



**Master's Thesis**

# DT@IT Method Toolbox

**.. to improve empathy for team members and end users to foster human centered design in software products**

## DT@IT Methoden Toolbox

.. um die Empathie für Teammitglieder und Endanwender zu verbessern und um ein am Menschen orientiertes Design von Softwareprodukten zu fördern

**by**
Philipp Pajak

**Supervisors**
Prof. Dr. Hasso Plattner
Dr. Matthias Uflacker
Franziska Häger, M.Sc.
*Enterprise Platform and Integration Concepts Chair*

Hasso Plattner Institute
University of Potsdam

October 22, 2018

**Eidesstattliche Erklärung**

Hiermit versichere ich an Eides statt, dass ich die vorliegende Masterarbeit ohne fremde Hilfe angefertigt und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Alle Teile, die wörtlich oder sinngemäß einer Veröffentlichung entstammen, sind als solche kenntlich gemacht. Die Arbeit wurde noch nicht veröffentlich oder einer anderen Prüfungsbehörde vorgelegt.

Ort, Datum, Name: _____

Philipp Pajak

# Abstract

End users are not "specialized workforces" anymore, and expect that software should just work. There is an emerging need for better software, but Information Technology (IT) companies have difficulties providing a better User Experience (UX). Although usability is an important topic, there is no optimal way for collaboration between developers and designers yet. Research shows that Design Thinking (DT) can help to improve collaboration in interdisciplinary teams and further fosters human-centered thinking. In this thesis, we discuss current research on this topic including the adoption of DT in IT companies, agile software development processes already using DT, and creativity methods for developers. Our research goal is to create a DT@IT Toolbox for day-to-day agile development. We conducted a study with five different companies that applied DT methods during everyday work or in a workshop setting. Based on previously defined criteria, we have selected 12 methods that were tested by selected participants of the involved software development teams. We evaluated how the methods can be used in agile software development to foster the creation of human-centered software, the applicability of the methods in IT and whether members of a software development team are able to understand them. Further we evaluated the effect of the methods on team empathy, end user empathy and product understanding. In our study the participants were able to understand and apply most methods, and what is more important, also benefit from them. Further our findings show that applying DT methods in agile work day indeed fosters team empathy, end user empathy and product understanding. Still, there are some open questions towards using these practices effectively, as participants did not start using the methods after the evaluation phase on their own. With this thesis we seek to contribute to current research on DT in the IT context and show how DT can be used not only before but also during development of software.

# Zusammenfassung

Endbenutzer sind keine ßpezialisierten Arbeitskräfte"mehr, sondern erwarten, dass Software funktionieren sollte. Es gibt einen wachsenden Bedarf an besseren Programmen, jedoch haben IT-Unternehmen Schwierigkeiten Software mit besserer Nutzererlebnis anzubieten. Auch wenn die Benutzerfreundlichkeit ein wichtiges Thema ist, sind bisher keine optimalen Möglichkeiten der Zusammenarbeit zwischen Entwicklern und Designern bekannt. Die Forschung zeigt, dass Design Thinking (DT) dazu beitragen kann, die Zusammenarbeit in interdisziplinären Teams zu verbessern und ein am Menschen orientiertes Denken weiter zu fördern. In dieser Arbeit stellen wir den aktuellen Stand der Forschung zu diesem Thema vor: die Einführung von DT in IT-Unternehmen, agile Softwareentwicklungsprozesse, die bereits DT verwenden, und Kreativitätsmethoden für Entwickler. Unser Forschungsziel ist es, eine DT@IT Toolbox für die tägliche agile Entwicklung zu entwickeln. Wir haben eine Studie mit fünf verschiedenen Unternehmen durchgeführt, die DT Methoden in der täglichen Arbeit oder in einem Workshop angewendet haben. Anhand von uns definierten Kriterien, haben haben wir 12 Methoden ausgewählt, die von den Teilnehmern getestet wurden. Wir haben evaluiert, wie die Methoden in der agilen Softwareentwicklung eingesetzt werden können, um die Erstellung von am Menschen orientierter Software zu fördern. Weiterhin haben wir die Anwendbarkeit der Methoden in der IT untersucht und ob Mitglieder eines Software-Entwicklungsteams sie verstehen können. Weiterhin haben wir die Wirkung der Methoden auf Team-Empathie, Endbenutzer-Empathie und Produktverständnis bewertet. In unserer Studie konnten die Teilnehmer die meisten Methoden problemlos verstehen und anwenden - insbesondere waren sie aber auch von Nutzen für die Mitglieder. Unsere Ergebnisse zeigen, dass die Anwendung von DT Methoden im agilen Arbeitsalltag tatsächlich die Team-Empathie, die Endbenutzer-Empathie und das Produktverständnis fördert. Es gibt jedoch noch einige offene Fragen zur effektiven Nutzung der Methoden, da die Teilnehmer die Methoden nach der Evaluierungsphase nicht selbstständig eingesetzt haben. Mit dieser Arbeit streben wir einen Beitrag zur aktuellen Forschung über DT im IT-Kontext an und zeigen, wie DT nicht nur vor, sondern auch während der Entwicklung von Software eingesetzt werden kann.

# Acknowledgements

This final product has been benefited from the worthy support of different academic, professional and personal contributors to whom, without following an order of importance, I want to express my recognition and gratitude. I would like to express my deepest appreciation to my main supervisor, Franziska Häger, for her valuable and constructive suggestions during the planning and development of this research work. My deep gratitude to Dr. Matthias Uflacker who provided me with very valuable feedback. I am particularly grateful for the assistance given by the Hasso-Plattner-Institute, especially for showing me that Information Technology is much more than the usual technical challenges that it faces. Through my institute not only have I got in touch with Design Thinking, but I was also supported in fully understanding its potential. I would like to thank the company *Design-IT* for their assistance with the collection of my data, and of course, the other four companies that allowed me to conduct a workshop with their employees. The support provided by my family and friends, by having deep conversations on the topic, was greatly appreciated. Finally, my special thanks are dedicated to my wife, who not only supported me, but also helped me with her knowledge about academic writing.

# Contents

# Acronyms

**API** Application Programming Interface

**CME** Continuing Medical Education

**DT** Design Thinking

**GUI** Graphical User Interface

**HCAW** Human-Centered Agile Workflow

**HCD** Human-Centered Design

**IT** Information Technology

**MVP** Minimum Viable Product

**QR** Quick Response

**SD** Service Design

**UCD** User Centered Design

**UI** User Interface

**UX** User Experience

**XP** Extreme Programming

# 1. Introduction

The usability of a software is becoming more important as end users are not “specialized workforces” anymore but people without any further schooling in using a specific application. People tend to expect that "[s]oftware should be easy-to-use and self explanatory without a necessity to read comprehensive manuals or even attend dedicated trainings" [59, p.2]. We can clearly see that trend in successful consumer software, which mostly has a strong focus on usability. On the other hand "[f]or managers, usability is a major decision factor" [78, p.72] as training costs can be lowered and users can perform their tasks more efficiently.

Many small and medium sized enterprises would like to buy products with better user experience, as "70% of software clients in Germany confirm that software usability issues have negative impacts on their productivity" [59, p.44]. On the other hand "a discrepancy exists between the required usability and that currently offered by manufacturers" [59, p.44-45], which means that a lot of companies are not able to provide software with a stronger focus on Human-Centered Design (HCD).

Big software companies like SAP already realized the "need to ease the communication between the developers and usability experts" [68, p.131], but communication between designers and developers is not an easy topic. Many developers are unaware of the importance of communication between developers and designers: "[...] it is not unusual to find development teams that think they can design the system and then have the ’usability team’ make it usable [...]" [28, p.23]. Usability professionals turn that statement around and tend to say that they should design the system first, and later the developers must implement the system. [78, p.73] The optimal way of collaboration is somewhere in between and requires a better understanding of each other’s way of working, close communication and integrated processes.

What is more important, it is further profitable to focus on usability topics as many authors claim and studies show [60][6][26] that investing in usability pays off: For every dollar spent the studies estimate a benefit between 10 and 100 dollars.

## 1.1. Why Design Thinking?

One successful approach for human-centered problem solving is DT. It is already widely used in non-software projects to turn *wicked problems* (see section 2.2) into ideas that emphasize the human as an important stakeholder in the provided solutions. According to [59], DT can be used in software development for a better understanding of end user's needs and thus "the software industry can benefit [from DT...] in order to create innovative software products"[59]. A few attempts of integrating DT into IT software development approaches already exist, because "design thinking embodies tackling those 'fuzzy' aspects of a design problem, which are in purely engineering-led approaches left aside" [57].

To reach its goal, DT comes not only with a problem solving process but also contains various methodologies and a marked mindset. The methods concentrate on fostering creativity, failing early and creating rapid prototypes. Researchers propose to integrate DT with agile software development to "tackle shortcomings of established IT development approaches"[57]. It may even "contribute to a better working atmosphere for software developers and potentially result in more innovative software products [... as it] promotes out-of-the-box thinking and encourages creativity"[59]. Especially the latter two are important techniques which can reduce enormous development effort that might be thrown away, because it addressed the wrong problem and had to be reworked in the development or even maintenance phase. But also by "integrating the user as early as possible, the likelihood of expensive succeeding adaptations is reduced"[59]. The cost of correcting a problem could be from ten to 100 times higher, when done after the design phase[26]. Furthermore, an important part of DT is to go for feedback, which is a substantial part of creating human-centered software. "Close interaction is also a key aspect of design activities" [43] and an integral part of DT. Combined with radical collaboration in multidisciplinary teams it can help to improve communication for people of different backgrounds [59, p.129], thus between developers and designers, too.

To complete the picture, we would like to show a definition of innovation that can be seen in figure 1.1: We already have technology and business in the IT industry, but we are lacking human values for the optimal spot of innovation. A combination of agile software development practices together with DT might help us to reach that point as both approaches have a lot in common[57], but agile techniques lack "techniques for understanding the users' needs and developing respective user interface concepts"[43].

## 1.2. Outline

We want to shortly outline the following chapters of this thesis. In chapter two we describe the background of our work by introducing DT and agile software development. Further in the next chapter, we summarize the related work that is relevant for our research. This includes

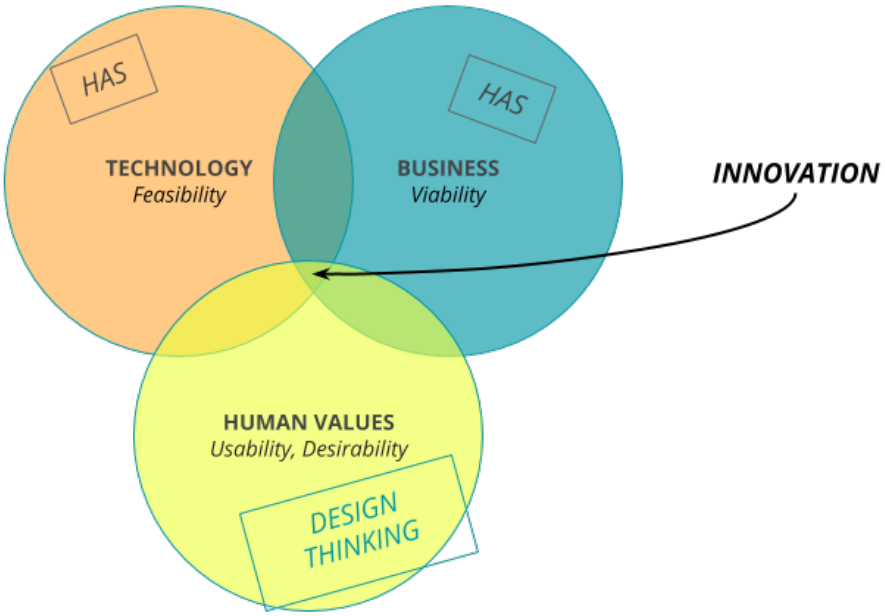



Figure 1.1.: IT Industry Already Has Business and Technology, but Lacks Human Values that are Needed for the Optimal Spot of Innovation; Based on the DT Feasibility, Viability and Desirability Model[12]

adoption of DT in the IT Industry, existing processes that merge DT with agile methodologies and creativity methods for developers. Chapter four presents our research goal with its respective hypotheses and research questions. Further, we describe the practical phase and the method selection. The methods that we have selected for our toolbox will be presented and described in chapter five. In chapter six, we evaluate the results of our study and in the last chapter we discuss our findings, present our conclusions and propose future work.

