Combining Design Thinking and Agile Development to Master Highly Innovative IT Projects

Leonard Przybilla[1], Maximilian Schreieck[1], Kai Klinker[1], Christoph Pflügler[1], Manuel Wiesche[1] and Helmut Krcmar[1]

Abstract: Agile development methods have become mainstream. Notwithstanding the improvements they bring about in implementation, they are of little help for deciding what exact features are needed to address the core needs of customers: they mostly rely on the competence and domain knowledge of the product owner. This is an issue of paramount importance in innovative projects with high ambiguity such as digitization projects because such projects require a detailed understanding of customers and their needs. In order to address this gap, we propose to follow a Design Up Front approach and to integrate the Design Thinking methodology, which aims at human-centered innovation, with agile development. Drawing on 25 student and research projects, we report key learnings concerning human aspects, knowledge management, and challenging of assumptions. Moreover, we offer practical recommendations for the integration of the two methodologies.

Keywords: Agile development; Design Thinking; Design Up Front; Requirements Engineering

1 Introduction

Agile methods have attained widespread popularity that continues to grow [Ve17]. Many advantages of agile methods have been identified and investigated in empirical studies. As an example, agile practices have been demonstrated to increase project success [SP15]. As a contingency, adjusting team characteristics can influence different dimensions of performance [LX10]. In addition, agile practices are found to interact with other phenomena such as subgrouping [La17, PWK18]. A core aspect of agile methods is reacting flexibly to changes and the admission that the course of a project cannot be fully controlled in advance [Di12]. To embrace this fact and effectively attain the project goals, an iterative approach in which the developed artifacts are repeatedly presented to the customer and evaluated is used [Di12].

The preceding description does, however, implicitly state a condition for an agile project to be successful: The goal of development has to be sufficiently known. Agile approaches such as Scrum typically employ roles for managing responsibilities and workload within the development team [Ki07]. Formally, the product owner is responsible for what should be developed, while the rest of the team is mainly focused on how the aim can be achieved. Agile development is thus beneficial in cases when the project goals are known by the product owner, but is of little use if the goal as such is unknown. To illustrate this potential gap, Augustine et al. [Au05] propose to first describe the product vision in the development team – without addressing how this vision has been developed. Extant literature underlines the benefits and new challenges of adequate requirements engineering in agile

[1] Technische Universität München, Lehrstuhl für Wirtschaftsinformatik, Boltzmannstr. 3, 85748 Garching, {leonard.przybilla, maximilian.schreieck, kai.klinker, christoph.pfluegler, wiesche, krcmar}@in.tum.de

development [In15]. In the development of an update release or an incremental improvement of an existing product, specifying the goal of development may be straightforward. In the case of radical innovation in a domain characterized by high uncertainty, the project goal itself is not yet known and cannot be as readily established–which may render the advantages of agile methods moot. An exemplary domain that illustrates this lack of goal clarity is the digital transformation of industrial value creation: In theory, there is a multitude of application areas, yet what exactly a valuable solution should look like is unknown [An18]. In this context of high uncertainty, the fundamental question is what features are exactly of value to the customer and user.

While there is research on user-centricity in agile development, it is mostly focused on achieving adequate usability and improved user experience in interacting with the UI (e.g., Da Silva et al. [Si11]). The pivotal question of what should be developed can hardly be answered by these approaches. A possibility to align the goal of a highly innovative project with maximum utility for users is offered by the Design Thinking methodology. While combining Design Thinking with agile development has been proposed before [AMC13], we focus on the integration of Design Thinking and Scrum and the handover between the two methodologies. To this end, we report on learnings and evidence drawn from 25 heterogeneous projects in an academic context.

2 Related Works

In the following, we will give a brief overview of agile development methodologies, Design Thinking, and research on combining user-centered development with agile methodologies.

