THE BEST OF THREE WORLDS - THE CREATION OF INNODEV A SOFTWARE DEVELOPMENT APPROACH THAT INTEGRATES DESIGN THINKING, SCRUM AND LEAN STARTUP

Dobrigkeit, Franziska (1); de Paula, Danielly (2)
1: Hasso-Plattner-Institut, Germany; 2: National University of Ireland Galway, Ireland

Abstract

Agile development has been important to software engineering for decades. However, limitations in existing agile methods such as Scrum and eXtreme Programming still persist and have given rise to efforts aiming to integrate agile software development with other iterative and innovative practices such as Deisgn Thinking or Lean Startup. This paper aims to identify best practices and valuable proposals from such integration efforts, in order to create a new process model, which aggregates the core elements and key aspects of the former works. The research is based on a literature review and a cross-case analysis of two already existing models, MoIT and DT@Scrum. Relevant aspects of both models are presented and discussed in relation to other relevant research in this area. The analysis shows that an integration of Scrum practices with Design Thinking and Lean Startup could be favorable. Furthermore, the combined process model InnoDev is presented.

Keywords: Design Thinking, User centred design, Software development, Multi- / Cross- / Trans-disciplinary processes, Process modelling

Contact:
Franziska Dobrigkeit
Hasso-Plattner-Institut
Enterprise Platform and Integrations Concepts Group
Germany
franziska.dobrigkeit@hpi.de

Please cite this paper as:
Surnames, Initials: *Title of paper*. In: Proceedings of the 21st International Conference on Engineering Design (ICED17), Vol. 8: Human Behaviour in Design, Vancouver, Canada, 21.-25.08.2017.

1 INTRODUCTION

How to organize agile development in order to deliver faster, better, and cheaper solutions is an ongoing discussion in software engineering circles (Dybå and Dingsøyr 2008). Limitations found in agile methods, such as Scrum, include problem understanding and solution finding (Lindberg et al. 2011), scaling (Vilkki 2010), and lack of attention towards design by (Dybå and Dingsøyr 2008). These limitations can lead to severe consequences such as: a company launching the "wrong" products, resulting in poor market reception, or necessary rework requiring extra engineering hours and investments (Verganti 1997). In order to improve problem understanding and solution, as well as attention towards design, researchers suggests the integration of Design Thinking with agile software development (Lindberg et al. 2011). Design thinking is a set of practices that helps organizations to solve complex problems by reducing bias, encouraging innovation and inspiring people to become more creative (Liedtka 2011). Such an integration has been researched in different settings, such as large business organizations (Häger et al. 2015), startup-like environments (de Paula and Araújo 2016) and educational settings (Lindberg et al. 2011).

Lean Startup is considered as a complementary approach to Design Thinking and agile software development that helps to create scalable plans (Grossman-Kahn and Rosensweig 2012; Hildenbrand and Meyer 2012; de Paula and Araújo 2016). Lean Startup is a "set of practices for helping entrepreneurs increase their odds of building a successful startup" (Ries 2011, p.37) by improving productivity, time-to-market, product quality and customer satisfaction (Rodríguez et al. 2012). According to (Grossman-Kahn and Rosensweig 2012), using the three approaches together makes particular sense because Design Thinking provides a roadmap to creative and human-centered solutions, agile methods optimize the process and enable the team to move quickly and Lean Startup provides a framework to validate and measure the product life cycle. However, a seamless integration of Design Thinking and Lean Startup into agile development processes of software development companies is still subject to research (Lindberg et al. 2011; Rodríguez et al. 2012). To increase the likelihood of a successful integration of Design Thinking, Lean Startup and agile into one process, this study aims to develop a new conceptual model by singling out the relevant dimensions that may improve the new software development process from two already existing models: DT@Scrum proposed by Häger et al. (Häger et al. 2015) and MoIT proposed by de Paula (de Paula and Araújo 2016). DT@Scrum is "an approach that combines Design Thinking and Scrum in order to create an agile software development process that can deliver the innovative customer-oriented products and services required by competitive companies" (Häger et al. 2015, p.2). MoIT, on the other hand, is an approach that combines Design Thinking, Scrum, and Lean Startup in order to teach startups on how to develop innovative products. We first extract detailed knowledge of the two models' composition using the principle of Method Engineering (Ter Hofstede and Verhoef 1997). Second, a cross-case analysis is performed by pattern matching as suggested by Yin (Yin 2003). The results of the comparison are further analysed to identify relevant aspects that can be aggregated into a new conceptual model, called InnoDev aimed at enhancing the innovativeness in IT development.

