinemann. S. 237-265.

[So15] Sonntag A. (2015): Instrument: Paarweiser Vergleich. Universität Hamburg. https://www.inf.uni-hamburg.de/de/inst/ab/itmc/research/completed/promidis/instrumente/paarweiser-vergleich, zuletzt geprüft am 10.05.2020.

[Šp14] Špundak M. (2014): Mixed Agile/Traditional Project Management Methodology – Reality or Illusion? Procedia - Social and Behavioral Sciences (119). S. 939-948.

[Ti17] Timinger H. (2017): Modernes Projektmanagement: Mit traditionellem, agilem und hybridem Vorgehen zum Erfolg. 1. Aufl. Weinheim: Wiley-VCH.

[TS16] Timinger H. und Seel C. (2016): Ein Ordnungsrahmen für adaptives hybrides Projektmanagement. projektMANAGEMENT aktuell (4). S. 55-61.

Von datenbasierter zu datengetriebener Geschäftsmodellentwicklung: Ein Überblick über Software-Tools und deren Datennutzung

Sebastian Gottschalk, Enes Yigitbas

Software Innovation Lab, Universität Paderborn

Zukunftsmeile 2, 33102 Paderborn

{sebastian.gottschalk,enes.yigitbas}@uni-paderborn.de

Abstract: Die kontinuierliche Weiterentwicklung des eigenen Geschäftsmodells ist für eine Organisation von entscheidender Bedeutung, um wettbewerbsfähig und somit nachhaltig erfolgreich zu bleiben. Während für die Entwicklung neuer Geschäftsmodelle häufig Workshops und einfache Software-Tools zur Visualisierung genutzt werden, wurden in der Forschung bereits erste Ansätze von datengetriebener Geschäftsmodellentwicklung (GME) vorgestellt. Diese Ansätze nutzen dabei Daten, Informationen oder auch Wissen aus internen und externen Unternehmensquellen, um den GME-Prozess zu unterstützen. Innerhalb dieses Beitrags zeigen wir einige Ansätze aus der aktuellen Literatur und analysieren wie ihre Datennutzung den GME-Prozess unterstützt. Weiterhin stellen wir mit dem BMDL Feature Modeler ein Tool vor, welches den GME-Prozess mit Expertenwissen unterstützt.

1 Einleitung

Ein Geschäftsmodell beschreibt die Logik wie ein Unternehmen Werte für seine Kunden schaffen kann [1]. Um wettbewerbsfähig nachhaltig erfolgreich zu bleiben, müssen Organisationen ihr Geschäftsmodell kontinuierlich weiterentwickeln [2]. Dies ist auch ein Ergebnis des GE Innovation Barometers 2018 [3], einer Umfrage unter Führungskräften zu Innovationsthemen. In diesem gaben 64% der Befragten an, dass sie Schwierigkeiten damit haben effektive Geschäftsmodelle für neue Ideen zu entwerfen, um diese nachhaltig profitabel zu machen. Ein Vergleich dieser Studie mit den Ergebnissen von 2015 zeigt, dass diese Herausforderung in den letzten Jahren noch zugenommen hat (vgl. 59% von 3000 Führungskräften im Jahr 2015). Ein Grund dafür ist, dass Kunden mehr und mehr Lösungen für ihre wahrgenommenen Bedürfnisse anstatt reiner Produkte möchten [4]. Dies führt dazu, dass ein gutes Geschäftsmodell für ein Produkt wichtiger als die neuste Technologie sein kann [5].

Der Prozess der Geschäftsmodellinnovation ist eine kreative Aufgabe, für welche in der Regel die Zusammenarbeit mehrerer Stakeholder in und außerhalb der Firma benötigt wird [6]. Eine Möglichkeit stellen dabei sogenannte Innovationsworkshops (beispielsweise mittels Design Thinking [7]) dar, in welchen die Teilnehmer versuchen, die Bedürfnisse der Kunden zu verstehen und daraus konkrete Geschäftsmodelle mit einem gezielten Wertversprechen abzuleiten. Zudem werden konkrete Handlungsempfehlungen für die Validierung und Umsetzung des Geschäftsmodells erarbeitet. Um den Ideengenerierungsprozess zu unterstützten, werden in der Regel Tools zur Unterstützung eingesetzt. Beispiele hierfür sind das Business Model Canvas, welches ein Geschäftsmodell visuell in neun Komponenten unterteilt [1] oder Business Model Pattern Cards [8], welche existierende Pattern für erfolgreiche Geschäftsmodelle aus bestehenden Unternehmen extrahiert haben.

Sowohl in der Wissenschaft [9] als auch in der Praxis [10] wurden dabei immer mehr *datenbasierte* Software-Tools vorgestellt, mit welchen sich der Geschäftsmodellentwicklungsprozess unterstützen lässt. Die in der Praxis entwickelten Software-Tools basieren dabei häufig auf der

reinen *Visualisierung* verschiedener Versionen des Geschäftsmodells. Hierzu setzen sie häufig auf das weit verbreitete Business Model Canvas (BMC) [1] und lassen teilweise kollaborativ verschiedene Stakeholder an der Erstellung teilnehmen [10]. Weiterhin ermöglichen diese Software-Tools eine Diskussion zwischen den einzelnen Teilnehmern auf Basis von Kommentar- und Chatfunktionen. Im Gegensatz dazu wurden in der Wissenschaft vermehrt *datengetriebene* Software-Tools vorgestellt, welche den Prozess der Geschäftsmodellentwicklung auf Basis von Daten unterstützen. Diese Daten lassen sich durch ihre Bedeutung zu Informationen und im Anschluss durch deren Vernetzung zu Wissen verarbeiten. Dieses Wissen wiederum kann durch ein Software-Tool genutzt werden, um die gezielte *Transformation* eines Geschäftsmodells zu unterstützen.

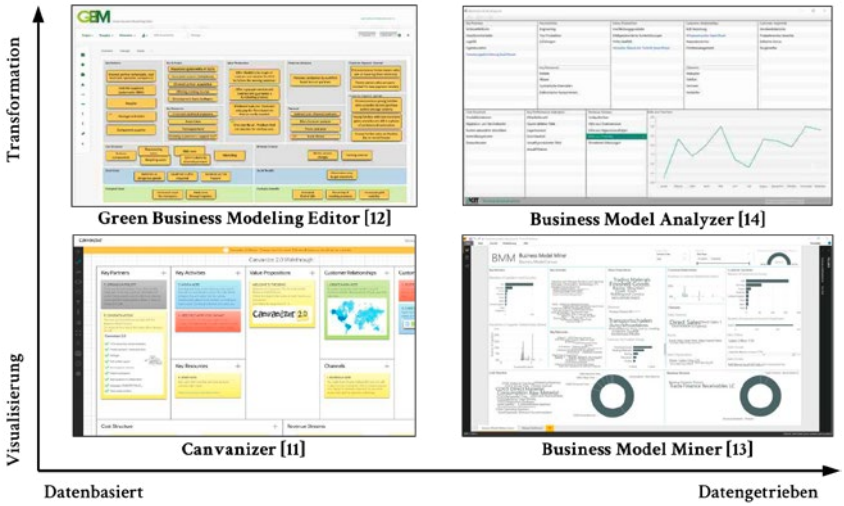

Abbildung 1: Einordnung ausgewählter Ansätze von Geschäftsmodellentwicklungstools

Ausgewählte Ansätze für solche Software-Tools sind dabei in Abbildung 1 eingeordnet. *Canvanizer* [11] ist ein Software-Tools aus der Praxis, welches unterschiedliche Canvas-Strukturen visualisiert und eine kollaborative Bearbeitung durch mehrere Nutzer erlaubt. Der *Green Business Modeling Editor* [12] unterstützt die Entwicklung von nachhaltigen Geschäftsmodellen. Hierzu werden dem BMC weitere Komponenten hinzufügt und Informationen über verschiedene Konfigurationen von Geschäftsmodellen hin zu einer Transformation gesammelt. Der *Business Model Miner* [13] erstellt eine Visualisierung des Geschäftsmodells auf Basis von existierenden Unternehmensdaten. Hierzu analysiert und aggregiert das Tool die Daten des ERP-Systems der Organisation. Der *Business Model Analyzer* [14] unterstützt dabei die datengetriebene Transformation des aktuellen Geschäftsmodells zu einem gewünschten Zielzustand. Basierend auf dem BMC werden hierzu semantische Beziehungen zwischen den einzelnen Elementen des Geschäftsmodells modelliert und relevante KPIs des Geschäftsmodells definiert.

Ziel dieses Beitrages ist es dabei, den Lesern einen Überblick über die datengetriebene Geschäftsmodellentwicklung anhand von ausgewählten Ansätzen zu geben. Hierzu gehen wir zunächst auf den wissenschaftlichen Hintergrund zu Geschäftsmodellen und Geschäftsmodellentwicklungstools ein. Darauf aufbauend stellen wir eine Wissenspyramide mit einer Unterteilung zwischen internen und externen Unternehmensquellen vor, aus dessen wir die Datennutzung

mit ausgewählten Ansätzen aus der Literatur aufzeigen. Weiterhin stellen wir mit dem BMDL Feature Modeler [15] einen eigenen Ansatz zur Nutzung von Expertenwissen im Geschäftsmodellierungsprozess vor. Hierbei gehen wir besonders auf die Modellierung des Expertenwissens und dessen anschließender Nutzung ein. Schlussendlich fassen wir unsere Ergebnisse zusammen und geben ein Ausblick auf mögliche weitere Entwicklungen.

2 Geschäftsmodellentwicklung

Innerhalb dieses Kapitel gehen wir auf das Thema der Geschäftsmodelle sowie den damit verbundenen Geschäftsmodelentwicklungstools ein.

2.1 Geschäftsmodelle

Ein Geschäftsmodell beschreibt das Konzept, wie eine Organisation Werte schafft, liefert und sichert [1]. Auf dieser Basis ist das Geschäftsmodell eine detaillierte Beschreibung der Firmenstrategie [16], welche zwischen der Geschäftsstrategie und den Geschäftsprozessen agiert [17]. Ein Alignment dieser drei Bereiche ist dabei in Abbildung 2 zu sehen. Eine *Geschäftsstrategie* beschreibt dabei die Positionierung einer Organisation innerhalb seines Umfeldes. Um diese langfristige Positionierung zu sichern, beschreibt das *Geschäftsmodell* daher, wie Werte aus dieser Positionierung heraus für den Kunden geschaffen werden können. Diese Wertschöpfung wiederum wird durch die eigentlichen *Geschäftsprozesse* der Organisation erreicht.

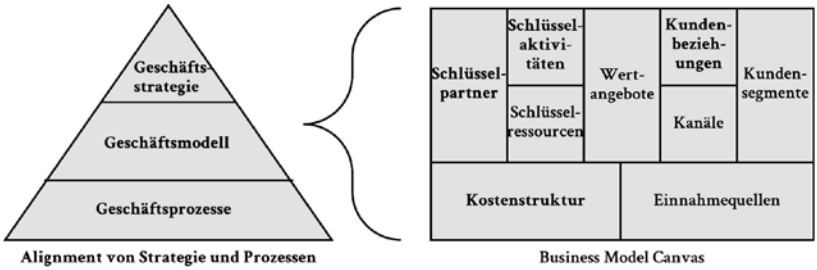

Abbildung 2: Geschäftsmodell als Vermittler zwischen der Strategie und den Prozessen

Mit dieser Vermittlerrolle erreicht das Geschäftsmodell dabei steigendes Interesse sowohl von der Praxis [3] als auch aus unterschiedlichen Bereichen der Wissenschaft wie beispielsweise der Informationssysteme, dem Innovationsmanagement oder Strategieentwicklung [4]. Um ein Geschäftsmodell dabei visuell darzustellen, wurden unterschiedliche Geschäftsmodellmodellierungsprachen wie beispielweise das Business Model Canvas (BMC) [1] oder die e3-Value Modeling Language [18] entwickelt. Das BMC, welche auch in Abbildung 2 zu sehen ist, bildet dabei den de-facto Standard für die Geschäftsmodellmodellierung. Hierbei wird das Geschäftsmodell in die neun Komponenten der Wertangebote, der Kundensegmente, der Kundenbeziehungen, der Kanäle, der Einnahmequellen, der Schlüsselpartner, der Schlüsselaktivitäten und -ressourcen sowie der Kostenstruktur unterteilt. Diese Komponenten lassen sich nun durch einzelne Elemente (z.B. Abo-Modell bei Einnahmequellen) verfeinern, was in Innovationsworkshops häufig durch die Nutzung von Haftnotizen auf einem ausdruckten Canvas gelöst wird. Während diese Visualisierung dabei jedoch nur einen einzelnen Zeitpunkt des Geschäftsmodells darstellen können, wurde mit dem Begriff der dynamischen Geschäftsmodellmodellierung eingeführt [19]. Unter diesem Begriff werden jedoch häufig nur Standard-Ansätze der

Entscheidungsunterstützung aus anderen Bereichen (z.B. System Dynamics, Szenario Planung) auf den Bereich der Geschäftsmodellentwicklung übertragen. Um dabei sowohl die statischen als auch die dynamischen Ansätze effizient nutzen zu können, werden Geschäftsmodellentwicklungstools benötigt.

2.2 Geschäftsmodellentwicklungstools

Diese software-gestützten Geschäftsmodellentwicklungstools bieten dabei unterschiedliche Level an Unterstützung, um neue Geschäftsmodelle zu entwickeln und existierende Geschäftsmodelle zu verbessern. Ein frühes Beispiel aus der Literatur ist dabei der e3-Value Editor [18]. Basierend auf der e3-Value Modeling Language wird mit diesem ein Wertnetzwerk von mehreren Akteuren modelliert. Innerhalb dieses Modells können nun Wertflüsse zwischen den einzelnen Akteuren definiert werden, welche zum Kalkulieren einer finanziellen Einschätzung verschiedener Szenarien genutzt werden kann. Ein weiteres Beispiel ist der BM|DESIGN|ER [20], welcher sich auf die Visualisierung des Business Model Canvas konzentriert. Hierzu stellt das Tool einen Editor bereit, um Unternehmen mit verschiedenen Geschäftsmodellen einzutragen, wobei sich die Geschäftsmodelle über den zeitlichen Verlauf ändern können. Ähnliche Ansätze lassen sich auch bei in der Praxis untersuchten Tools wiederfinden [10].

Zudem wurden in der Wissenschaft mehrere Ansätze vorgestellt, welche über die eigentliche Visualisierung des Geschäftsmodells hinausgehen. Diese Ansätze, welche sich häufig als Entscheidungsunterstützungssysteme verstehen, nutzen dabei verschiedene Daten aus unterschiedlichen Quellen, um den Prozess der Geschäftsmodellierung aktiv zu unterstützen. Beispiele hierfür sind die Analyse von Daten in ERP-Systemen [13], die systematische Transformation auf Basis von KPIs [12] oder die Crowd-Validierung von Ideen für Geschäftsmodelle [20]. Häufig werden hier jedoch nur die wesentlichen Eigenschaften der Systeme in Form von Design-Prinzipien vorgestellt, sodass diese Tools nicht von anderen Wissenschaftlern und Unternehmern genutzt werden können. Hierbei wird davon ausgegangen, dass sich Geschäftsmodellmodellierungstools in der Zukunft immer mehr zu Entscheidungsunterstützungssystemen entwickeln werden [22]. Eine offene Fragestellung ist jedoch bis zu welchen Grad sich der Prozess der Geschäftsmodellentwicklung automatisieren lässt [9].

3 Überblick über datengetriebene Geschäftsmodellentwicklungstools

Um die Geschäftsmodellentwicklung zu unterstützen, wurden dabei in der Forschung erste datengetriebene Ansätze vorgestellt. Wie in Abbildung 3 zu sehen, lässt sich dabei zwischen Daten, Informationen und Wissen unterscheiden. *Daten* sind dabei Symbole und Zeichen, welche durch das Sammeln und Messen von Beobachtungen entstehen. Diese müssen nun mit einem entsprechenden Kontext verknüpft werden, um Informationen zu erhalten. Durch die Vernetzung von *Informationen* zu einem bestimmten Sachverhalt und deren Interpretation kann zudem *Wissen* generiert werden.

Sowohl Daten, Informationen als auch Wissen können dabei aus internen oder externen Unternehmensquellen kommen. *Interne Unternehmensquellen* nutzen dabei Daten aus Systemen wie dem ERP oder der Workflow Engine, um Informationen über Vertriebskanäle oder Key- Performance-Indikatoren zu erhalten. Diese können im Anschluss zu Wissen über beispielsweise das aktuelle Geschäftsmodell oder mögliche Transformationen vernetzt werden. *Externe Unternehmensquellen* nutzen beispielsweise die Daten von Crowd-Plattformen oder dem Semantic Web, um Informationen über Kunden oder Märkte zu erhalten. Diese können im Anschluss zu Wissen über beispielsweise validierte eigene Geschäftsmodelle oder erfolgreiche Geschäftsmodelle von Wettbewerbern vernetzt werden.

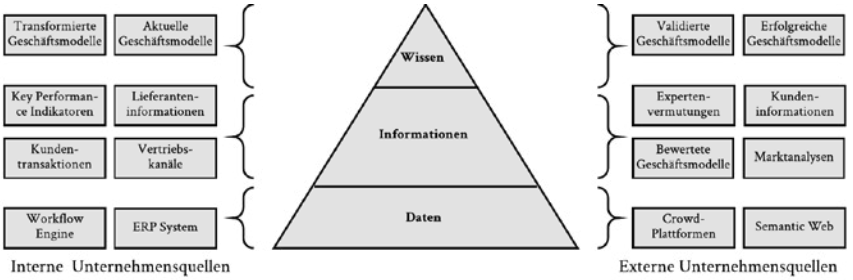

Abbildung 3: Beispiele für Daten, Informationen und Wissen aus Unternehmensquellen

3.1 Interne Unternehmensquellen

Bei den internen Unternehmensquellen werden die für die Geschäftsmodellentwicklung benötigten Daten innerhalb der Organisation gewonnen und aufbereitet. Diese können beispielsweise aus den organisationseigenen Informationssystemen gewonnen oder durch Kollaboration verschiedener Mitarbeiter entwickelt werden. Ansätze in der Literatur sind hier der Business Model Miner, der Business Model Analyzer oder das Business Model – Business Process Alignment Tool.

Der *Business Model Miner* [13] sammelt und konsolidiert dabei Geschäftsmodell-bezogene Daten aus verschiedenen Organisationssystemen (z.B. ERP-System, SCM-System, CRM-System) und präsentiert diese aufbereitet auf Basis von Templates. Als konkreten Anwendungsfall wird dabei auf die Daten des ERP-Systems einer Organisation zugegriffen und diese zu unterschiedlichen Informationen wie den Erlösinformationen einzelner Produkte, den bestandenen Vertriebskanälen, den Kundentransaktionen oder den Lieferanteninformationen weiterverarbeitet. Diese werden nun zu den neun Komponenten des Business Model Canvas in Beziehung gesetzt, um Wissen über das aktuelle Geschäftsmodell bereitzustellen.