2.1 Agile Development

Driven by the dissatisfaction with then-current methods of software engineering, a group of people put forth a manifesto for agile software development [Be01], which now forms the basis of agile development methods. Among other propositions, the manifesto calls for prioritizing people over process, usable software over documentation, and to integrate customers in development. Building on these propositions, several different agile development methods such as Crystal, eXtreme Programming, or Scrum, have been developed [Di12]. Within these methods, multiple agile practices such as daily stand-ups or retrospectives at the end of sprints are harnessed [Ve17]. In the following, we will mostly focus on Scrum as a commonly used implementation of agile development.

Scrum heeds the call for customer integration put forth in the agile manifesto for example through repeated testing and evaluation of artifacts by customers at the end of sprints [RJ00]. The basic notion of agile response to change also transcends to requirements engineering: Instead of collecting all requirements beforehand, based on several case studies it is reported that they are detailed iteratively throughout the project [RCB10]. Agile requirements engineering relies on several practices that on the one hand address and ameliorate challenges of traditional requirements engineering but on the other hand result in new challenges that are only partly addressed by agile practices [In15]. As an example, Inayat et al. [In15] raise the issue of a lack of traceability brought about by little documentation, considering non-functional requirements, or the inability of customers to make decisions. While no panacea, Design Thinking as a methodology to achieve human-centered innovation can help especially with the latter challenge.

2.2 Design Thinking

Design Thinking is at its core focused on developing human-centered innovation, that is, innovation that addresses needs and wants of the user, while ensuring technical feasibility, and economic viability [Br08b]. As a methodology, Design Thinking encompasses several individual methods and tools, which originally have been used by designers and are now applied to managerial problems [Br08b, JWC13, Ma09]. The methodology is appropriate for wicked problems that cannot be pinpointed and to which no optimal solution can be found [Bu92], e.g. for improving health care provision using digital means [Kl17, Pr18]. Typically, a Design Thinking project is divided into work on the problem space and the solution space. As can be discerned in figure 1, both are first broadened before converging, which leads to a double-diamond shape [De15]. Attributing equal size to the problem and solution spaces signifies that much time is spent on furthering the understanding of the actual problem to be solved.

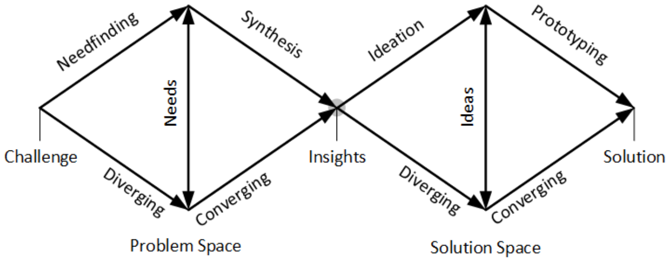

Fig. 1: Design Thinking Project as Double Diamond, adapted from [De15]

Formulating a challenge initiates the diverging part of the problem space through needfinding, which is transferred to the solution space through synthesis. The solution space is opened up through ideation and converges towards a final solution through prototyping and testing activities. A multitude of different methods can be harnessed within these phases. For example, interviews [KB09] or ethnographic observations [EFS11] can be used in needfinding, whereas development of first rough prototypes and their iterative refinement towards a detailed prototype characterize the solution space [He18].

2.3 Human-centered Software Development

Integrating human-centered development with software engineering has been described in several areas. In addition to calls for using Design Thinking in Requirements Engineering [Ve13a], there are several overviews of human-centered or usability-centered agile development [Br08a, SNP14, SPC14, Si11]. A key result of these literature reviews is the identification of a core of methods and practices that are utilized repeatedly. An example from the area of project organization is the Design Up Front principle. This principle entails that resources are dedicated to investigating users before development starts [Si11]. Examples of common artifacts include personas or user stories [Br08a], whereas customer integration and testing are used for evaluation [Br08a]. The main focus of extant research is on the usability of a solution and the related UX design. Defining the core functionality is, however, not within this scope. Further adding to the lack of guidance in this critical phase, while Design Up Front has been proposed as a potential integration mechanism, most evaluations and tests are conducted at the end of development [SNP14].

