The Collective Process Framework DTScrum for Integrating Design Thinking into Scrum

Daniel Gadner and Michael Felderer

Introduction

It is a well-known fact that the accelerating pace of change in today's digital world comes along with disruptive technologies, volatile requirements, wicked problems, and sophisticated demand. This fast-paced environment is often referred to as the time of digitalization, where humankind must inherently transform the traditional ways of product development practices, internal organizing logic, and customer interactions to stay competitive. In order to describe this transition, Gray and Rumpe (2015) use the business-oriented definition of digitalization from Gartner: "Digitalization is the use of digital technologies to change a business model and provide new revenue and value-producing opportunities; it is the process of moving to a digital business." (p. 1319). Despite the benefits that digitalization brings along, it is difficult to maintain the course in the ever-increasing flood of data resulting in unpredictable situations and complex problems (Gray and Rumpe 2015). Hand in hand with such digital transformation efforts goes the diffusion of digital technologies, and therewith, the digital disruption. Technological diffusion is known for its far-reaching effects, as, for example, the rapid pace of change in market needs (Abrell et al. 2016). Stoneman and Battisti (2010) define the technological diffusion "[. . .] as the process by which the market for a new technology changes over time and from which production and usage patterns of new products and production processes result." (p. 733). In this context, firms must learn to adapt to a volatile environment and react quickly to change. The challenge is that today's customers are

D. Gadner (✉)
Internetstores, Stuttgart, Germany

M. Felderer
University of Innsbruck, Innsbruck, Austria

Blekinge Institute of Technology, Karlskrona, Sweden
e-mail: michael.felderer@uibk.ac.at

J. Hehn et al. (eds.), *Design Thinking for Software Engineering*, Progress in IS,
https://doi.org/10.1007/978-3-030-90594-1_5

very well-informed about the capabilities of new technology and are consequently more demanding. Competition is hard as customer needs evolve quickly. It is getting difficult for firms to anticipate the forces of change. The corporate environment is becoming confusing (Fahey 2016). As a result, the timely reaction to changing requirements is more important than ever. It can be best managed by using agile frameworks for digital product development (Fonseca and Domingues 2017). Those often come along with practices that enable quick accommodation to change rather than following a defined plan. Especially in the software engineering domain, agile means short development cycles resulting in fast reactions to changing customer needs (Abrell et al. 2016). Though many companies are jumping on the bandwagon of agile software development practices with their fast feedback loops and flexible functioning, they must not forget, resulting in complex problems. The so-called wicked problems are one of the reasons why firms fail to survive in the time of digitalization. Here, tackling such a class of problems plays to the strength of Design Thinking. The background section will cover the functioning of it, as well as the principles of the Scrum framework. In addition, potential obstacles and challenges of an integration will be exemplified. Based on input retrieved from already conducted expert interviews and practical observations, a conceptual model was developed to overcome beforehand identified challenges. This integrative model is called Collective Process Framework and starts with the Multidisciplinary Knowledge Café, diverging into two process areas running in parallel. The term “Collective” is here used as it describes the acting in the context of software development processes in a cooperative way. In other words, the Design Thinking process, as well as the Scrum process, were emblematic put next to each other to produce a product of collective effort. This means, one area focuses on irrational beliefs and lateral thinking, whereas the other one sticks to causal reasoning and organized thinking. In this context, Plattner et al. (2012) mentioned that “[. . .] Design Thinking focuses on those ‘fuzzy’ aspects of a design problem, which are in purely engineering-led approaches left aside and is thus suggested as a useful supplement to a problem perception and solving in ‘traditional’ IT development approaches.” (p.231). Although, one might argue about the contradicting focus each process pursues on its own. On the one hand, Scrum has a focus on iterative development more likely to respond to change rather than planning for it. On the other hand, Design Thinking has the goal to discover change opportunities that solve customer problems and thus imply movement towards an idealized state. This situation provides an additional motivation to integrate Scrum and Design Thinking in this contribution to benefit from synergies and minimize potential problems.

This contribution is structured as follows. We first present the necessary background on Scrum and Design Thinking. Then, we discuss obstacles and challenges of the integration of Scrum and Design Thinking. As main contribution, we present our collective process framework DTScrum. Finally, we conclude this contribution.

Background

The central frameworks, Scrum and Design Thinking, are both very helpful when it comes to complex problem-solving in a dynamic environment. Scrum, on the one hand, can be named as an effective approach to developing customer-focused solutions. That is because, referring to Vetterli et al. (2013), the main task of Scrum is to implement the user stories which represent the customer requirements in clear form. On the other hand, Design Thinking can be named as a useful process framework to identify real customer problems, redefine complex problems, challenge assumptions and develop novel product designs. Plattner et al. (2012) mentioned that requirements, especially for software systems, should be collected and defined in collaboration with the user, often via the means of Design Thinking.