# 2. Background

Software Development is a rather new engineering discipline that initially took over most methodologies from traditional engineering fields. With the new possibilities of engineering in a virtual world, and because of recurring problems due to changing requirements, these methodologies had to change and still are in a constant movement.

Basic knowledge about DT and Agile Software Development is required for this thesis. In the following two sections there is an introduction into those two topics.

## 2.1. Agile Software Development

The Agile Manifesto [3] was signed by 17 software development practitioners in early 2001 - it wrapped up several methodologies that began to emerge in the late 1990s under the term "Agile"[2]. This was a revolution in software development after the domination of the traditional waterfall model for decades. The waterfall model is based on engineering methods that come from the manufacturing and construction industries of the 1960s where changes during the implementation phase are radically expensive and sometimes even impossible. As no development process existed in the early days of software development, the traditional approach from other engineering industries was taken over and just slightly adapted[65, p.11].

The Agile Manifesto consists of the following four main values that all participating software practitioners could agree on:

- *Individuals and interactions* over processes and tools
- *Working software* over comprehensive documentation
- *Customer collaboration* over contract negotiation
- *Responding to change* over following a plan

Besides those values there is also a radical change in the development processes, making them lighter[15] and thus much more flexible[76]. Members of a team are self-organizing

and, in addition, developers should collaborate with business people during the whole project[3]. The team is working on incremental, frequent updates for the software that they are building [3]. Feedback should be gathered every time the software is presented to the stakeholders to discover and correct errors and wrong assumptions[3]. What is more important, the software should be always in a working state so it could be shown to the stakeholders at any time[3]. Day to day agile development is based on so called user stories that should "divid[e] up the work to be done into functional increments"[2] that "contribut[e] to the value of the overall product"[2]. Additionally, changing requirements should be accepted and welcomed at any time to provide most value to the stakeholders[3]. The development process is iterative, so after each development cycle the team starts the process again to incrementally update the software. [2]

Agile builds a base for frameworks such as Scrum or Extreme Programming (XP). As an example, a Scrum team is a small group of people, mostly working on a full-time basis[77]. To ease communication in the team there are daily meetings to discuss finished work and potential blockers[73]. At the end of a development cycle, which is called "sprint" in the Scrum process, there is a retrospective with the whole team to eliminate any obstacles that might be in the way and adjust processes to be more efficient, if needed[32].

Although the agile methodology has several advantages over the waterfall model, it still has further challenges to solve. Agile teams have a strong focus on technical aspects [86] rather than targeting the needs of end users[81]. According to Griffith[35], most project failures originate from building software that has no market, thus emphasizing the need of human-centered solutions. Furthermore, agile assumes that the team already has a product vision without further specifying how to obtain it[22]. Additionally, it was shown that agile methodologies like Scrum do not consider design in a reasonable manner[24] and software teams in general lack the ability to empathize with the end user[86]. Agile methodologies further do not provide approaches to understand end users' needs[81], as "[the] end user experience is not the primary aim of agile methodologies"[81]. Agile also avoids divergent thinking and puts no emphasis on interdisciplinary teams[22], and thus another limitation of agile teams is problem understanding and solution finding[24].

## 2.2. Design Thinking

DT is a process, mindset and set of methods to facilitate the innovation process by introducing human values [47, p.6]. It is a "solution-based approach to solving problems" by "challeng[ing] assumptions" and "identifying alternative strategies and solutions that might not be instantly apparent" [19]. With its emphasis on empathy [47, p.13] and action it helps to create solutions that are more human-centered. If necessary, problems are additionally redefined to address the right problem and thus fulfill the needs of end users.

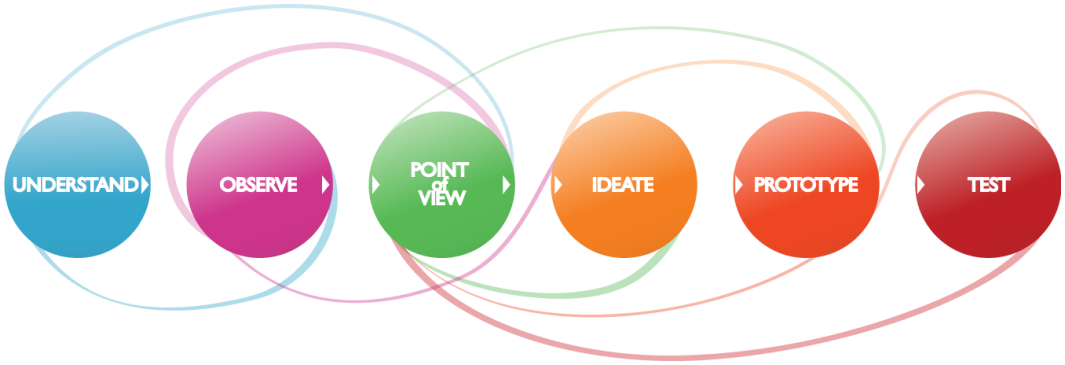



Figure 2.1.: Iterative Design Thinking Process [71]

There are various definitions of the DT process, depending on the source it is defined in different amount of steps, but the process stays the same. Thus only the weighting of each part of the process is different. In this thesis, we will refer to the process as defined by the Hasso-Plattner-Institute School of Design Thinking [71].

As shown in figure 2.1 the DT process consists of six steps which are interconnected iteratively. This means that it is possible to go back and forth through the process, if necessary. For instance, while prototyping the team might realize that it is necessary to observe the user again because the team has not gained enough insights yet.

A detailed description of the DT process can be found in [71]; in this section we will present a short summary. The first step of the DT process is the *Understand* step. The team tries to understand the so called "Design Challenge" and research existing solutions and problems. The task is further specified and the team finally selects the target user group. After the intellectual phase of understanding, the *Observe* step is where the focus shifts to action. The team has to prepare and conduct interviews, but most importantly also observe the target end users to gain useful insights. In this phase, the team becomes more aware of the cultural, economical and personal circumstances of the people that they are actually designing for and thus gains empathy for their target users. The next step, *Point of View*, is where the team comes together to exchange insights, interprets them and finally defines the point of view on the initially defined design challenge. It should be defined in a way that corresponds to the actual needs of the potential users. Because innovation problems are so complex, redefining the problem statement has to be part of the process. So called *wicked problems*, are highly complex problems that do not have clear solutions. As there are no right or wrong solutions known, different paths are possible and DT offers a way to approach a good solution. On the other hand, redefining the problem is also crucial to be able to satisfy the real need of the end users. In the forth step, *Ideate*, it is important to start diverging and to produce as much ideas as possible in a short amount of time. It is not crucial whether the ideas are actually viable, as even unconventional and not feasible ideas can give the necessary spark for the right idea afterwards - thus any judgment should be deferred. The ideation phase is concluded

by selecting few, but valuable ideas. The *Prototype* step, is where the team has to decide for potentially useful ideas and prototype them. This means, that it should be prepared in a way that is tangible for the user to make it easy to gather feedback. It is not important whether the prototype is perfect; it can be a simple paper model or even a role play, but it should be fast to build and resemble the experience of using the real product or service. The last step is the *Test* - and it has to be performed with real end users. This step is used to gather qualitative feedback from the people that the team is designing for, to be able to learn and enhance the prototype. It is very likely that after the test ideas need to be changed drastically or even abandoned completely, but exactly this is what the Design Thinker wants to achieve in this step: Test and verify the idea to make sure that the process is going in the right direction. [71]

DT incorporates a lot of different mindsets to facilitate the innovation process. As with most DT related topics, there is no specific set of mindsets and every DT specialist might emphasize different mindsets. This is why we present only some mindsets of those we think that are relevant to this thesis or to understand the basics of DT.

*"Radical collaboration"* ensures that teams include people from various backgrounds, so that they can inspire each other and create creative and unique solutions[70]. Another important mindset *"show don't tell"* emphasizes that DT should be visual - one rule in DT is for example to add a small drawing to each post-it, so one can easier grasp the meaning of it[70]. Post-its are a common tool that is used in DT[21]. There is a *"bias toward action"*, which means that Design Thinkers rather do things like going out on the field or building prototypes to make sure that whatever they are building is not a theoretical construct but was inspired by end users, verified by a prototype and finally tested again by the end users[70]. The *"prototyping culture"* in DT is so important, because a prototype allows to communicate an idea much better to the team or end users. Many design pitfalls do not become evident until a tangible prototype has been built[70]. Another very important mindset is that DT should be *"human-centered"*, which is the most important part of any innovation process - to include the human being as the most important source for requirements and feedback[70]. Last but not least, if the team is lost, the mindset *"mindful of process"* helps them to get back on the path: The DT process helps the team to recognize where they are and what might be the next step[70].

Furthermore it should be noted that agile and DT have different understandings regarding whom the product should be developed for. While agile methodologies like e.g. Scrum focus on the customer, DT focuses on the end user[30]. For the terms *end user* and *customer* we want to use the definition by Beyer et al.: "We make a distinction between the [end user] and the customer; the [end user] is the individual who interacts with the system being designed directly. They use it to accomplish their job. But [...] a customer may be [an end user] or may depend on the output of the system, prepare input for a system, decide on the need for a system, or approve the purchase of a system. A customer [...] is anyone standing between you and acceptance of your system in practice"[5].

# 3. Related Work

DT has already been successfully used for software development and also integrated into agile processes. Nevertheless, existing research mostly concentrates on generic approaches or on pre-development processes. This chapter gives an overview of ongoing research on how to integrate DT with agile software development approaches and the existing challenges in that regard. Not all examples that we provide are based on DT, but some also include HCD, UX and Service Design (SD). This is reasonable, as the concepts do not differ much on the methodological level and we are conducting research on DT methods for developers.

## 3.1. Adoption of Design Thinking in IT Companies

Frye and Inge[30] have studied how product owners and scrum masters perceive the integration of DT with Lean Software Development and Scrum. Their results include benefits, but also challenges in combining those methodologies. One of their most important findings is that while DT can generate waste, it is capable of eliminating much more waste if used correctly. Most waste that is generated in software development is due to development of unnecessary features - DT makes developing the wrong idea less probable, e.g. "by seeking early user feedback before actual development"[30]. Some of these actions have been titled as "necessary waste" that is considered as tolerable and required. Still "interviewees perceived that DT does create some waste in certain parts [... as] the amount of data collected increased dramatically and it was difficult to decide the correct path to follow"[30]. On the other hand, there is also waste generated by ideas not being technically feasible - DT actually encourages wild ideas that are not necessarily feasible - and interviewees would like to "avoid spending too much effort on ideas that have low success chances"[30]. Also the size of the teams has an impact on the generation of waste: Bigger teams seem to lead to more discussions and therefore make it harder to come to an agreement. Smaller team sizes have been identified as a better fit to incorporate DT[30].

Further research[10] has been conducted by Bravo and Adler at a Brazilian innovation consultancy that has been working together with a retail company to use DT in a Split Process Model for application development. The innovation consultancy has developed a Minimum Viable Product (MVP) using DT methodologies, which was later implemented by the retail company. Besides the application being a success by increasing key performace indicators by more than

50%, the researchers found out that DT "can help companies develop new technological solutions that meet end-users' needs and bring financial results in a shorter timeframe"[10]. Additionally, changes in the company culture could be observed as departments started to benefit from mutual learning and a rise in interdepartmental communication could be observed.

Likewise, de Paula et al.[23] described the creation of a mobile application with DT, UX and usability guidelines that was commissioned by Blackberry Latin America. The application has been created in roughly a month by undergraduate students. The same students that have been conducting the inspiration and ideation phases of DT were also implementing the final application. The application was a high success, as it has received very high rankings, especially compared to alternative products available on the market. The researchers conclude that undergradute students with no experience in design methods can successfully use DT with UX and usability guidelines for software development.