Drawing on the assertion that combining human-centered methods with agile development is advantageous, we will report on using a Design Up Front approach to improve customer value [Br08a]. In this realm, we draw on earlier suggestions to combine Design Thinking with Scrum. While Vetterli et al. [Ve13b] provide an extensive overview on how the two methodologies can be combined contingent on project factors, we report on empirical learning regarding the handover between Design Thinking and agile development, which has been described as problematic [He18].

3 Three Areas of Learnings to Address for Integration

We will report main learnings on the handover between Design Thinking and agile development. The learnings are derived from a heterogeneous sample of 25 projects in an academic context. Projects include both student projects aiming at radical innovation and research projects seeking to develop prototypical solutions for digitization of specific industries (e.g., [Sc18]). This heterogeneity of projects allows for deducing basic patterns with high generalizability. The key learnings can be grouped in the three categories: human aspects, knowledge transfer, and challenging assumptions (figure 2). These three categories emerged through inductive analysis of observations by the authors and are not meant to be mutually exclusive. Learnings are categorized based on the authors' perception of the most relevant aspect.

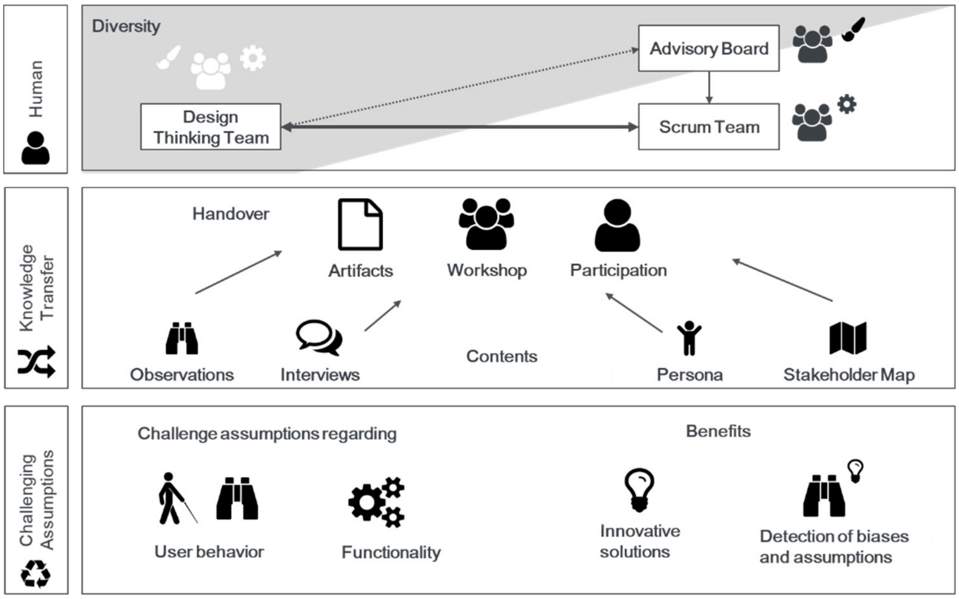

Fig. 2: Overview of Key Areas of Learnings

3.1 Human Aspects

This section summarizes learnings that pertain to organizational aspects in planning and managing teams and the interaction within them. The learnings thus should be considered at the planning stage before embarking on the project.

The main learning on a human level is **that at least one project member**, for example the product owner, should **participate in both the Design Thinking part as well as in the Scrum implementation phase**, mirroring the practice of having a designer on an agile team to improve outcomes [Br08a]. We observed that projects in which both phases were run completely independently, using only artifacts for knowledge transfer and handover, showed problems due to a lack of information. As an example of a communication challenge [In15], in one project the implementation team did not understand the product vision and the underlying customer needs that drive this vision. This was due to a lack of direct customer observations and not being able to retrace the evolution of the preceding prototypes [He18]. Having at least one group member take part in both phases helps tremendously with guarding the product vision as the output of Design Thinking throughout the project. Albeit arguably not as good as direct involvement in both phases, it is advantageous to have participants of the Design Thinking phase at least available in an **advisory board** so that the implementation team can ask for insights. This learning is closely related to the category of knowledge transfer.