2 LITERATURE REVIEW

Agile methods (e.g. Kanban, Scrum) have been recommended for software development due to its benefits in relation to reducing the development time, increasing the flexibility and quality of the product (Erickson et al. 2005, p.89). In particular, Scrum focuses on project management in situations where it is difficult to plan ahead; with mechanisms such as feedback loops, self-organizing teams and sprints as its core elements (Schwaber and Beedle 2001). However, a comprehensive systematic literature review conducted by (Dybå and Dingsøyr 2008) concluded that a limitation that has repeatedly been mentioned in the literature related to Scrum is the lack of attention to design. In accordance with this finding, (Lindberg et al. 2011) discuss problem understanding and solution finding as another limitation of IT teams in general and especially agile teams. In order to overcome this restriction and encourage more interdisciplinary collaboration, there have been serious efforts (Lindberg et al. 2011; Hildenbrand and Meyer 2012) to introduce design methods, especially Design Thinking to IT development. In these efforts Deign Thinking is commonly used as a means to requirements analysis and elicitation before the actual agile development. At the core of Design Thinking are four key elements: the iterative process including various methods and tools supporting each phase, multidisciplinary teams, creative space and

a designer's mindset (Wölbling et al. 2012). According to (Beverland and Farrelly 2007), Design Thinking offers a potent way to develop superior products and facilitate product appropriateness by enhancing team collaboration and improving idea generation. Similarly, it has been argued that incorporating Design Thinking into the software development process can result in cost savings due to reductions in redesign work, as well as shortening the length of the process itself (Lindberg et al. 2011). By making a comparison between Design Thinking and agile practices, it is possible to identify some strong parallels such as team collaboration (Schwaber and Beedle 2001), direct iterative learning (Larman 2004) and the ability to solve wicked problems.

Even though a number of studies on how to integrate Design Thinking with agile practices have been conducted, some authors suggest that there are limitations in using only those two approaches. The main limitation mentioned is that neither Design Thinking nor agile practices offer support on how to track growth and how to scale the product after its launch (Vilkki 2010; Grossman-Kahn and Rosensweig 2012). For instance, in the context of SMEs and startups, scalability represents the most important architectural aspect and should be addressed as soon as possible (Thorpe et al. 2005). Therefore, a balance must be found between flexibility and repeatability in their organizations' knowledge management and processes in order to plan for scale (Paternoster et al. 2014). In this regard, several authors propose to use Lean Startup to address those limitations (Grossman-Kahn and Rosensweig 2012; Hildenbrand and Meyer 2012; de Paula and Araújo 2016). A core component of Lean Startup is the build-measure-learn life cycle, which provides guidance on how to develop a product that meets its value proposition, the MVP – Minimum Viable Product without waste (Ries 2011). In addition, the lean life cycle includes actionable metrics, AARRR - Acquisition, Activation, Retention, Revenue and Referral, which can be used to assess the product performance according to the users acceptance (Maurya 2012). For instance, to measure acquisition it is possible to use a specific test environment that allows for quantitative measuring of user feedback, such as a landing page. Another core element of the Lean Startup is the concept of pivot. "A pivot is a special kind of change designed to test a new fundamental hypothesis about the product, business model, and engine of growth" (Ries 2011, p.168).

Table 1. Design Thinking-agile models for software development

Model	Specialty	Focus	Target group
(Grossman-Kahn and Rosensweig 2012)	Lean Startup was integrated and tested in a laboratory.	identify the solution + deliver a prototype.	Startups.
(Hildenbrand and Meyer 2012)	The use of lean thinking concepts throughout the development process.	identify + implement the solution.	Inexperienced teams.
(Häger et al. 2015)	Scrum is used to structure Design Thinking activities.	identify + implement the solution.	Large software organizations.
(de Paula and Araújo 2016)	Lean Startup was integrated and tested with undergraduate students.	identify, implement + scale the solution.	Startups.