Scrum

Jeff Sutherland and Ken Schwaber coined the term "Scrum" in the context of agile software development in 1990 and declared that constant evaluation of requirements, plans, and results is the rudiment for a common and natural understanding of how to respond quickly to change. Moreover, Scrum practitioners should embrace change. It is better to respond to change rather than planning for it. All team members ought to discuss on daily basis topics such as what had been already delivered and what kind of tasks are still open. Hereby the Scrum framework promotes an open meeting culture which should assure constant and effective communication inside the team and between all involved parties. This also contributes to a common understanding about the same line of vision on what to achieve, which is mandatory for success (Nachbagauer and Ortner 2015). Next, the scope is not fixed. More the scope is initially mirrored in kind of a project plan, which is called the product backlog. It can evolve and can be seen as a living artifact. Scrum practitioners must always keep in mind that the next set of features to work on could change because of the unpredictable customer and market behavior. Therefore, the product backlog should be updated with each completion of a development cycle. Such a cycle is called an iteration or sprint and describes the time frame during which the team completes predefined user stories. Often, it is depicted as a circle because there is not only one but many iterations. Part of each sprint is a daily meeting, called daily Scrum. As mentioned before, this should foster a communicative culture within the Scrum team (Schwaber 2004). The result of each sprint is called an increment of functionality. Here, the whole team is responsible for the fast and agreeable delivery of the software product. Working functionality is delivered to the customer as soon as possible in terms of small and frequent repetitions (Dzamashvili Fogelström et al. 2010). After each sprint, the team should always look back on how they reached the actual state of delivery and analyze its way of working. All in all, Scrum can be named an iterative, incremental process. Or in other words, Scrum is an agile project

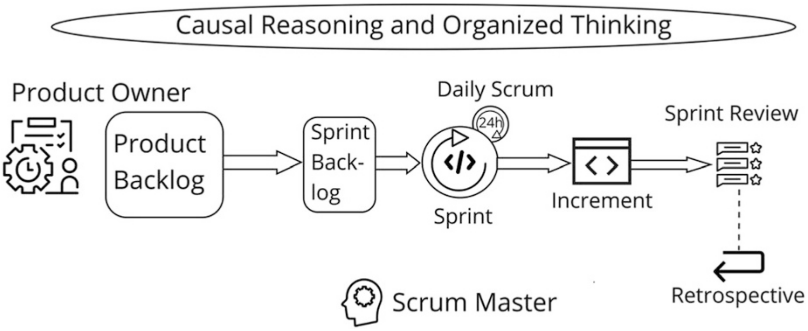

Fig. 1 Scrum process area

management framework mainly used for software development projects (see Fig. 1 for an overview of the Scrum process), which includes change embracing properties (Schwaber 2004).

Design Thinking

In accordance with the research of Gerstbach (2016), it can be stated that most of today's people associate the physical figure of a product with a successful design. Even though successful design evolves out of meaningful and effective development of products incorporating the needs and wishes of users. Hereby the main task of a designer must be the investigation of customer behavior in order to solve the right problem (Gerstbach 2016). In respect to the Collective Process Framework, Damien Newman's illustration of the design process is used to exemplify the characteristics of the Design Thinking process (see Fig. 2 for an overview of the Design Thinking process). The look-alike scribble is called design squiggle and represents the way from research over developing prototypes to finally establish a clear design. Furthermore, it shows that the way to the desired solution might be uncertain in the beginning, but in the end, it will evolve into a single point of clarity. In other words, every solution is born out of uncertainty (Naiman 2017). If necessary, one might even need to take a step back or start all over again; because failure is knowledge and knowledge means success. The path to a successful design consequently unfolds in iterative loops moving back and forth across phases. "With this acknowledgment, it is perhaps also worth noting that whether the process is intentional or not, it will assume a significance upon the final design outcome" (Schneider et al. 2013, p. 117). As we know by now that there is no ultimate single solution to a problem, four factors of success, each individually weighted dependent on the respective situation will be described now. At first, everybody's own intuition is decisive. Due to the explorative character of the Design Thinking process, undiscovered problems and opportunities are more likely to be revealed by trusting the gut instinct. Furthermore, no generated idea is nonsense. Every unique way to a solution should be chased.