Der *Business Model Analyzer* [14] unterstützt die Transformation eines aktuellen Geschäftsmodells hin zu einem gewünschten Geschäftsmodell durch Nutzung von semantischen Beziehungen zwischen den einzelnen Geschäftsmodell-Elementen und Quantifizierung der Auswirkungen einer Geschäftsmodelländerung. Hierzu nutzt dieser ebenfalls die Daten aus verschiedenen Organisationssystemen, um Informationen über einzelne Teile des Geschäftsmodells zu sammeln. Diese werden mit Informationen zu KPIs und möglichen Geschäftsmodell-Alternativen angereichert, um eine Wissensdatenbank zur Geschäftsmodelltransformation zu erstellen.

Das *Business Model-Business Process Alignment Tool* [25] gleicht die Geschäftsmodelle mit den Geschäftsprozessen ab, um den Entscheidungsprozess für mögliche Änderungen zu verbessern. Hierbei werden dem gesamten BMC, einzelnen Komponenten oder einzelnen Elementen jeweils Teile der Geschäftsprozesse zugeordnet. Der Ansatz besteht aus zwei Tools, welche jeweils Daten über ihren Zustand halten und diese in aufbereiteter Form als Informationen austauschen. Diese Daten werden zusammengesetzt, um vernetztes Wissen über die Anordnung zu erhalten.

3.2 Externe Unternehmensquellen

Bei den externen Unternehmensquellen werden die für die Geschäftsmodellentwicklung benötigten Daten außerhalb der Organisation gewonnen. Dies kann beispielsweise die Kollaboration mit Endnutzern über Crowd-Plattformen oder die Nutzung von Expertenwissen sein. Ansätze in der Literatur sind hier die Business Model Idea Generators, das Crowd-based Business Model Validation System oder der Smart Business Modeler.

Der Begriff *Business Model Idea Generators* [26] definiert eine neue Art von Informationssystem, welches Geschäftsmodellideen durch eine Kombination von menschlichen Ideen und Computerwissen generiert. Hierzu erstellt das System iterativ mögliche Geschäftsmodelle, lässt diese durch Nutzer bewerten und rekombiniert gut bewertete Teilideen zu neuen Geschäftsmodellen. Hierzu stellen die Nutzer Daten in Form von Bewertungen zur Verfügung, welche zu Informationen über mögliche Geschäftsmodelle und schließlich zu Wissen über validierte Geschäftsmodelle aggregiert werden.

Das *Crowd-based Business Model Validation System* [21] ermöglicht dem Unternehmer einen schnellen und skalierbaren Zugang zu Nutzern, Experten und Investoren, um durch Feedback die Unsicherheiten im eigenen Geschäftsmodell zu reduzieren. Hierzu stellt es das zu bewertende Geschäftsmodell auf Crowd-Plattformen zur Verfügung und sammelt die entsprechenden Daten ein. Ähnlich wie bei den Business Model Idea Generators stellen die Nutzer Daten in Form von Feedback zur Verfügung, welche zu Informationen aggregiert und zu Wissen vernetzt wird.

Der *Smart Business Modeler* [27] ist ein Tool zur Geschäftsmodellentwicklung, welches den Prozess des Geschäftsmodellerstellung mittels Geschäftsmodellpattern unterstützt. Hierzu kann der Nutzer aus mehreren Pattern-Packs diejenigen auswählen, welche für sein Geschäftsmodell relevant sind. Hierbei haben die Autoren die Daten zu nachhaltigen Geschäftsmodellen in der Literatur analysiert und hieraus Informationen abgeleitet. Diese Informationen werden dem Nutzer im Anschluss als Wissen in Form von Pattern-Packs zur Verfügung gestellt.

4 Nutzung von Expertenwissen durch den BMDL Feature Modeler

Innerhalb unserer Forschung in einem Teilprojekt des Sonderforschungsbereich 901 „On-The-Fly Computing“[1] arbeiten wir an Methoden und Tools zur systematischen Gestaltung und Entwicklung Plattform-basierter Geschäftsmodelle. Innerhalb dieser Forschung wurde mit dem BMDL Feature Modeler[2] (BMDL = Business Model Decision Line) ein Software-Tool entwickelt, welches den Geschäftsmodellentwickelungsprozess mithilfe des Konzeptes von Software-Produktlinien unterstützen kann. Ziel ist es dabei, dass Wissen zur Geschäftsmodellentwicklung explizit zu formulieren und es somit unternehmensintern und unternehmensextern wiederverwendbar zu machen. Hierzu wird das gesamte Geschäftsmodellwissen in Modellen gespeichert, voraus sich entsprechende konkrete Geschäftsmodelle ableiten lassen.

Hierzu haben wir das Domänenwissen zu möglichen Entscheidungen des Geschäftsmodells als Feature Model definiert, woraus sich nun einzelne Geschäftsmodelle als Instanzen des Feature Models ableiten und die dahinter liegenden Hypothesen mit Experimenten überprüfen lassen [15]. Um diese Validierung von Hypothesen zu unterstützen, haben wir darauf aufbauend eine Modellierungssprache für Experimente entworfen [23]. Weiterhin haben wir einen Ansatz entwickelt, um den Prozess mithilfe von konsolidiertem Expertenwissen zu unterstützen [24].

Innerhalb dieses Beitrages möchten wir auf genau diese Unterstützung von Expertenwissen eingehen, weil hierdurch der Prozess der Geschäftsmodellentwicklung datengetrieben verbessert werden kann. Innerhalb dieses Ansatzes, siehe Abbildung 4, stellen dabei unterschiedliche Experten ihr Wissen zu Geschäftsmodellen in modellierter Form zur Verfügung (*Erstellen von Expertenwissen*). Der Business Developer einer Organisation kann nun Teile dieses Wissens aus externen Unternehmensquellen auswählen und mit Wissen aus internen Unternehmensquel-

[1] Webseite des Sonderforschungsbereiches: https://sfb901.uni-paderborn.de/

[2] Online-Version des BMDL Feature Modeler: https://sebastiangtts.github.io/bmdl-feature-modeler

len zusammenführen (*Konsolidieren des Expertenwissens*). Im Anschluss kann er dieses konsolidierte Wissen nutzen, um eigene Geschäftsmodelle zu generieren (*Erstellen von Geschäftsmodellen*) und verschiedene Geschäftsmodelle mit dem eigenen zu vergleichen (*Vergleich von Geschäftsmodellen*). Hierzu möchten wir in diesem Beitrag tiefer auf die Modellierung des Expertenwissens und dessen Nutzung eingehen.

Abbildung 4: Ausschnitte aus dem BMDL Feature Modeler [15]

4.1 Modellierung von Expertenwissen

Um sowohl das Wissen der Experten als auch der Organisation zu modellieren, nutzen wir Feature-Modelle. Feature-Modelle stellen dabei eine kompakte Repräsentation aller möglichen Produkte einer Software-Produktlinie dar. Hierzu werden die einzelnen Features in einer Baum-Hierarchie angeordnet, sodass sich durch Selektion einzelner Features ein entsprechendes Produkt konfigurieren lässt. Dieser Konfigurationsraum lässt sich nun mittels entsprechenden Feature-Typen (verpflichtende und optionale Features), Eltern-Kind-Beziehung innerhalb der Features eines Baumes (OR und XOR Beziehungen) und Beziehungen zwischen Featuren verschiedener Bäume (erforderliche und ausschließende Features) weiter einschränken.

Innerhalb unseres Ansatzes nutzen wir Feature-Modelle, um das domänen-spezifische Wissen als Konfigurationsraum zu modellieren und daraus einzelne Konfiguration in Form von Geschäftsmodellen abzuleiten. Hierzu werden die Geschäftsmodelle hierarchisch (Canvas, Komponente, Element, Subelement) dargestellt und alle Einschränkungen von normalen Feature-Modellen nutzbar gemacht. Weiterhin haben wir einige Informationen zu diesen Modellen hinzugefügt, um diese über mehrere Tools hinweg durch eine domänen-spezifische Sprache austauschbar zu machen. So lässt sich zu jedem Experten-Modell ein Name, eine Beschreibung, die aktuelle Version sowie eine Lizenz hinzufügen. Weiterhin kann ein Autor mitsamt seiner Organisation, seiner E-Mail-Adresse sowie seiner Webseite referenziert werden. Um die einzelnen Features besser zu verstehen, wurden zusätzlich zu den Namen noch einzelne Beschreibungen hinzugefügt. Zudem wurden unterstützende und schadende Beziehungen (z.B. Auswählen von Privatsphäre und Werbeeinahmen) zwischen den einzelnen Features hinzugefügt, um mögliche Erfolgskriterien und Schwachstellen eines Geschäftsmodells sichtbar zu machen. Zu guter Letzt, lassen sich einzelne Konfiguration direkt im Expertenwissen speichern, was zur

Speicherung von Geschäftsmodell-Pattern als auch Beispielsorganisationen genutzt werden kann.

4.2 Nutzung von Expertenwissen

Um dieses Expertenwissen nun nutzbar zu machen, muss es zunächst konsolidiert und mit dem Wissen der Organisation verknüpft werden. Hierzu haben wir ein semi-automatisches Verfahren entwickelt, welches die Modelle verknüpft und mögliche Konflikte zwischen den Modellen erkennt. Diese müssen in Anschluss von Nutzer des Tools behoben werden. Das so konsolidierte Wissen kann dabei in unterschiedlichen Phasen der Geschäftsmodellentwicklung genutzt werden:

- **Durchsuchen von Geschäftsmodell-Elementen:** Während der Erstellung von neuen Geschäftsmodellen kann das Wissen als Datenbank für mögliche Elemente von Geschäftsmodellen genutzt werden. Zudem kann die Konformität zwischen dem Konfigurationsraum und den Konfigurationen überprüft werden.
- **Vorschlagen von Geschäftsmodell-Pattern:** Auf Basis der gewählten Geschäftsmodellelemente können zusätzliche Elemente empfohlen werden, welche zur Nutzung eines Geschäftspatterns notwendig sind. Zudem können mögliche Erfolgskriterien und Schwachstellen eines Geschäftsmodells analysiert werden.
- **Vergleichen von Geschäftsmodellen:** Zuletzt können zudem die Beispielorganisationen aus dem Expertenwissen genutzt werden, um einen kompetitiven Vorteil der eigenen Organisation herauszustellen. Diese Beispiele können zudem genutzt werden, um vergleichbare Geschäftsmodelle in anderen Märkten zu identifizieren.

Durch die Nutzung des Expertenwissens in unterschiedlichen Phasen kann der Nutzer dabei ganzheitlich bei der Entwicklung neuer Geschäftsmodelle unterstützt werden. Weiterhin kann neu erlangtes Wissen innerhalb der Entwicklung auf für die zukünftige Entwicklung der Geschäftsmodelle gespeichert werden.

5 Zusammenfassung und Ausblick

Ziel dieses Beitrages war es, dem Leser einen Überblick über die datengetriebene Geschäftsmodellentwicklung zu geben. Hierzu haben wir allgemeine Hintergrundinformationen zu Geschäftsmodellen sowie Geschäftsmodellentwicklungstools zur Verfügung gestellt. Weiterhin haben wir die Beziehungen zwischen Daten, Informationen und Wissen erläutert und einige Ansätze aus der Literatur auf diese Beziehungen hin analysiert. Darauf aufbauend haben wir mit dem BDML Feature Modeler ein eigenes Software-Tool vorgestellt, welches das Wissen von verschiedenen Experten konsolidiert und mit diesem den Geschäftsmodellentwicklungsprozess unterstützt.

Innerhalb der Anfertigung dieses Überblicks haben wir dabei herausgefunden, dass es sich bei der datenbetrieben Geschäftsmodellentwicklung um ein noch relatives junges Forschungsfeld handelt, zu welchem allerdings schon einige unterschiedliche Ansätze entwickelt wurden. Hierbei sehen wir das Potenzial das in Zukunft Geschäftsmodelle auf Basis von internen und externen Unternehmensquellen effizienter und effektiver als bisher entwickelt werden können. Allerdings gibt es derzeit noch keinen systematischen Überblick über die verschiedenen Tools zur datengetriebenen Geschäftsmodellentwicklung. Hierdurch bauen die bestehenden Ansätze zum größten Teil nicht auf den Erkenntnissen der anderen Ansätze auf, was eine Weiterentwicklung hin zu effektiveren Tools erschwert. Weiterhin sind die in den Ansätzen entwickelten Tools oftmals nur in abstrakten Design Prinzipien beschrieben und die Tools nicht frei verfügbar,

sodass die gewonnenen Erkenntnisse noch schwer in der Forschung und Praxis weiterverwendet werden können.

Aus diesem Grund sind wir derzeit dabei mittels einer systematischen Literaturanalyse einen Gesamtüberblick zur daten-gestützten Geschäftsmodellentwicklung zu erhalten. Aus den gefundenen Ansätzen werden wir dabei eine Referenzarchitektur für entsprechende Tools entwickeln und ein Mächtigkeitsmodell über deren Entscheidungsunterstützung definieren. Dies wiederum soll sowohl Forscher als auch Praktiker dabei unterstützen, ihre Geschäftsmodelentwicklungstools zu verbessern.

Literaturverzeichnis

1. Osterwalder, A.; Pigneur, Y.: Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers. Wiley, Hoboken, 2010.
2. Ebel, P.; Bretschneider, U.; & Leimeister, J: Leveraging virtual business model innovation: A framework for designing business model development tools. Information Systems Journal 26(5), S. 519-550, 2016.
3. General Electric Inc.: GE Global Innovation Barometer 2018, General Electric Inc., 2018.
4. Teece, D.: Business Models, Business Strategy and Innovation. Long Range Planning 43 (2-3), S. 172–194, 2010.
5. Chesbrough, H.: Business Model Innovation: Opportunities and Barriers. Long Range Planning 43 (2-3), S. 354–363, 2010.
6. Eppler, M.; Hoffmann, F.; Bresciani, S.: New business models through collaborative idea generation. International Journal of Innovation Management 15 (06), S. 1323–1341, 2011.
7. Dorst, K: The core of 'design thinking' and its application. Design Studies 32 (6), S. 521-532, 2011.
8. Gassmann, O.; Frankenberger, K.; Csik, M.: The Business Model Navigator: 55 Models That Will Revolutionise Your Business. Financial Times Prent, 2014.
9. Bouwman, H. et al.: Business model tooling: where research and practice meet. Electronic Markets 30 (3), S. 413–419, 2020.
10. Szopinski, D. et al.: Software tools for business model innovation: current state and future challenges. Electronic Markets 60 (11), S. 469-494, 2019.
11. Canvanizer: Brainstorm better concepts. Together with your team, URL: https://www.canvanizer.com
12. Schoormann, T.; Behrens D. Knackstedt, R.: The Noblest Way to Learn Wisdom is by Reflection: Designing Software Tools for Reflecting Sustainability in Business Models. Proceedings of the ICIS, AIS, 2018.
13. Fleig, C.; Augenstein, D.; Maedche, A.: Tell Me What's My Business - Development of a Business Model Mining Software. Information Systems in the Big Data Era, Springer, S. 105–113, 2018.
14. Augenstein, D.; Fleig, C.; Maedche, A.: Development of a Data-Driven Business Model Transformation Tool. Designing for a Digital and Globalized World, Springer, S. 205–217, 2018.
15. Gottschalk, S.; Rittmeier, F., Engels, G.: Hypothesis-driven Adaptation of Business Models based on Product Line Engineering. Proceedings of the 22nd Conference on Business Informatics (CBI), IEEE, S. 134–143, 2020.
16. Casadesus-Masanell, R; Ricart, J.: From Strategy to Business Models and onto Tactics. Long Range Planning 43 (2-3), S. 195–215, 2010.

17. Al-Debei, M.; El-Haddadeh, R.; Avison, D.: Defining the Business Model in the New World of Digital Business. Proceedings of AMCIS, AIS, 2008

18. Gordijn, J.; Akkermans, H.: Designing and evaluating e-business models. IEEE Intelligent Systems 16 (4), S. 11–17, 2001.

19. Moellers, T. et al.: System dynamics for corporate business model innovation. Electronic Markets 29 (3), S. 387–406, 2019.

20. Fritscher, B.; Pigneur, Y.: Supporting Business Model Modelling: A Compromise between Creativity and Constraints. Task Models and Diagrams for User Interface Design, Springer, S. 28–43, 2010.

21. Dellermann, D.; Lipusch, N.; Ebel, P.: Developing Design Principles for a Crowd-Based Business Model Validation System. Designing the Digital Transformation, Springer, S. 163–178, 2017.

22. Osterwalder, A.; Pigneur, Y.: Designing Business Models and Similar Strategic Objects: The Contribution of IS. Journal of the Association for Information Systems 14 (5), S. 237–244, 2013.

23. Gottschalk, S.; Yigitbas, E.; Engels, G.: Model-Based Hypothesis Engineering for Supporting Adaptation to Uncertain Customer Needs. Business Modeling and Software Design, Springer, S. 276–286, 2020.

24. Gottschalk, S.; Kirchhoff, J.; Engels, G.: Model-Based Hypothesis Engineering for Supporting Adaptation to Uncertain Customer Needs. Business Modeling and Software Design (erscheint), Springer, 2021.

25. Schoormann, T. et al.: Towards Aligning Business Models with Business Processes: A Tool-based Approach. Proceedings of Modellierung 2020, GI, S. 12-27, 2020.

26. John, T.: Supporting Business Model Idea Generation Through Machine-generated Ideas: A Design Theory. Proceedings of the ICIS, AIS, 2016.

27. Lüdeke-Freund, F. et al.: Research on Sustainable Business Model Patterns: Status quo, Methodological Issues, and a Research Agenda. Sustainable Business Models, Springer, S. 25–60, 2019.

Kritische Betrachtung agiler Transformation am Beispiel von Scrum: Perspektiven und Faktoren mit Einfluss auf den Erfolg in der Praxis

Sarah Hochstrat, Oliver Linssen

ifid - Institut für IT-Management & Digitalisierung

FOM Hochschule für Oekonomie & Management gemeinnützige GmbH, Essen
Studienzentrum Düsseldorf

sarah_hochstrat@msn.com, oliver.linssen@fom.de

Abstract: Agile Vorgehensweisen haben sich in der Praxis, vor allem im Projektumfeld etabliert. Um die Vorteile der Agilität bezüglich Überlebensfähigkeit, Flexibilität, Innovationsfähigkeit oder Geschwindigkeit auch in die Organisation zu tragen, durchlaufen Unternehmen eine agile Transformation. Trotz einiger Erfolge existieren offensichtlich Hürden, woran Transformationen scheitern. Diese Arbeit untersucht mittels aktueller Literatur und durch eine Datenerhebung anhand von Interviews in einem praktischen Fall die Probleme der agilen Transformation. Es werden insbesondere die Faktoren des Scheiterns einer Transformation herausgearbeitet und es wird ein Ansatz entwickelt, der zur Früherkennung fehlerhafter Durchführungen dienen soll.