The characterization of new theories on how to integrate Design Thinking into the agile process has been progressing in the literature. Table 1 summarizes some of the existing models. (Grossman-Kahn and Rosensweig 2012) propose a design-led, multidisciplinary model to build innovation capacity through the integration of diverse innovation methodologies such as Design Thinking, Lean Startup and agile practices. By validating the model with a team from the Nordstrom Innovation Lab, the authors suggest that software development teams should be guided by a clearly defined set of end goals and mindsets, rather than a rigid adherence to specific tools or processes. Similarly, (Hildenbrand and Meyer 2012) introduced the concept of lean thinking and developed a model using Design Thinking and agile methods to optimize the training experience for software professionals and their coaches. The authors suggest that lean thinking is closely intertwined in many ways with Design Thinking and they complement each other very well. (Häger et al. 2015) present DT@Scrum, a process model for large organizations that seamlessly integrates Design Thinking and Scrum. Unlike the other models, the authors use agile concepts, such as sprints and backlogs, to plan and structure the Design Thinking activities. (de Paula and Araújo 2016) developed a model using agile, lean start up and Design Thinking by combining the models of (Grossman-Kahn and Rosensweig 2012) and (Hildenbrand and Meyer

2012). It is based on previous research (de Paula and Araújo 2016) and aims to identify, implement and scale solutions in a startup environment.
Even though there are many models, a generally accepted model has not yet emerged. In order to come to an agreement on the core concepts for software development, it is required to balance the existing contributions by making a comparison between the most recent models in order to identify and analyse relevant aspects from each model.

3 RESEARCH METHOD

This study entails a qualitative comparative cross-case analysis (Miles and Huberman 1994) of two software development models, DT@Scrum proposed by (Häger et al. 2015) and MoIT proposed by (de Paula and Araújo 2016) in order explore how Design Thinking, Lean Startup and Scrum can be successfully integrated into the software development process. Qualitative comparative analysis can be used to accumulate, organize, and interpret the studies, with the aim of achieving a level of understanding that transcends the results of the individual studies (Ragin 1987). For this study, the qualitative comparative cross-case analysis was adapted from (Yin 2003) and consists of three main activities: to develop a detailed analysis of the selected models, compare concepts and relations that emerged from the analysis and develop a new model based on the comparison analysis.

3.1 Research questions

In this article, the following central research question guided the next steps: How can Design Thinking, Lean Startup and agile practices be successfully integrated into one software development process? The following specific research question was used to guide the comparison analysis and synthesis of results: What are the relevant aspects of a software development process used by a startup environment that could be used by large organizations and vice versa?

3.2 Case selection

In order to answer the specific research question, we aim to compare to models designed for an organizational environment and a start-up environment. The case selection combined criterion sampling and convenience sampling (Patton, 1990). The two models were included in this study because of their different targets. While DT@Scrum aims to combine Design Thinking and Scrum to be used by software teams in large organizations, MoIT aims to support startups to develop innovative products using Design Thinking, agile practices and Lean Startup. This variation increases the likelihood to generate theoretical insights from a cross-case analysis (Yin 2003). Additionally, both cases were easily accessible due to existing personal connections, and there were no confidentiality issues to collect the data and publish the results. The table below summarizes the characteristics of the selected models.

Table 2. The characteristics of the selected models

	DT@Scrum	**MoIT**
Target group	Large companies	Startups
Product	Innovative Software	Innovative Software
Characteristics	Combines Design Thinking and Scrum.	Combines Design Thinking, agile practices and Lean Startup.

3.3 Data Collection

Two main data sources were used for MoIT: a published paper (de Paula and Araújo 2016) that details the technical aspects of the model, and observational first-hand experiences of one of the authors who had participated in the creation and evaluation of the model. For DT@Scrum, there were also two main data sources: a published paper (Häger et al. 2015) reporting the technical aspects of the model and its evaluation, and also the details from one of the authors who had participated in the development of the model.

3.4 Data Analysis

For this study, qualitative cross-case analysis (Miles and Huberman 1994; Yin 2003) was conducted by comparing and synthesizing the two models. Following an approach by (Müller and Thoring 2012)

Method Engineering (ME) was chosen to extract detailed knowledge of the two models' composition and allow for comparison. ME is concerned with the detailed description, design, adaption, and evaluation of methods, using engineering principles (Ter Hofstede and Verhoef 1997). By providing a detailed analysis of a method, it is possible to allow other researchers to reproduce the method and test its utility claims. Similar to (Thoring and Müller 2011), we use a subset of Business Process Modelling Notation (BPMN), as shown in Figure 1, to describe the constructs associated with each model. While BPMN is not usually used to model iterative processes, (Thoring and Müller 2011) have shown that it can be useful to model the phases, basic tasks, outcomes and decisions relevant to a process model, in their case Design Thinking, thereby allowing for a detailed comparison of these elements, as was done in (Müller and Thoring 2012).

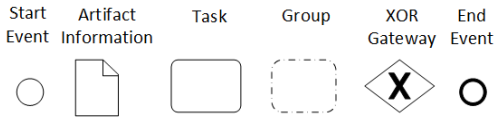

Figure 1. Used symbols of BPMN.