Participation of several people in both project phases has the additional benefit of **attenuating potential decision-making biases** that may sprawl if only one person links the two phases. In the case of a single person link, the balance between human desirability, technical feasibility, and economic viability as the cornerstone of Design Thinking [Br08b] may be distorted. Including several people forces them to ponder the underlying project objectives and thus to reduce the chance of biases.

A second aspect on the human level to consider is **team composition**. In the projects in our sample, **functional diversity** had a very positive effect–especially during the Design Thinking phase. It is worth highlighting the valuable contributions made by participants from domains far from the problem domain that greatly advanced the projects. As an example, the perceptions and ideas of nursing scientists were of prime importance for a case of industrial maintenance: With their experience in time-critical flexible care they could relate very well to the situation of use. This led to a very focused feature set and stopped feature creep. Another example are the valuable contributions of mechanical engineers to service projects. The returns on functional diversity diminish over the course of the project. In the end of the Design Thinking phase and in the agile implementation phase, we found diversity to play a lesser role. It is from our experiences thus possible to organize teams in a cascading manner with diverse disciplines participating in the beginning of projects and diversity receding over the course of the project.

3.2 Knowledge Management

Adequate management and transfer of knowledge have emerged as main topics of concern in the projects comprised in our sample. These are especially of relevance in the handover between Design Thinking and agile implementation. While knowledge management is enabled by human factors discussed in the preceding section, the focus in the following is on learnings concerning a shorter timeframe that should be considered while executing projects.

In order to enable a successful and effortless **handover** between Design Thinking and agile implementation, it is recommended to **conduct a handover workshop** in addition to passing on **artifacts** created in the process. This is particularly important if only part of the implementation team participates in the Design Thinking phase–see the preceding learnings on human factors. Doing so can build trust between the participants, which in turn helps with implementing the product idea. A key activity for this workshop is to specify the functions of the envisioned product and their implementation details. Naturally, this is not to say that a complete waterfall-style project plan should be worked out. One option for feature specification would be having Design Thinking participants report on their findings and experiences in detail while simultaneously implementation team members use these reports to fill their backlog. In this setting, if questions arise, the Design Thinking participants would be readily available for clarification. This opportunity of direct interaction is crucial if only few people are involved in both project phases. If the entire team participates in both phases, another aspect should be prioritized: **Changing the team's mindset**: it should progress from the completely open, explorative mindset of Design Thinking to the agile but focused approach to implementation. As much as early fixation on a solution is to be avoided in Design Thinking, questioning core features in implementation should be avoided as well. Agile implementation should focus on the "how" of implementation–"what" to implement should have been established in the Design Thinking phase.

The most important piece of knowledge to be safeguarded and managed is **the detailed and concise specification of the product vision**–the "what" that is to be implemented in an agile manner. To achieve adequate quality, the paramount value of iterative user studies has been observed throughout the projects. Detailed **observations** are well worth the large amount of time invested. As a positive example, in a project with craftsmen the project team was able to identify early on that the planned solution would generate no value but that there was a worthwhile alternative. To further stress this point, two negative examples are provided by projects in which developers engaged only in a short user study before implementing requirements on their own without another test. The resulting prototypes were deemed useless by the users, which could have been known earlier through repeated testing on the conceptual level. While user stories are a common tool in agile development for achieving usability [Br08a, Si11], we strongly warn against deriving them from anything else than real customer needs. To achieve this aim, the methods **ethnographic studies, in-depth interviews, and focus groups** have proven especially useful in our sample of projects, (e.g., [SWK17, Sc16]). **Personas** as a condensed representation of stakeholder characteristics [PG03] allowed for detailed yet concise transfer of knowledge. This enables the team to evaluate their implementation against the needs of the customer without necessarily running out for testing.