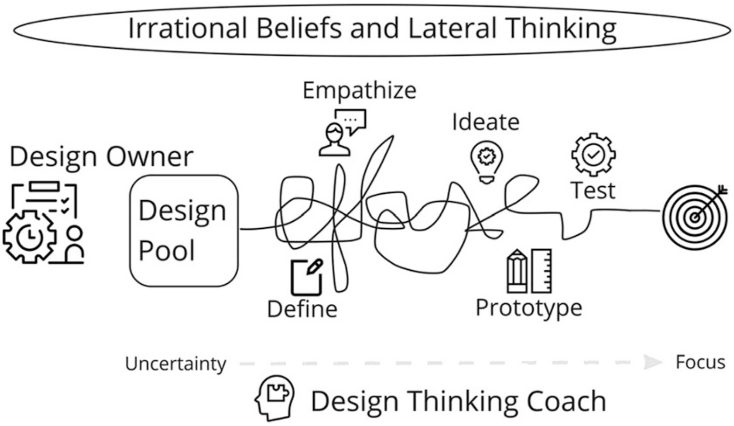

Fig. 2 Design thinking process area

That leads already to the second property, which is the trial-and-error principle. Besides the negative aspect of trial and error, which is nearly impossible to predict the duration of the ideation phase, this principle is a great way to gain knowledge. There are many ways to solve a wicked problem. Especially prototyping should be used to reveal the right problem and possible solutions to it. All ideas obtain the opportunity to be sufficiently tested and improved till one idea stands out from the others. Resulting the third factor of success is the direct implementation of the ideas in the form of prototypes. Now the last property is necessary to identify the limits of the human-centered design approach. Those boundaries should not be exceeded. Within the defined scope Design Thinking can unfold its comprehensive capabilities to innovate. The boundaries are congruent with the desirability, viability, and feasibility dimension. As a result, the appropriate balancing between the feasibility of the technology, the viable business dimension, and the desired customer needs is essential. Dependent on the situation every aspect should be given more weight, sometimes less weight, always with the human being in mind (Gerstbach 2016).

Obstacles and Challenges of the Integration

The successful integration of a proper design understanding in the industry sector, marketing branch, or public had already been an important topic several years ago. However, little is yet known on how to make effective use of Design Thinking in the context of agile software development methods. Buchanan (1992), states hereof that, "Without appropriate reflection to help clarify the basis of communication among all the participants, there is little hope of understanding the foundations and value of Design Thinking in an increasingly complex technological culture." (p. 8). Thus, as a part of former research, interviews with experts in Design Thinking and/−or Scrum had been organized and conducted to define such a basis of communication and further to identify other obstacles and challenges. In this section, the barriers which may occur on the way to a proper integration will be pointed out, and a process framework showing how to overcome them, on a conceptual level, will be depicted next. For exemplification, own experiences will be included as well.

Resource Allocation

Of course, the first thing which might come into mind when mentioning possible challenges of a practical integration is the topic of resources. In most of today's companies, resource restrictions are a critical issue. It is pretty rare that anyone finds themselves in the fortunate situation of being able to focus 100% on a specific problem without any other distractions. Especially, that if all team members are committed to this one project to the full extent. As Design Thinking relies on the involvement of the user as well as the knowledge of people from different domains, it is often difficult to agree on resource capacities. So, depending on the ratio of wicked problems to more tangible problems, you may need to focus your resources either on one or another mindset. Design Thinking itself tries to obtain the user feedback and expertise of other professional roles as often as possible. Therefore, the resources must need to be allocated before the project starts, i.e., in the form of a sensible resource allocation meeting. This will be described later on in terms of the process framework, our starting point for any integrative Design Thinking—Scrum projects. Here, it is important to fine-tune the occupation of resources and the point in time when they are presumably available. So the emphasis on the work must be based on the degree of resource utilization in an appropriate ratio to discover complex problems and ordinary problems. Too many resources would have been occupied with several user research topics and ideation sessions before the actual development phase, in case of Design Thinking is used as an upstream process in front of the Scrum steps. According to this, another challenge is the limited budget for a human-centered design project, as well as the pressure from the management to develop something potentially shippable in a short period. It is, therefore, important that the management understands that agreeing to incorporate Design Thinking in software development projects is a comparatively more time-consuming commitment without guarantee for a successful product idea as an outcome. Instead, every insight gained by the problem discovery and definition phase is a small step toward understanding your customer demand to its full extent. Thus, the resource investment seems to be justifiable.