Second, the cross-case analysis was performed by pattern matching as suggested by (Yin 2003). The results of the comparison were further analysed to identify relations between the models. The main similarities and differences were synthesized and illustrated in Table 3. Subsequently, a new conceptual model - InnoDev - was developed that aggregates relevant aspects of each model. InnoDev, aims to improve the software development process.

4 COMPARATIVE ANALYSIS

To reverse-engineer the two models, we used the BPMN to describe the constructs associated with each model. The methods and tools and the aspired outcome of each step are presented. For simplicity, the iterative and interconnected nature of the Design Thinking activities as well as the Scrum sprints have been omitted.

4.1 MoIT

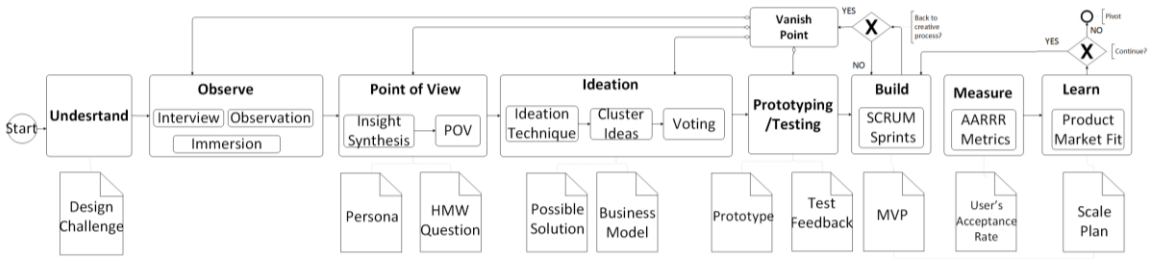

Figure 2. BPMN description of MoIT.

The MoIT process, as depicted in Figure 2, starts with the standard phases of the Design Thinking process as described by (Wölbling et al. 2012). As the model addresses inexperienced startups, it prescribes methods to use and outcomes to achieve for each of the steps. For example, the model prescribes creating a Persona and a How-Might-We- Question as part of the step Point of View in order to simplify the process for novice Design Thinking teams. During the Design Thinking phase, various forms of prototypes are developed. For team-internal discussion low-fidelity prototypes, like paper UIs are suitable while testing with users require more medium-fidelity prototypes, which can be tested in the relevant context. For example, mobile games should be tested on a phone by using wireframes or mock-ups. Following the Design Thinking phase, the team moves on to Build the product following Scrum practices and lean concepts. During this phase, it is important to already implement ways of collecting data according to AARRR, a metric-based evaluation technique that is commonly used in Lean Startup. The goal is to find some validation of a 'product-market fit' and to answer the question if the developed product is something that people want (Maurya 2012). During the build phase, the MoIT model also introduces the concept of "micro problems" that should be addressed using Design Thinking. A micro problem could take place during product implementation, and is derived from the main problem. Micro problems faced by the team during the Scrum sprints, should be solved by using the

"vanishing point" to go back to the Design Thinking phase. By the end of the Build phase, the team will have launched the MVP. In the following measure phase the user's acceptance can be monitored and evaluated according to the defined AARRR metrics. During the Learn phase the team will use these measurements to decide whether to continue to develop the product or pivot to a new idea. If the team decides to pivot, they discard the current product and start over. If the team decides to continue, they should analyse the user's' feedback and identify problems and solve them, or add new features accordingly.

4.2 DT@Scrum

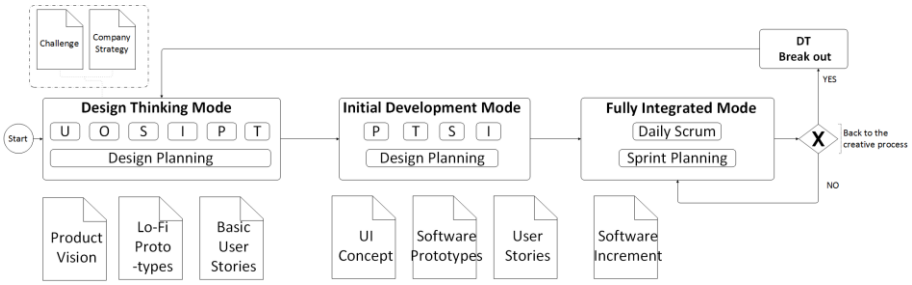

Figure 3. A detailed description of DT@Scrum.