For creating and analyzing personas, details that at first sight seem unimportant can make or break value and acceptance of the solution. As an example, in a project with truck drivers it was found that drivers differ on many more dimensions than originally thought. In the following, general characteristics without direct relevance to the challenge turned out to be most important for developing a solution. Moreover, it was very helpful to include not only direct users but also to survey additional stakeholders. To attain this goal, the broader context of use should be identified and observed. To keep different personas manageable and provide structure it is helpful to organize them in a **stakeholder map** [CGL15]. In addition to including more stakeholders for analysis, it has been very beneficial in several projects to **benchmark related processes or problems** in other–sometimes completely unrelated–domains. As an example, the well-known "Tupperware parties" provided helpful information on onboarding mechanisms of platforms.

3.3 Challenging Assumptions

In nearly all projects it was surprisingly important **to challenge or even reverse assumptions** of the team–some of which had been made implicitly. To achieve this goal, the Dark Horse phase proposed in Design Thinking has been very beneficial. In the Dark Horse phase, teams are challenged to relax or even invert their assumptions in order to develop solutions that at first sight seem infeasible [Bu13]. This approach is especially well-suited for wicked problems that can only be treated after the problem has been understood [Bu92]–such as innovation projects. As discussed in literature [Bu13], two beneficial results have been observed: First, the **solutions developed in this phase can turn out to be feasible** and move directly into the final product idea. Second, trying to overcome assumptions helps with **reducing biases and fixation on existing solutions** that are due to overly restrictive assumptions. It is, however, extremely important to document the nature of these results for handover if no personal involvement in both phases is given. If this is insufficient, implementers may discard ideas altogether as infeasible. As an example, in a project on vehicle lighting, overturning the assumption that light has to be emitted from headlights was part of the final prototype.

In addition to testing assumptions relating to the solution space and degree of innovation, **assumptions on addressing and communicating** with customers are to be challenged. Again, challenging these assumptions can present handover problems if the underlying rationale is not evident to implementers. In one project, for example, the targeted group in principle welcomed the solution idea. This was, however, not evident in early field tests during the concept phase since the team had chosen a description based on scientific data that customers obviously did not understand and consequently rejected. It is thus recommendable to also challenge assumptions pertaining to customer communication in testing since they can directly affect results and thus indirectly idea generation.

4 Recommendations for Integrating Design Thinking with Agile Development

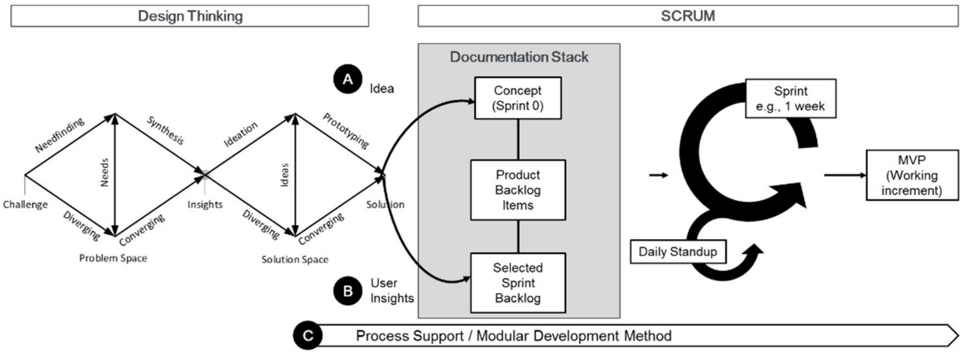

Fig. 3: Overview of Integration of Design Thinking (based on [De15]) with an Exemplary Scrum Process

In addition to the key learnings described in the preceding section, we would like to offer some practical recommendations for integrating Design Thinking as a methodology for product scope definition with agile methods for implementation. From the experiences made, it is recommendable for **highly innovative projects to conduct a complete Design Thinking project before the Scrum process**. This allows for clearly identifying the goal of development. As shown in figure 3, **three key items are handed over from the upfront Design Thinking Project to the Scrum process**: the core idea of what should be developed (A) supported by the user insights (B) explaining, justifying and providing traceability. The user insights provide the explanations of why this idea came into existence and help the team in prioritizing aspects and thus in managing the spring backlog. Moreover, we have found upfront Design Thinking to support the agile implementation from a process perspective (C): Experienced or at least documented customer needs can preempt lengthy discussions on the direction to take.