1 Einleitung

Mit der Idee, Agilität im Projektmanagement einzuführen, sollte eine Art „Revolution" herbeigeführt werden, um den Prozess für die sich ständig ändernden Anforderungen der heutigen Zeit anzupassen [33]. Die Autoren des Agilen Manifests wollten „*bessere Wege erschließen, Software zu entwickeln*" [1]. Doch die Vorgaben des Agilen Manifests [1] und des Scrum Guides [62] mit den dort versprochenen Vorteilen in Bezug auf Zusammenarbeit, Kommunikation, besserer Reaktion auf Veränderung und Funktionalität [33] können dazu führen, dass Theorie und Praxis auseinanderklaffen: "*But the reality is troubling, because much of what is done is faux-agile, disregarding agile's values and principles*" [18].

Wenn Agilität und die damit angestrebten Vorteile auch auf der Ebene des gesamten Unternehmens angewendet werden sollen, wie es bereits bei einigen Unternehmen zu sehen ist [5], [48], [56], erweitert sich das denkbare Fehlerpotential [61], [56], [19]. Der Vorgang der Transformation und ihr Erfolg kann in unterschiedlichen Unternehmen und in unterschiedlichen Branchen [19] variieren. Das wissenschaftliche Schrifttum liefert aktuell nur wenige für die Praxis wertvolle Antworten, wie eine agile Transformation erfolgreich werden kann [15], [20], [38].

1.1 Ziel der Veröffentlichung

Ziel diese Arbeit[1] ist, die Gründe für das Scheitern agiler Transformationen zu analysieren. Dabei sollen die in der Literatur genannten Vorgaben, was zu einer erfolgreichen agilen Transformation gehört, mit den genannten Ursachen des Scheiterns bzw. mit praktischen Ansätzen verglichen werden. Es werden Faktoren des Scheiterns herausgearbeitet, deren Beachtung die Wahrscheinlichkeit des Erfolgs einer agilen Transformation erhöhen können.

Untersucht werden soll weiterhin, welche Faktoren dazu führen, dass agile Transformationen in der Praxis scheitern und wie sich der Prozess des Scheiterns frühzeitig erkennen und positiv beeinflussen lässt.

[1] Dieser Aufsatz ist eine Kurzfassung von [31].

1.2 Aufbau der Veröffentlichung

Kapitel 2 beschreibt auf Basis vorhandener Literatur das Konzept der agilen Transformation, welche den agilen Ansatz von der Projekt- auf die Unternehmensebene überträgt. Als erster Teil der hier vorgestellten Untersuchung enthält Kapitel 3 eine Literaturanalyse der Anforderungen an den Erfolg einer agilen Transformation. Dabei werden kritische Erfolgsfaktoren definiert und die potentiellen Ursachen eines Scheiterns herausgearbeitet. Nach einer einführenden Analyse praxisorientierter Literatur beschreibt Kapitel 4 die hier durchgeführte Untersuchung der agilen Transformation in einem konkreten Unternehmen. Insbesondere werden dabei die Merkmale von Erfolg und Misserfolg in der Praxis bestimmt. Kapitel 5 führt die Erkenntnisse zusammen und entwickelt daraus Misserfolgsfaktoren und ein Modell zur Kontrolle der Erfolgstendenz während der Transformation. Kapitel 6 fasst die Ergebnisse zusammen und leitet daraus eine Empfehlung ab.

2 Die Idee, den agilen Ansatz auf die Unternehmensebene zu übertragen

Bei allen Unterschieden im Detail kann man agiles Projektmanagement wie folgt charakterisieren:

„*Agilität wird primär als Vorgehensmodell betrachtet, welches sich auf die Maximierung des Kundennutzen fokussiert. Im Rahmen eines iterativen Prozesses werden regelmäßig fertige Produkt ausgeliefert. Das Feedback des Kunden wird dazu verwendet, sich immer weiter zu entwickeln, zu verbessern und gleichzeitig unnötigen Aufwand zu reduzieren.*“ (nach [31, S. 4], [5], [19])

Überträgt man diese Charakterisierung der Agilität auf Unternehmensebene, soll dies Unternehmen ermöglichen, flexibel und transformierbar zu sein [5], [36]. Es soll helfen, starre Strukturen und Prozesse in Unternehmen aufzulösen, die sich am Wasserfallmodell (im Sinne von linearen Prozessen ohne iterativ-inkrementelle Komponente) orientieren [22], [55], [56].

Die versprochenen Vorteile sind vielfältig. Sie reichen von besserer Geschäftsunterstützung durch die IT, verbesserten Umgang mit sich ändernden Prioritäten und kontinuierlich durchgeführte Verbesserungsprozesse [2] bis hin zu langfristiger Anpassungsfähigkeit und Innovationsfähigkeit [56]. Nach [36] werden ökonomische (finanzielle Flexibilität), operationelle (Produktionsflexibilität), organisationale (Organisationsmodell, Menschen und Arbeitsflexibilität) oder strategische Dimensionen (Kultur, Leadership und Dynamik) damit erreicht.

Was als Scheitern oder Erfolg betrachtet wird, liegt durchaus im Auge des Betrachters. Es hängt von der jeweiligen Blickrichtung auf die Transformation ab, was als Erfolg gilt. Projekte müssen in erster Linie die vom Kunden festgelegten Ziele erfüllen. Erfolgreiche Unternehmen erreichen durch Agilität eine dauerhafte Anpassungsfähigkeit gegenüber den Anforderungen ihrer Umwelt. Dabei soll nach Perkin [56] ein Gleichgewicht zwischen dem alltäglichen Betrieb und Veränderungen gehalten werden (“*balanced approach to exploitation and exploration*“ [55, S. 86]) und gleichzeitig soll festgelegt werden, wie weit die Transformation in das Unternehmen vordringt (siehe z.B. [44]).

3 Anforderungen an den Erfolg agiler Transformation in der Literatur

3.1 Vorgehen

Für die Untersuchung der erfolgsbedrohenden Merkmale wird ein Systematic Literature Review nach von Brocke [7] durchgeführt. Relevante Quellen wurden in folgenden Datenbanken

recherchiert: IEEE, ACM, WISO, BASE, Emerald Insight, ScienceDirect, EBSCO, SpringerLink, Researchgate, Cornell University Library und Google Scholar. Als Suchbegriffe wurden unter anderem verwendet: „Scrum“, „project management“, „agile software development“, „agile Softwareentwicklung“, „Projektmanagement“, „agile transformation“, „agile adoption“. „scrum anti pattern“, „fail“, „failure“. Selektiert wurden Quellen, die sich entweder mit den Merkmalen einer erfolgreichen agilen Transformation beschäftigten oder mit Quellen, die sich kritisch mit agiler Transformation auseinandersetzten. Nicht betrachtet wurden Quellen, die sich mit der agilen Transformation jenseits von Scrum beschäftigen.

Im ersten Schritt der Untersuchung werden die Anforderungen an eine erfolgreiche agile Transformation aus verschiedenen Blickwinkeln betrachtet.

Um dieses Ziel zu erreichen, werden im ersten Untersuchungsabschnitt Aussagen zu Anforderungen für den Erfolg betrachtet. Anhand dieser Anforderungen werden im nächsten Schritt Erfolgsfaktoren hergeleitet. Am Ende werden diese Faktoren den Aussagen aus der Literatur bezüglich des Scheiterns agiler Transformationen und deren Ursachen gegenübergestellt.

3.2 Anforderungen aus verschiedenen Perspektiven

Wie bereits festgestellt, wird sich ein erfolgreiches Unternehmen dauerhafte Anpassungsfähigkeit gegenüber seiner Umwelt aneignen. Doch der Weg zum Erfolg ist für viele Unternehmen unklar. Aus der Quellenanalyse wurde deutlich, dass Anforderungen aus unterschiedlichen Richtungen auf die Transformation einwirken. Zur Orientierung wurde im Laufe der Analyse das Framework in Abbildung 1 entwickelt, welches sich auf drei Blickrichtungen konzentriert.

Die erste Blickrichtung (in der Abbildung mit 1 gekennzeichnet) stammt aus der Umwelt und ist auf die Organisation gerichtet. Dies beinhaltet beispielsweise die Gesetzgebung und die Gesellschaft mit ihren Anforderungen zur Einhaltung von Werten, Normen und Gesetzen, aber auch den Kunden [42]. Der Kunde kann unter anderem fordern, dass er eine höhere Wertschöpfung oder mehr Flexibilität vom Unternehmen erhält [5], [16]. Auch Anforderungen aus der Literatur wurden hier subsumiert. Der Scrum Guide [62], das Agile Manifest [1] und viele weitere Werke beinhalten in großer Zahl Vorgaben und Best Practices, die man einhalten soll. Dies betriff die Methodik, das Management, die Technik, die Prozesse, bis hin zu sozialen Aspekten [5], [27], [44], [55], [56]. Beispiele hierfür sind die Einhaltung von motivationalen (z.B. Leidenschaft, Selbstverpflichtung, Humor etc.), methodischen (Transparenz, Fokus, Kreativität etc.) oder sozialen Werten (Respekt, Empathie etc.), die man in Werken wie [5], [9], [44], [49], [63], [62] in verschiedener Ausprägung finden kann. Eine weitere Forderung ist der Kulturwandel, der für die agile Transformation als unabdingbar scheint [5], [17], [36], [56].

Die externen Anforderungen lösen innerhalb des Unternehmens neue Anforderungen vom Management an die Organisation aus. Dies ist die zweite Blickrichtung (in der Abbildung mit 2 gekennzeichnet), die man vereinfacht als „hierarchisch von oben nach unten“ bezeichnen kann. Dies betrifft zumeist die Elemente, die mit der anzustrebenden Agilität direkt oder indirekt in Berührung kommen, wie Scrum-Projekte und ihre Schnittstellen. Den Anforderungen zu sichtbarem Geschäftserfolg mit Aussicht auf höheren Profit, Nachhaltigkeit, verringerten Risiken, Prognosesicherheit, Geschwindigkeit etc. [5], [13] folgen weiterführende Überlegungen, was alles agil werden muss, um als Unternehmen agil zu sein [56]. Dies erfordert in den betroffenen Bereichen Wissen und Fähigkeiten im Bereich Technologie, Methodik, Social Skills, die auf unterschiedliche Rollen verteilt werden sollen [23], [30], [40], [42], [55], [54], [70].

Aus diesen Anforderungen des Managements entstehen schließlich neue Anforderungen der ausführenden Einheiten nach oben und nach außen, was hier als dritte Blickrichtung (in der Abbildung mit 3 gekennzeichnet) verstanden werden soll. Egal, ob Projektteams, Fachbereich, Tester, Entwickler oder Architekten - um die Anforderungen von Unternehmen und Umwelt erfüllen zu können, wird eine Grundlage benötigt. Hier wird nach [31] das Management, die

Organisation, Technologie, Methodik und, sofern es möglich ist, die Umwelt gleichermaßen gefordert. Die passenden Rahmenbedingungen sind hier eine weitere zu erfüllende Bedingung [27], [21], [26], [42]. Management muss hier als Enabler fungieren, um passende Schnittstellen, Prozesse, Kooperationen, Werkzeuge etc. für das agile Arbeiten zur Verfügung zu stellen (siehe z.B. [5], [6], [10], [15], [56]).

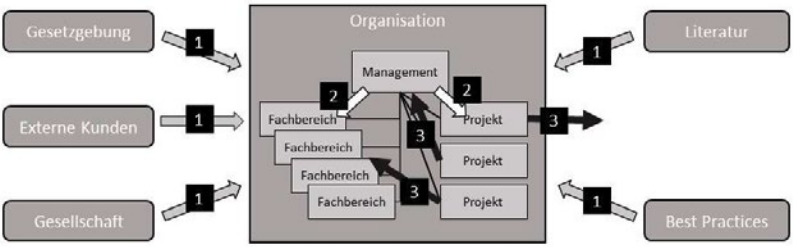

Abbildung 1: Perspektiven in- und außerhalb einer Organisation. Angelehnt an [31, S. 16].

3.3 Definition kritischer Erfolgsfaktoren

Eine bloße Aufzählung der vielfältigen Anforderungen ist für den Erfolg nicht hinreichend, wenn man diese nicht einordnet und damit strukturiert. In der Literatur finden sich eine Vielzahl von Vorschlägen für Erfolgsfaktoren, von denen hier nur einige genannt werden können:

Bates und Yates [3] verwenden Faktoren aus dem sozialen und umwelttechnischen Bereich, während Hasebrook et al. [28] den sog. „Diamanten der agilen Organisation" vorstellen. Bonk und Hedfeld [6] unterscheiden zwischen der unternehmenskulturellen, prozessualen und organisationsstrukturellen Dimension. In Iqbal et al. [34] verweisen die Autoren auf das „Six-point star-model" aus dem PMBOK4.0. Dieser Detailgrad kann sich noch weiter steigern, denn Paterek [55] benennt bis zu zwölf Bereiche besonderer Aufmerksamkeit. Campanelli et al. [8] beschreiben 23 Erfolgsfaktoren in fünf Gruppierungen. Dikert et al. [15] ordnet 29 Erfolgsfaktoren in 11 Kategorien ein. Weitere Ansätze kann man in [51], [50], [60], [54] und in vielen weiteren Quellen finden, die in [31] untersucht wurden.

Um die Vielzahl der Ansätze zu konsolidieren und zu strukturieren, werden sie den von Hamdani und Butt (vgl. [27] und in Abbildung 2 verwendeten Erfolgsfaktoren „Projekt", „Menschen", „Technologie", „Organisation", „Prozesse", „Management" und „Umwelt" zugeordnet.

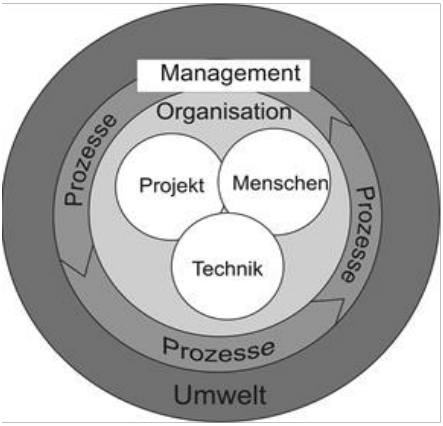

Abbildung 2: Erfolgsfaktoren agiler Transformation. Nach [27, S. 983].

- Projekt: Die Aufgabe von Projekten besteht darin, ihre projektspezifischen Ziele zu erreichen. Daher müssen die in Projekten verwendeten Methoden zur Aufgabenstellung passend gewählt sein, sowie passend technische Ressourcen und Humanressourcen eingesetzt werden. Der Erfolg in Projekten hängt stark von den Rahmenbedingungen ab, in denen sich das Projekt befindet. Vgl. [21], [34], [41], [44], [55], [68].
- Menschen: Menschen sind die wichtige Ressource von Projekten und Unternehmen, weil von ihrem Wissen und ihren Fähigkeiten der Projekterfolg abhängt. Dabei muss sowohl die Individualität dieser Ressource beachtet werden, als auch muss ihre Motivation auf hohem Niveau erhalten werden. Dies betrifft nicht nur Angestellte, sondern alle beteiligten Individuen. Vgl. [15], [21], [30], [41], [44], [68], [69].
- Technik: Technik erleichtert die Aufgabenerfüllung und -organisation. Von Management- und Entwicklungssystemen über Werkzeuge zur Kommunikation und Kollaboration bis hin zu Werkzeugen für den Betrieb sollte für agile Projekte und die agile Organisation die benötigten Werkzeuge angeschafft werden. Vgl. [21], [41], [44], [53], [68].
- Umgebung/Organisation: Die Organisation hat großen Einfluss auf den Verlauf der Transformation. Die Menschen, Projekte und Technologien sind in einer Umgebung bzw. einer Organisation eingebettet. Diese Umgebung bzw. die Organisation reicht vom direkten Projektumfeld hin zu anderen Unternehmensstandorten, den übrigen Elementen der Aufbauorganisation bis zur Unternehmenskultur. Je mehr die Agilität in die Organisation eingebettet ist, desto einfacher sind die darin ablaufenden Prozesse und die Arbeitsumgebung der Mitarbeiter zu gestalten. Vgl. [8], [17], [20], [40], [41], [42], [44], [55], [56].
- Prozesse: Kommunikations-, Kooperations- und Kollaborationsprozesse müssen eingeführt werden, deren Transparenz, Frequenz der Ausführung und Inhalt dem agilen Ansatz entsprechen. Dies wird durch ein Wissensmanagement und agilen Werten und Methoden unterstützt. Es muss sichergestellt sein, dass die Prozesse flexibel angepasst werden können und ein kontinuierlicher Verbesserungsprozesses existiert. Außerdem ist es essentiell, beim Gestalten von neuen Prozessen ein Rahmenwerk zu verwenden, welches zu den Anforderungen passt. Vgl. [5], [10], [15], [21], [28], [32], [54], [55], [56], [68], [69], [70].
- Management: Auf jeder Ebene der Aufbauorganisation muss sich auch das Management entsprechend der agilen Methodik anpassen. Es kann und muss hinter der Transformation

stehen und diese unterstützen. Dazu gehört ein agiler Führungsstil, in dem auch Motivation, sowie die Strategie- und Visionsbildung berücksichtigt wird. Gerade das Management in den oberen Ebenen der Aufbauorganisation ist aufgefordert Rahmenbedingungen (Finanzen, Risiko, Kultur, Umgebung etc.) zu gestalten und ein zum Unternehmen stimmendes agiles Changemanagement anzustoßen. Vgl. [15], [28], [42], [56], [68].

- Umwelt: Anhand der Erfüllung der Anforderungen externer Stakeholder kann bestimmt werden, wie erfolgreich sich ein Unternehmen am Markt behaupten kann. Gerade hier steht die Einbindung der Kunden in die agilen Prozesse. Eine agile Umgebung erfordert agile Unternehmen und umgekehrt. Vgl. [21], [41], [42], [55].

3.4 Ursachen des Scheiterns

Mit *"Scrum is broken"* [65] fasst Mike Sutton die Situation in manchen Unternehmen zusammen. In keiner untersuchten Quelle wurde die Frage, warum agile Transformationen scheitern, zufriedenstellend beantwortet. Deshalb wurden in einer weiteren Untersuchung insgesamt 53 Quellen auf Ursachen, Mängel, Antipattern oder im Umkehrschluss auf Lessons Learned und Empfehlungen untersucht, systematisiert und unter Schlagworten zusammengefasst. Das Ergebnis dieser Systematisierung in Form einer Übersicht ist in Tabelle 1 zu sehen.