Competing Views and Different Kinds of Problems

According to the different kinds of obstacles to face, it is necessary to mention that despite all the similarities between Design Thinking and Scrum, they are very different in nature. The Design Thinking's problem-solving approach considers socially ambiguous aspects that build upon heuristics and situational reasoning, whereas the corresponding problem-solving patterns of Scrum rely more on the orthodox engineering design paradigms and analytical thinking (Plattner et al. 2012). With this thought in mind, we must be aware of the competing views between the design domain and the software engineering domain. The different purposes and distinct kinds of problems to be treated by the frameworks could be a reason for conflict when trying to collaborate with each other. Also, the oppositional mindset of the Design Thinking team and the more analytical thinking Scrum people can be seen as a challenge here. Consequently, the question arises of how to use two frameworks together when both have a diverse purpose. "Whereas Design Thinking allows dealing with the ambiguity of design problems as wicked problems, the thinking of IT engineers instead supports the effective technical realization." (Plattner et al. 2012, p. 230). Also, in practice, you can feel the tension between those opposed characters. The most problematic thing is when the more open-minded approach of Designers clashes with more analytical-oriented Scrum practitioners. Both somehow feel a thread in the counterpart's way of thinking, as it might disturb its own way of working. Scrum practitioners might feel held back by the Design Thinking counterpart as the Design Thinking people might feel annoyed by the disbelief of developers in the value creation of the Design Thinking steps. Here, it is necessary that all team members should be sensitized to the opposing mindset. Every project member should have an open mind toward new thinking and working mode. However, the Scrum framework's capability to support the path of understanding the problem and, in turn, defining certain requirements is limited. Approaches, like Scrum, are not yet able to cope with a class of problems demanding techniques reaching from further exploring problems to fully understand them. Especially when thinking of problems occurring in the fast-paced technological world, the characteristics of complex problems, especially in the context of the human-centered design approach, are of high relevance. Especially today's software development projects demand more and more for human-centered design approaches to tackle what is often referred to as wicked problems, i.e., unknown and inherently volatile requirements. Nevertheless, identifying requirements for a solution to tackle the right problem is a very sophisticated process. This might also be the reason why firms fail to survive in the time of digitalization. That is to say, while software development approaches aim at transferring customer needs into rapid development cycles as a means to develop software products iteratively and incrementally, we often still need to first shift our attention to framing and changing the actual problem. To achieve this, Design Thinking should be used to challenge assumptions, redefine complex problems, and even explore new possibilities and paths for problem solving. On the other hand, Scrum is a bit more dedicated to rapidly develop solutions for tamed or ordinary problems, based on user stories that represent the customer requirements in clear form. For example, if you have a problem that is very well defined, you would not necessarily need to conduct a full-blown Design Thinking project, as it would be too big of an investment and a waste of resources. This is because here, you already know about the problem space and the resulting customer pain to be solved. For example, in web development, it is already common knowledge that an "add-to cart" button on a product detail page must be shown on the first view and designed in a prominent way to boost the conversion. Here you can benefit from already existing knowledge and do not necessarily need to start from scratch. So, in this case, one can directly start with the development sprint.

Coordination and Communication

Important for this kind of collaborative procedure of using Design Thinking and Scrum together is constant and regular communication and coordination between all participants. This also involves stakeholder management. Especially in terms of Design Thinking, with all its gatherings of people with diverse backgrounds also the different types of personalities working together must be considered. This can create a conflict between the participants, which, at worst, could also have a bad influence on future cooperation. In order to prevent this potential conflict, an emphasis must be put on effective communication. Nevertheless, a balance between meeting overload and too little communication, which might result in knowledge silos, is crucial. It is important to never underestimate the power of gathering. As soon as several people meet to work together on a concrete topic, problems can be defined, and ideas to solve them are collected. Here, the ones responsible for the output of any implementation must be informed about the current state. People who will be working on the solution to the problem in the near future do not only want to receive a list of requirements to be worked on. Rather they prefer to be part of the journey from the very beginning. Otherwise, people might feel excluded and think they missed their chance to contribute their thoughts and ideas to it. Sometimes it is just the simple human behavior of being affronted after somebody has told them what and how to do something, which puts a successful Design Thinking Scrum project, or any project in general, at risk. This can also result in a lack of motivation and decreases the commitment to the project. When it comes to organizational structure, hierarchical constraints must be mentioned as a possible factor that can also decrease the effectiveness of applying a design methodology such as Design Thinking in the software development domain. An example of something which restricts creative work is design standards given by management. Therefore, "[. . .] a balance needs to be found between corporate requirements and creative freedom" (Häger et al. 2015, p. 264). Further, it is crucial that everybody being part of interdisciplinary teams should know their role and associated responsibilities within the project. It is a completely different situation working together with someone on a regular basis, knowing the team's internal structure and hierarchies, in comparison to someone who has just recently been added/begun working with it. This can create tension, especially when it comes to an overlap of responsibilities. That is also why ownership is a very important part of this topic. If you are responsible for a specific piece of the project, you are less likely to feel threatened by other one's actions to affect your work.