The most important recommendation that can be derived from our experience is the **willingness to experiment** and to leave the beaten path. Even when having established a Design Thinking-infused process, it is recommendable to reflect on the practices applied. In nearly every project, we ponder the specifics and evaluate which methods could be advantageous at least once every couple of months.

We recognize the scarcity of resources and propose to **tailor integration** according to project needs. For highly innovative projects, it is recommended to conduct an entire Design Thinking project and to have all developers engage in the corresponding activities as described above. While at first sight this procedure violates the principle of delivering working code fast [Be01], implementing a solution that is not what is really needed and has to be adjusted in final testing or fails altogether is even more costly than doing thorough needfinding in the beginning to grasp the core problem. Involvement of all developers helps with more agile implementation since the shared problem understanding arguably reduces the need for formal documentation and thus implements an agile principle [Be01].

This being said, not all projects need or provide the opportunity for the same level of Design Thinking activities: Projects with defined goals that are not wicked may not need an entire Design Thinking project [HU18]–especially since excessive application of techniques can lead to negative outcomes [WSK13]. Since all projects in our sample were highly innovative and exhibited the corresponding level of ambiguity, not all of our learnings and recommendations may be generalizable to other project settings. If, however, some aspects of the project scope are not clearly defined it is beneficial to engage in at least some Design Thinking upfront. Even having just a few people spend some time, e.g. half a day, on structured needfinding, rapid prototyping, and testing can in our experience go a long way towards improving project outcomes. If only a limited subset of developers takes part in Design Thinking, thorough documentation regains paramount importance to starve off communication lapses.

5 Conclusion

Agile methods have been established as mainstream approaches to development. While they work well in implementing solutions, they fail to answer the question of what should be implemented. This is especially problematic for highly innovative projects such as digitization efforts in which the exact project goal is not yet known (see e.g. [SW17, SWK18]). To overcome this problem, we propose to follow a Design Up Front approach using the Design Thinking methodology. We report on key learnings on integrating Design Thinking and agile methodologies obtained in 25 student and research projects–focusing on issues of handover between the phases. They pertain to the three categories human aspects, knowledge management, and challenging assumptions. These have been observed to be important for the fruitful integration of Design Thinking with agile development. In addition to the case-based learnings we offer practical recommendations for tailoring and integrating the two methodologies.

Acknowledgement

Support for this project was provided by the German Federal Ministry for Economic Affairs and Energy (01MD15001D) and the German Federal Ministry of Education and Research (01FJ15058).

References

[AMC13] Adikari, S. ; McDonald, C. ; Campbell, J.: Reframed Contexts: Design Thinking for Agile User Experience Design. In: International Conference of Design, User Experience, and Usability, LNCS 8012, Springer Berlin Heidelberg, Berlin u.a., S. 3–12, 2013.

[An18] Andersson, P. ; Movin, S. ; Mähring, M. ; Teigland, R. ; Wennberg, K.: Managing Digital Transformation, Stockholm School of Economics Institute for Research, 2018.

[Au05] Augustine, S. ; Payne, B. ; Sencindiver, F. ; Woodcock, S.: Agile Project Management: Steering from the Edges. Communications of the ACM 12(48)/2005, S. 85–89, 2005.

[Be01] Beck, K. ; Beedle, M. ; Cockburn, A. ; Cunningham, W. ; Grenning, J. ; Highsmith, J. ; Jeffries, R. ; Kern, J. ; u. a.: Manifesto for Agile Software Development, 2001.

[Br08a] Brhel, M. ; Meth, H. ; Maedche, A. ; Werder, K.: Exploring Principles of User-Centered Agile Software Development: A Literature Review. In: Proceedings of the 2008 Agile Conference (AGILE 2008), S. 63–72, 2008.

[Br08b] Brown, T.: Design thinking. Harvard Business Review 6(86)/2008, S. 84–92, 2008.