Ursachen des Scheiterns	Ausprägungen gängiger Fehler	Quelle
Unzureichende Top-Down Führung bei Bottom-Up-Ansatz	- Nicht-agile/fehlerhafte Denkweise: Management verharrt in plangetriebenen Strukturen; Leadership hört nicht auf, wenn Team reif genug ist; zu viel Kontrolle; starker Fokus auf Effizienz und Ausführung; mangelndes Vertrauen. - Einführung aus falschen Gründen (z.B. aufgrund eines Trends bzw. einer Mode); Interpretation von Agilität als Allheilmittel. - Fehlerhafte Durchführung: Keine klaren Ziele und Vorstellung; fehlende bzw. nicht verständliche Strategie in der Transformation; kein Erzeugen des Gefühls der Dringlichkeit und Zweck der Veränderung; kein Hervorheben des Stellenwerts der Auswirkungen der Veränderung; keine Beachtung der Komplexität des Veränderungsprozesses (Investition, Anstrengung, Zeitaufwand); kein Vor-Leben; Versuch die Verantwortung der Transformation nach unten zu delegieren. - Fehlende Unterstützung: Keine Maßnahmen gegen Veränderungsresistenz im Management und in den darunterliegenden Ebenen; keine Sinnstiftung; kein Engagement des Top-Managements. - Politische Machtkämpfe.	[5], [9], [15], [18], [17], [20], [28],[27], [30], [33], [44], [47], [48], [56], [58], [64], [70]
Bottom-Up-Ansatz stößt an Grenzen	- Fehlende Befugnisse und Freiheiten der Mitarbeiter, Maßnahmen umzusetzen; treibende Mitarbeiter stoßen an interne hierarchische Grenzen.	[6], [15], [67]
Partielle /unvollständige Transformation	- Fehlende Integration von Menschen: Problem bei der Integration von Nicht-Entwicklern in die Transformation. - Unvollständige Übernahme von Agilität: Einsatz nur ausgewählter Prinzipien und Methoden, Einführung nur dann, wenn es opportun erscheint; Vermischung aus agilen und plangetriebenen Prinzipien; Konflikte an Schnittstellen; ungeeignete Mischung von klassischen und agilen Methoden, Verfahren, Ansätzen. - Anpassung der Agilität: Veränderungen nur auf dem Papier; Anpassung der Methodik statt Anpassung des Unternehmens. - Transformation wird als abgeschlossen betrachtet.	[15], [18], [47], [62], [64]
Big-Bang-Transformation	- Einführung von Agilität mit Zwang und ohne Vorlaufzeit.	[5], [18], [56], [64]

Fehlinterpretationen und Probleme mit agilen Vorgaben und Prinzipien	- Fehlinterpretation: Einführung streng nach Buch vs. Missachtung von Prinzipien; Interpretation als „Kochrezept“ oder als Managementansatz oder als technischer Ansatz; Irrglauben, Agilität bestehe nur aus dem agilem Manifest und seinen Prinzipien; unterschiedliche Umsetzung und Interpretation in verschiedenen Teams. - Verwendung unausgereifter Lösungen oder unvollständigen Lösungen: Einhalten von unvollständigen Anleitungen aus der Literatur. - Schlechte Umsetzung: Unzureichende Umsetzungen der Prinzipien; zu bürokratische und kontrollbasierte Umsetzung der Methoden; zu wenig kontinuierlicher Verbesserungsprozess und Einschleichen von Anti-Pattern. - Betrachtung von Agilität als Allheilmittel.	[12], [15], [18], [19], [20], [23], [28], [30], [32],[33], [35], [38], [47], [53], [50], [59], [64]
Unpassende Organisationsstrukturen und -kulturen	- Nicht-agile Denkweise: Striktes Festhalten an alten Paradigmen, Verhalten und Prozessen (gerade auf höheren Entscheidungsebenen); fehlende Selbstreflexion. - Unpassende Rahmenbedingungen und unpassende Kultur: Bürokratie; zentrale Steuerung; Bereichsegoismus; Hierarchisch orientierte Organisation; Stellenwert einer Kulturveränderung wird ignoriert; Schwerfälligkeit von Veränderungsprozessen wird ignoriert; alte Rollenbilder werden nicht überdacht; Kontrollkultur dominierend.	[5], [9], [12], [15], [17], [28], [27], [33], [35], [36], [47], [53], [55], [56], [58], [61], [67]
Unpassende Projektstrukturen und -umfeld	- Unpassende Rahmenbedingungen: Viele lineare Projektstrukturen innerhalb und außerhalb agil geplanter Projekte; Wechsel zwischen Projekten und Aufgabe aus der Matrixorganisation; räumlich verteilte Projektmitglieder (evtl. mit Zeitunterschieden); Arbeitsumgebung der Teams ist an alten Prozessen ausgerichtet; Abwesenheit der Kunden; keine cross-funktionalen Teams; unpassende Teamgrößen. - Nicht-agile Denkweise: Denken in Projekten statt in Produkten. - Unklare Führung: Agile Prozesse werden nicht klar definiert.	[15], [18],[27], [44], [47], [55], [57], [58], [67]
Unzureichendes Wissensmanagement	- Fehlende Wissensressourcen: Fehlendes Wissen und Erfahrung auf allen und insb. den entscheidenden Ebenen bezüglich Agilität; Angst vor fehlenden Skills; Angst vor Arbeitsplatzverlust; fehlende Social Skills; Wissenssilos. - Ausbildungsmängel: Mangel an Coaches und externer Unterstützung; unpassende Trainings und Schulungen; wenig vorliegendes Wissen bei den einzelnen Rollen bezüglich Methodik, Prozesse, Prinzipien etc. (auf allen Ebenen); kein Coaching auf strategischer, operationaler und taktischer Ebene; Kein Lernen aus negativen Erfahrungen	[15], [22], [24],[27], [47], [55], [56], [67], [70]
Unzureichendes Kostenmanagement, Controlling und Risikomanagement	- Unzureichendes Kostenmanagement: Unsicherheit korreliert mit Kosten; Kosten- statt Kundenfokussierung; fixe Budgets. - Keine oder fehlerhafte Erfolgsmessung: Fehlende klare Metriken bzw. fortgeführte Verwendung von veralteten und unpassenden Metriken; keine Messungen; fehlende Erfolgskontrolle des Veränderungs- und Verbesserungsprozesses. - Keine Qualitätssicherung: Unzureichende Qualitätssicherungsmechanismen (z.B. fehlende Testautomatisierung). - Kein oder übervorsichtiges Risikomanagement (Experimente sind unzulässig).	[15], [26], [34], [47], [55], [56], [64]
Mangelhafte Kommunikation und Kollaboration	- Mangelhaftes Feedback: Keine Feedback-Zyklen; demotivierendes Feedback zur falschen Zeit. - Keine oder zu geringe Kommunikation und Kollaboration: Mängel in teaminterner und team-übergreifender Kommunikation; fehlende Kommunikation mit Kunden; fehlende Kommunikation mit Scrum Master oder Product Owner; fehlende Kommunikation mit dem Management; fehlende Einbindung der Anwender. - Zu viele Geheimnisse: Fehlende Transparenz bei aktuellen Problemen, bei Vorhaben und bei erfolgten Fortschritten.	[5], [15], [18], [24], [26],[27], [55], [58], [59]
Keinen Support	- Mit Interessenten an der Transformation (Kunden, Mitarbeiter, mögliche externe und interne Unterstützer) wird nicht gesprochen; keine Management-Unterstützung (sowohl oberes als auch unteres Management); Bottom-Up-Ansatz stößt an hierarchische Grenzen.	[5], [6],[27], [47], [57], [64]

Fehlende Humanressourcen	- Fehlendes Personal; fehlende Experten.	[11], [57], [67]
Schlechte technologische Ressourcen und Umgang	- Unpassende Rahmenbedingungen: Veraltete und stark vernetzte Unternehmensarchitektur; Altsysteme von mehreren Teams abhängig. - Unpassender Umgang und unpassender Einsatz von Technologie: Probleme bei der Beibehaltung technischer Konsistenz; fehlende Unterstützungssysteme und Werkzeuge; Werkzeuge und Systeme passen nicht zum agilen Ansatz; nicht vorhandene technische Exzellenz.	[18], [27], [53], [57], [64]
Schlechter Umgang mit Anforderungen	- Hoher Zeitdruck; kein ausreichendes Anforderungsmanagement.	[12], [15], [57], [59]
Ungünstige Startposition	- Keine Pilotprojekte: Schlechte bis unzureichende Auswahlkriterien und Rahmenbedingungen für die ersten agilen Projekte.	[52], [56]

Tabelle 1: Ursachen des Scheiterns agiler Transformation. Zusammenfassung der Literaturanalyse in [31, S. 34 ff].

Die in [31] durchgeführte Literaturanalyse ergab, dass in den gesichteten Quellen auf vielfältige Fehler hingewiesen wird, die während der agilen Transformation gemacht werden können. Schäden können entstehen, wenn die Rahmenbedingungen in der Organisation nicht auf die agile Arbeit ausgerichtet sind. Dies geschieht meist durch Desinteresse, Unwissenheit, fehlendes Sicherheitsgefühl oder auch durch Gewohnheit. Dadurch mangelt es oft im Management und bei den Mitarbeitern an vorantreibenden Kräften und an gezielten Aktivitäten, um passende Rahmenbedingungen zu schaffen. Darunter leiden Führungsansätze, Organisations- und Projektstrukturen, Kommunikations- und Arbeitsprozesse, Ressourcenbereitstellungen und die Motivation, sich mehr in Richtung Agilität zu bewegen.

Die in der Tabelle 1 genannten Ursachen des Scheiterns (siehe erste Spalte) besitzen einen direkten Zusammenhang mit den Erfolgsfaktoren des vorangegangenen Unterkapitels 3.3, denn durch diese Ursachen weicht man vom Weg zum Erfolg ab. Ein Beispiel hierfür ist eine unzureichende Führung. Diese hält sich aus agilen Bestrebungen raus, denkt nicht agil genug oder blockiert möglicherweise die agile Transformation. Als Folge fehlen ausreichenden Ressourcen und die Organisations- und Projektstrukturen passen sich nur schwerfällig den neuen Anforderungen an. Als Folge ist die Organisation nicht in der Lage, agile Prozesse einzuführen. In dieser Umgebung sinken Bestrebungen zum Ziel und die Unterstützung der Menschen (Mitarbeiter, Kunden) kontinuierlich. Das kann zur allgemeinen Demotivationen, mehr Zwang und Verwirrung bis hin zur Aufgabe des Vorhabens führen.

Jede dieser Ursachen berührt einen oder mehrere Faktoren negativ. Das Management spielt hier eine große Rolle, das es die Rahmenbedingungen setzt und großen Einfluss auf die Mitarbeiter hat, was wiederum stark das Endergebnis beeinflusst.

4 Untersuchung der agilen Transformation in der Praxis: Literaturanalyse und Untersuchung eines praktischen Falls

4.1 Vorgehen

Um die Erkenntnisse der Literaturauswertung abzurunden, werden im weiteren Verlauf Praxisbeispiele untersucht. Im ersten Schritt werden Unternehmensbeispiele aus praxisorientierten Quellen und Erfahrungsberichte der Literatur auf Erfolgs- und Misserfolgsaspekte untersucht. Im zweiten Abschnitt werden mittels semistrukturierter, leitfadenorientierter Interviews Daten in einem Unternehmen erhoben, welches sich in einer agilen Transformation befindet. Die Inhalte der Interviews wurden durch eine qualitative Inhaltsanalyse nach Mayring [45] bzw. nach

Mayring und Fenzl [46] hinsichtlich Fortschritt und Gefährdungen des Fortschritts ausgewertet. Schwerpunkt dieses Kapitels ist herauszufinden, welche Mängel und Störfaktoren und ihre Ursachen eine agile Transformation in der Praxis stören können.

4.2 Beispiele erfolgreicher und nicht-erfolgreicher Transformationen in der Literatur

In der Vergangenheit gab es schon viele Versuche, Agilität in ein Unternehmen einzuführen. Einige waren erfolgreich bzw. kommunizierten Erfolg. Andere waren es nicht. Bekannte Beispiele für erfolgreiche Ansätze sind Amazon und Microsoft [14], die Haufe Group [4], Ericsson [52], Gap Inc. [25], Samsung Electronics [37] usw. Auch wenn diese Unternehmen unterschiedlichen Branchen und Bedingungen angehören, haben sie doch wichtige Gemeinsamkeiten. Zum Einem folgt auf den Bottom-up-Ansatz durch die Mitarbeiter die Top-down-Unterstützung durch das Management. Zum anderen haben diese Unternehmen ein neues Mindset etabliert und einen Organisationsstruktur- und Kulturwechsel durchlebt. Dazu gehört eine umfassende Kommunikation in allen Richtungen, eine strategische Verankerung der Agilität und ein adäquater Zeithorizont für die Veränderung, der Jahre betragen kann. Fehlerfrei sind diese Ansätze jedoch nicht. Neu ist, Fehler als Chance zum Lernen zu akzeptieren und Experimente willkommen zu heißen wie in [4], [14], [37].

Einige Beispiele für weniger erfolgreiche Transformationen finden sich bei General Electrics [14], Pyxis Technologies [66] oder in den Studien bzw. Erfahrungsberichten von O'Connor [11], Mann und Maurer [43] oder Krasteva und Ilieva [39]. Gründe für das Scheitern können je nach Unternehmen unterschiedlich gelagert sein, aber auch hier gibt es Gemeinsamkeiten Topdown Ansätze können an Mitarbeiter scheitern, wenn sie das agile Mindset nicht annehmen. Auch sorgt eine ständige Fluktuation von Wissensträgern bzw. schlechte Ausbildung und dadurch fehlende Expertise für Mängel bei den ausführenden Projekten. Zu große und standortmäßig verteilte Projektteams sorgen zudem für Schwierigkeiten bei der Umsetzung. Ebenfalls fehleranfällig ist die Arbeit mit zu vielen Altlasten wie Legacy Code oder den kontinuierlichen Wechsel von Vorgaben und Anforderungen von Projekten und von der Organisation, denen man nicht gewachsen ist.

Die Beispiele haben gezeigt, dass Projekt- und Unternehmenserfolg, menschliche Motivation und das Engagement des Managements oft einhergehen. Fehlerhafte Ansätze können zu Kosten, Aufgabe der Transformation bis hin – wie im Fall von General Electrics [14] – zur Abspaltung von Unternehmensteilen führen, während Erfolge es bis hin zu Marktwertsteigerungen bringen können. Natürlich werden lieber Erfolge kommuniziert. Die Veröffentlichung von Misserfolgen findet wesentlich seltener statt. Um hier zu weiteren Erkenntnissen zu gelangen, wurde ein konkretes Beispiel einer agilen Transformation intensiver untersucht.

4.3 Untersuchung der agilen Transformation in einem konkreten Fall

Im folgenden Beispiel wird ein (hier anonymisierten) Unternehmen im Bankensektor untersucht, das schon seit vielen Jahren besteht und bisher seine Projekte, seine Aufbauorganisation und seine Kultur an wasserfallartigen Ablaufstrukturen orientiert hat. Um das Jahr 2018 wurden erste Bestrebungen sichtbar, das gesamte Unternehmen einer agilen Transformation zu unterziehen, um sich den verändernden Herausforderungen der Umwelt (Kunden, Gesetzgeber und Wettbewerb) anzupassen. Ein großes Veränderungsprogramm zur Umstrukturierung der technischen Systemlandschaft wurde bewusst mit agilen Vorgehensweisen aufgesetzt, wobei Scrum verwendet wurde.

Die Untersuchung fand in Form von semistrukturierten, leitfadengeführte Experteninterviews statt, die mit einer Qualitativen Inhaltsanalyse nach Mayring [45], [46] ausgewertet wurden. Dazu wurden im Zeitraum Januar 2020 bis März 2020 acht Personen in unterschiedlichen Rol-

len (Programmleiter, Workstreamleiter, Scrum Master, Chief Product Owner, Fachlicher Tester, Architekt und Anwendungsentwickler) zu ihren Erfahrungen befragt. Im Mittelpunkt der Fragen stand:

- Die eigene Rolle in der agilen Transformation
- Erfahrungsschatz des Interviewten
- Betrachtungsweise der Agilität im Unternehmen
- Einschätzung zum Fortschritt des aktuellen Umsetzungsgrades
- Einschätzung zu Mängeln des aktuellen Umsetzungsgrades
- Hindernisse der agilen Transformation aus der Perspektive des Interviewten
- Verbesserungsansätze
- Vom agilen Ansatz abweichende Vorgehensweise im Unternehmen

Die Ergebnisse der Interviews aus [31] lassen sich wie folgt zusammenfassen: Das Unternehmen versucht, Agilität für sich als ein neues Gebiet zu erobern und befindet sich im Aufbruch zwischen den gewohnten, am Wasserfall orientierten Abläufen und dem neuen, unbekannten „agilen Neuland". Schon Jahre vor dem Veränderungsprogramm gab es vereinzelte Ansätze zu agilen Projekten, doch erst vor wenigen Jahren „entdeckte" das Management ihren Wert. Gesteigerte Marktfähigkeit bei schwindenden Branchenerlösen und preissensitiven Kunden, wartbarere Systeme, höhere Attraktivität für Mitarbeiter, Ablösung alter, langsamer bürokratischer Prozesse etc. sind Argumente für die Agilität. Alle acht Interviewpartner erachten die Veränderung deshalb als notwendig. Das daraufhin aufgesetzte Veränderungsprogramm sollte als „Motor unserer Entwicklung, unserer agilen Transformation" (Zitat Workstreamleiter) fungieren und als ein erster großer Schritt für das Unternehmen, stückweise agiler zu werden. Dies wurde Teil der neuen Unternehmensstrategie (nach Programmleiter, Workstreamleiter).

Im Laufe des Veränderungsprogramms wurden durchaus Fortschritte gemacht. Unter anderem passte man Prozesse und Regelwerke den neuen Anforderungen an (z.B. monatliche Sprint-Releases statt quartalsweiser Releases.) und etablierte Workstreams zur Unterstützung der Projekte in Sachen Skill-Aufbau und Infrastruktur. Positive Auswirkungen auf die Kommunikation oder der Fehlerminimierung bei Softwareeinsätzen konnten nach ca. zwei Jahren Laufzeit des Veränderungsprogramms festgestellt werden.

Doch die Transformation stieß an Grenzen. Viele Fachbereiche fühlten sich nicht „mitgenommen" und standen der Veränderung kritisch gegenüber. Auch das Management handelte – nach Aussage der Interviewpartner – noch zu wenig agil, um geeignete Rahmenbedingungen für eine agile Organisation zu schaffen. Die Einbettung des agilen Ansatzes in einer nicht-agilen Organisation schaffte unpassende Rahmenbedingungen, die sich durch Mängel bzw. durch Anzeichen für eine Veränderung in die falsche Richtung bemerkbar machten. In [31] wird geschlussfolgert, dass die Transformation besonders Schwachpunkte im Übergang von agilen zu nicht-agilen Einheiten besitzt. Von den Projekten über Fachbereichen bis hin zum Management existieren Probleme in der Kommunikation, Koordination, Umsetzung, Führung sowie Ressourcenverfügbarkeit zwischen den Ebenen. Dies drückt sich durch Konflikte und Kompromisse aus.

Die Tabelle 2 fasst die Erkenntnisse der Interviews zusammen und ordnet sie den Kategorien prozessuale, methodische und soziale Symptome des Scheiterns zu. Prozessuale Symptome beschreiben Anzeichen, die auf eine geringe Änderungs- und Verbesserungsgeschwindigkeit in der Transformation hindeuten, während die methodischen Symptome Konfliktpotential zu den agilen Prinzipien und Werten und der daraus resultierenden geringen Wertschöpfung darstellt. Soziale Symptome zeigen sich in den Anforderungen und Verhalten der beteiligten Personen.