The Collective Process Framework DTScrum

Especially today's software development projects demand more and more for human-centered design approaches to tackle what is often referred to as wicked problems, i.e., unknown and inherently volatile requirements. To achieve this, Design Thinking should be used to challenge assumptions, redefine complex problems, and even explore new possibilities and paths for problem solving. Now, as there are already several kinds of research work that agree on the idea of putting a Design Thinking approach together with Scrum, we are tackling the issue of really synthesizing those different mindsets and working modes. In this sense, the conceptual model which we intentionally describe as a *Collective Process Framework* should be understood as a visual roadmap of:

- Artifacts
- Roles and responsibilities
- Activities

Toward the development of a software-intensive solution using both Design Thinking and Scrum. The model is called the collective process framework, with reference to the work by Malone et al. (2009), who investigated the effectiveness of individuals working together in a collaborative environment. Collective here means that both, Scrum and Design Thinking, are functioning on their own but are connected in several ways throughout the overall solution discovery and development process in order to profit from each other's respective strengths. Therefore, the collective process framework emphasizes a collaborative working mode and mindset. This is achieved by putting the Design Thinking phases in parallel to the Scrum framework, whereby feedback cycles and coordination meetings are promoted in order to maintain a constant connection between the Design Thinking work and the Scrum work. Based on Malone et al. (2009), "The phrase we find most useful is collective intelligence, defined very broadly as groups of individuals doing things collectively that seem intelligent." (p. 2). Consequently, the term "Collective" should, on the one hand, be used to describe the multidisciplinary characteristic of the process framework, on the other hand, to describe the in parallel functioning Design Thinking and Scrum processes. In other words, the Design Thinking process, as well as the Scrum process, were emblematic put next to each other to produce a product of collective effort. Now having a closer look at the conceptual model, the spectator will notice so-called process areas, currently four in number. The following represents the skeleton of the model:

- Multidisciplinary Knowledge Café
- Design Thinking process area
- Scrum process area
- Product Backlog Design Matching

Multidisciplinary Knowledge Café

The first area must be seen as the groundwork for a successful design implementation approach and, therefore, starts with a multidisciplinary knowledge café that involves:

- Bringing people with different backgrounds together
- Connecting diverse points of views
- Preparing a rough roadmap in order to plan and allocate resources appropriately upfront
- Identifying and classifying problems

So, the term multidisciplinary here is used in reference to the Design Thinking mindset, which emphasizes knowledge assimilation regarding putting different points of view together. Incorporating people with diverse fields of expertise is an advantage. Here, Buchanan used the term liberal arts to describe a vision of an encyclopedic education of various natural sciences and mathematics, philosophy, and social sciences. As a result, Design Thinking in the twentieth century must be recognized as the new liberal art of technological culture. Design is an integrative discipline to complement art and sciences (Buchanan 1992). In this case, the advocates of the new liberal art say that "[. . .] the proper study of mankind is the science of design, not only as of the professional component of a technical education but as a core discipline for every liberally educated person." (Simon 2019, p. 138). This is first, effective in a way that every aspect of a problem is mainly covered, and second, it will be talked about any concerns and ideas regarding future work. Thus, such a creative and open-minded approach results either in the identification of a concrete problem set or in the revelation that there, in fact, is no real problem, which can save a considerable amount of time. Trying to solve something which you are not really sure about exist is a complete waste of time. Besides the basic principles of the Multidisciplinary Knowledge Café, namely fine-tuning resource capabilities, connecting diverse perspectives, sharing collective discoveries, and acknowledging different opinions, its main purpose is the identification of subproblems and their classification. An emphasis must be put on the implication that it will be distinguished between wicked problems and well-known, or "ordinary" problems. The separation into different kinds of subproblem is important as the next process areas of the collective process framework have different capabilities to handle the outputs of the Multidisciplinary Knowledge Café. In short, it will be distinguished between ordinary problems and wicked problems. For example, the purpose of Design Thinking is directed towards complex problem solving and novel idea generation, whereas the Scrum process often deals with ordinary or ill-defined problems. Thus, the Design Thinking part deals with the treatment of wicked problems and innovation opportunities, collected in the so-called "Design Pool" which is the counterpart of the Scrum processes product backlog. Wicked problems are real-world problems that are difficult or impossible to solve for some reason. There are no solutions in the sense of definitive and objective answers. One could say, wicked problems refer equally to problems of design and planning. The scrum approach is a bit more dedicated to rapidly develop a solution for tamed or "ordinary" problems. This class of problems is known for its well-defined requirements where the solution is clear. Known algorithms can be applied straight to come up with the solution. Even though the wording might imply it, "ordinary" problems are not less worthy of being solved than wicked ones. But it would not be sufficient to re-discover the same problem again and again. Therefore, it is important to differentiate between those. For example, if you have a problem that is very well defined, you would not necessarily need to conduct a full-blown Design Thinking project, as it would be too big of an investment and kind of a waste of resources. This is because here you already know about the problem and the resulting customer pain. For example, in web development, it is already commonly accepted that the basic structure of any product detail page includes an image of the product, the title, the price, variation selection, and the add-to-cart button. Maybe in the past, a problem existed, that customers have not really had added products to the cart. Now we know that it is decisive to show a clickable element, also known as the "add-to-cart" button, on the first view as it is the main element on such a template to boost the conversion. In this case, you can benefit from already existing knowledge and do not necessarily need to start from scratch. Resulting, one can directly start with the implementation sprint. To make use of both, the collective process framework is not really about integrating one framework into another. It is about finding ways of planning both together because they can also work on their own very well. An emphasis must be put on a collaborative working mode and mindset. This is achieved by putting the Design Thinking phases in parallel to the Scrum framework, whereby feedback cycles and coordination meetings are promoted in order to maintain a constant connection between the Design Thinking work and the Scrum work. Such collective functioning is vital for the ultimate implementation of a successful product design. It is depicted in the conceptual model as process areas two and three, which will be described now.