DT@Scrum, as depicted in Figure 3, proposes three phases that gradually move from Design Thinking to software development: the Design Thinking Mode, the Initial Development Mode and the Fully Integrated Mode. The Design Thinking Mode is used to explore the problem and solution spaces and delivers a product vision that solves at least one identified problem. It follows the standard phases of the Design Thinking process as described by (Wölbling et al. 2012) in an iterative manner. Following this phase, the Initial Development Mode allows the team to refine the concept, by implementing and testing UI concepts, technology stacks and first features thus ensuring the viability, desirability and feasibility of the concept. The final phase, the Fully Integrated Mode, is used to gradually develop the software system according to the concept. During this phase, the model introduces DT-Breakouts, a concept which allows the team to go back into a Design Thinking working mode in order to define or refine features or solve blockers that they came across during development. According to the progress throughout the three-phase process, the artefacts and especially the prototypes used during the phases move from being of low fidelity (e.g. paper UI or vision poster) in the first phase to a medium fidelity (e.g. clickable wireframes or first proof of concept implementations) in the second phase to high-fidelity prototypes (e.g. actual software on final technology stack) in the final phase. In large organizations, management approval can be required at certain points of a project. In this way, the transitions between the three phases form a natural gateway that can be connected to an approval meeting, allowing management to discuss the project with the team.

Additionally, Scrum is proposed as the overall process framework underlying all three phases. Accordingly, DT@Scrum proposes Design Planning as an adaption of Scrum to Design Thinking activities in order to let design teams get a feeling for the duration and value of Design Thinking activities, and to enable them to better structure their creative work. Accordingly, it proposes to structure the Design Thinking phase in several one or two-week sprints, including backlog creation, sprint planning, and evaluations in retrospective meetings.

DT@Scrum proposes measures to ensure continuity in the team. First, the Design Thinking team should be composed of members from all relevant departments, e.g. sales, development, design. Second, the transition between modes allows to make changes with regards to the team and either scale the team or switch team members. Third, DT@Scrum proposes to keep at least one member of the original design team in the role of the Product Owner, to ensure knowledge is passed on correctly.

4.3 Cross-Case Analysis

This subsection presents a detailed comparison of MoIT and DT@Scrum based on the aforementioned data sources (papers and thesis) and the detailed description from the previous subsection. Table 3 provides an overview and comparison of relevant aspects of both innovation strategies. We compare the goals, methods, specific process steps and the respective target groups.

4.3.1 Similarities

Both models have the same goal, to develop innovative software products. However, the target group is different. While MoIT is mainly focusing on inexperienced teams in startups, DT@Scrum is seeking to improve the software development process for large organizations. Both models are using Design Thinking and Scrum as approaches to structuring the development process. However, each model has a unique feature not represented in the other model, which enhances the respective model with specific tools and methods. Both models make use of Design Thinking throughout the development process and propose to use Design Thinking in case of blockers in the development process. Using Design Thinking during the implementation phase is also defended by (Hildenbrand and Meyer 2012) who claim that a team should make use of Design Thinking methods and techniques during the Scrum sprints when they approach new and unclear user stories. Although the names are different (Vanishing Point in MoIT and Design Thinking Break out in DT@Scrum), both models believe that Design Thinking application is ad-hoc. Therefore, a close observation of the development process is needed to quickly react to blockers with the adequate Design Thinking tool. DT@Scrum proposes a role responsible for this task, the Process Master, trained in Scrum and Design Thinking and acting as a Design Thinking coach and Scrum Master, thus facilitating the complete process (Häger et al. 2015).

Table 3. Comparison of important aspects of both models

What	MoIT	DT@Scrum
Goal	Innovative software products	Innovative software products
Approach	Design Thinking, Scrum and Lean Startup	Design Thinking and Scrum
Metrics	Strong focus	Not a focus
Pivot	Strong focus	Not a focus
Design Planning	Not a focus	Strong focus.
Gateways	Not a focus	Can be implemented between phases
Business Model	Implemented before prototyping	Optional before development
DT Methods	Prescribed	Teams Choice
DT in implementation	Yes (“Vanishing point”)	Yes (“Design Thinking break-out”)
Target group	Inexperienced IT startup teams	IT teams in large organizations