Dies betrifft sowohl die persönliche Gedankenwelt als auch die Interaktion der Personen untereinander.

Symptome des Scheiterns	
Prozessual	- Keine schnellen/sichtbaren Fortschritte der Transformation. - Keine kontinuierliche Verbesserung sichtbar. - Release-Termine können nicht eingehalten werden. - Sich einstellende Zweifel an Erreichung des Ziels. - Unternehmen verharrt bei alten Verfahren. - Schnittstellenprobleme von und zu anderen Bereichen und Teams (Test, Release, Infrastruktur etc.). - Bruch zwischen Linien- und Projektorganisation. - Ressourcen für den Fortschritt fehlen. - Infrastruktur hinkt hinter den Anforderungen der Entwicklung hinterher. - Mitarbeiterseite strebt stärker nach Agilität als die Managementseite.
Methodisch	- Time-to-Market verkürzt sich nicht. - Gering geschätzte Reaktions- und Innovationsfähigkeit. - Fehlende Automatisierung: Keine automatisierten Tests, kein Continuous Integration, kein Continuous Deployment und kein Continuous Delivery. - Ineffektive Meetings: Langatmige Diskussion, geringer Informationsgehalt, Wiederholungen, Zeitverschwendung. - Ineffektive Sprintgrößen: Als zu lang wahrgenommen, zu kurzer Testzeitraum. - Prozesse: Länger andauernd als gewünscht, Hürden durch hohe Bürokratie z.B. bei Releases, Schwierige Koordination mit nicht-agilen Prozessen, fixe Termine, Angst vor Vorgaben aus nicht-agilen Bereichen. - Kommunikation und Kollaboration: Austausch findet selten statt und stoppt an Projektgrenzen, Probleme bei Kommunikation über mehrere Standorte hinweg, zu wenig Transparenz. - Projektorganisation: Wenige Teams zur Selbstorganisation in der Lage, zu große Teams. - Umsetzung: Probleme bei Interpretation von Story Points, Stories werden nicht umgesetzt, Skalierungs-Frameworks stoßen auf Ablehnung, Angst vor unvollständigen Ansätzen, Qualität, Aussehen oder Funktionalität fallen knappem Zeitbudget zum Opfer. - Ressourcen: Fehlende Humanressourcen fehlende technische Ressourcen, Angst vor Streichungen, Wissens- und Erfahrungslücken und mangelnde Verfügbarkeit. - Mindset: Arbeiten nach Wasserfallmodell wird vermisst, starke Fokussierung auf Team- statt auf Unternehmensziele.
Sozial	- Psychologie: Ängste, Sorgen und Ablehnung gegenüber agilem Mindset, wenig Vertrauen in geänderte Verfahren, wachsendes Gefühl von Fremdbestimmung, sinkende Einsatzbereitschaft in den Teams, Resignation in den Projekten und agile Interessierten, Abschreckung neuer Mitarbeiter. - Wissen: Diskrepanz zwischen Bereichen bezüglich Wissen, abhängig von der Beteiligung an den Projekten (bzgl. Sprache, Technik, Arbeitsweise). - Führung: Viele Führungskräfte führen weiterhin traditionell, wenig Verständnis für Toolbedarf, wenig Akzeptanz von Rollen und Verantwortungsbereichen, Kontrollmechanismen schränken Teams stark ein, agile Bereiche fühlen sich nicht ausreichend unterstützt.

Tabelle 2: Symptome des Scheiterns aus der Praxis. Zusammenfassung aus [31, S. 76 ff].

Diese Symptome haben Ursachen, die in den Experteninterviews herausgearbeitet wurden. Besonders auffällig ist, dass viele Symptome auf wenige Ursachen zurückgeführt werden konnten. Tabelle 3 fasst die Erkenntnisse über die Ursachen aus der Literaturanalyse und aus Beobachtungen der Autorin zusammen und ordnet sie den Ursachenkategorien Organisation, Management, Menschen, Changemanagement, Prozesse, Umwelt, Methoden und Ressourcen zu [31]. Dabei beeinflusst insbesondere das Management mit seiner Steuerung des Changemanagements zusammen mit der Umwelt die Rahmenbedingungen für die agile Transformation. Damit

werden auch die anderen Ursachen für Fehler ausgelöst. Es entsteht eine Kaskade, wo ein Missverständnis des Managements (z.B. Agilität als selbstlaufende Lösung anzusehen) unzureichende Budgets zu Folge haben kann, was dann unzureichende Infrastrukturen, fehlendes verfügbares Personal, mangelnde Fortbildung etc. zur Folge hat. Am Ende der Kette stehen die Menschen, die die Transformation vorantreiben sollen und dadurch demotiviert werden.

Mögliche Ursachen des Scheiterns	
Organisation	- Verteilte Teams mit wenig Kontakt- und Kollaborationsmöglichkeiten. - Einbettung in eine nicht-agile, streng hierarchische Organisation (hybrider Ansatz). - Matrixorganisation zwischen Linie und Projekte mit raschen Aufgabenwechsel. - Starke Trennung von Fach- und Technikbereichen. - Stark verankerte und lang gelebte Kultur linearer Prozesse (Wasserfallartiges Vorgehen). - Kein Zusammenhalt durch gemeinsame Aufgabe.
Management	- Führungskultur: Kontrollkultur, wenig Vertrauen, Top-Down-Vorgaben statt Selbstorganisation, Formalien, Bürokratie, hierarchischer Führungsstil, Misstrauen (Selbstständigkeit, Lernen, Netzwerken). - Delegation von Verantwortung nach unten, Coaches werden als alleinige Lösung betrachtet. - Agiles Mindset noch nicht vollständig etabliert, akzeptiert oder gelebt. - Ziel und Strategie der Transformation nicht ausreichend kommuniziert. - Ungenügende bzw. verspätete Schaffung passender Rahmenbedingungen und Instabilität derselben.
Menschen	- Misstrauen gegenüber Methodik, Technik etc. - Gefahr der Demotivation durch fehlende Rahmenbedingungen und Trägheit der Umstellung; Schuldzuweisungen an die Methodik statt Blick auf die eigenen Fehler. - Festhalten am traditionellen, Wasserfall-orientierten Mindset. - Sorgen vor Kontroll- und Positionsverluste, Sorgen vor Fehlern und Strafen. - Unzureichendes Wissensmanagement: ignorierte Wissenslücken, Wissenssilos. - Auf einzelne Mitarbeiter als individuelle Wesen wird nicht eingegangen.
Changemanagement	- Strenge Sparmaßnahmen und Budgetkürzungen, Ressourcenstreichungen. - Teams und treibende Kräfte werden mit der Transformation (Technologie, Methode, Infrastruktur etc.) allein gelassen, Bottom-Up-Transformation wird mit schwacher Top-Down-Unterstützung kombiniert. - Transformation wird als beendet angesehen „weil man bereits agil ist“. - Symbolische Innovation: nur das Wording wird verändert, die Substanz verbleibt aber unverändert. - Starke Kontrollkultur. - Wenig Mut zur Veränderung.
Prozesse	- Dominanz unflexibler Prozesse nicht-agilem Ursprungs. - Abhängigkeiten von externen, nicht-agilen Lieferanten und anderen nicht-agilen Teams. - Fehlender Ansatz/Wille zur Automatisierung. - Festhalten an fixen Daten z.B. für Releases, obwohl dies sachlich nicht notwendig ist. - Verschieben von Problemen auf spätere Zeitpunkte. - Eine große Transformation ohne ausreichende Erfahrung und Wissensträger ausrollen.
Umwelt	- Gesetzliche Anforderungen, Kontrolle von außerhalb (Aufsichtsbehörden). - Übermäßige vorgeschriebene, jedoch unflexible Dokumentations- und Sorgfaltspflichten. - Zeitdruck durch vorgegebene Termine.
Methode	- Wenig Wissen und Fehlinterpretation von Scrum. - Missachten von Best Practices und Empfehlungen. - Unklarheiten bzgl. der Scrum-Rollen, Weglassen von Rollen. - Es werden keine ähnlichen Fälle oder Best Practices betrachtet.

Mögliche Ursachen des Scheiterns	
Ressourcen	- Unzureichende Infrastruktur für die agilen Prozesse, viele Infrastrukturen richten sich nach alten Prozessen, viele technische Schulden. - Ressourcen für die Transformation werden zwecks anderer (nicht-agiler) Vorhaben abgezogen (z.B. Anforderungen von Kunden oder Gesetzgeber), Fluktuation in den Teams. - Wenige Wissensträger vorhanden.

Tabelle 3: Mögliche Ursachen des Scheiterns aus der Praxis.
Zusammenfassung aus [31, S. 78 ff].

Aus den Experteninterviews und den erfolgreichen Ansätzen in der Literatur erwachsen auch Vorschläge, wie man in der aktuellen Situation ansetzen könnte, um diese Kaskade der Auswirkungen zu unterbrechen. Dabei stehen Projekt- und Unternehmenserfolg in Verbindung. In [31] wurden alle resultierenden Verbesserungsansätze aufgezählt. Diese variieren in dem Punkt, wo die Verbesserung stattfinden soll. Im Folgendem werden – aufsetzend auf den gesammelten Aussagen aus Tabelle 2 und Tabelle 3 – die möglichen Ansatzpunkte der Verbesserungen zusammengefasst. Hier seien einige Beispiele genannt (Kernaussagen aus [31] mit Ergänzungen aus [4], [11], [14], [25], [37], [39], [43], [52], [66]):

- Allgemein muss der agile Gedanken im Unternehmen verbreitet werden. Es muss ein kontinuierlicher Verbesserungsprozess eingeführt werden. Hier muss die Gelegenheit genutzt werden, alte Prozesse grundlegend zu überdenken.
- Projekten gegenüber sollte mehr Vertrauen aufgebaut werden und innerhalb dieser muss das Verantwortungsgefühl gestärkt werden. Zudem könnte man Verzögerungen reduzieren, indem man nicht-agile Schnittstellen entkoppelt oder kleinere Teams mit Zuständigkeit für Module bildet.
- In der Projektmethodik und in der Projektorganisation sollten sich Denk- und Arbeitsweisen möglichst von vorhandenen Tools und Dokumenten und nicht-agilen Elementen lösen. Eine offene Fehlerkultur und eine unternehmensgerechte Anpassung der Methodik – wie das Ausprobieren neuer Ansätze – können beim hybriden Übergang helfen.
- Die Unternehmensorganisation muss so verändert werden, dass sich die Fach- und Technikbereich annähern. Das bedeutet, dass durch geschickte Umorganisation und ein allgemeiner Kulturwandel ein engeres Verständnis erzielt und die Zusammenarbeit zwischen diesen Bereichen erleichtert werden soll.
- Technik, wie ein stärkerer Tooleinsatz (für Kommunikation, Management, Deployment etc.) oder Testumgebungen unterstützen die agilen Prozesse. Dazu kann man auch die angepasste Neugestaltung von Arbeits- und Meetingräumen sehen.
- Das Management muss die Transformation in der Unternehmensstrategie verankern. Auch hier sollte ein Umdenken stattfinden und klare Ziele für das Unternehmen festgelegt und kommuniziert werden (Will das Unternehmen komplett agil sein?). Alte Strategien sollten bezüglich Kompatibilität zur Agilität überdacht werden (z.B. zu externen Lieferanten). Eine Schulung vom obersten bis zum unteren Management inklusive der Beratung über die Bedeutung von Agilität und Transformation muss in Betracht gezogen werden.
- Für die Kommunikation in der Aufbauorganisation muss das Instrument der Diskussion als Konzept normalisiert werden. Die Kommunikation mit anderen Unternehmen und die Suche nach gemeinsamen Lösungen kann Wissen und Erfahrungen vermehren. Auch *Quick Wins* sind es wert, kommuniziert zu werden.
- Im Change-/Transformationsprozess muss die Verbreitung der Agilität über Projektgrenzen hinaus erfolgen. Fachbereiche müssen bereits in frühen Stadien einbezogen werden.

Prozesse, Organisation und Kultur sind wichtige Elemente des Transformationsprozesses. Mit ihnen soll die Transformation alle Bereiche des Unternehmens erreichen, wo eine Transformation Sinn macht, und dort den Denkhorizont erweitern (z.B. im gesamten Projektumfeld mit seinen Schnittstellen). Ein verstärkter Fokus auf Mitarbeiterwünsche und die Einbeziehung unterschiedlicher Interessengruppen kann helfen, von Schwachstellen zu erfahren. Die Bereitschaft, notwendige Investitionen zu tätigen, bringt den Transformationsprozess voran.

Auch wenn viele Wege zum Scheitern führen, kann man dennoch an vielen Punkten steuernd eingreifen, sofern die Ursache-Wirkungsbeziehungen bekannt sind.

5 Entwicklung von Misserfolgsfaktoren

Der theoretische und praktische Teil der Untersuchung macht deutlich, dass auch das Scheitern „viele Gesichter hat". Da ein bestimmtes Verhalten die Erfolgsfaktoren sowohl positiv als auch negativ beeinflusst, findet sich deshalb in den sechs Ursachenkategorien Management, Menschen, Organisation, Prozesse, Methode, Ressourcen und Umwelt auch kontraproduktives Verhalten wieder. Der Unterschied zwischen dem kontraproduktiven Verhalten und der guten Umsetzung der Erfolgsfaktoren liegt hier im Umgang mit den Erfolgsfaktoren, wobei ersteres durch nicht-agile Aspekte wie Kontrollzwang, Geiz, Angst oder Faulheit der Transformation Schaden zufügen. In Anlehnung an die Unterteilung in Tabelle 3 kann man zu diesen Faktoren auch das Changemanagement zählen, denn dieses drückt die Art und Weise aus, wie das Management und die Mitarbeiter bei der Gestaltung der gesamten Transformation agieren. Deshalb können die Erfolgsfaktoren auch als Faktoren des Misserfolgs verwendet werden, denn wenn diese keine ausreichende Unterstützung finden, können sie sich gegenseitig in Richtung Misserfolg verstärken. Tabelle 4 beinhaltet die acht Faktoren und deren negativen Einfluss auf den Erfolg, inklusive der aus der Untersuchung resultierenden Einstufung des Einflussgrades anhand der Bedeutung auf den Erfolg. Die Spalte „Negativer Einfluss auf…" zeigt Beispiele der zu beeinflussenden Tätigkeiten der Kategorie und „Wirkung" nimmt Bezug auf die Effekte, die der Einfluss dieser Kategorie haben kann. Dabei werden alle Kategorien als bedeutsam/unverzichtbar (mindestens hoch) eingestuft, da jede von ihnen, wie in der vorangegangenen Untersuchung gezeigt, innerhalb der Transformation einen nicht unerheblichen Einfluss ausübt und immer präsent ist. Die Umwelt, das Management und die Menschen besitzen jedoch den größten Einfluss und sind daher als „sehr hoch" zu bewerten.

	Ursachen-Kategorie	Negativer Einfluss auf…	Wirkung	Einfluss auf Scheitern
Interne Rahmenbedingungen	1 Management	- Führungsstil - Strategie, Vision und Zielsetzung - Rahmenbedingungen der Transformation - Durchführung der Transformation - Top-down-Support	- Geringe Möglichkeiten der Gestaltung der Rahmenbedingungen - Unerwünschte Ergebnisse der Transformation - Geringe Motivation der Mitarbeiter - Negativer Einfluss auf Prozesse	sehr hoch
	2 Menschen	- Akzeptanz - Emotionale Einstellung - Individualität - Fleiß und Ehrgeiz - Bottom-up-Ansatz	- Geringe Möglichkeiten der Gestaltung der Rahmenbedingungen - Unerwünschte Ergebnisse der Transformation - Geringe Geschwindigkeit der Transformation	sehr hoch

			- Geringe Motivation der Mitarbeiter	
	3 Change-Management	- Art und Höhe des Budgets - Kontrollkultur - Startposition und Fortschritt	- Geringe Möglichkeiten der Gestaltung der Rahmenbedingungen - Geringe Änderungsgeschwindigkeit - Chaos im Ablauf - Unpassende/nicht-agile interne Rahmenbedingungen - Geringe Motivation der Mitarbeiter - Geringe Geschwindigkeit der Transformation	hoch
	4 Organisation	- Unternehmens- und Projektorganisation - Arbeitsumfeld - Kultur, Denkweise und Verhalten - Gestaltung der Aufbauorganisation nach agilen Prinzipien - Zusammensetzung der Bereiche	- Negativer Einfluss auf Arbeitsbedingungen - Veränderungsprozesse werden erschwert	hoch
	5 Prozesse	- Lernmöglichkeiten - Kommunikation - Anforderungsmanagement - Art und Reichweite der Transformation - Schnittstellen und Konflikte	- Konflikte - Wenig Transparenz - Unerfülltes Verbesserungspotential - Wenig erkennbarer Fortschritt - Unerwünschte Ergebnisse der Transformation - Wenig Motivation der Mitarbeiter - Geringe Geschwindigkeit der Transformation	hoch
	6 Methode	- Verständnis bzgl. Komplexität - Einhaltung und Akzeptanz - Zu suchende Vorbilder - Rollen und Verantwortung	- Chaos im Ablauf der Transformation - Unerwünschte Ergebnisse der Transformation - Wenig Motivation der Mitarbeiter	hoch
	7 Ressourcen	- Technische Ressourcen: Infrastruktur, Schnittstellen, Anforderungsgerechtigkeit, Verfügbarkeit - Humanressourcen: Verfügbarkeit, Wissensstand, Einteilung	- Gestörter Ablauf der Transformation - Geringe Motivation der Mitarbeiter - Geschwindigkeit der Transformation	hoch
Externe Rahmenbedingungen	8 Umwelt	- Gesetzgebung - Kunden - Zeitdruck	- Qualität der Ergebnisse - Geringe Akzeptanz der Produkte - Unpassende Rahmenbedingungen und nicht-agile Schnittstellen nach außen - Compliance-Verstöße	sehr hoch

Tabelle 4: Kategorien der Ursache des Scheiterns und deren Einflüsse [31, S. 87 ff]

Im Hinblick auf die Einschätzung der Kategorie des Scheiterns nach ihrer Einflussstärke sieht man, dass sowohl das Management, als auch die Umwelt als wichtigste beeinflussende Faktoren auf Strategie und Vision, auf Gestaltung der Rahmenbedingungen und auf die Art und Weise des Changemanagements einwirken. Diese wiederum beeinflussen die Motivation, das Verhalten und den Faktor Mensch. Er entscheidet, ob er die Transformation voranbringen oder blockieren möchte. Abbildung 3 stellt diesen Zusammenhang dar, wobei der Einfluss von oben nach unten, egal ob positiv oder negativ, abgebildet ist. Dieser Einfluss löst, wie in Tabelle 2 zu sehen, Symptome methodischer, sozialer oder prozessualer Art aus, die man als Indikatoren für den Erfolg betrachten kann. Der Faktor Mensch ist jedoch bei diesen Indikatoren stark involviert und prägt diese stark mit bzw. lässt sich durch diese stark prägen. Am Ende deuten die Indikatoren die Tendenz zum Erfolg oder Misserfolg an, indem man die in Abbildung 3 zu sehenden Ausprägungen dieser für sich auswertet (z.B. Welche Vorteile konnten wir bisher aus der Transformation schöpfen und was hätten wir uns zu diesem Stadium der Transformation erhofft?).