Lindberg et al.[57] are trying to answer the question whether and how DT and IT development could be combined. As a result, four different models have been distinguished in the study:

- *Split Process Model*; DT is handled separately and before the development process, without the development team
- *Champion Model*; Similar to the Split Process Model, a DT phase is conducted at the beginning, additionally a few members from the development team participate in the DT phase
- *Design to Development Model*; DT is especially used for the front-end development and is gradually replaced by IT development throughout the process
- *Toolbox Model*; DT is not directly integrated into the development process, but instead a bundle of methods is available to the developers to solve problems that require a more creative approach

Most of the research in this area concentrates on the Split Process Model, which means that a DT phase is conducted before actual development. Some researchers try to extend to a *Design To Development Model* to involve DT also in the actual development process. Nevertheless, not all have been tried out in real development teams and they lack a proper description on how to combine the actual development with DT. A selection of this research is presented in section 3.2.

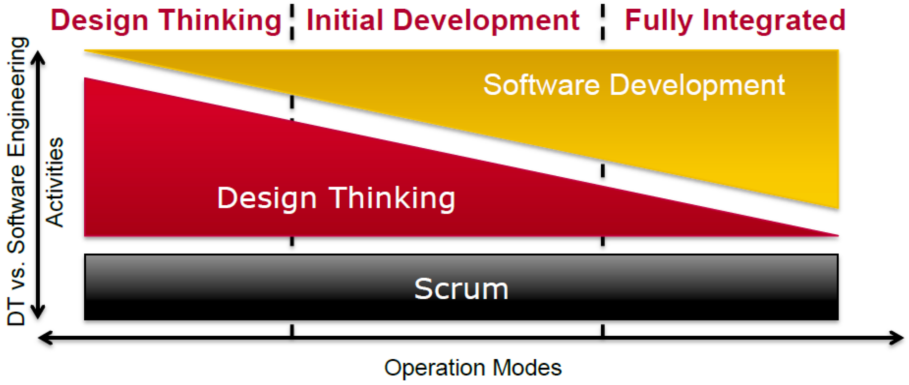



Figure 3.1.: DT@Scrum Process Model[40]

## 3.2. Existing processes that Merge Design Thinking and Software Development

In this section, we present existing processes that merge DT or a similar methodology such as HCD with an agile software development process. We have selected models that involve a human centered process with the same team that is developing the software, thus the same software engineers are participating in the creative process as well as in development.

### 3.2.1. DT@Scrum

DT@Scrum [40] is a Design to Development Model approach to combine DT and Scrum into one software development process, that would include DT activities in three phases, making sure that all the requirements are fully understood and implemented in a human centered way. As can be seen in figure 3.1, the ratio of DT and development is changing throughout the process, starting with mostly DT activities, the bias changes towards software development in the end. In the last phase of development, a so-called *DT Break Out* can be used to get back to the creative process in case of blockers. This approach was partially implemented in bachelor projects and a course at the Hasso-Plattner-Institute, but further evaluation is still pending.

### 3.2.2. Integrated Framework

The Integrated Framework for Design Thinking and Agile Methods for Digital Transformation (Integrated Framework)[39] is naming each agile iteration a Workable Module, as seen in figure 3.2. The requirements and design parts of the Workable Modules are applying the

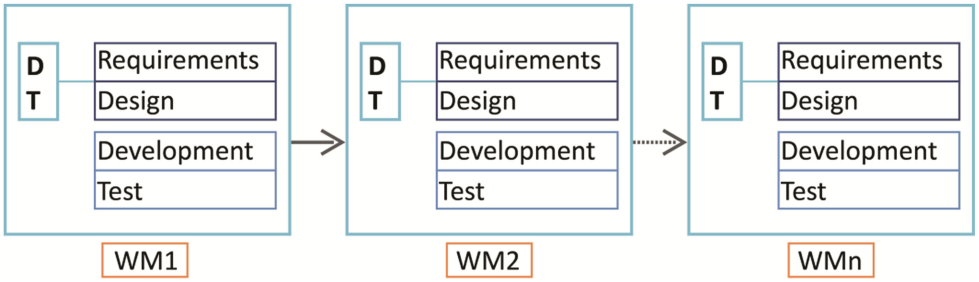



Figure 3.2.: Integrated Framework Process Model[39]

DT process. The outcomes of these steps will flow later into the development and test phases of the Workable Module. The principles of diverging and converging are used in all phases of the Workable Module. This framework tries to address the problem that "too much time is spent on understanding the problem before actual development process starts [... and] especially people reporting to higher hierarchy perceive DT as a risk"[39] - though there has been no tests to verify this yet.

### 3.2.3. DT@XP

There has been research to integrate DT into XP for solving wicked problems in software development. [81] The researchers are proposing the following practices to be included in the process:

- Integrate user stories with persona-based design
- Multidisciplinary teams for collaboration and creativity
- Prototype development
- User-centered design + user acceptance testing
- Agile usability testing throughout the development process

As can be seen in figure 3.3, DT is integrated into the exploration and planning phases of XP, by using so-called *DT-User stories* that are user stories as known from XP but built with personas and the DT empathy and define stages in mind. Later in the planning phase, those DT-User stories are used for usability evaluation, prototyping and user testing. An interesting approach of this process is to incorporate automated prototyping - web prototypes should be automatically created from a Unified Modeling Language based notation to minimize efforts in creating and updating prototypes. It is further emphasized that low-fidelity prototypes

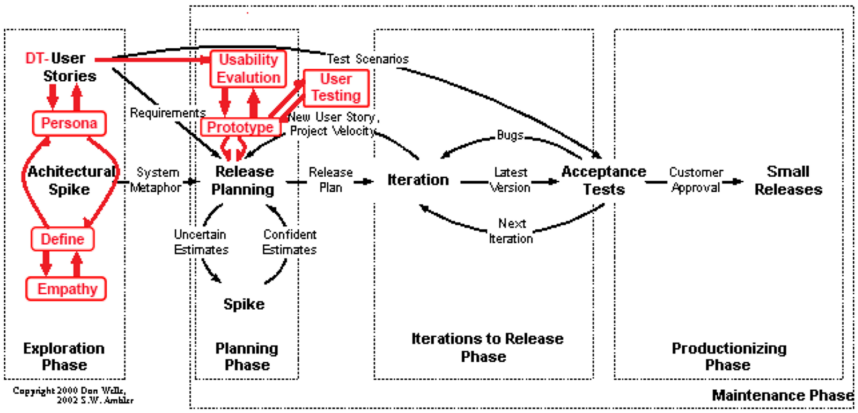



Figure 3.3.: DT@XP Process Model[81]

should be used to validate internally, while for external target users a high-fidelity prototype should be used. The proposed process has not been tested in real software projects yet.

### 3.2.4. Pet Empires

The Discovery by Design model for innovation developed by the Nordstrom Innovation Lab[36] was tested by de Paula et al. in IT development and was adjusted according to the results. The original model, as seen in figure 3.4, combines DT with Lean Startup and Agile by first applying DT for initial research, ideation and prototyping. An idea that has passed through various cycles of testing and prototyping may then later be turned into an MVP using the Lean Startup methodology together with agile for the actual development. After testing the unmodified model with three undergraduate students, the researchers found out that in the implementation phase the students did not consider the users' needs. Therefore the model was adapted to integrate DT in the whole process. Finally, the researchers conclude that "Design thinking should be used in the entire process of the Nordstrom Model"[22] and that the team should use "an interface that is closest to the final product"[22] to validate the user experience.

### 3.2.5. Converge

Converge[86] also combines DT, Lean Startup and Agile into a software development process. DT is used at the beginning of a project to empathize with the end user. As can be seen in figure 3.5, later in development DT is used to address so-called *knots*. These can appear at any moment of the project - they "concern the ability to understand the user, their limitations and needs"[86]. DT should be used in these cases "to come up with different creative

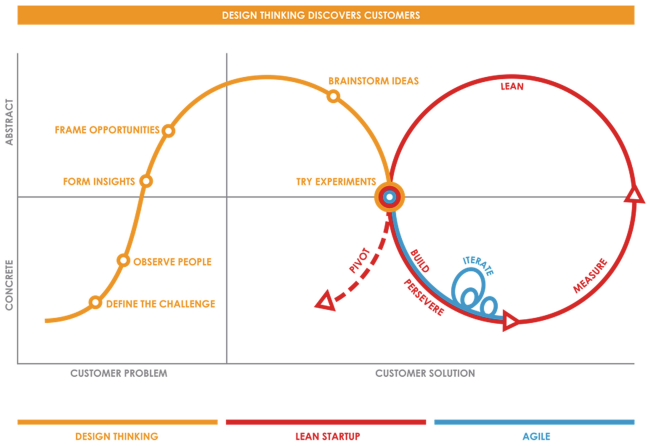



Figure 3.4.: Discovery by Design Process Model[36]


Figure 3.5.: Converge Process Model[86]

solutions beyond technical issues"[86]. The researchers conclude that it is possible to integrate DT with Lean Startup and Agile, and that the proposed model works well with software development and project conception. Furthermore, the study showed that a team of undergraduate students with little experience can create valuable applications that considered real user needs.

### 3.2.6. Human-Centered Agile Workflow

For the Human-Centered Agile Workflow (HCAW)[33], researchers have combined various sprints into so-called *cycles*, that each contain a Sprint 0 and further sprints. As can be seen in figure 3.6, the Sprint 0 consists of a cycle conception phase and furthermore includes the conception of the first and second sprint of the cycle. The goal of the cycle conception is to create epics by using suitable HCD methods in different phases. Later, the epics are allocated to sprints and roughly estimated. In the second phase of Sprint 0, the technical groundwork is set for the two next sprints. Additionally, a prototype is built at least for the next, if necessary also for the sprint after. After evaluating the prototype, user stories are written. At the end of Sprint 0, the user stories are estimated. After Sprint 0, the iterative sprints start, which consists of the actual implementation of the sprint, but also planning of the next two sprints in a similar manner as the second phase of Sprint 0. There is a strict time allocation suggested by the paper for the cycle conception, sprint 1 and 2 conception and every sprint. The model is currently in evaluation in three projects, but at the time of writing results were not available yet.

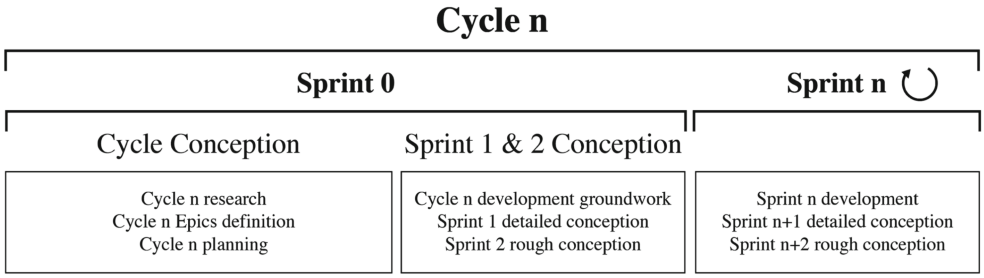



Figure 3.6.: HCAW Process Model[33]

### 3.2.7. Summary

An overview of all presented models can be seen in table 3.1. All processes have in common that they focus on DT in the beginning and then switch to programming during each agile iteration; as described by Lindberg et al.[57], this is the Design to Development model. Only two processes are also using DT later in development - Converge has so-called *knots* and DT@Scrum uses *DT Break Outs*. Both concepts are rather similar as they use DT to tackle blockers. Though two models have not been applied in practice, the other four could show that applying DT with agile methodologies is appropriate and leads to good and innovative results.

| *Process* | *Methodologies* | *Applied in Practice* | *Adoption Model* |
|---|---|---|---|
| DT@Scrum | DT, Agile (Scrum) | Partially, one course | Design to Development / In Development |
| Integrated Framework[1] | DT, Agile | No | Design to Development |
| DT@XP | DT, Agile (XP) | No | Design to Development |
| Pet Empires | DT, Lean Startup, Agile | Yes, one project | Design to Development |
| Converge | DT, Lean Startup, Agile | Yes, one project | Design to Development / In Development |
| HCAW | HCD, Agile (Scrum) | Yes, three projects | Design to Development |

Table 3.1.: Existing Process that incorporate Design Thinking

[1] Integrated Framework for Design Thinking and Agile Methods for Digital Transformation

## 3.3. Creativity Methods for Developers

Cajander et al.[13] have found out that "[there] is no clear picture of the responsibility for usability" and that "[usability] goals are unclear" in Scrum based projects. Furthermore Lárusdóttir et al.[53] describes "that the big picture of the user experience is often missing in Scrum projects". What is more important, the researchers conclude that UX specialists need to work closely with developers. On the other hand, Jurca et al.[51] recommend artifacts and practises to improve communication and collaboration between agile developers and UX designers.