[Bu13] Bushnell, T. ; Steber, S. ; Matta, A. ; Cutkosky, M. ; Leifer, L.: Using A 'Dark Horse'Prototype to Manage Innovative Teams. In: 3rd International Conference on Integration of Design, Engineering and Management for Innovation, 2013.

[Bu92] Buchanan, R.: Wicked problems in design thinking. Design issues 2(8)/1992, S. 5–21, 1992.

[CGL15] Chasanidou, D. ; Gasparini, A.A. ; Lee, E.: Design thinking methods and tools for innovation. In: International Conference of Design, User Experience, and Usability, Springer, S. 12–23, 2015.

[De15] The Design Process: What is the Double Diamond?, https://www.designcouncil.org.uk/news-opinion/design-process-what-double-diamond, 2015.

[Di12] Dingsøyr, T. ; Nerur, S. ; Balijepally, V. ; Moe, N.B.: A decade of agile methodologies: Towards explaining agile software development. Journal of Systems and Software 6(85)/2012, S. 1213–1221, 2012.

[EFS11] Emerson, R.M. ; Fretz, R.I. ; Shaw, L.L.: Writing ethnographic fieldnotes, University of Chicago Press, 2011.

[He18] Hehn, J. ; Uebernickel, F. ; Stöckli, E. ; Brenner, W.: Designing Human-Centric Information Systems: Towards an Understanding of Challenges in Specifying Requirements within Design Thinking Projects. In: Multikonferenz Wirtschaftsinformatik 2018, Lüneburg, 2018.

[HU18] Hehn, J. ; Uebernickel, F.: Towards an understanding of the Role of Design Thinking for Requirements Elicitation – Findings from a Multiple-Case Study. In: 24th Americas Conference on Information Systems, New Orleans, 2018.

[In15] Inayat, I. ; Salim, S.S. ; Marczak, S. ; Daneva, M. ; Shamshirband, S.: A systematic literature review on agile requirements engineering practices and challenges. Computers in Human Behavior (51)/2015, S. 915–929, 2015.

[JWC13] Johansson-Sköldberg, U. ; Woodilla, J. ; Çetinkaya, M.: Design Thinking: Past, Present and Possible Futures. Creativity and Innovation Management 2(22)/2013, S. 121–146, 2013.

[KB09] Kvale, S. ; Brinkmann, S.: InterViews: Learning the Craft of Qualitative Research Interviewing, SAGE, 2009.

[Ki07] Kim, Y.: Analyzing scrum agile software development with development process, social factor, and project management lenses. In: Association for Information Systems - 13th Americas Conference on Information Systems, AMCIS 2007: Reaching New Heights, 2007.

[Kl17] Klinker, K. ; Fries, V. ; Wiesche, M. ; Krcmar, H.: CatCare : Designing a serious game to foster hand hygiene compliance in health care facilities. In: Twelfth international conference on Design Science Research in Information Systems and Technology, S. 20–28, 2017.

[La17] Lassak, S. ; Przybilla, L. ; Wiesche, M. ; Krcmar, H.: Explaining How Agile Software Development Practices Moderate the Negative Effects of Faultlines in Teams. In: ACIS 2017 - Australasian Conference on Information Systems, Hobart, 2017.

[LX10] Lee, G. ; Xia, W.: Toward Agile: an Integrated Analysis of Quantitative and Qualitative Field Data on Software Development Agility. MIS Quarterly 1(34)/2010, S. 87–114, 2010.

[Ma09] Martin, R.: The Design of Business: Why Design Thinking is the Next Competitive Advantage, Harvard Business Press, Boston, 2009.

[PG03] Pruitt, J. ; Grundin, J.: Personas : Practice and Theory. Proceedings of the 2003 conference on Designing for user experiences /2003, S. 1–15, 2003.

[Pr18] Przybilla, L. ; Klinker, K. ; Wiesche, M. ; Krcmar, H.: A Human-Centric Approach to Digital Innovation Projects in Health Care: Learnings from Applying Design Thinking. In: 22nd Pacific Asia Conference on Information Systems (PACIS), Yokohama, 2018.