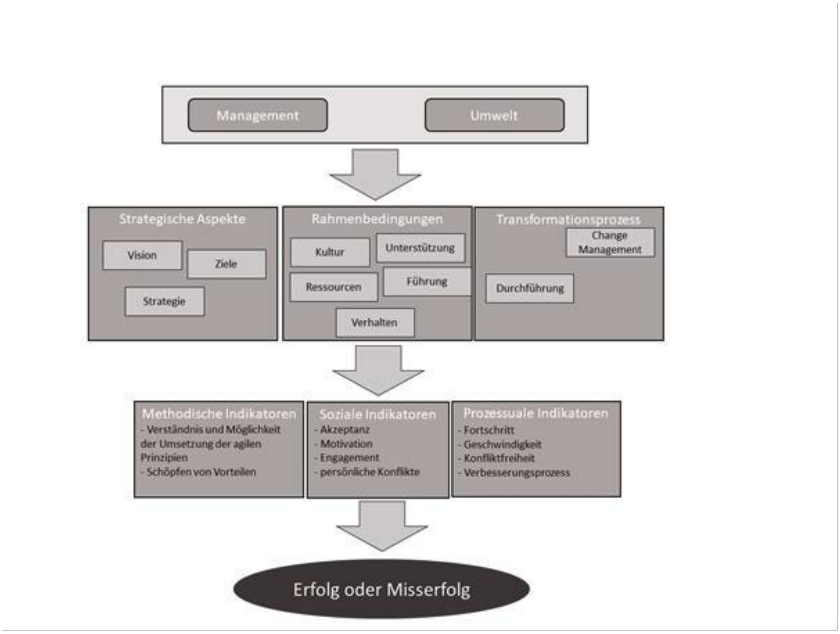

Abbildung 3: Kette der Einflussfaktoren auf eine agile Transformation.
Angelehnt an [31, S. 89].

Zur Ursachenforschung kann man diese Ursache-Wirkungskette von unten nach oben zurückverfolgen und sich daran orientieren, wo es im eigenen Unternehmen gut oder schlecht „läuft". Dem Management kommt am Ende die wichtigste Rolle zu. Es muss lenken, gestalten und motivieren. Verschiedene Verbesserungsansätze entstehen durch den Blick aus verschiedenen Perspektiven auf die Strategie, das Changemanagement und die Rahmenbedingungen, die das Management mit am stärksten beeinflusst.

6 Ergebnisse und Ausblick

In diesem Aufsatz wurde mithilfe einer Literaturauswertung und der Analyse eines praktischen Falls Misserfolgsfaktoren und eine Ursache-Wirkungskette für den Erfolg agiler Transformationen entwickelt. Anhand der Ergebnisse kann geschlossen werden, dass mit der Aufmerksamkeit auf den aktuellen Fortgang des Transformationsprozesses, der Beobachtung von Hindernissen und Fehlern, sowie deren Ursachen und Behebung, ein essentieller Schritt getan ist, um die Erfolgswahrscheinlichkeit einer agilen Transformation zu erhöhen. Dabei kann man sich an drei Indikatoren orientieren, um insgesamt die Art der Auswirkungen von acht (Miss-)Erfolgsfaktoren auf die Gesamtorganisation und auf die Projekte zu beobachten. Da jede Transformation anders verlaufen kann, kann allerdings keine generelle Empfehlung entwickelt werden, die in jeder Situation anwendbar ist. Aber dieser Aufsatz liefert einen ersten Leitfaden, wie ein Unternehmen eine passende Lösung für sich finden kann, indem es sich selber gezielt beobachtet und eigene Lösungsansätze entwickelt. Für die weitere Forschung sollten weitere praktische Fälle auf diese Weise untersucht werden, um die Übertragbarkeit der Ergebnisse auf andere Unternehmen zu prüfen und vertiefen zu können.

Die Zukunft der Agilität und ihrer Anwendung ist noch offen. Durch neue Trends wie z.B. DevOps fließen immer wieder neue Variationen und Ideen ein und beeinflussen heutige und zukünftige Transformationen. Da Agilität einen ständigen Wandel mit sich bringt, kann die Lösung von heute schon morgen wieder unpassend sein und Agilität ist für viele organisatorische Probleme keine Lösung. Sowohl Unternehmen, wie auch die Forschung, sollten sich auf diese Dynamik einstellen.

Referenzen

1. agilemanifesto.org (2001). Manifest für Agile Softwareentwicklung. Abgerufen am 9. August 2019 von Agilemanifesto.org. http://agilemanifesto.org/iso/de/manifesto.html. Abraham, N.; Bibel, U.; Corleone, P.: Formatting Contributions for LNI. In (Glück, H.I. Hrsg.): Proc. 7th Int. Conf. on Formatting of Workshop-Proceedings, New York 1999. Noah & Sons, San Francisco, 2001; S. 46-53.
2. Barroca, L., Dingsøyr, T. & Mikalsen, M. (2019). Agile Transformation: A Summary and Research Agenda from the First International Workshop. In R. Hoda (Hrsg.). Proceedings of the Agile Processes in Software Engineering and Extreme Programming – Workshops (XP 19 Workshops) 21-25 May 2019, Montréal, QC, Canada (S. 3 – 9). https://doi.org/10.1007/978-3-030-30126-2_1. https://arxiv.org/ftp/arxiv/papers/1907/1907.10312.pdf (pdf S. 1 – 6).
3. Bates, C. & Yates, S. (2008, Mai 13). Scrum Down: A Software Engineer and a Sociologist Explore the Implementation of an Agile Method. In L.-T. Cheng, C. de Souza, Y. Dittrich, M. John, O. Hazzan, F. Maurer, H. Sharp, J. Singer, S. E. Sim, J. Sillito, M.-A. Storey, B. Tessem, & G. Venolia (Chair), *Proceedings of the 2008 international workshop on Cooperative and human aspects of software engineering (CHASE '08), Leipzig, Germany (*S. 13 – 16). https://doi.org/10.1145/1370114.1370118.
4. Baur, M. & Jakob, S. (2018). Wenn es auf Geschwindigkeit und Flexibilität ankommt Produkte, Projekte, Portfolios, Experimente – die agile Transformation der Haufe Group. *Projekt Magazin,* 09 2018, 1 – 17. https://www.projektmagazin.de/artikel/produkte-projekte-portfolios-experimente-die-agile-transformation-der-haufe-group_1128343.
5. Böhm, J. (2019). *Erfolgsfaktor Agilität: Warum Scrum und Kanban zu zufriedenen Mitarbeitern und erfolgreichen Kunden führen*. Springer Vieweg. https://doi.org/10.1007/978-3-658-25085-0.
6. Bonk, L. & Hedfeld, P. (2019). Agile Transformation und Innovationsfähigkeit. *ERP Management,* 1/2019, 41-44. https://www.wiso-net.de/document./ERP__89B61255E61606E8C12583C1005610B1. (pdf S. 1 – 3).

7. Vom Brocke, J., Simons, A., Niehaves, B., Riemer, K., Plattfaut, R. & Cleven, A. (2009). Reconstruction the Giant: On The Importance of Rigour in Documenting the Literature Search Progress. Association for Information Systems AIS Electronic (Hrsg.) *Proceedings of the 17th European Conference on Information Systems (ECIS 2009) 8-10 June 2009, Verona, Italy* (S. 2206-2217). http://aisel.aisnet.org/ecis2009/161.

8. Campanelli, A. S., Neto, F. S. & Parreiras, S. (2017). *Assessing Agile Transformation Success Factors*. In B. Aysolmaz & H. A. Reijers (Hrsg.) Use Cases for Understanding Business Process Models. *Proceedings of the 29th International Conference (CAiSE 2017) 12-16 Juni 2017, Essen, Germany* (S. 364-379) (pdf S. 1-6). https://doi.org/10.1007/978-3-319-59536-8_23. https://arxiv.org/pdf/1711.04188.pdf. (pdf S. 1 – 6).

9. Carew, P. J., Glynn, D. (2017). Anti-patterns in Agile Adoption: A Grounded Theory Case Study of One Irish IT Organisation. *Global Journal of Flexible Systems Management,* 18(1), 275 – 289. https://doi.org/10.1007/s40171-017-0162-8.

10. Conboy, K. & Caroll, N. (2019). Implementing Large-Scale Agile Frameworks: Challenges and Recommendations. *IEEE Software,* 36(2), 44 – 50. https://doi.org/10.1109/MS.2018.2884865.

11. O'Connor, C. P. (2010, October 17-21). Letters from the Edge of an Agile Transition. W. R. Cook, S. Clarke & M. Rinard (Chair), *Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion (OOPSLA'10) Reno/Tahoe/Nevada, USA* (S. 79 – 84). https://doi.org/10.1145/1869542.1869557.

12. Denning, S. (2016). How to make the whole organization "Agile". *Strategy & Leadership,* 44(4), 10 – 17. https://doi.org/10.1108/SL-06-2016-0043.

13. Denning, S. (2019a). The ten stages of the Agile transformation journey. *Strategy & Leadership,* 47(1), 3 - 10. https://doi.org/10.1108/SL-11-2018-0109.

14. Denning, S. (2019b). Lessons learned from mapping successful and unsuccessful Agile transformation journeys. *Strategy & Leadership*, 47(4), 3 - 11. https://doi.org/10.1108/SL-04-2019-0052.

15. Dikert, K., Paasivaara, M.& Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations: A systematic literature review. *The Journal of Systems and Software,* 119, 87 – 108. https://doi.org/10.1016/j.jss.2016.06.013.

16. Ebert, C. & Paasivaara, M. (2017). Scaling Agile. *IEEE Software, 34,* 98 – 103. https://doi.org/10.1109/MS.2017.4121226.

17. Fischer, O. (2018). Erfahrungsbericht der Fiducia IT AG - Den Kulturwandel zum agilen Unternehmen aktiv gestalten. Projektmagazin (Hrsg.) *Spotlight Agil im Großen – Frameworks für den Wandel,* S. 65 – 78. https://www.projektmagazin.de/spotlight/agil-im-grossen-frameworks-fuer-den-wandel.

18. Fowler, M. (25. August 2018). *The State of Agile Software in 2018- transcription of my keynote at Agile Australia*. Abgerufen am 24.10.2019 von https://martinfowler.com/articles/agile-aus-2018.html.

19. Fuchs, C., Barthel, P., Winter, K. & Hess, T. (2019). Agile Methoden in der digitalen Transformation – mehr als ein Konzept für die Softwareentwicklung. *Wirtschaftsinformatik & Management,* 11(4), 169 – 207. https://doi.org/10.1365/s35764-019-00192-8.

20. Gandomani, T. J., Zulzali, H., Gahni, A. A. A., Sultan, A. D. M. (2013). Effective factors in agile transformation process from change management perspective. In IEEE (Hrsg.) *Proceedings of 3rd International Conference on Advance Information System, E-Education & Development (CAISED 2013) January 2013, 2013 Kuala Lumpur, Malaysia* (S. 1 – 6). https://arxiv.org/ftp/arxiv/papers/1302/1302.2747.pdf.

21. Gandomani, T. J., Zulzali, H. & Nafchi, M. Z. (2014). Agile Transformation: What is it about? In D. N. A. Jawawi, S. Sulaiman, N. Sa'adon, & R. Mohamad (Hrsg.) *Proceedings of 8th Malaysian Software Engineering Conference (MySec 2014) 23-24 September 2014, Langkawi, Malaysia* (S. 240 - 245). https://doi.org/10.1109/MySec.2014.6986021.

22. Gandomani, T. J., Zulzali, H., Abdul, A. A., Sultan, A. B. & Parizi, R. M. (2015). The impact of inadequate and dysfunctional training on Agile transformation process: A Grounded Theory study. *Information and Software Technology,* 57 (2015), 295 – 309. https://doi.org/10.1016/j.infsof.2014.05.011.

23. Gelperin, D. (2008, Mai 10). Exploring Agile. P. Kruchten & S. Adolph (Chair), *Proceedings of the 2008 international workshop on Scrutinizing agile practices or shoot-out at the agile corral (APOS '08) Leipzig, Germany* (S. 1 – 3). https://doi.org/10.1145/1370143.1370144.

24. Ghosh, G. K. (2012). Challenges in Distributed Scrum. IEEE (Hrsg.) *Proceedings of the 2012 IEEE 7th International Conference on Global Software Engineering (ICGSE 2012) 27-30 August 2012, Porto Alegre, Rio Grande do Sul, Brazil* (S. 200). https://doi.org/10.1109/ICGSE.2012.46.

25. Goodman, D. & Elbaz, M. (2008). "It's not the pants, it's the people in the pants" Learnings from The Gap Agile Transformation – What Worked, How We Did it, and What Still Puzzles Us. L. Bernard, A. Friis-Christensen, H. Pundt & I. Compte (Hrsg.) *Proceedings of the Agile 2008 Conference 5-8 May 2008,* Girona, Spain (S. 112 – 115).

26. Gren, L., Torkar, R. & Feldt, R. (2014). Work Motivational Challenges Regarding the Interface Between Agile Teams and a Non-Agile Surrounding Organization: A case study. In IEEE (Hrsg.) *Proceedings of the 2014 Agile Conference (AGILE'14) 28 July 2014 – 1 August 2014, Kissimmee, Florida, USA* (S. 11 – 15). https://doi.org/10.1109/AGILE.2014.13.

27. Hamdani, M. & Butt, W. H. (2017). Success and Failure Factors in Agile Development. In IEEE (Hrsg.) *Proceedings of the 2017 International Conference on Computational Science and Computational Intelligence* (CSCI 2017) 14-16 December 2017, Las Vegas, USA (S. 981 – 986). https://doi.org/10.1109/CSCI.2017.169.

28. Hasebrook, J., Kirmße, S & Fürst, M. (2019). *Wie Organisationen erfolgreich agil werden: Hinweise zur erfolgreichen Umsetzung in Zusammenarbeit und Strategie*. Springer Nature.

29. Helfferich, C. (2019). Leitfaden- und Experteninterviews. In N. Baur & J. Blasius (Hrsg.) (2019*) Handbuch Methoden der empirischen Sozialforschung* (S. 669 – 686). Springer VS.

30. Hochmüller, E. & Mittermeir, R. T. (2008, Mai 10). Agile Process Myths. P. Kruchten & S. Adolph (Chair), *Proceedings of the 2008 international workshop on Scrutinizing agile practices or shoot-out at the agile corral (APOS '08) Leipzig, Germany (S. 5 – 8). https://doi.org/10.1145/1370143.1370145.*

31. Hochstrat, S. (2020). *Kritische Betrachtung agiler Transformation am Beispiel von Scrum: Perspektiven und Faktoren mit Einfluss auf den Erfolg in der Praxis* (Master-Thesis). FOM Hochschule, Düsseldorf.

32. Hoda, R. & Noble, J. (2017, May 20-28). Becoming Agile: A Grounded Theory of Agile Transitions in Practice. In M. A. Ferrarion & B. Pernici (Chair), *Proceedings of the 39th International Conference on Software Engineering (ICSE '17)* Buenes Aires, Argentinia (S. 141 – 151). https://doi.org/10.1109/ICSE.2017.21.

33. Hohl, P., Klünder, J., van Bennekum, A., Lockhard, R., Gifford, J., Münch, J., Stupperich, M. & Schneider, K. (2018). Back to the future: origins and directions of the "Agile Manifesto" – views of the originators. *Journal of Software Engineering Research and Development,* 6 (15), 1 – 27. https://doi.org/10.1186/s40411-018-0059-z.

34. Iqbal, J., Omar, M. & Yasin, A. (2019). The Impact of Agile Methodologies and Cost Management Success Factors: An Empirical Study. *Baghdad Science Journal, 16 (Special Issue on the Proceedings of the7th International Conference on Computing and informatics (ICOCI2019) 27-29 March 2019, Bangkok, Thailand)*, 496 – 504.

35. Jeffries, R. (2. März. 2016). *Scrum Fails, I'm Told, Endlessly*. Abgerufen am 17. August 2019 von https://ronjeffries.com/articles/2015-03-02-scrum-fails/.

36. Karvonen, T., Sharp, H. & Barroca, L. (2018). Enterprise Agility: Why Is Transformation so Hard? J., Garbajosa, W. Xioafeng, & , Ademar, A. (Hrsg.) *Proceedings of the 19th International Conference on Agile Software Development (XP 18) 21-25 May 2018, Port Portugal* (S.131 – 145). https://doi.org/10.1007/978-3-319-91602-6_9.

37. Kim, S., Lee, H. Kwon, Y., Yu, M. & Jo, H. (2016). Our Journey to becoming Agile - Experiences with agile transformation in Samsung Electronics. In A. Potanin, G. C. Murphy, S. Reeves & J. Dietrich (Hrsg.) *Proceedings of 23rd Asia-Pacific Software Engineering Conference (APSEC) 6-9 December 2016, Hamilton, New Zealand* (S.377-380). https://doi.org/10.1109/APSEC.2016.60.

38. Komus, A. (2013). Agile Methoden in der Praxis – Studie zur Anwendung und Zufriedenheit. *HMD – Praxis der Wirtschaftsinformatik,* 50(2), 84-91. https://doi.org/10.1007/BF03340799.

39. Krasteva, I. & Ilieva, S. (2008, May 10). Adopting an Agile Methodology — Why It Did Not Work. P. Kruchten & S. Adolph (Chair), *Proceedings of the 2008 international workshop on Scrutinizing agile practices or shoot-out at the agile corral (APOS '08)*. Leipzig, Germany (S. 33 – 36). https://doi.org/ 10.1145/1370143.1370150.

40. Kuusinen, K., Gregory, P. Sharp, H. & Barroca, L. (2016). Strategies for doing Agile in a non-Agile Environment. In ACM (Hrsg.) *Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '16) September 2016, Ciudad Real, Spain* (Artikel Nr 5). https://doi.org/10.1145/2961111.2962623.

41. Layman, L., Williams, L. & Cunningham, L. (2004, November). Motivations and Measurements in an Agile Case Study. E. Damiani, G. Succi & M. Marchesi (Chair), *Proceedings of the 2004 workshop on Quantitative techniques for software agile process (QUTE-SWAP '04) New Port Beach, California* (S. 14 - 24). https://doi.org/10.1016/j.sysarc.2006.06.009.

42. Lee, J.-C. & Chen, C.-Y. (2019). Investigating the environmental antecedents of organizations' intention to adopt agile software development. *Journal of Enterprise Information Management,* 32(5), 869 – 886. https://doi.org/10.1108/JEIM-06-2018-0119.

43. Mann, C. & Maurer, F. (2005). A case study on the impact of scrum on overtime and customer satisfaction. IEEE (Hrsg.) *Proceedings of Agile Development Conference (2005) (ADC'05) 24-29 July, Denver, CO, USA* (S. 70 -79). https://doi.org/10.1109/ADC.2005.1.