Diverge into Design Thinking and Scrum

The class of problems must be structured, categorized, and prioritized. Therefore, the second and third process areas start with backlog management. Here it is important to get a better overview of the work to be done. The different class of problems which have been discussed beforehand are going to be revised. On the one hand, there is the Scrum framework, focusing on organized thinking and causal reasoning, and on the other hand, there is the Design Thinking approach, more of lateral nature. First, coming to the Scrum process, which stands for the principles of cause and effect. It is all about thinking which event leads to one another. Especially, communication within distributed systems and its determination of the sequence when several instructions should take place is crucial. So, this way of thinking leads to a causal order in which the output of one action leads to and, or is needed by the following operation. As a result, processes can be planned in a more holistic way. It emphasizes analyzing and categorizing conceptual information using a systematic and logical thought process. In comparison to this incremental way of thinking, the Design Thinking approach is more about the opposite. Here, lateral thinking stands for analysis by intuition. Mental leaps are no shame, and not every interim finding must make sense in the first place. Especially odd ideas are welcomed. Further, initial situation and boundary conditions must not be seen as immutable, rather as an opportunity to shape the context for its own purpose to open up new perspectives. You should force yourself to believe in something which you normally would not believe due to objective evidence retained by intrapersonal cognitive structures. In this sense, Design Thinking ends with a good understanding and confidence of ideas.

Irrational Beliefs and Lateral Thinking

Process area two is showing the before-mentioned illustration of Damien Newman's design squiggle, which refers to the design process and it is way from research over developing prototypes to finally establish the clear design (Naiman 2017). Here, irrational beliefs and lateral thinking minimize the risk of developing a solution towards a non-existing problem and uncertainty about the right problem by engaging customers through a series of prototypes to learn, test, and refine concepts. Learning from failures of previous iterations is essential. Now, the roles corresponding to the Design Thinking work within this collective process framework will be explained. Here, the role of the Design Owner is the counterpart of the Product Owner and, as a result, responsible for the Design Outcome, for instance, the mock-up. Important to mention here is that the Design Owner decides with the consultation of the Design Thinking team which elements of the Design Pool should be part of the next design step and if going back or forth to any other step might be beneficial. Another option would be an innovation opportunity that seems profitable so that more investigation must be made. Either way, the Design Owner triggers the design sprint, which in turn will be run iteratively by the Design Thinking team, including the Design Thinking Coach, and in some cases, subject matter experts. The Design Thinking Coach can be seen as the counterpart to the Scrum master.

Causal Reasoning and Organized Thinking

For the correct application of Scrum, the applicant must remember the major principles of agile software development. First, the development of features should happen right away. Working functionality is delivered to the customer as soon as possible in terms of small and frequent repetitions. Additionally, embracing change is a crucial factor. Therefore, Scrum practitioners should be aware that it is better to respond to change rather than plan for it. Next, the scope is not fixed. More the scope is initially mirrored in kind of a project plan, which is called the product backlog, which is one of the four key building blocks of the Scrum process. It can evolve and be updated between the completion of each iteration. It is kind of a learning process. The other elements are the increment of functionality, the 24-hour inspection meeting, and the time frame of an iteration cycle called a sprint. As explained yet, Scrum is an iterative, incremental process. It all starts with the product backlog. This artifact is a list of requirements that are mostly requested by the customers and, thus, by the Product Owner. An iteration of development activities is represented as a circle because there is not only one iteration but many, one after another. The result of an iteration is the potentially shippable product increment. So, if a final increment can be presented to the customer, everything starts from the beginning. The problem with Scrum is that most organizations nowadays do not allow the direct collaboration of users and the development team. The principles of Design Thinking could help the Product Owner or any other role involved in the product development process to establish a better understanding of how to tame complex problems, foster innovative thinking, and have a customer-focused mindset.