[PWK18] Przybilla, L. ; Wiesche, M. ; Krcmar, H.: The Influence of Agile Practices on Performance in Software Engineering Teams: A Subgroup Perspective. In: Proceedings of the 2018 ACM SIGMIS Conference on Computers and People Research, SIGMIS-CPR'18, ACM, Niagara Falls, S. 33–40, 2018.

[RCB10] Ramesh, B. ; Cao, L. ; Baskerville, R.: Agile requirements engineering practices and challenges: an empirical study. Information Systems Journal 5(20)/2010, S. 449–480, 2010.

[RJ00] Rising, L. ; Janoff, N.S.: The Scrum Software Development Process for Small Teams. IEEE Software August/2000, S. 26–32, 2000.

[Sc16] Schreieck, M. ; Pflügler, C. ; Dehner, C. ; Vaidya, S. ; Bönisch, S. ; Wiesche, M. ; Krcmar, H.: A concept of crowdsourced delivery for small local shops. In: GI-Jahrestagung, 2016.

[Sc18] Schreieck, M. ; Pflügler, C. ; Wiesche, M. ; Krcmar, H.: Ideengenerierung durch Accelerator und Entwicklerportale BT - Management digitaler Plattformen: Konzeption und Realisierung eines offenen Ökosystems für intelligente Mobilitätsdienste in der Smart City. In: Springer Fachmedien Wiesbaden, Wiesbaden, S. 359–386, 2018.

[Si11] Da Silva, T.S. ; Martin, A. ; Maurer, F. ; Silveira, M.: User-centered design and agile methods: A systematic review. In: Proceedings - 2011 Agile Conference, Agile 2011, S. 77–86, 2011.

[SNP14] Salvador, C. ; Nakasone, A. ; Pow-Sang, J.A.: A systematic review of usability techniques in agile methodologies. In: Proceedings of the 7th Euro American Conference on Telematics and Information Systems - EATIS ’14, 2014.

[SP15] Serrador, P. ; Pinto, J.K.: Does Agile work? - A quantitative analysis of agile project success. International Journal of Project Management 5(33)/2015, S. 1040–1051, 2015.

[SPC14] Salah, D. ; Paige, R.F. ; Cairns, P.: A systematic literature review for agile development processes and user centred design integration. In: Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering - EASE ’14, 2014.

[SW17] Schreieck, M. ; Wiesche, M.: How Established Companies Leverage IT Platforms for Value Co-Creation – Insights from Banking. In: Proceedings of the 25th European Conference on Information Systems (ECIS), Guimarães, S. 1726–1741, 2017.

[SWK17] Schreieck, M. ; Wiesche, M. ; Krcmar, H.: Governing nonprofit platform ecosystems– an information platform for refugees. Information Technology for Development 3(23)/2017, S. 618–643, 2017.

[SWK18] Schreieck, M. ; Wiesche, M. ; Krcmar, H.: Multi-Layer Governance in Platform Ecosystems of Established Companies. In: Academy of Management Proceedings, Academy of Management, 2018.

[Ve17] Versionone: 11th Annual State of Agile Report. 2017.

[Ve13a] Vetterli, C. ; Brenner, W. ; Uebernickel, F. ; Petrie, C.: From palaces to yurts: Why requirements engineering needs design thinking. IEEE Internet Computing 2(17)/2013, S. 91–94, 2013.

[Ve13b] Vetterli, C. ; Uebernickel, F. ; Brenner, W. ; Haeger, F. ; Kowark, T. ; Krueger, J. ; Mueller, J. ; Plattner, H. ; u. a.: Jumpstarting Scrum with Design Thinking. University of St.Gallen, St.Gallen, 2013.

[WSK13] Wiesche, M. ; Schermann, M. ; Krcmar, H.: When IT risk management produces more harm than good: The phenomenon of „mock bureaucracy“. In: Proceedings of the Annual Hawaii International Conference on System Sciences 2013, S. 4502–4511, 2013.