44. Maximini, D. (2018). *Scrum – Einführung in der Unternehmenspraxis* (2. Auflage). Springer Gabler. https://doi.org/10.1007/978-3-662-56326-7.

45. Mayring, P. (2000). Qualitative Inhaltsanalyse. Forum Qualitative Sozialforschung, 1(2), Artikel 20, 1 – 10. www.qualitative-research.net/index.php/fqs/article/view/1089/2383.

46. Mayring, P. & Fenzl, T. (2019). Qualitative Inhaltsanalyse. N. Baur & J. Blasius (Hrsg.)(2019). *Handbuch Methoden der empirischen Sozialforschung* (S. 633 – 648) (2. Auflage). Springer VS. https://doi.org/10.1007/978-3-658-21308-4_42.

47. Mitchell, I. (22. März 2018). *Twenty Top Fails in Executive Agile Leadership*. Abgerufen am 26. August 2019 https://www.scrum.org/resources/blog/twenty-top-fails-executive-agile-leadership.

48. Nafchi, M. Z., Zulzali, H. & Gandomani, J. (2014). On The Current Agile Assessment Methods and Approaches. In IEEE (Hrsg.) *Proceedings of the 8th Malaysian Software Engineering Conference (2014) 23-24 September 2014, Langkawi, Malaysia (MySEC 2014)* (S. 251 – 254). https://doi.org/10.1109/MySec.2014.6986023.

49. Oomen, S., De Waal, B., Albertin, A. & Ravesteyn, P. (2017). HOW CAN SCRUM BE SUCCESFUL? COMPETENCES OF THE SCRUM PRODUCT OWNER. Association for Information Systems AIS Electronic (Hrsg.), *Proceedings of the 25th European Conference on Information Systems (ECIS) 5-10 June 2017, Guimarães, Portugal* (S. 131 – 142). http://aisel.aisnet.org/ecis2017_rp/9.

50. Paasivaara, M. (2017, Mai 22-23). Adopting SAFe to Scale Agile in a Globally Distributed Organization. In S. Marczak, A. Sarma, D. Cruzes (Chair), *Proceedings of IEEE 12th International Conference on Global Software Engineering (ICGSE'17), Buenos Aires, Argentinia* (S. 36 – 40). https://doi.org/10.1109/ICGSE.2017.15.

51. Paasivaara, M. & Lassenius, C. (2016). Challenges and Success Factors for Large-scale Agile Transformations: A Research Proposal and a Pilot Study. In ACM (Hrsg.) *Proceedings of the Scientific Workshop (XP '16) 24 May 2016,* Edingburgh, Scotlandm UK (Artikelnr. 9). https://doi.org/10.1145/2962695.2962704.

52. Paasivaara, M. Behm, B., Lassenius, C. & Hallikainen, M. (2018). Large-scale agile transformation at Ericsson: a case study. *Empirical Software Engineering, 23(5),* S. 2550-2596.

53. Parizi, R. M. & Gandomani T. J. & Nafchi, M. Z. (2014). Hidden Facilitators of Agile Transition: Agile Coaches and Agile Champions. In IEE (Hrsg.) *Proceedings of 8th Malaysian Software Engineering Conference (MySEC 2014) 23-24 September, Langkwi, Malaysia (S. 246 – 250).* https://doi.org/10.1109/MySec.2014.6986022.

54. Paterek, P. (2017). Agile Transformation in Project Organization: Knowledge Management Aspects and Challenges. AGH University of Science and Technology (Hrsg.) *Proceedings of the 18th European Conference on Knowledge Management (ECKM2017)* (S.1170 – 1179). https://www.researchgate.net/publication/320002919_Agile_Transformation_in_Project_Organization_Knowledge_Management_Aspects_and_Challenges.

55. Paterek, P. (2019). AGILE TRANSFORMATION CHANGES FROM THE PERSPECTIVE OF PROJECT TEAM VALUES. In University of Latvia (Hrsg.) *Proceedings of the 8th International Scientific Conference on Project Management in the Baltic Countries "Project Management Development – Practice and Perspectives" 25-26 April 2019, Riga, Latvia (S. 162 – 174).*

56. Perkin, N. (2020). *Agile Transformation – Structures, Processes and Mindsets for the Digital Age*. Kogan Page Inc.

57. Perkusich, M., de Almeida, H, O. & Perkusich, A. (2013, March 18-22). A Model to Detect Problems on Scrum-based Software Development Projects. S, Y. Shin & J. C. Maldonado (Chair), *Proceedings of the 28th Annual ACM Symposium on Applied Computing (SAC'13) Coimbra, Portugal* (S. 1037 – 1042). https://doi.org/10.1145/2480362.2480560.

58. Power, K. (2016). Sensemaking and Complexity in Large-Scale Lean-Agile Transformation: A Case Study from Cisco. In IEEE (Hrsg.) *Proceedings of the 49th Hawaii International Conference on System Sciences (HICSS 2016) 5-8 January 2016, Grand Hyatt, Kauai, Hawaii* (S. 5417 – 5426). https://doi.org/10.1109/HICSS.2016.669.

59. Rajpal, M. (2016). Lessons Learned from a Failed Attempt at Distributed Agile. . H. Sharp & T. Hall (Hrsg.) Agile Processes in Software Engineering and Extreme Programming. *Proceedings of the 17th International Conference on Agile Software Development (XP 2016)* 24-27 *May 2016, Edingburgh, UK* (S. 235 - 243). https://link.springer.com/book/10.1007/978-3-319-33515-5.

60. Ranganath, P. (2011). Elevating teams from ‚Doing' agile to 'Being' and 'Living' agile. In IEEE (Hrsg.) *Proceedings of the 2011 Agile Conference (AGILE'11) August 2011* (S.187-197). https://doi.org/10.1109/AGILE.2011.40.

61. Sahota, M. (2012). *An Agile Adoption And Transformation Survival Guide*. lulu.com.

62. Schwaber, K. & Sutherland, J. (2017). *The Scrum Guide - Der gültige Leitfaden für Scrum: Die Spielregeln*. Deutsche Ausgabe. https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-German.pdf.

63. Schwaber, K. (1995, October 15-19). SCRUM Development Process. R. J. Wirfs-Brock (Chair), *Proceedings of the Tenth Annual Conference on Object-Oriented Programming Systems, Languages, and Applications (1995) (OOPSLA'95) Austin, Texas(S.1 – 23)*. In Business Object Design and Implementation (1997), 117-136.

64. Smart, J. (2018). To Transform to Have Agility, Don't Do a Capital A, Capital T Agile Transformation. *IEEE Software, 35 (6),* 56 – 60. https://doi.org/10.1109/MS.2018.4321245.

65. Sutton, M. (20. Februar 2015). *Why Scrum is designed for misuse*. Mhsutto.de. Abgerufen am 20.09.2019 von https://mhsutton.me/scrum-designed-misuse/.

66. Therrien, I. & LeBel, E. (2009). From Anarchy to Sustainable Development: Scrum in Less Than Ideal Conditions. IEEE (Hrsg.) *Proceedings of the 2009 Agile Conference (AGILE'09) August 2009, Chicago, IL, USA (*S. 289 – 294). https://doi.org/10.1109/AGILE.2009.73.

67. Vaske, H. (2019). Die agile Transformation fordert Unternehmen heraus. *Computerwoche (18.02.2019, Nr. 08*. https://www.wiso-net.de/document/CW__20190218040386160103847524 34.

68. De Waal, A. (2018). Success factors of high performance organization transformations. *Measuring Business Excellence,* 22 (4), 375 – 390. https://doi.org/10.1108/MBE-08-2018-0055.

69. Wright, D. (2018). Best Practices for Large-Scale Agile Transformations. University of Oregon (Hrsg.) Capstone Report. University of Oregon Applied Information Management Program 2018. https://scholarsbank.uoregon.edu/xmlui/bitstream/handle/1794/23896/Wright_2018. pdf.

70. Yüce, Ö. (2013). Changing the Way People Think. ITNOW, 55(4), 54 – 55. doi: 10.1093/itnow/bwt086.

Workshop “Evaluation of Service-APIs – ESAPI 2020“
Motto: APIs als Klebstoff einer allumfassenden Digitalisierung
detaillierter Bericht

Sandro Hartenstein[2], Konrad Nadobny[1, 2, 3],
Steven Schmidt[2, 4], Andreas Schmietendorf[1, 2]

[1]OvG-Universität Magdeburg, [2]HWR Berlin,

[3]Bayer AG, [4]Deutsche Bahn AG

1 Motivation und Themen des Workshops

Die Gartner Group[1] geht davon aus, dass im Jahr 2021 mehr als 60% aller Anwendungsentwicklungen von eingesetzten Web-APIs profitieren. Diese mit Hilfe klassischer Internettechnologien zur Verfügung gestellten Web-APIs bieten die Möglichkeit eines konsistenten Zugriffs auf fachlich begründete Informationen und Funktionen aber auch auf komplette Geschäftsprozesse. Neben einer unternehmens- und branchenübergreifenden Integration existierender Softwarelösungen wird dabei auch die Zielstellung einer kompositorischen und damit agilen Softwareentwicklung verfolgt. Aufgrund der ggf. „ad hoc“ zusammengesetzten Lösungen muss auch der Betrieb mit diesen Herausforderungen umgehen können. Daher kommt der Themenstellung „DevOps“ als Klammer zwischen Entwicklung und Betrieb eine besondere Bedeutung zu.

Der ESAPI-Workshop im Jahr 2020 fokussierte die folgenden Themen:

- Bewertung von Vertrauen und Sicherheit bei Web-APIs.
- Branchenspezifische Ansätze zur Spezifikation von Web-APIs.
- Lowcode bzw. Codeless Softwareentwicklung mit Web-APIs.
- Effiziente Ansätze zur „API-fizierung“ von Altanwendungen.
- Risiken bei über Web-APIs bezogenen KI-Algorithmen.
- Vor- und Nachteile von GraphQL-basierten Web-APIs.
- Elemente eines DevOps-orientierten API-Managements.
- Serverless bereitgestellte Web-APIs – Fiktion oder Wirklichkeit?

Der ursprünglich als Präsenzveranstaltung geplante Workshop wurde im Jahr 2020 erstmals online durchgeführt. Mehr als 40 Teilnehmer hatten sich im Rahmen der virtuellen Konferenz zusammengefunden. Erstmals wurde die Veranstaltung von der Bayer AG in Berlin gehostet.

[1] Quelle: Zumerle, D. et al. 2019. API Security. What You Need to Do to Protect Your APIs [online]. Verfügbar unter https://www.gartner.com/en/documents/3956746/api-security-what-you-need-to-do-to-protect-your-apis

2 Beiträge zum Workshop

Im Vorfeld wurde ein entsprechender Call for Paper innerhalb der Community verteilt, auf dessen Basis 11 Beiträge für den Workshop ausgewählt wurden. Um den speziellen Herausforderungen einer Online-Veranstaltung zu genügen, wurden die Beiträge als Keynote, als 10minütige Themen-Pitches oder mit Hilfe parallel bereitgestellter Poster präsentiert.

Anja Fiegler, Andreas Schmietendorf
: Entwicklung smarter Anwendungen mit Hilfe cloudbasiert angebotener KI-Algorithmen;

Niko Zenker, Daniel Paschek, Marvin Leine
: Einsatz einer gewichteten Graphendatenbank zur Abbildung komplexer Unternehmensarchitekturen;

Konrad Nadobny
: Vergleich von Enterprise API-Management-Lösungen;

Steven Schmidt
: Schaffung eines vertrauenswürdigen, öffentlichen WLAN - Herangehensweise und Teilergebnisse;

Michael Petry, Volker Reers, Frank Simon
: Reaktive, minimal destruktive API-Härtung am Beispiel von GraphQL;

Jens Borchers
: Zero Trust-Architektur und -Kultur;

Daniel Kant, Andreas Johannsen
: Exemplarische API-Schwachstellen bei IoT-Geräten auf der Grundlage von OWASP API Security TOP 10;

Gabriel Landa, Sandro Hartenstein
: Bitcoin Blockchain via Satelliten;

Kadir Ider
: Effective Privacy Management Concepts: Increasing Privacy Control by Reducing Complexity;

Maximilian Müller, Matthias Dobkowicz, Andreas Johannsen, Allan Fodi
: Konzeption eines Objektkonfigurators zur Erstellung von Auszügen einer Objektbibliothek;

Sandro Hartenstein
: Entwicklung vertrauenswürdiger Web-APIs.

3 Ergebnisse der Breakout-Diskussionen

Der Tradition des Workshops entsprechend galt es, ein World Cafe erstmals virtuell, mit Hilfe von „Breack Out Sessions durchzuführen. Die Teilnehmer wurden zunächst in drei Gruppen aufgeteilt und dann jeweils einem Diskussionsraum zugeteilt. Jedem dieser Diskussionsräume war ein fester Moderator zugeteilt, welcher den Austausch leitete und die Ergebnisse auf einem gemeinsamen Whiteboard dokumentierte. Nach 15 Minuten wechselten die Gruppen den Diskussionsraum, so dass sie sich nun zu einem weiteren Thema austauschen konnten. Dabei bauten sie auf den Ergebnissen der vorherigen Gruppe auf. Nach weiteren 15 Minuten wurde der Diskussionsraum abermals gewechselt.

Massive APIfizierung von Legacy Applikationen

Das Thema der massiven APIfizierung von Legacy Applikationen wurde kontrovers diskutiert, wobei das Ergebnis in der Abbildung dargestellt ist.

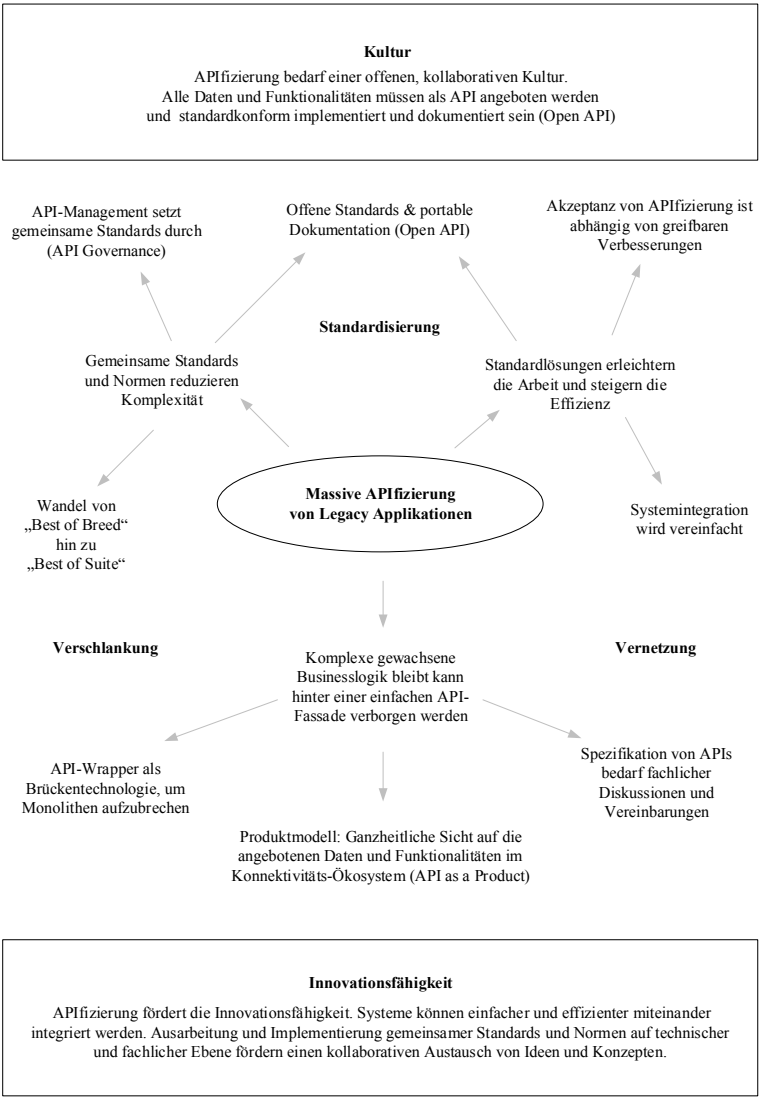

Abbildung 1: Ergebnis der Diskussion

Das Thema gliedert sich in die Teilbereiche Standardisierung, Vernetzung und Verschlankung. Im Laufe der Diskussion wurde zudem herausgearbeitet, dass auch Aspekte wie Kultur und Innovation in diesem Kontext eine große Rolle spielen. So bedarf es einer offenen,

kollaborativen Kultur mit dem Ziel, dass möglichst alle Daten und Funktionalitäten als API angeboten und standardkonform implementiert und dokumentiert werden (Open API). Die Zielvorstellung ist somit ein Konnektivitäts-Ökosystem, in dem alle Akteure sich einfach und effizient austauschen können. Systeme sind im Idealfall in Echtzeit integrierbar und ermöglichen ein Agieren ohne Medienbrüche und Inkonsistenzen.

Die Motivation für eine API-getriebene IT-Architektur im Allgemeinen und die dementsprechend benötigte massive APIfizierung von Altsystemen im Speziellen begründet sich im folgenden Sachverhalt:

Etablierung gemeinsamer Standards und Normen zur Reduktion der Komplexität. Auf dieser Grundlage lassen sich Standardlösungen einfacher und vor allem effizienter bereitstellen. Darüber hinaus bietet sich die Möglichkeit, historisch gewachsene Systemlandschaften zu entwirren (d.h. entkoppeln) und die Strukturen sukzessive zu modernisieren.

In Bezug auf Legacy-Applikationen muss beachtet werden, dass die Systeme oftmals über komplexe, historisch gewachsene Businesslogiken verfügen, die nicht verloren gehen dürfen. Mithilfe eines API-Wrappers können diese zum Beispiel hinter einer API-Fassade verborgen werden, so dass das Altsystem modernisiert und endkoppelt werden kann. Die ursprüngliche Kernfunktion bleibt dabei erhalten, so dass es wie gehabt weiter betrieben werden kann. API-Wrapper können somit als Brückentechnologie genutzt werden, um monolithische Systeme zunächst mit einer Standardschnittstelle zu ertüchtigen und dann nach und nach aufzubrechen. Dies erleichtert nicht nur die Systemintegration, sondern ermöglicht auch einen Wandel weg von lokal optimierten Lösungen (Best of Breed) hin zu ganzheitlichen, integrierten Systemlandschaften (Best of Suite).

Die Akzeptanz einer API-getriebenen Transformation ist abhängig von greifbaren Verbesserungen. Aus technischer, betrieblicher und organisatorischer Sicht ist zunächst ein professionelles API-Management zum Durchsetzen der gemeinsamen Standards unerlässlich (API Governance). In Bezug auf Daten und Funktionalitäten müssen diese Standards auf fachlicher Ebene diskutiert und vereinbart werden. Diese gemeinsamen Standards, Regeln und Normen reduzieren im Nachgang die Komplexität und vereinfachen den späteren Austausch. Idealerweise werden dabei bereits existierende Industriestandards, wie zum Beispiel domänenspezifische Datenmodelle, implementiert.