In the following sections we will describe creativity methods for developers that address these and further problems of agile software development.

### 3.3.1. Design Thinking in Academic Context

An approach described by Domschke et al.[25] was to teach DT to software engineering students. Two different courses took part in the study - one five day block course with software engineering students and a nine months project with interdisciplinary team members. In contrast to the interdisciplinary teams, the homogeneous teams were not open towards some parts of DT methodology, like empathizing with the end user and understanding their needs, as they did not result in technological outcomes[25]. Thus, the researchers suggest that teaching DT leads to better results in interdisciplinary teams[25]. Furthermore, autonomous teams create better solutions[25].

### 3.3.2. Enter the Design Studio

A case study was conducted by Ungar and White[85] to find out benefits and challenges of a design studio approach in the context of agile development and User Centered Design (UCD). A design studio first produces rough sketch designs according to common guidelines, then all the participants gather their ideas and discuss them. Finally all ideas are merged into a common design concept. In the study, developers have conducted research and were part of the design process, thus "giving the ownership of design to the entire team"[85]. The researchers discovered that the "design studio facilitates knowledge transfer and role sharing"[85] and further "developers can learn about design [... and] designers can learn more about the challenges faced by developers"[85]. Ungar and White conclude that design and UCD knowledge was "spread throughout the organization"[85] and in general the quality of the designs has improved.

### 3.3.3. Service Design Prototyping in Software Development

Sauvola et al.[76] try to integrate SD Prototyping into software development. SD is another HCD approach that has its focus on innovating services. The proposed model involves SD at three stages: before, in and after development. In the first phase, experience prototyping is used to gain customer insights and analyse the customer journey. The outcome is the product backlog of the MVP. Only after that the actual development process starts. After each sprint there is a UX validation of the developed features to integrate them into a common vision. After the development has finished, but the product has not been launched yet, there is a UX review to evaluate the product or service with customers and other external stakeholders. Also, it might be needed to educate relevant stakeholders with the use cases of the software. The proposed process has been tried out in four companies, and according to the researchers, could show that "SD methods enhance the software development process and benefit both the developers and users by enabling companies to develop customer-centric products and services that are useful, desirable and competitive in the market"[76]. Furthermore, the participants perceived that "instant feedback, faster and more accurate decision-making, continuous learning as well as focusing the development effort on things that bring value to users"[76] are benefits of using SD experience prototyping methods.

### 3.3.4. UX Toolbox

The *UX Toolbox* proposed by Pedersen et al.[68] serves as a blueprint for the Toolbox Model approach which is applied to propose methods that would allow software developers to perform UX tasks and seamlessly integrate them into an agile development process. The toolbox consists of selected and modified UX methods that were enriched with guidelines and templates. Furthermore, it includes material to support a one-day training that according to the researchers should be sufficient to ensure that software developers can apply the methods without UX specialists. As a result, the thesis states that the developers were interested in learning UX competences and could perform their work without the need of UX specialists. It is mentioned though that daily management faced problems to provide sufficient resources for the research.

## 3.4. Summary

We have seen various attempts as described in section 3.1 that try to incorporate DT with agile software development - these show that DT indeed is a relevant topic in IT. Further, in section 3.2, we showed that research is conducted in order to define processes that combine agile with DT or a similar methodology. These processes teach DT to the same team that is responsible for developing the software, but they do not specify how exactly DT is used in

development. Using individual methods or toolboxes in combination with workshops proved to be a good approach as we have outlined in section 3.3. Research shows that, developers are open to DT if certain aspects are considered:

- Do not spend much time on upfront problem solving before development starts[39]
- Smaller team sizes have been identified as a better fit to incorporate DT[30]
- Teaching DT leads to better results in interdisciplinary teams[25]

We want to fill the knowledge gap in this area and show how a DT@IT Toolbox can address existing challenges. A DT Method Toolbox does not exist in current research yet, though various of the presented processes in section 3.2 could benefit from it.

# 4. Research Methodology

The research methodology for this thesis is explained in this chapter. We start by describing the research goal itself with the corresponding hypotheses and research questions. Building upon that, the methodology for selecting the methods is defined. Finally, we outline the practical phase including the test setup, questionnaires and interviews and their evaluation.

## 4.1. Research Goal

Software development companies have problems providing software that is human centered. As described in section 1, we also know that there is a high demand for software with better usability. In this section, we want to introduce two hypotheses about how DT could help to tackle this problem. The two hypotheses lead to three research questions that were used by us to verify the hypotheses. As a final goal - to enhance team collaboration and user centricity - we created a DT@IT Toolbox: A collection of methods and mindsets in an easily understandable format, so that it would be not only useful for experienced DT teams, but could be also applied in teams without the need to hire an external expert. The toolbox aims to improve empathy for team members and end users, as well as to foster prototyping and creativity by using adjusted DT methods. Furthermore, this thesis extends upon existing research in this area. It highlights how the DT breakouts in the last phase of a process such as InnoDev or DT@Scrum, as described in chapter 3, can be realized and which DT methods are helpful for developers during agile development.

### 4.1.1. Hypotheses

> Establishing Design Thinking practices for giving feedback and exploring and sharing ideas fosters empathy for team members and thus leads to a better shared understanding and less miscommunication.
>
> Hypothesis 1

With the first hypothesis we are assuming that it is possible to raise empathy inside the team by regularly applying DT methods that:

- help giving constructive feedback
- ease the ideation process and sharing of ideas

We also assume that by raising the empathy, a better communication between different team members like developers and designers could lead to:

- a better shared understanding of the product
- less miscommunication

More cooperation within the team could lead to a higher efficiency of the development teams by eliminating communication issues and establishing shorter communication paths. Furthermore, a lack of information about the product that is being developed needs to be tackled, so that new knowledge that is available to the team is also distributed across its members.

> Making use of Design Thinking techniques that bring developers into contact with the user, together with prototyping methods, help developers to gain empathy for the user and bring a human centered perspective to the development team.
>
> Hypothesis 2

The second hypothesis is about improving empathy for the end user by regularly applying DT methods that:

- introduce end users to the team
- facilitate the process of prototyping

Furthermore, getting to know the end user and rethinking requirements from their point of view could also make the end product more human centered. This ensures that the needs of the end users are considered in more parts of product development - specifically in the actual implementation.

The two hypotheses were summarized in figure 4.1: On the left side, the end users are providing their knowledge, feedback and needs to the team, and on the other side the team itself is spreading that knowledge and deepening it by feedback, ideation and a shared understanding. By improving the communication inside the team, the newly gained knowledge from the end user should be spread easier across team members.


Figure 4.1.: The relation of the first and second hypotheses

### 4.1.2. Research Questions

The main research goal can be divided into three research questions, that are based on the two hypotheses in section 4.1.1.

Which methods from Design Thinking can be used in an agile development process

- to support the work of developers in a human centered way?
- to support empathy for team members and end users?

Research Question 1

To answer this question we have first selected twelve methods used in DT that are known to support both empathy and thinking in a more human centered way. These twelve methods were then evaluated in multiple groups.

How to adjust the methods to suite the needs of the engineering profession?

Research Question 2

As the field of application is very different, it might be necessary to adjust the methods for usage in software development teams. Some methods had to be changed slightly to also reflect e.g. team empathy, and not only end user empathy. Whether the methods were adjusted successfully and if further adjustment is necessary was evaluated by providing questions about applicability in method evaluation questionnaires, which is explained in section 4.3.2.

How to gather these methods in a toolbox, so that software developers can be easily trained to effectively apply them in appropriate situations?

Research Question 3

The selected methods should be gathered in a toolbox, like the UX Toolbox presented in section 3.3. Therefore for each selected method we wrote an application guideline with examples specific to software development. These examples should make it easier for the software development team to understand the methods and be able to pick the right method at the right moment.

## 4.2. Selecting Methods

### 4.2.1. Criteria for Methods

First, we had to define criteria to find suitable methods for evaluation. The methods should be widely applicable, to ensure that they are not too specialized and can be useful in a variety of situations that apply to software development. The focus should be on day to day agile work to ensure that we could achieve much even with a small method toolbox. Ideally the application of the method takes less than 60 minutes. Furthermore, methods should be easily understandable rather than too complex so that the team can apply them without further schooling and can concentrate on the actual task. Activities which might seem uncomfortable to the participants should be avoided, such as performing a theater role play game, creating a "Just Play Prototype" or making an advertising film. Performing these tasks already requires a high empathy in the team and might work better for teams that already have experience in DT activities. Most importantly, the methods should support situations or interactions that we want to foster - these are described in section 4.2.2.

To sum up the criteria - the methods:

- are widely applicable
- are ideally shorter than 60 minutes in application
- are easily understandable
- do not include activities that might seem uncomfortable
- are known to support situations or interactions that we want to foster

### 4.2.2. Categories

One of the criteria for selecting methods for evaluation is whether they support specific situations or interactions that we want to foster. For that, we have created categories that directly

or indirectly map to the DT process and are derived from our hypotheses mentioned in section 4.1.1. The table 4.1 lists each category with the hypotheses that they were derived from.

| *Category* | *Hypotheses* |
|---|---|
| Understand Product | H1 / H2 |
| Empathize with Team | H1 |
| Empathize with End User | H2 |
| Communicate | H1 |
| Ideate | H1 |
| Prototype | H2 |
| Feedback | H1 |

Table 4.1.: Mapping our Hypotheses to Categories

#### 4.2.2.1. Mapping to the Design Thinking Process

There is a set of methods that can be used to support each step of the DT process - as described in section 2.2. Most methods are not bound to a single step, but can be useful in multiple steps of the process - still they are considered to fit more in a specific part of the process. E.g. certain methods are used to empathize, ideate or prototype and thus fit into different steps of the DT process.

| *DT Process Step* | *Category* |
|---|---|
| Understand | Understand Product |
| Observe | Empathize with Team / End User |
| Point of View | Communicate |
| Ideate | Ideate |
| Prototype | Prototype |
| Test | Feedback |

Table 4.2.: Mapping Steps of the DT Process to Categories Based on the Hypotheses

*Direct Mappings*

Some of these mappings are clear from the first sight, as they have the same name - it is easy to understand that "Ideate" and "Ideate" directly map to each other, as well as "Prototype".

"Test" and "Feedback" are not necessarily the same, but testing in Design Thinking involves gathering a lot of feedback, so these two also directly map to each other.

*Indirect Mappings*

The step "Point of View" is mapped to "Communicate", since the possibility to understand someone else's point of view, and most importantly, the ability of expressing thoughts from a different perspective, enhance communication. In this step, it is also very important to communicate all insights and findings of the whole team properly to finally formulate a point of view that is understandable to everybody and can be agreed upon. To empathize with your team or the end user, the steps "Understand" and "Observe" can help. This becomes clear when we look at other definitions of the DT process which combine these two steps into one that is called "Empathize", e.g. as in the definition of the Hasso-Plattner-Institute of Design Standford [70]. Usually, the aim of the research phase, which includes both the step of "understanding" and that of "observing", is to bring the team to a common expert level [37]. Finally, the "Understand" step is beneficial to understanding the product as it involves many activities to research or understand the actual given problematic - in the case of a software development team that could be for example the already existing product, or at least the work in progress.