Converge

An important success factor is the same line of vision, which in turn is crucial to synthesize all tasks. Therefore, it is not enough to just let the two processes run in parallel in order to benefit from a collective process approach. It must be ensured that both frameworks benefit from the advantages of their counterparts and still utilize their own strengths. Important for such a collaborative procedure is constant communication and coordination with each other. That is why we want to introduce the fourth process area. In this context, the Design Thinking output, like, for example, nonfunctional prototypes, should be used as an input for further development cycles. This means that the insights about customer needs and tamed problems must be included in the development sprints of the Scrum team. “Ideally a requirements specification could include a description of basic assumptions, needs, and knowledge of the problem domain, needed for designing and implementing an information system.” (Bubenko 1995, p. 160). Now, it is all about bringing the work back and learning from each other’s achievements. Important for such a collaborative procedure is constant communication and coordination with each other. The

Multidisciplinary Knowledge Café, which brought together different professions and personalities in the first run, can also be used to converge the beforehand diverged working teams. So, for this converging step, an artifact such as a backlog can be used again as a merging tool to bring the people back together and collect the different achievements, ideas, and problems in a shared area. This meeting is called "Product Backlog Design Matching" and offers attendees the possibility to discuss newly developed functionality, improved mock-ups, tamed problems, and clearly defined innovation opportunities. Further, recently appeared complex problems that cannot be implemented due to non-existing clear definition are part of this pool. The attendees should feel like to be cushioned against the "daily" work (reality). This melting pot is the heart of past work and the foundation for future achievements. This means that the insights about customer needs and tamed problems must be included in the development sprints of the Scrum team. It is important to hand over the knowledge of each other's tasks and achievements. When having a look at the parallel functioning processes on a holistic level, there is still the connection to the end-user. This is also where you get the knowledge on which new features and functions should be integrated into the next sprints to solve the right problem. Further, issues can be discussed so that misunderstandings can be excluded, and maybe another point of view will reveal an alternative way to solve an issue. The vision which has been developed in the Multidisciplinary Knowledge Café, therefore, can evolve over time by keeping the core problem and goal in mind. It will be noticed that the knowledge hand-over is a critical but also indispensable milestone during the solution development process. Critical, because there is so much knowledge that is implicitly within the Design Thinking team and its possible loss of achievements and insights due to the poor hand-over to the other or a completely new team, which might happen due to capacity or resource constraints. Mainly this is critical because Design Thinking does not document that much. And indispensable because ideas exist, which you need to implement to experience it and to see if the idea is working. In other words, the team is performing some Design Thinking stories. Already existing user stories on the part of the Scrum product backlog can be adapted or removed, and new user stories can be established. As the Product Backlog Design Matching finishes, the "daily" work begins again.

Conclusion

All in all, we presented the Collective Process Framework, which unifies Design Thinking and Scrum on a conceptual level. The framework, visualized in Fig. 3, comprises four process areas, and it should be understood as a visual roadmap of general process items, including:

- Artifacts, such as mock-ups and ideas and their relationship development artifacts such as the product backlog.