Vertrauen in Public WIFI-Infrastrukturen

In diesem World Cafe wurden zentrale Fragen zur Einstellung der Diskussionsteilnehmer gegenüber der Vertrauenswürdigkeit öffentlicher WLANs diskutiert. Einstiegspunkt war dabei die Grundsatzfrage, inwiefern überhaupt öffentlichen WLANs vertraut wird. Das Feedback über die verschiedenen Diskussionsrunden hinweg war stark diversifiziert. Im Wesentlichen wird öffentlichen WLANs nicht vertraut. Häufige Antworten haben aber teilweise nach Anbieter bzw. angebotenem Serviceumfang unterschieden, oder die Entscheidung einer eigenen, kurzen Prüfung verschiedener Datensicherheitseigenschaften vorbehalten.

Im Hinblick auf die Fragestellung zur Relevanz der Vertrauenswürdigkeit für die Nutzung des jeweiligen WLAN-Angebotes müsse nach Meinung der Teilnehmer grundsätzlich zwischen beruflicher und privater Verwendung unterschieden werden. Im privaten Kontext war die Auswirkung auf die tatsächliche Nutzung häufiger irrelevant, im beruflichen oder professionellen Kontext gab es jedoch starke Abhängigkeiten.

Der dritte Diskussionsgegenstand bewegte sich im Bereich vertrauensschaffender Maßnahmen, welche der Service aufweisen müsste, um als relativ vertrauenswürdig zu gelten. Hierbei war in allen Diskussionsrunden der grundsätzliche Konsens erkennbar, dass sich Maßnahmen nicht auf eine rein technische Dimension beschränken dürfen. Kommunikative Aspekte zur transparenten Darstellung von Nutzungsrisiken und entsprechenden - gegebenenfalls

betreiberseitigen – Lösungen sind hier sehr häufig genannt worden. Auch die Rolle des Staates als Aufklärer über diese Sachverhalte kam zum Tragen, ebenso wie eine Zertifizierung einer relativen Vertrauenswürdigkeit seitens einer unabhängigen Institution. Eine Anmeldung im öffentlichen WLAN mit einem durch den Betreiber gestellten Zertifikat auf dem eigenen Endgerät ist abhängig von der Anbieterreputation zur Etablierung einer verschlüsselten Kommunikation dabei weitläufig akzeptiert.

Die abschließende Frage für die Teilnehmer des World Cafes befasste sich mit einer potenziell gesteigerten Nutzungsrate öffentlicher WLANs bei eventueller Umsetzung der zuvor diskutierten Maßnahmen und Eigenschaften. Hier war das Feedback überwiegend positiv. Eine Anmerkung bestand darin dass ein Belohnungssystem für die Nutzung eines sichereren Angebots einen zusätzlichen Anreiz darstellen könnte.

Herausforderungen beim KI-Bezug via Web-APIs

Das Angebot von webbasierten APIs, die KI-Algorithmen zugänglich machen, wächst täglich. Aufgrund der zumeist cloudbasierten Bereitstellung dieser Ansätze wird häufig auch von einer Demokratisierung der KI gesprochen, da die technischen Hürden für einen Einsatz von Algorithmen der künstlichen Intelligenz enorm sinken. Entsprechende Angebote finden sich z.B. bei Microsoft im Rahmen der Azure-Plattform oder auch bei der IBM im Rahmen der Bluemix-Plattform. Die Bedenken von Seiten der Anwender, entsprechende Angebote produktiv zum Einsatz zu bringen, sind in Deutschland allerdings enorm. In anderen Regionen wie z.B. im asiatischen oder auch nordamerikanischen Raum steht man dem Einsatz weitaus unkritischer gegenüber. Dementsprechend profitieren innovative Lösungen im Zusammenhang mit fachlichen Anwendungsszenarien, die eher durch den Endanwender bzw. durch potentielle Kunden getrieben werden. Die eher abwartende Haltung in Deutschland impliziert die Gefahr, den Anschluss zu verlieren.

Die folgenden Ausführungen charakterisieren die wesentlichen Eckpunkte der innerhalb des World Cafes durchgeführten Diskussion:

Die Erwartungen der Web-API-Consumer (d.h. Entwickler) sind zum einen eine hohe Security-Grundabsicherung. In diesem Zusammenhang wird typischerweise auf die Authentifizierung und Autorisierung, die netzwerkorientierte Verschlüsselung, das Versionsmanagement sowie ein transparentes Vertragsmanagement Bezug genommen. Zum anderen erwarten die Anwender spezielle Transparenz hinsichtlich des konkret eingesetzten KI-Algorithmus. Die Nachvollziehbarkeit, Verständlichkeit, Genauigkeit und das Vertrauen in die vortrainierten Modelle sind wichtige Anforderungen der Nutzer. Sie können und sollten vom Service Provider adressiert werden. Das Vertrauen in die trainierten Modelle und die Absicherung gegen „böses“ Training sollte von unabhängigen Dritten geprüft und mit Hilfe anerkannter Zertifikate bestätig werden. In diesem Zusammenhang sollten sich auch Standards hinsichtlich des API-Managements bzw. der Spezifikation (Beschreibung) etablieren. Grundsätzlich wurde durch die Teilnehmer festgestellt, dass eine ausschließlich technische Sicht auf die Vertrauenswürdigkeit von Web-APIs nicht ausreicht. Darüber hinaus bedarf es einer Entmystifizierung eingesetzter KI-Algorithmen.

4 Tagungsband und weitere Informationen

Auch für das Jahr 2021 ist die Durchführung eines ESAPI-Workshops vorgesehen. Aktuell gehen wir davon aus, das dieser in Köln (angefragter Gastgeber: Zurich Versicherungsgruppe Deutschland, HS Köln) durchgeführt werden kann. Weiterführende Informationen werden zeitnah unter der folgenden URL im Internet bereitgestellt:

https://blog.hwr-berlin.de/schmietendorf/

Abbildung 2: Tagungsband zum Workshop ([Schmietendorf/Nadobny 2020])
Quelle: https://www.shaker.de/de/content/catalogue/index.asp?lang=de&ID=8&ISBN=978-3-8440-7515-1

5 Quellenverzeichnis

[Schmietendorf/Nadobny 2020] Schmietendorf, A.; Nadobny, K. (Hrsg.): ESAPI 2020 – 4. Workshop Evaluation of Service-APIs, Berlin – 03. November 2020, 140 Seiten, in Berliner Schriften zu modernen Integrationsarchitekturen, Shaker-Verlag, Düren, November 2020, ISBN 978-3-8440-7515-1

Dank

Unser Dank gilt den Referenten und Teilnehmern, aber auch den Partnern (HWR Berlin, OvG-Universität Magdeburg), Sponsoren (Bayer AG Berlin, Deutsche Bahn AG, Delivery Hero) und Unterstützern im Programmkomitee, die eine solche Veranstaltung ermöglicht haben. Ein herzlicher Dank geht auch an die beteiligten Medienpartner SIGS DATACOM GmbH aus Köln und an den Shaker Verlag GmbH aus Aachen.

Der Fachausschuss und die Fachgruppen WI-VM, WI-PM, WI-PrdM stellen sich vor

Fachausschuss WI-MAW:
Management der Anwendungsentwicklung und -wartung

Anwendungssysteme sind aus Sicht der Wirtschaftsinformatik Aufgabenträger im Rahmen der Erfüllung der betrieblichen Gesamtaufgabe. Ihre Aufgabenstellungen werden aus den Unternehmenszielen und den strategischen Zielen der Informationsverarbeitung abgeleitet. Die Entwicklung von Anwendungssystemen erfolgt nicht "kontextfrei", sondern i.A. in einem bestimmten betrieblichen Umfeld. Dies bedeutet zum einen, dass sich das einzelne Anwendungssystem in bereichsübergreifende bzw. unternehmensweite Daten- und Funktionsmodelle oder Objektmodelle einordnen muss. Zum anderen existieren häufig bereits Anwendungen für andere betriebliche (Teil-)Aufgaben, mit denen das System zusammenarbeiten muss.

Der Fachausschuss beschäftigt sich aus dieser Sicht mit der Planung, der Entwicklung, der Einführung, dem Einsatz und der Wartung betrieblicher Anwendungssysteme. Im Vordergrund stehen Vorgehensweisen, Prinzipien und Methoden für die Anwendungsentwicklung im betrieblichen Umfeld sowie ihre Unterstützung durch Softwarewerkzeuge. Im Einzelnen setzt sich der Fachausschuss mit Themen wie den folgenden auseinander:

- Integration von Anwendungssystemen in eine existierende betriebliche DV-Landschaft;
- Sicherung der Investitionen in das Wirtschaftsgut Software; Bewertung von Vorgehensmodellen, Methoden und Werkzeugen zur Anwendungsentwicklung sowie Einsatzerfahrungen;
- Management von Softwareentwicklungsprojekten (Projektplanung, -durchführung und -kontrolle, Projektorganisation, Projektmanagementsysteme, Kosten/ Wirtschaftlichkeit),
- Software Produktmanagement, Configuration Management, Change Management, Migration Management, Reengineering.

Mitgliederzahl: ca. 500

FA-Sprecher

Prof. Dr. G. Herzwurm
Universität Stuttgart
Lehrstuhl für Allgemeine Betriebswirtschaftslehre und Wirtschaftsinformatik II (Unternehmenssoftware)

stellv. FA-Sprecherin

Dr.-Ing. Birgit Demuth
Technische Universität Dresden
Institut für Software- und Multimediatechnik

Fachgruppe WI-VM:
Vorgehensmodelle für die betriebliche Anwendungsentwicklung

Betrachtungsgegenstand der Fachgruppe sind die als "Vorgehensmodelle" bezeichneten Beschreibungen der Aufbau- und Ablauforganisation von Projekten zur Entwicklung und Wartung von Anwendungssystemen. Solche Beschreibungen helfen, die Durchführung von Projekten innerhalb eines Unternehmens oder darüber hinaus zu standardisieren und zu verbessern. Der Begriff Anwendungssystem sei hier sehr weit gefasst: von technischen über betriebswirtschaftliche bis zu organisatorischen Systemen.
Um eine effektive und effiziente Gestaltung der Vorgehensmodelle und damit der Projekte zu erreichen, ist die Berücksichtigung der Schnittstellen zur Betriebswirtschaftslehre einerseits, insbesondere der Organisations- und der Managementlehre, und dem Software Engineering andererseits wesentlich.
Das Thema "Vorgehensmodelle" wird daher von der Fachgruppe aus verschiedenen Blickrichtungen betrachtet:

- Grundlagen: Begriffsdefinitionen, Bestandteile, (formale) Beschreibung von Vorgehensmodellen, Vorgehensmodell-Typen.
- Inhaltliche Bausteine: Konzepte, Methoden, Phasen, Projektmanagement, Qualitätssicherung.
- Werkzeugunterstützung: Vorgehensmodell-Driver, Meta-Modelle, Data-Dictionaries.
- Ökonomische, soziale und psychologische Aspekte: Einführung und Betrieb von Vorgehensmodellen, organisatorisches Umfeld.
- Beispiele aus der Praxis: Standard-Vorgehensmodelle in Organisationen, Branchen und für Anwendungstypen, spezielle Vorgehensmodelle von Unternehmen.
- Standardisierung von Vorgehensmodellen: V-Modell XT, Hermes

Die Fachgruppe fördert einen intensiven Gedankenaustausch durch die Pflege persönlicher Kontakte und unterstützt einen offenen und kritischen Dialog zwischen Wissenschaft und Praxis. Ein weiteres Ziel der Fachgruppe ist die Erarbeitung von Empfehlungen und Stellungnahmen zu den technischen, wirtschaftlichen, organisatorischen und sozialen Aspekten bei Auswahl und Einsatz von Vorgehensmodellen - dies insbesondere vor dem Hintergrund nationaler, europäischer und internationaler Normungs- und Standardisierungs-bestrebungen.
Weitere Informationen über Vorgehensmodelle und die Arbeit der Fachgruppe sind im Internet zu finden unter www.vorgehensmodelle.de.

FG-Sprecher

Dr. rer. nat. Masud Fazal-Baqaie
Next Data Service AG
Berlin

stellv. FG-Sprecher

Dr. rer. nat. Enes Yigitbas
Fachgruppe Datenbanken- und Informationssysteme
Universität Paderborn

Fachgruppe WI-PM:
Projektmanagement

Die Fachgruppe befasst sich mit dem Einsatz, der Verbreitung sowie der Weiterentwicklung des Projektmanagements. Neben Vertretern aus den Hochschulen sollen vor allem Praktiker die Arbeitsschwerpunkte der Fachgruppe definieren, Ergebnisse erarbeiten und Erfahrungen weitergeben. Für die Aufgabengebiete des Projektmanagements sollen Methoden, Werkzeuge und Techniken untersucht werden. Neben den klassischen Aufgabengebieten wie beispielsweise Projektorganisation, Aufwandschätzung, Projektverfolgung und Projektsteuerung stehen folgende Themen im Vordergrund:

Bedeutung und Dimensionierung des Projektmanagements.
Die Bedeutung des DV-Projektmanagements als entscheidender Faktor für den Erfolg oder das Mißlingen von DV-Projekten wird von vielen Entscheidungsträgern unterschätzt. Daher sollte die grundsätzliche Bedeutung sowie der Nutzen einer angemessenen Ausstattung des Projektmanagements mit eigenen Ressourcen transparent gemacht werden.

Human Factors.
In zahlreichen Projekten liegen die größten Projektrisiken bei den sogenannten Human Factors (oder "weichen" Faktoren). Der Umgang mit solchen Risiken erfordert Kompetenz bei Themen wie Motivation, Führung, Teamfähigkeit, Überwindung "politischer" Widerstände u.a.m.

Programm Management.
Immer öfter gefordert wird das Management eines Portfolios von Projekten, wobei nicht alle Projekte des Portfolios eigentliche DV-Projekte zu sein brauchen. Solche Projektportfolios können beispielweise als Folge einer veränderten Unternehmensstrategie entstehen und sollen dann einen größeren Veränderungsprozess bewirken. Hauptaufgabe eines Programme Managements ist dabei die zielorientierte Steuerung der Abarbeitung des Projektportfolios, wobei insbesondere unternehmerische Gesichtspunkte zu beachten sind.

FG-Sprecher

Prof. Dr. Martin Engstler
Hochschule der Medien Stuttgart

stellv. FG-Sprecher

Alexander Volland
Union IT-Services GmbH
Frankfurt am Main

Fachgruppe WI-PrdM:
Software Produktmanagement

Effizientes und effektives Management softwareintensiver Produkte ist zu einer kritischen Kernkompetenz von Unternehmen geworden. Unternehmen sind mit einer stetig wachsenden Anzahl von Herausforderungen konfrontiert, die durch unterschiedliche Lebenszyklen von Systemen und unterschiedliche Kritikalität im Systemeinsatz in immer mehr – und neuen – Anwendungsfeldern entstehen. Hybride Systeme, z.B. im Internet-of-Things, in Automobilen, Flugzeugen, Drohnen, medizinischen Geräten oder in der Unterhaltungselektronik geben Software eine nie dagewesene Bedeutung. Zusätzlich entstehen durch die vielfältigen Initiativen im Rahmen Digitalisierung neue Arbeits- und Geschäftsmodelle und eröffnen vollkommen neue, durch Software getriebene Möglichkeiten zur Innovation.

In diesem dynamischen Umfeld findet softwaregetriebene Innovation an der Schnittstelle zwischen Informatik/Software Engineering und Wirtschaft statt, zwischen Forschung und industrieller Praxis. Das Produktmanagement umfasst hierbei die Entwicklung, Wartung und Evolution klassischer Softwarelösungen im gesamten Produktlebenszyklus, aber insbesondere auch innovative softwarebasierte Innovation. Die Fachgruppe befasst sich einerseits mit Konzepten, Methoden und Werkzeugen der Informatik/Wirtschaftsinformatik zur Gestaltung des Produktmanagements und der Produktinnovation. Andererseits wird insbesondere auch ein starker Fokus auf die praktische Anwendbarkeit theoretischer Konzepte gelegt.

Die Fachgruppe fördert auf dem genannten Gebiet den intensiven Gedankenaustausch, die Pflege persönlicher Kontakte und die Zusammenarbeit interessierter Personen und Gruppen. Dazu zählt u.a. die gegenseitige Information über Veranstaltungen, Projekte und Veröffentlichungen.

FG-Sprecher

PD Dr. Marco Kuhrmann
Universität Passau
Fakultät für Informatik und Mathematik

stellv. FG-Sprecher

Prof. Dr. Jürgen Münch
Hochschule Reutlingen
Herman Hollerith Zentrum

Mitglieder des Fachausschusses Management der Anwendungsentwicklung- und wartung (GI-MAW)

Die Mitglieder des Leitungsgremiums des Fachausschusses finden Sie unter:

https://fa-wi-maw.gi.de/fachausschuss/leitungsgremium

Impressum

Der Rundbrief des Fachausschusses *Management der Anwendungsentwicklung und -wartung (WI-MAW)* ist das Publikationsorgan des Fachausschusses sowie der Fachgruppen

WI-VM *Vorgehensmodelle für die betriebliche Anwendungsentwicklung*
WI-PM *Projektmanagement*
WI-PrdM *Software Produktmanagement*

Der Rundbrief erscheint einmal jährlich elektronisch. Durch den Rundbrief sollen wichtige Erfahrungen, neue Erkenntnisse und aktuelle Informationen unter den Mitgliedern ausgetauscht werden. Rundbriefbeiträge von Mitgliedern und Interessenten sind daher besonders willkommen. Es können Beiträge zu folgenden Rubriken eingereicht werden:

- Fachbeiträge: *Erfahrungsberichte; Theoretische Beiträge; Projektberichte (auch über laufende Projekte)*
- Informationen: *Buchbesprechungen; Tagungsberichte; Vorstellung von Arbeitsgruppen;*
- Leserbriefe: *Veranstaltungen; Call for Papers; Einladungen; Programme*

Es wird gebeten, Beiträge in elektronischer Form (Word) an die Rundbriefredaktion zu senden. Ein Ausdruck sollte keine Seitennummerierung enthalten, wegen der Verkleinerung auf DIN A5 jedoch eine Schrift von mindestens der Größe wie Times Roman 12.

Die Beiträge können in deutscher oder englischer Sprache abgefasst sein. Mit der Zusendung eines Beitrags ist das Einverständnis zur Veröffentlichung im Rundbrief verbunden. Jeder Beitrag wird ohne Begutachtung veröffentlicht.

Herausgeber Fachausschuss *Management der Anwendungsentwicklung und -wartung*
Auflage 500
Redaktion Christian Kop
Institut für Artificial Intelligence and Cyber Security
Universität Klagenfurt
A-9020 Klagenfurt
E-mail: christian.kop@aau.at
Tel.: +43 463 2700 3735
Fax: +43 463 2700 993735

Redaktionsschluß für das nächste Heft: 31.01.2022