### 4.2.3. Sources

Various method card sets have been developed to ease the selection and usage of methods during the DT process. We have used two card sets and a starting guide from the Hasso-Plattner-Institute of Design at Stanford as our main sources:

- Design Thinking Prototyping Cardset [61]
- IDEO Method Cards [46]
- D-School Stanford Bootcamp Bootleg [70]

For example, the 'Design Thinking Prototyping Cardset', which was created within the *Hasso-Plattner-Institute Stanford Design Thinking Research Program*, consists of 36 methods for DT novices [50]. On the other hand, the 'IDEO Method Cards' have been developed as a showcase and faciliator to inspire the DT process with 51 methods [45]. The D-School Stanford Bootcamp Bootleg[70] is a guide with mindsets, a description of the DT process and 39 methods which aim to support the design thinking practice. We chose these sources as they are specifically targeted towards beginners. Thus the selection contains easy to understand methods that help DT beginners to get started and thus might be more suitable for our use-case.

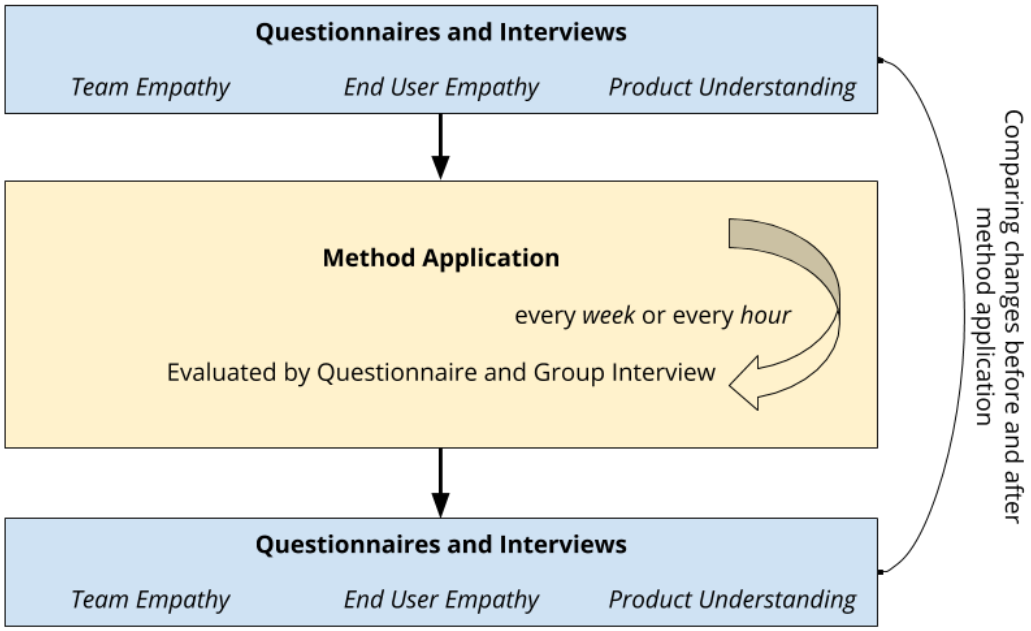



Figure 4.2.: The overall test setup

## 4.3. Practical Phase

### 4.3.1. Test Setup

The practical phase was conducted in five evaluation groups that were testing the methods in their work context. The overall test setup can be seen in figure 4.2. At the very beginning, questionnaires and interviews for the three topics "Team Empathy", "End User Empathy" and "Product Understanding" were conducted to get an initial overview of these topics. Later the methods were applied by the group by resolving everyday problems using one specific method over a certain time span. Each method has an own document that contains a description of the method, how to use it and some examples. The group was assigned a very general task, e.g. "Please use the method at least two times and prepare to present the results to the group.". After applying the method, the group presented their results and a group interview was conducted to get the general feelings and impressions. Finally, a questionnaire was filled out.

Depending on the group, three or twelve methods were evaluated per group, either every hour or every week. After the method evaluation phase, the same questionnaires from the very beginning and interviews were conducted again to be able to find out whether any of the three topics changed after using the DT methods.

The setup differed between the core evaluation group and further evaluation groups because of time restrictions - the groups are further described in detail in this section. The blue colored steps seen in figure 4.2, that is the initial and final questionnaires and interviews, were only conducted with the core evaluation group - this group also has evaluated all twelve methods. All the other groups have evaluated three methods, these were chosen by the groups themselves.

The core evaluation group is a team of seven programmers, two designers and one assistant that is working in multiple projects. They are working at *DESIGN-IT*, the company has eleven members in total and is creating custom business applications for multiple partners, like the school- and grouptravel platform `https://www.db-gruppen.de` by *DB-VERTRIEB GmbH* a subsidiary of *Deutsche Bahn AG*, the biggest German railway company.

Besides the core evaluation group, there has been further evaluation groups, that have been testing selected methods in their daily work. All companies are from the region Berlin-Brandenburg in Germany and their software departments are co-located.

At *Company I*[1] a group of six programmers has evaluated three methods in a four hour workshop. The attendees work in the fields of data analysis, articifial intelligence, machine learning and computer vision. The company is part of a German research organization with various institutes in Germany with over 25000 employees. The institute that participated in our study has about 430 employees.

In the same way, at *Company II*, a group of eight participants did a four hour workshop with three methods. They were from different teams including one sales-manager, one designer and a mixed group of front-end and back-end developers. DT is announced and promoted a lot by the company, but they did not use it for the software development team yet. The company has about 50 employees and develops solutions for team collaboration and security.

Another workshop in the same setting took place at *Company III* with a group of nine software developers from different teams. Compared to the other groups, the age spectrum of the participants was wider. The software department of the company has 24 employees and there are 91 employees in total. The main product of the company is a project management software.

At *Company IV* a group of eight participants did three workshops an hour each. Each workshop has evaluated one method in a three week schedule. The participants were a mixed team of developers, UX designers and agile coaches. The company has about 170 employees in total and is building business applications.

An overview of all companies and teams participating in our study can be found in table 4.3.

[1] For confidentiality reasons we will refer to these four companies as *Company I*, *Company II*, *Company III* and *Company IV*

| *Company* | *Employees* | *Team Size* | *Participants* | *DT Knowledge* |
|---|---|---|---|---|
| Design-IT | 11 | 8 | Developer, Designer, Team Assistant | None |
| Company I | 430 | 6 | Developer | None |
| Company II | 50 | 8 | Developer, Designer, Sales Manager | DT is promoted in the company and some participants were familiar with the concept |
| Company III | 91 | 9 | Developer | None |
| Company IV | 170 | 8 | Developer, UX Designer, Agile Coach | Some participants were familiar with the concept |

Table 4.3.: Overview of Companies and Teams Participating in the Study

### 4.3.2. Questionnaires and Interviews

Different questionnaires were used to evaluate the hypothesis and research questions. The empathy questionnaires are based on the Toronto Empathy Questionnaire[82] from 2009. This is a tool to measure the general empathy of a person. For our study, it was important to measure the empathy towards a special user group, instead of the general empathy. This is why the questionnaire was adapted to suit those cases[2]. The general structure was kept: there are 16 statements that have to be answered on a scale from zero, which means "never", to four, which means "always". The sum of all scores is the total score for the questionnaire[82]. The structure of the statements has also been kept and the meaning was adjusted to specifically address the target user group. As an example, the phrase "When a friend starts to talk about his/her problems, I try to steer the conversation towards something else" from the Toronto Empathy Questionnaire was adjusted to:

**Team Empathy Questionnaire:** "When a team member starts to talk about his/her problems, I try to find an easy way out by delegating the issue to somebody else or by proposing a quick-fix"

**End User Empathy Questionnaire:** "When an end user starts to talk about his/her problems with the software, I try to steer the conversation towards something else, e.g. I believe that the problem lies with the user not the software"

[2] Both questionnaires can be found in the Appendix, the team empathy questionnaire in A.1 and the end user empathy questionnaire in A.2.

We developed a product understanding questionnaire that contains eleven questions which address important aspects of a product, it can be found in Appendix A.3. The questions are biased on the end user point of view and general market positions. E.g. one question is "Describe key functionalities of the product, and name the benefits for the end user". The questionnaire consists of eight open format questions together with three closed format questions of the types differential scale and choice of categories [55]. One of those questions is a "Yes" or "No" question type. We have used mostly open format questions, because product understanding is a topic that has too many alternative choices that cannot be compiled into closed format questions [55]. The closed format questions have been just used for choosing the product that the participants are describing, and giving a subjective feeling about their product understanding, e.g. "How familiar do you feel with the product?". The questionnaire should show whether the core research group was able to deepen product understanding from the point of view of the end user and customer, by comparing the results of the questionnaire before and after the method evaluation phase.

The Method Evaluation Questionnaire is based on our research questions and hypotheses, all questions can be found in Appendix A.4. The questionnaire tries to find out whether and how the method was useful to the applicant, in which situations it could be used and if it was understandable. Furthermore, questions like "The method gave me new insights from my team members" were asked to find out if the method was useful for fostering empathy within the team - a similar question was asked for empathy towards end users. Similarly, further questions were asked to measure if the method had any effect on communication, ideation or shared understanding. The responses are a mixture of open and closed formats, as described by Leung [55]. This gives us the opportunity to have statistically comparable results, but also gives the participants the freedom to express their feelings and thoughts about the method, that could lead to further insights. The questionnaire includes eight closed format questions with differential scales and further six open format questions. The questionnaire was filled out by all participants that tried out a specific method in their everyday work context. The table 4.4 shows a list sections of the questionnaire together with our research questions or hypotheses that they are meant to validate.

### 4.3.3. Core Group Interviews

Besides the questionnaires, the core group had additional interviews before the practical phase and after it. The interviews were conducted as focused interviews as described by Flick [29]. Focused interviews have a predefined set of questions, but in contrast to structured interviews they allow to alternate or change questions if necessary to get more details about certain experiences. The scope of focused interviews is mostly very wide, to be able not to only collect foreseen insights, but also the ones that might be non-foreseen. This type of interview is according to Flick especially useful to focus on a specific subject, e.g. "jointly experienced situations" [29] like the method evaluation phase was for the core group.

| *Section* | *Question* | *Research Questions / Hypotheses* |
|---|---|---|
| General Method Experience | How did you like the method in general?<br>Feel free to describe your general experience<br>I liked about the method...<br>I disliked about the method... | RQ1 |
| Communication | The method allowed me to express my thoughts better | H1 |
| Team Empathy | ... gave me new insights from my team members<br>... helped me to understand my team members better | RQ1 / H1 |
| End User Empathy | ... helped me to understand the end users better | RQ1 / H2 |
| Ideation | ... helped me to find a better solution | H1 |
| Shared Understanding | ... helped to find a problem in our solution | H1 |
| Applicability | I could apply this method in the following situations<br>I could not apply this method in the following situations ... because ... | RQ2 |
| Understandability | How easy was it to understand the method?<br>How did you like the way of presenting the method? | RQ3 |

Table 4.4.: Mapping our Hypotheses and Research Questions to Sections of the Method Evaluation Questionnaire

An interview guideline that is based on the three main topics "Team Emapthy", "End User Empathy" and "Product Understanding" was used, it can be found in Appendix A.5. The final interviews also included a section about the "Methods" to get a general feeling about the methods and their usefulness. One of the goals of focused interviews is to "maximize the scope of the topics and to give interviewees an opportunity to invoke points of view that had not been anticipated"[29], so it is a good tool to[29]:

- get a good overview of the situation before the method evaluation
- gather broad feedback about the method application phase

### 4.3.4. Data Evaluation

The open questions of the Method Evaluation Questionnaire, Group Method Interviews and Solo Interviews were evaluated by using coding as defined by Flick[29]. For this we have created categories, that fit the free text from the questionnaires and the transcribed text from the interviews. For each of these categories we have then created a coding guideline, which we have developed according to our data by comparing each category for similar statements. Lastly, each interview was coded by using these coding guidelines for each category. The solo interviews have been recorded and later transcribed, for the group method interviews we wrote down notes during the interview. The closed questions of the Method Evaluation Questionnaire and Product Understanding Questionnaire were evaluated by comparing the average and mean values of the results. All questions are comparable, because all answers are on the same scale in the range from one to five. Tendencies and deviations were further evaluated by applying standard deviation and visualizing the data in box plots.