4.3.2 Differences

Some of the smaller differences between the two models are due to their different target groups. Concerning the implementation of Design Thinking, MoIT prescribes the method and tools to be used during the process, while DT@Scrum mentions various methods and tools but does not prescribe which ones have to be implemented. Furthermore, DT@Scrum proposes to use the transitions between the different process phases as possible points for management approval, which can be necessary in large enterprises. The more significant differences between the two innovation strategies, derive from the stronger project management focus in DT@Scrum and the integration of Lean Startup into the development process in MoIT. MoIT suggests the use of metric-based evaluation techniques from the Lean Startup literature, to measure user’s acceptance or activity. The literature warns against using “vanity metrics” and recommends the use of actionable metrics such as AARRR that can be linked to the specific business (Ries 2011; Maurya 2012). For generating those metrics and then test the hypothesis, an experiment has to be designed (e.g. evaluating the customer acquisition costs by minimal landing pages at a small scale). In addition, the development of the business model is a crucial aspect recommended by MoIT since startups are still at their early stages in the business environment. The model makes use of the Lean Canvas proposed by (Maurya 2012) and based on the work of (Ries 2011) that helps to systematically align stakeholders, value propositions, required resources, cost and revenue structure, channels, etc. for a startup business model.

According to (Häger et al. 2015) communication between implementation and Design Thinking teams should start early during the projects in order to allow for a realistic assessment of the feasibility of ideas and understanding of the ideas during development. Therefore, they propose to compose the Design Thinking team of members from all relevant areas in the company and keep on at least one of the original Design Team members as the Product Owner during development. Furthermore, DT@Scrum recommends Scrum as the overall framework for both development and Design Thinking activities.

Design Planning adopts methods already known from Scrum to Design Thinking, thus DT@Scrum presents a unique perspective on how to plan Design Thinking activities without compromising the creative process. MoIT does not consider Design Planning, instead it recommends only little structure during the creative process to ensure a good learning experience for the inexperienced teams. However, Design Planning as proposed by DT@Scrum would only provide little structure that is controlled by the design team itself.

4.4 Conclusion of Analysis

We believe, there is potential to improve the software development process by converging the two strategies. The models share an initial Design Thinking phase that is followed by agile sprints to build the software and thus follow a similar general setup allowing for an integration of the approaches. While the minor differences between the models result from the different target groups, the two major differences, the use of sprints and planning during the Design Thinking phase in DT@Scrum and the addition of Lean Start-Up principles in MoIT reflect aspects which are not considered in the respective other model. Both aspects make sense in a software development process no matter whether the target group are start-ups or large organizations. Therefore, we propose a new model aggregating these features.

5 INNODEV

InnoDev, as illustrated in Figure 4, is a three-phase software development process combining elements from Design Thinking, Scrum and Lean Start-Up. Similar to DT@Scrum and MoIT, the initial Design Thinking phase follows the Design Thinking process as described by (Wölbling et al. 2012) to explore the problem and solution spaces and define a product vision addressing at least one of the identified problems. In the following initial development phase, this vision is refined and developed into a proof-of-concept prototype following the idea of an MVP. UI concepts are tested and implemented, technology stacks considered and tested and the most important features are implemented to ensure viability, desirability and feasibility of the concept. Additionally, ways of collecting data to monitor the user's acceptance according to the AARRR metrics are implemented. In the final Development Phase, the MVP will be tested and extended according to the original concept or feedback gained during the build-measure-learn-life-cycle. In this phase, the team will run agile sprints combined with lean practices to establish a build measure learn lifecycle. Depending on the outcome of the learn phase the team might decide to pivot their project or continue to the next sprint. Similar to DT@Scrum and MoIT, InnoDev proposes to make use of Design Thinking tools in an ad-hoc manner in case of blockers or problems related to the product. During all phases, the team makes use of the sprint and backlog concepts from Scrum to plan and structure their activities, thus providing transparency over Design Thinking, design and development activities and being able to move forward while staying flexible to changing requirements.

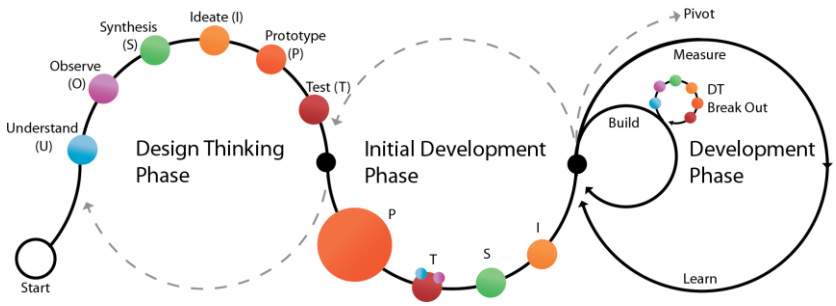

Figure 4. The new process model: InnoDev

We consider InnoDev to be general model applicable to different company settings (e.g. Start-Ups, SMEs, or large organizations). However, differences between these target groups exists and some of those are reflected in specific features of the compared processes as our analysis has shown. Therefore, specific needs that exist in a company should be targeted by adapting InnoDev to the context of the company. For example, the Design Thinking Phase can prescribe methods and tools, similar to MoIT, for an inexperienced team while a more experienced team can choose methods that they find suitable.