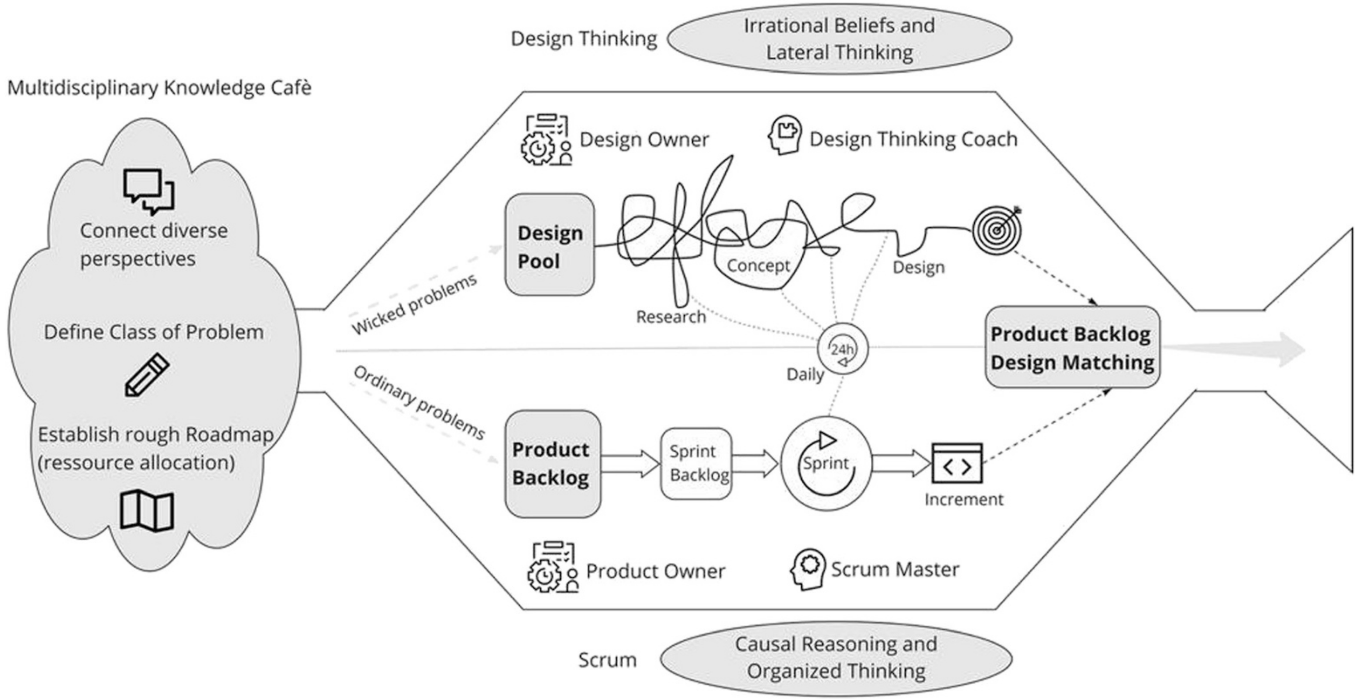

Fig. 3 Collective process framework

- Roles, such as the design owner building a counterpart to the Product Owner.
- Activities, such as the Product Backlog Design Matching.

This roadmap serves, as we argue, as one important step toward the development of a software-intensive solution using both DT and Scrum. Both frameworks benefit from each other and still utilize their own strengths. Here, Design Thinking is frequently identified as an engaging process and methodical framework for approaching complex problems in a multidisciplinary way. Such a design work often tames complex problems and results in novel design solutions. Scrum is a simple framework for effective team collaboration in a dynamic environment, in which communication and collaboration are essential elements. Further, it provides guidance for effective product development in a dynamic environment. Now, due to the lack of social aspects within the more analytically arranged software engineering domain, Design Thinking must help to integrate a proper understanding of humanity in the software development process. So, this shall contribute to the long-term objective of using Design Thinking within the Software Engineering domain. We consider the refinement of the process areas by empirical studies, especially case studies in an industrial context, as future work, and we cordially invite interested researchers and practitioners to join our endeavor.

References

Abrell T, Pihlajamaa M, Kanto L, Vom Brocke J, Uebernickel F (2016) The role of users and customers in digital innovation: insights from b2b manufacturing firms. Inf Manag 53(3): 324–335

Bubenko JA (1995) Challenges in requirements engineering. In: Proceedings of 1995 IEEE international symposium on requirements engineering (RE'95), pp 160–162

Buchanan R (1992) Wicked problems in design thinking. Des Issues 8(2):5–21

Dzamashvili Fogelström N, Gorschek T, Svahnberg M, Olsson P (2010) The impact of agile principles on market-driven software product development. J Softw Maint Evol Res Pract 22(1):53–80

Fahey L (2016) John c. camillus: discovering opportunities by exploring wicked problems. Strategy & Leadership

Fonseca LM, Domingues JP (2017) How to succeed in the digital age? Monitor the organizational context, identify risks and opportunities, and manage change effectively. Manag Mark Chall Knowl Soc 12(3):443–455

Gerstbach I (2016) Design Thinking im Unternehmen: Ein Workbook für die Einführung von Design Thinking. GABAL Verlag GmbH, Offenbach

Gray J, Rumpe B (2015) Models for digitalization. Softw Syst Model 14:1319–1320

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, Leifer L (eds) Design thinking research: building innovators. Springer, Cham, pp 263–289

Malone TW, Laubacher R, Dellarocas C (2009) Harnessing crowds: mapping the genome of collective intelligence. MIT Sloan Research Paper 4732(09)

Nachbagauer A, Ortner G (2015) Globale Projekte managen: Neue Wege für die weltweite Projektarbeit. Symposion Publishing, Düsseldorf

Naiman L (2017) Design Thinking as a strategy for innovation. The European Business Review. www.europeanbusinessreview.com/design-thinking-as-a-strategy-for-innovation. Accessed 16 Feb 2019

Plattner H, Meinel C, Leifer L (2012) Design thinking research. Springer-Verlag, Berlin Heidelberg

Schneider J, Stickdorn M, Bisset F, Andrews K, Lawrence A (2013) This is service design thinking. BIS Publishers

Schwaber K (2004) Agile project management with Scrum. Microsoft Press, Redmond

Simon HA (2019) The sciences of the artificial. MIT Press

Stoneman P, Battisti G (2010) The diffusion of new technology. In: Bronwyn HH, Rosenberg N (eds) Handbook of the economics of innovation, vol 2. Elsevier, pp 733–760

Vetterli C, Brenner W, Uebernickel F, Petrie C (2013) From palaces to yurts: why requirements engineering needs design thinking. IEEE Internet Comput 17(2):91–94