The general evaluation of the core group has been done using the empathy questionnaires, product understanding questionnaire, group method interviews and solo interviews. The evaluation of the core group is used to answer the hypotheses one and two. The Empathy Questionnaires as described in section 4.3.2 were evaluated by comparing the total score before and after method application. The total score is calculated in the same way as in the Toronto Empathy Questionnaire[82]. The questionnaires have a test-retest reliability and have been already used by other researchers for comparing empathy levels before and after certain events[67][42].

To answer the research question one we have used the method evaluation questionnaire and the group interview for each method. Furthermore we have also looked for mentions of the methods in the solo interviews that revealed more insights.

The second research question was answered by the questions "I could apply this method in the following situations" and "I could not apply this method in the following situations ... because ..." from the method evaluation questionnaire. The method group interviews and solo interviews were also used to see if any insights on general application of the methods have been revealed - the solo interviews additionally contained a specific section that asked how often methods have been used and what the specific problems are to get them into daily work.

The evaluation of understandability, which answers research question three, was answered by two questions about understandability from the method evaluation questionnaire. Additionally, insights from solo and group method interviews were used.

An overview of all questionnaires and interviews that we used together with the respective research question or hypotheses that they answer, can be found in table 4.5. Our research was conducted according to a mixed method approach. This approach combines collecting both

qualitative and quantitative data and using various sources for the evaluation[17]. According to Creswell et al. a mixed method approach is suitable if performing only "one [source] may provide an incomplete understanding"[16] and when there is a need for a second source to explain the first[16]. A mixed method approach is also good for research questions that require both exploration and explanation[16], as in our case. Further the authors state that "[the] strength of one method may offset the weaknesses of the other"[16] and "[using] multiple sources of data simply provides more evidence for studying a problem than a single method of data"[16].

| *Questionnaire / Interview* | *H1* | *H2* | *RQ1* | *RQ2* | *RQ3* |
|---|---|---|---|---|---|
| Method Evaluation Questionnaire | | | X | X | X |
| Product Understanding Questionnaire | X | | | | |
| Team Empathy Questionnaire | X | | | | |
| End User Empathy Questionnaire | | X | | | |
| Group Method Interviews | X | X | X | X | X |
| Solo Interviews | X | X | X | X | X |

Table 4.5.: Overview of Questionnaires and Interviews with the Respective Hypotheses and Research Questions they answer

# 5. Selected Methods

The selected methods for the DT@IT Toolbox are presented in this chapter. All the methods were assigned to one or more categories as specified in section 4.2.2.1. In this chapter each method is presented and described. The method sheets that we have created to present the methods to the evaluation groups can be found in Appendix B.

The methods were selected using the criteria defined in section 4.2.1 and the sources described in section 4.2.3. A list of all selected methods, with their corresponding category and hypotheses can be found in table 5.1.

| *Method* | *Categories* | *Hypotheses* |
|---|---|---|
| *Personal Inventory* | Empathize with Team / Empathize with End User | H1 / H2 |
| *Character Profiles* | Empathize with End User | H2 |
| *Customer Journey Map* | Understand Product / Empathize with End User | H1 / H2 |
| *Five Whys?* | Communicate | H1 |
| *Letter to Grandma* | Communicate / Understand Product | H1 / H2 |
| *A Beginner's Mind* | Ideate | H1 |
| *30 Second Sketch* | Ideate | H1 |
| *Powers of Ten* | Ideate | H1 |
| *Feedback Capture Grid* | Feedback | H1 |
| *Five Finger Feedback* | Feedback | H1 |
| *Empathy Tools* | Prototype | H2 |
| *Paper Prototype* | Prototype | H2 |

Table 5.1.: List of Selected Methods with corresponding Categories and Hypotheses

## 5.1. Personal Inventory

This method works by interviewing peers for their favorite items[46], but we have adjusted this method to the circumstances of developing a software in a team and with the customer or end user. A similar adjustment to this method has been done by Jalleh in a study to co-design an air ambulance cabin with a medical flight crew[49]. The adjusted method works in two ways. Everyone participating shares a picture of his or her working space with annotations, additionally, the working stack should be described - e.g. by presenting it to the team. Optionally, the participants can add a picture of their favorite thing in the office. Additionally, they should ask their customers and/or end users to send them pictures of their working space.

The Agile Manifesto sets "[i]ndividuals and interactions over processes and tools"[2], still we think that especially when different tools are used within one company, there should be an exchange about these tools and practices. Besides of that, team members of different areas, like programmers and designers, use very different tools, because of their different tasks and focus.

Seeing the workspace of people that are using the software, can give the participants an overview on how and where the end users are using the software. Martin and Hanington say about Personal Inventories that they "allow the designer to see and understand the relevance of objects in a user's life from the participant's point of view, to inspire design themes and insight"[41]. As the working space depends on the type of software that is being developed it might reveal insights about possibilities and difficulties.

## 5.2. Character Profiles

Character Profiles, also called Personas, are fictional characters[18], that are "based on observation of real people"[46]. A finite amount of personas can be created per project[52], "who in turn [represent] a group of real consumers with similar characteristics"[62].

> "Personas are not real people; rather, they are imaginary examples of real users they represent"
>
> Cross 2003

Each Character Profile has a name, image[18] and a further description about what the character likes, dislikes, its occupation and other narrative information[62]. The profile also includes needs and goals with regard to the project[62]. Combining the data from observing and interviewing the target group with a narrative "brings the persona to life"[62].

Creating the personas is not a focus in day to day agile software development, as this task should be done in a separate research phase.[41] For our toolbox working with a persona is an interesting approach though, so the character profiles can be used in everyday development.

This method is a good tool to get an impression of the end users, as it creates empathy with the end user[62]. Furthermore, character profiles focus development on the end users and helps to prioritize the audience[62], thus considering different types of end users, for which personas exist. A persona challenges assumptions about end users and prevents self-referential design[62]. Using Character Profiles in a software development team could make communication about end users easier, as a persona is a medium for communication[38] that benefits from the shared knowledge[62].

## 5.3. Customer Journey Map

A Customer Journey Map, also known as User Journey Map[52] or Journey Map[70], describes the journey that the customer takes in order to consume a product or service on a timeline[52]. One has to define all the steps that the customer takes - even the ones that do not have direct touchpoints with the creator[70]. E.g. the journey could start by having a need, over finding a product that would fulfill this need and ending up in consuming a product or service and finally recommending it[75, p. 2]. The gathered data is then visualized[52] and can be propagated by storytelling. An example of a Customer Journey Map can be found in the Appendix B.3.1.

Customer Journey Maps describe the journey from the end users point of view[54] detailing important activities and associated emotions and thus reveal insights about where things work out great, what pain-points exists and new opportunities to tackle[52]. For Agile Teams this method can help to identify usability issues within the software and surrounding services such as contacting support. Thus, the Customer Journey Map allows developers to see their product through the eyes of an end user. Furthermore, a visual Customer Journey Map enables the developers to discuss and collect data around the customer experience and thus we expect it to foster product understanding as well as user empathy. Though the creation of a Customer Journey Map can take considerably longer than an hour, we still decided to include this tool because of the mentioned benefits. Furthermore, it is also possible to map smaller features that would make it possible to perform this method in less time.

## 5.4. Five Whys

Five Whys is a simple tool, to find the root cause of a problem and to uncover useful insights[72]. It is "quick and easy to use [... and] leads the team to a better understanding of the issue"[72]. The method was invented by Toyoda Sakichi to explore the cause-and-effect relationships[79] in industrial manufacturing, but in general it is applicable to any field. In its simplest form, the method works by repeatedly asking "Why?" until one is satisfied with the answer.

In most cases root causes are hidden behind symptoms, and the first reaction is to discover those symptoms. As symptoms are caused by further causes, fighting the symptoms can only give you short-lived results, as the root cause still exists[66].

Likewise, with information based problems one needs to take care of finding the root cause: e.g. a specific customer desire might be based on a root desire that could be solved in a much more efficient way. So to know that the work is based on correct assumptions, asking and digging deeper could lead to a better understanding of your conversation partner. A similar approach is found in agile user stories, that structure requirements in a way that not only includes what the user wants to do, but also why[31]. Thus the development team can decide how it implements the feature, e.g. instead of getting the requirement to put a "print button" in the layout, a user story would explain that "[as] a manager, I want to be able to print out a budget overview so that I can find out which departments need my support".

## 5.5. Letter to Grandma

“Letter to Grandma” is a method to improve sharing and understanding of requirements and tasks[1]. It allows a knowledge transfer that focuses on the essential aspects[61] without omitting core functionalities and benefits[1]. It is about writing a letter that describes requirements or tasks in a way that one would write to his or her grandmother[61].

When sharing information the knowledge provider holds the required information and transfers this knowledge to one or various knowledge recipients. The task of knowledge sharing requires the knowledge provider to translate the knowledge in a format that allows to encode the information in a way that can be interpreted by the knowledge recipient[58, p. 10]. The most common format of knowledge transfer is to translate the knowledge into words. A big problem of knowledge transfer lies in this translation, especially when handling tacit knowledge. While explicit knowledge is easy to codify, tacit knowledge cannot be easily exchanged in that manner.[58, p. 3] Thus information is lost while encoding and also while decoding information by the recipient.

Writing a letter to grandma helps to minimize these problems when dealing with require-

ments or tasks to be shared[1]. The letter could also be addressed to somebody that has never dealt with the project - or to give an extreme example: an alien[11]. All these recipients have in common, that they have no prior knowledge and thus need a description of the essentials enriched with further information that is only known to the team. This last part is a very important one: While the knowledge provider might think that some piece of information is already known to the team - or not worth to mention - the recipients may be missing that specific part to be able to complete the knowledge transfer. Concentrating on the essentials makes sure that one sticks to the important parts and does not overwhelm the recipient with unnecessary details, but gives a good overview of the requirements[1]. Furthermore, it is useful to foster universal comprehensibility[1]. Because of that, we think that it might be good e.g. to explain epic user stories to the team or to introduce a new person to a new project.

## 5.6. A Beginner's Mind

The general idea for this method comes from Zen Buddhism and is called "shoshin"[83], which means "beginner's mind" in English. This state of mind provides an attitude of openness and a lack of preconceptions[83]. In contrary there also exists the "expert's mind" that describes the state when the constraints of the subject are obvious and there is only a few, very clear solutions[83]. To apply this method one can either try to adopt a beginner's mind oneself or ask a real beginner - either somebody outside of the team that ideally has not much knowledge about the project, or one can go out on the streets and ask somebody - e.g. at lunch. Belshee found out that knowledge sharing improved by using A Beginner's Mind with Pair Programming, keeping the pair in a beginner's mind by having a new partner every 90 minutes[4].

> In the beginner's mind there are many possibilities; in the expert's mind there are few.
>
> Shunryu Suzuki[83]

Our experiences with a topic include assumptions, perceptions and stereotypes that we take for granted[70]. By applying A Beginner's Mind we open ourselves to unexpected discovery and innovation because we are not blinded by our prior assumptions[4] and have a better view of the details and the bigger picture. In general, uncertainty gives us the courage to try things out anyway, regardless whether they will work or not - because we don't know better yet[4]. Apart from the discussed benefits, with a mind of a beginner one is able to empathize better with other beginners. People are more creative, learn faster and achieve more[4]. Belshee states that A "Beginner's Mind is a very efficient way to solve programming problems"[4].

## 5.7. 30 Seconds Sketch

In the early ideation stage this method is used similarly like brainstorming to generate many ideas in a short time, but giving it additionally a visualization[61]. A sketch can be seen as a simple and low fidelity prototype - it is quick[61] and thus easily disposable. The method simply works by creating sketches in no more than 30 seconds on paper[61]. Depending on the phase of development, one can adjust how deep to dive into the solution space[34]. This exercise can be performed with multiple people and also iteratively by drawing multiple sketches in a row[34].

Sketches in general are a good tool to express, develop and communicate ideas[34]. By visualizing an idea one can make it much more tangible for their peers, which enables to gather better critique and feedback on them[34]. We think that it is easy to abandon an idea from a sketch because of the tiny effort one does not get too attached to it. This is important to be able to explore different concepts and thus seek for alternatives by thinking through ideas [18, p.25].