Similarly, the transition between the three phases of InnoDev can take on the form of a management approval meeting as proposed by DT@Scrum. If this is not necessary (e.g. in a Start-Up or SME), the team can make the transition decision by itself. While the development of a business model is a crucial aspect no matter the companies size, suitable tools vary. Therefore, the creation of new or the adaption of existing business models should be started during the Initial Development Phase and can be supported by suitable tools, e.g. the lean startup canvas for start-ups or the business model canvas for larger organizations. Our suggestions do not present a complete list of possible adaptions but rather stem from our former analysis and can be extended for other needs.

6 DISCUSSION

This study aimed to develop a new conceptual model by singling out the relevant dimensions from two already existing models that may improve a new software development process. The specific research question intended to identify relevant aspects from a software development process used by startups that could be used by large organizations and vice versa. Based on our analysis, we suggest that large organizations could benefit from the use of Lean Startup. Based on the build-measure-learn life cycle, a company could enhance its software development process by making use of lean concepts such as MVP, actionable metrics, and pivot. Although the universal application of lean principles to software development is still under debate (Staats et al. 2011), our findings are in agreement with findings in (Poppendieck and Cusumano 2012), which showed that some characteristics of software products, such as its value proposition and malleability, open new opportunities for combining agile and lean practices in a software domain. For instance, the use of checklists or specific test environments that allow for quantitative measuring of user feedback (such as landing page design, smoke-test, etc.) would help the team to track growth and scale the product based on “Innovation Metrics. Moreover, pivoting seems to be a promising technique to strengthen the software development process. Early testing of hypotheses, might save time and resources and could result in a better output of successful project results. According to (Ries 2011), pivots are a permanent fact of life for any growing business and even after a company achieves initial success, it must continue to pivot. Accordingly, we consider pivoting as an important aspect of the software development process regardless of the company’s size.
In addition, we suggest that startups could benefit from the use of design planning. The insights from DT@Scrum on design planning might allow the startup to make more informed decisions with regard to implementation and evaluation. Due to the fact that startups develop software under highly uncertain conditions with severe lack of resources, specifying the design activities scope and resources required would help the startup to avoid waste. Since software startups are often familiar with Scrum, Design Planning would not require a change of mindset, thereby facilitating the model's adoption. Our findings contribute to the ongoing discussions (Lindberg et al. 2011; Hildenbrand and Meyer 2012; Häger et al. 2015) of whether planning Design Thinking activities has positive effects on software development. Aggregating these ideas resulted in our newly proposed model, InnoDev.

7 CONCLUSION

This paper provides a structured analysis and comparison of the two innovation strategies - DT@Scrum and MoIT - with the goal to propose one synthesized model that aggregates relevant aspects of each model. The detailed analysis of both innovation strategies contributes to better understand the particular aspects of software development for two different organizational settings: startups and large organizations. The findings provide a deep understanding of the relevant dimensions that can be used to improve the new software development process. Based on the findings, this study suggests that the creation of design planning and a scalable plan should be considered as an important aspect of the software development process regardless of the company’s size. This research concludes that InnoDev has potential to be applied in different settings (startups, SMEs, large organizations, etc.). However, even though the new model was developed based on best practices, it is necessary to implement it in a software development project in order to verify whether our assumptions are correct. Based on that, one limitation of this study is not trying to apply the new model in organizations. To address this limitation, future work will validate the application of the suggested model in one or more case studies. The present work contributes to the academic discussions on design and software development in two ways. First, we present a new conceptual model for software development based on a comparison analysis of software development process from two different organizational setups.

REFERENCES

Beverland, M., Farrelly, F.J. (2007) 'What Does It Mean to Be Design-led?', Design Management Review, 18(4), 10–17.

Dybå, T., Dingsøyr, T. (2008) 'Empirical studies of agile software development: A systematic review', Information and Software Technology, 50(9–10), 833–859.

Erickson, J., Lyytinen, K., Siau, K. (2005) 'Agile modeling, agile software development, and extreme programming: the state of research', Journal of database Management, 16(4), 88.