## 5.8. Powers of Ten

The method Powers of Ten is a reframing technique based on the short film of the same name from the year 1977[27]. The movie shows a picnic scene in Chicago with different magnitudes of distance from 1 meter up to $10^{24}$ meters and then down to $10^{-16}$ meters. It takes you up into the deep universe and then back into the hand of the man that is at the picnic until you only see atoms. Powers of Ten is an exercise that allows to change the point of view by varying magnitudes of context for insight generation[70].

The first interesting thing about Powers of Ten is that there is no linear, but exponential growth. For humans it is hard to think exponentially because our brains are trained to think linearly[20]. The second thing is changing the point of view - a problem can be seen from very different magnitudes of a certain dimension. Varying the point of view allows to find the right framing for different issues of a certain problem - basically by zooming in and out the current scene[70].

## 5.9. Feedback Capture Grid

The Feedback Capture Grid is a method to gather feedback in a structured grid[70]. As seen in figure 5.1, it is based on four quadrants labeled with easy to understand symbols, that are

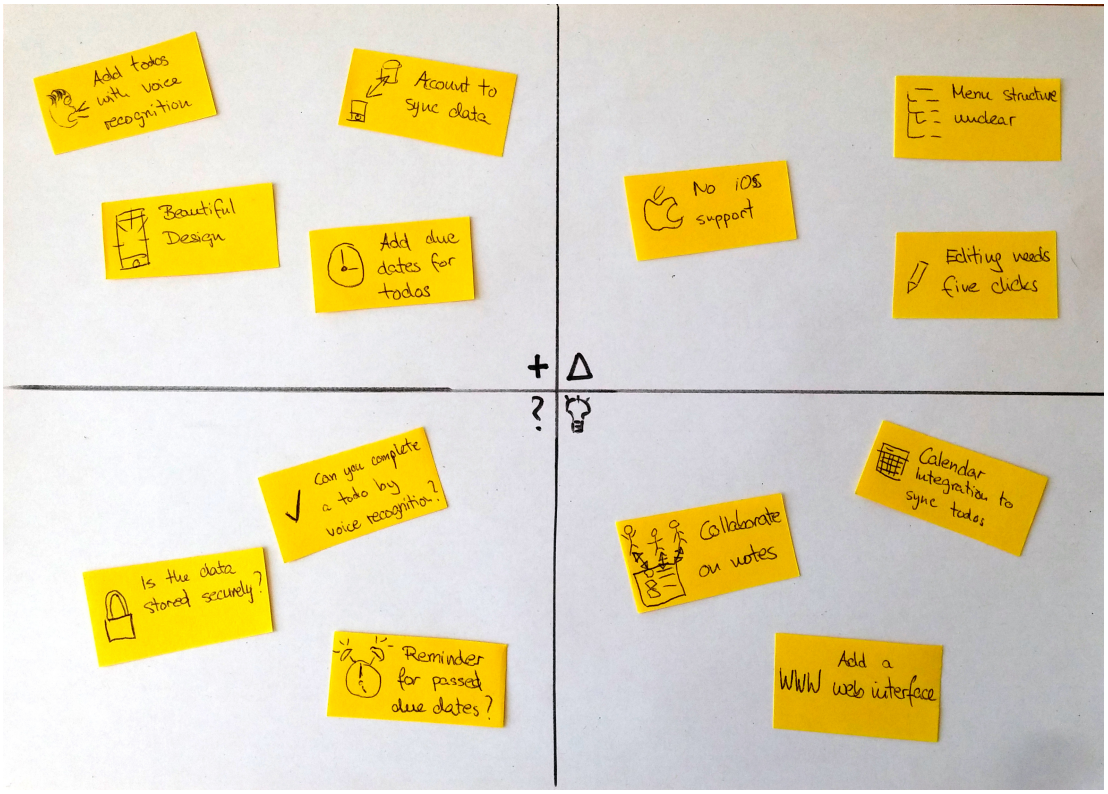



Figure 5.1.: Feedback Capture Grid Example from the Method Sheet

used as categories for the feedback[70]. Each quadrant is filled with feedback by the participants and should be categorized by: likes, dislikes, ideas and questions[70].

This method helps to be systematic about feedback and forces to think about different aspects[56]. It is useful for on demand feedback or post-mortem unpacking[70]. The method is seen as an easy way to present results and allows to see common themes and areas that need more development[74]. Because this method is performed as a written exercise, it facilitates documentation[56] by capturing the results e.g. in a photo.

## 5.10. Five Finger Feedback

Five Finger Feedback is an unconventional alternative to other feedback gathering methods[9]. The method template gathers differentiated information and the fingers serve as a symbolic medium for critical statements[9]. The method is assigning each finger a specific meaning and those associations are then used to give structured feedback[9]. The gathered feedback is structured and the exercise can be performed written or oral[9].

Every finger is associated with a type of feedback that should be given[9]:

**Thumb** Expressing what was liked

**Index Finger** What was noticed, or something to point out

**Middle Finger** Expressing what was not so good, or could be done better

**Ring Finger** What was emotionally interesting

**Little Finger** What was missing, or did not get enough attention

The method fosters constructive feedback and considers different aspects[7]. Furthermore, it gives clarity, can improve communication in groups and allows every participant to formulate his or her statements[8]. This method is not described in the mentioned sources in section 4.2.3, it was described by Brauneck et al.[9]. Despite that, we included the method as it is our favorite one for gathering feedback and we have used many times in workshops and other hands-on situations.

## 5.11. Empathy Tools

Patricia Moore described in her book[63] how she has used empathy tools as a young woman to put herself in the shoes of elderly people. She has used tools such as glasses for blurring her vision, earplugs so she does not hear well and uneven shoes to be forced walking with a stick. Empathy Tools is a term used to describe physical objects and cognitive or social techniques that provide designers with a feeling and deeper understanding of users with various abilities[46]. The method is used in inclusive design, which means to gain more empathy and understand better other user's capabilities[44]. Inclusive design e.g. could be achieved by experiencing a product in a way that an end user with certain disabilities would[46]. Empathy Tools is an informal method that has no defined process to be taken[48]. It is about discovering the end users disabilities to get a deeper understanding of the special conditions that the end users might have[48]. This could be done e.g. "by tools like clouded glasses and weighted gloves"[46]. It is difficult to empathize with people that have different abilities, or live in a different personal or social situation[44], but it helps a lot to live those by using tools that allow to experience a similiar condition[44].

Especially in the field of software development, it is crucial to see the differing abilities in using technical devices: While the team developing the software has mostly very advanced and from most non-developers differing abilities, their end-users often just have basic abilities. People with minimal or almost no technical abilities are still common, as "widespread assessment of IT literacy is just beginning to emerge"[69] and according to the European Com-

mission "more than 100 million European citizens are at the risk of digital exclusion"[14]. Empathy Tools could serve as a good countermeasure to develop more human centered software by gaining insights from these extreme users. Finally, when thinking about how well something can be used one should also focus on how inappropriate design can further disable some people from using a product.

## 5.12. Paper Prototype

Paper Prototyping is a method to create low fidelity prototypes based on sketches made on paper[80] or a whiteboard. It should represent the screens and flow of the application in a fast, easy and fun way to get a first impression of your concept[61]. A very basic Paper Prototype consists of sketches of each screen, a person in charge of changing the screens and a tester that is interacting with the paper version of the interface[80]. There also exist prototyping kits and templates, which can be used to create basic User Interface (UI) elements quickly.[1] Paper Prototypes can be used in early testing or in brainstorming meetings and sessions. A big waste in software development is spending time on unneccessary features[30] - low fidelity prototypes, like Paper Prototyping, could help to find problems beforehand[80]. This method is very inexpensive in time and effort[84] - on paper, one can explore many different versions without spending too much time, and if necessary also discard any idea that might not be appropriate. Compared to digital prototypes, paper has a higher flexibility in a faster and cheaper way, because it does not need any technological knowledge[84]. One can make changes and annotations on the fly - even while testing[84] - because there are no technical barriers that would distract from the actual task. Because of this, prototyping on paper also gives the chance to non-designers to participate in the process[64, p.166] and thus might facilitate communication of ideas and taking decisions in the team. Finally, paper prototyping can be a lot of fun[64, p.166] and because of that it might strengthen the team spirit.

[1] e.g. `http://sneakpeekit.com/` or `https://marvelapp.com/static/site/downloads/devices.pdf` last accessed October 22, 2018

# 6. Evaluation of Results

In this chapter we evaluate the results based on questionnaires, interviews and observations that we conducted during the practical phase. First, we explain for each method how the participants perceived the methods. The following section contains an evaluation for the applicability of these methods in IT to find out whether further adjustments of the methods would be necessary to be able to use them in an agile context. We also name the possible fields of application mentioned by the participants. In the third section, we evaluate the toolbox by determining if the participants could easily understand the methods and if they would be able to use them without further schooling. Finally, at the end of this chapter we evaluate the core group general results to see whether empathy levels or product understanding has improved in the team. The table 6.1 lists the sections of this chapter with the research questions that they are answering.

| *Section* | *Research Question* |
|---|---|
| Method Evaluation 6.1 | RQ 1 |
| Applicability in IT 6.2 | RQ 2 |
| Toolbox Evaluation 6.3 | RQ 3 |
| General Core Group Evaluation 6.4 | H1 / H2 |

Table 6.1.: Listing the sections of this chapter with the corresponding research questions

All quotations found in this chapter are based on the responses in the questionnaires and interviews that we have conducted during our study. We applied minor adjustments to responses only in places where grammar or typing mistakes would have led to misunderstandings - besides colloquial language was also left untouched. As not all quotations were originally in English, we have translated them to the best of our ability.

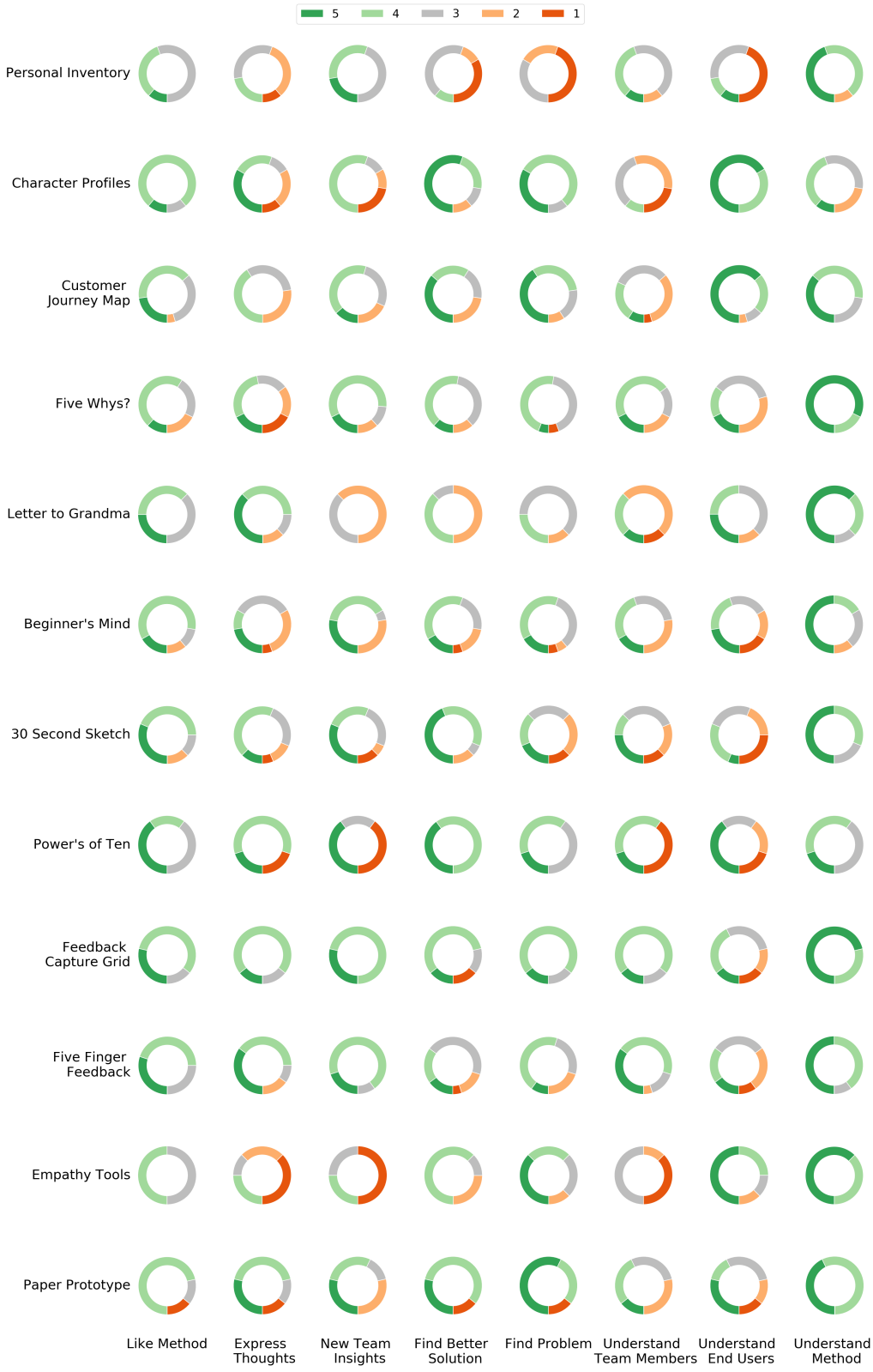



Figure 6.1.: Overview of how the participants answered the method evaluation questionnaires