Grossman-Kahn, B., Rosensweig, R. (2012) 'Skip The Silver Bullet: Driving Innovation Through Small Bets And Diverse Practices', LEADING THROUGH DESIGN, 815–829.

Häger, F., Kowark, T., Krüger, J., Vetterli, C., Übernickel, F., Uflacker, M. (2015) 'DT@Scrum: Integrating Design Thinking with Software Development Processes', in Plattner, H., Meinel, C. and Leifer, L., eds., Design Thinking Research, Understanding Innovation, Springer, 263–289.

Hildenbrand, T., Meyer, J. (2012) 'Intertwining Lean and Design Thinking: Software Product Development from Empathy to Shipment', in Maedche, A., Botzenhardt, A. and Neer, L., eds., Software for People: Fundamentals, Trends and Best Practices, Management for Professionals, Springer, 217–237.

Larman, C. (2004) Agile and Iterative Development: A Manager's Guide, Addison-Wesley.

Liedtka, J. (2011) 'Learning to use design thinking tools for successful innovation', Strategy & Leadership, 39(5), 13–19.

Lindberg, T., Meinel, C., Wagner, R. (2011) 'Design Thinking: A Fruitful Concept for IT Development?', in Meinel, C., Leifer, L. and Plattner, H., eds., Design Thinking, Understanding Innovation, Springer, 3–18.

Maurya, A. (2012) Running Lean: Iterate from Plan A to a Plan That Works, O'Reilly Media, Inc.

Miles, M.B., Huberman, A.M. (1994) Qualitative Data Analysis: An Expanded Sourcebook, Sage.

Müller, R.M., Thoring, K. (2012) 'DESIGN THINKING VS. LEAN STARTUP: A COMPARISON OF TWO USER-DRIVEN INNOVATION STRATEGIES', in Leading Innovation through Design Proceedings of the DMI 2012 International Research Conference 2012, 8-9 August, Boston.

Paternoster, N., Giardino, C., Unterkalmsteiner, M., Gorschek, T., Abrahamsson, P. (2014) 'Software development in startup companies: A systematic mapping study', Information and Software Technology, 56(10), 1200–1218.

de Paula, D.F.O., Araújo, C.C. (2016) 'Pet Empires: Combining Design Thinking, Lean Startup and Agile to Learn from Failure and Develop a Successful Game in an Undergraduate Environment', in International Conference on Human-Computer Interaction, Springer, 30–34.

Poppendieck, M., Cusumano, M.A. (2012) 'Lean software development: A tutorial', IEEE software, 29(5), 26–32.

Ragin, C.C. (1987) The Comparative Method: Moving beyond Qualitative and Quantitative Strategies, University of California Press,.

Ries, E. (2011) The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses, 1st ed. ed, Crown Business.

Rodríguez, P., Markkula, J., Oivo, M., Turula, K. (2012) 'Survey on agile and lean usage in finnish software industry', in Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement - ESEM '12, ACM, 139–148.

Schwaber, K., Beedle, M. (2001) 'Agile Software Development with Scrum'.

Staats, B.R., Brunner, D.J., Upton, D.M. (2011) 'Lean principles, learning, and knowledge work: Evidence from a software services provider', Journal of Operations Management, 29(5), 376–390.

Ter Hofstede, A.H., Verhoef, T. (1997) 'On the feasibility of situational method engineering', Information Systems, 22(6), 401–422.

Thoring, K., Müller, R.M. (2011) 'Understanding design thinking: A process model based on method engineering', in DS 69: Proceedings of E&PDE 2011, the 13th International Conference on Engineering and Product Design Education, London, UK, 08.-09.09. 2011.

Thorpe, R., Holt, R., Macpherson, A., Pittaway, L. (2005) 'Using knowledge within small and medium-sized firms: A systematic review of the evidence', International Journal of Management Reviews, 7(4), 257–281.

Verganti, R. (1997) 'Leveraging on systemic learning to manage the early phases of product innovation projects', R&D Management, 27(4), 377–392.

Vilkki, K. (2010) 'When agile is not enough', in Lean Enterprise Software and Systems, Springer, 44–47.

Wölbling, A., Krämer, K., Buss, C.N., Dribbisch, K., LoBue, P., Taherivand, A. (2012) 'Design Thinking: An Innovative Concept for Developing User-Centered Software', in Software for People: Fundamentals, Trends and Best Practices, Management for Professionals, Springer Berlin Heidelberg, 121–136.

Yin, R.K. (2003) Case Study Research: Design and Methods, Applied Social Research Methods, SAGE.

ACKNOWLEDGMENTS

The research is supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPQ) – Brazil.