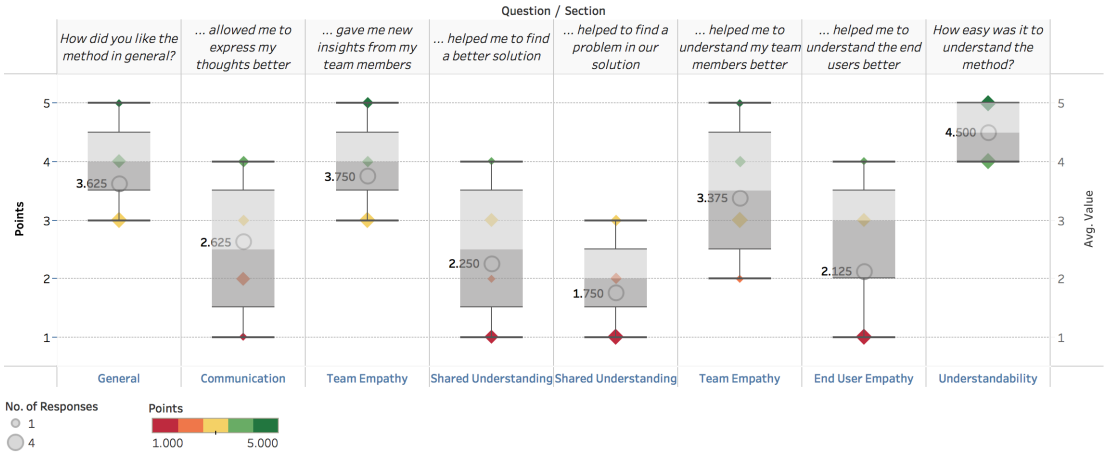



Figure 6.2.: Core Group Answers for *Personal Inventory* Method Questionnaire

## 6.1. Method Evaluation

An overview of the evaluation can be found in figure 6.1: Each circle represents how participants rated the questions (columns) of each applied method (rows). For each method the different evaluation groups are combined. Each pie chart shows the distribution of answers on the scale. As expected all the methods have different weak and strong points. No method proved to be beneficial in all aspects and the results showed that different problems should be solved by different methods.

### 6.1.1. Personal Inventory

The task for the first method presented to the core group was to upload two pictures of the working space, e.g. a photo of the desk and another of the favorite item in the office. Additionally, the team was asked to shortly present their workflow in the next meeting. On top of that, the customers of Design-IT were asked to provide photos of their working environments - unfortunately only two customers responded to our inquiry.

Compared to other methods, the Personal Inventory was on the lower end regarding how much the participants liked it (3.63) (see figure 6.2). In the group interview it was stated that the method might work better in early stages of team building. From another point of view, a colleague disagrees as he "enjoyed looking at everyone's different ways of doing their job".

Personal Inventory was less successful to understand end users (2.13) (see figure 6.2) because the core evaluation group failed to gather a sufficient amount of pictures of end users desks. During the core group interview various attendees complained about low customer partici-

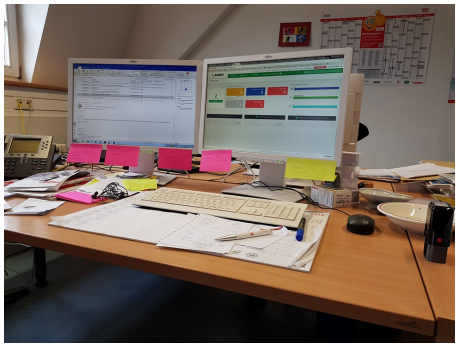


Figure 6.3.: An Example of a Customer's Desk

Figure 6.4.: An Example of a Customer's Home Office Desk

pation. In contrast, for the cases where pictures were provided, the method sustainably created empathy for end users and customers. As one member of the core group reported, after seeing the desk showed in figure 6.4, he "started feeling sorry for [the customer ... as he is] using three screens to make adjustments in WINGLET [...] and she only has one laptop screen, so this must be a pain for her, so that was eye-opening". The attendee further adds that when communicating with the customer he now thinks: "OK there are two ways to do that, and this way is going to be easier with a single monitor". Another quote from a participant emphasizes how Personal Inventory can further improve end user empathy:

> "When [a colleague] mentioned [...] that [the end users] wait like 15 seconds for the process page to load [...] when he made that comment, I remembered the picture in their office, they are sitting there, waiting [...] I imagine them in their office"
>
> Core Group Participant

After receiving the picture seen in figure 6.3, a second member of the team thinks that it "is interesting how much papers and extra helper [the customers] need to do their work". "It seems a good approach to take a step back into reality and not only [see] the [Graphical User Interface (GUI)], but [see] the GUI used by someone in a certain environment with possible limitations", believes another participant.

Nevertheless, some participants remark that "pictures are not enough, [...] a true workflow explanation should be also provided, with more details" explains a contributor. A participant would like to "[see] the customer environment, and them using our software too". His partner adds that he did not dislike "the method but its application, clients didn't gave a workflow [really]". On top of that, a co-worker believes that "this method could be useful the other way around [...] [the customer] could benefit from visualizing [the developer's] workflow, since she is being kind of difficult with estimations, [...] she's not taking into account the time spent reviewing, testing and other activities involved in the process".

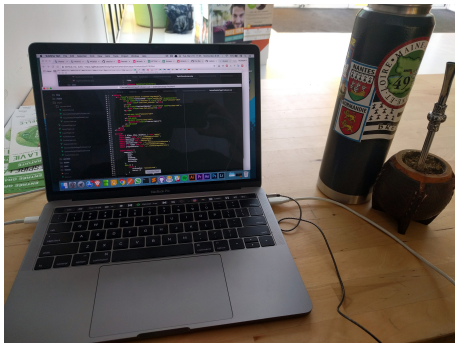


Figure 6.5.: An Example of a Remote Programmer's Desk

Figure 6.6.: An Example of a Favorite Item: *Couch*

On the contrary, the method was as expected rather successful in understanding team members (3.38) (see figure 6.2) and gathering insights (3.75) (see figure 6.2).

During the group interviews it was mentioned that showing the working environment to other team members helped to get to know how others work, especially between programmers and designers. A programmer of the team claimed that it "was interesting see designer's tools" - a teammate agrees and further points out, that Personal Inventory "gave [him] a little more empathy for the designers, because that aspect of their work [he] wasn't familiar with her process at all". But not only could developers gain empathy for designers, as an employee explained, Personal Inventory gave him "some insight about how [my team members] work, about how they manage to do things, [... he] realized that [his boss] used just a notepad". Referring to the desk of his remote colleague, which can be seen in figure 6.5, he adds that he "never thought about [...] the way [his colleague] works just with one screen, which is really hard for [him, as he needs] three". Likewise, a participant liked "[seeing] the favorite things of each teammate". Though most programmers believed that developer to developer exchange was not so interesting "as [developers] use almost the same tools and techniques". His partner further adds, that "there weren't any surprises". From another point of view, a collaborator assured that he "will ask [his teammate] about some tools that he uses". Besides, a co-worker mentioned that the method "gave insights about some remote colleagues work places". His teammate remarked that he could have used the Personal Inventory method to gain empathy towards a developer of another company, that he had to work with. They "were talking through whatsapp, it would have been very very easy for [the other developer] to send [him] a picture of his desk". He further explained that it would have been useful e.g. "because [he doesn't] know if [the other developer] uses a mac or linux or windows".

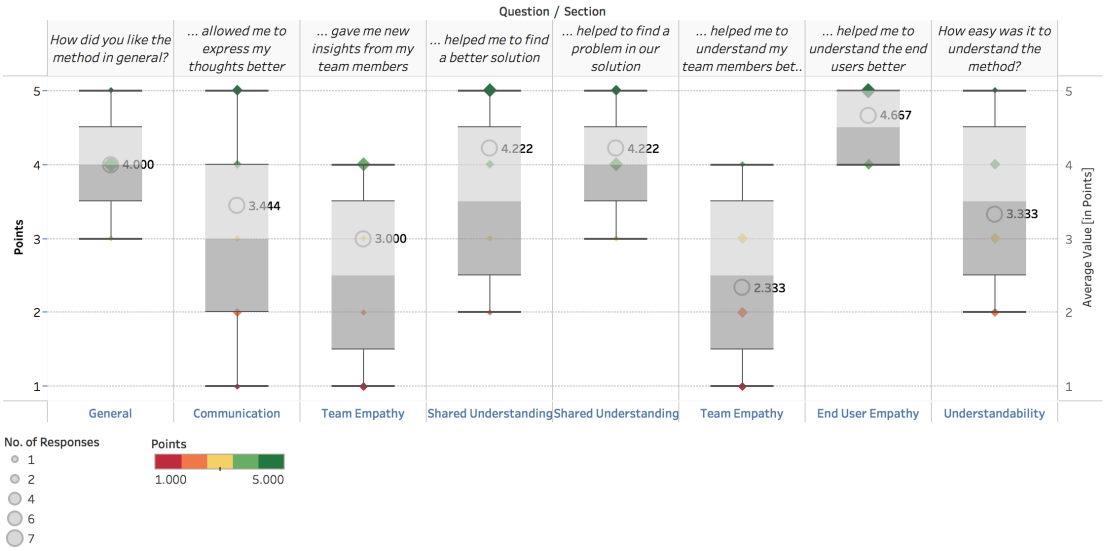



Figure 6.7.: Core Group Answers for *Character Profiles* Method Questionnaire

### 6.1.2. Character Profiles

The attendees were given Character Profiles for the projects that they were working on. These profiles were created by three members of Design-IT that have regular contact with end users and customers. They have conducted interviews and observations to collect the data for the profiles. The task for the core group was to use the Character Profiles in their daily work.

Most participants liked the method (4.00) (see figure 6.7) and it was among the top three favorites. In the group interview, a teammate assured that he will print the character profiles, because he believes that having the end user profile printed out helps. His co-worker claimed that she will use this method for the next project.

Both on the second place, Character Profiles were very suitable to find better solutions (4.22) (see figure 6.7) and to find a problem in a solution (4.22) (see figure 6.7). A participant explained that this method is beneficial "not only to empathize with the customer but also to come out with new requirements, thinking on possible solutions from the character's eyes". Another attendee further adds that "talking about an actual person triggers creativity".

The results of the method provided by the participants show that there were multiple issues found in the current solutions and in most cases also improvements were proposed. In one case a participant realized that "the way [the application is] presented now, [the persona] wouldn't be able to use it", thus he proposed to "add options for bigger fonts and spaced-out items". In another example, a group of two teammates worked on a web site where teachers can add information about their students. The participants tried to make clearer which

information has to be filled in the list. The feature was adapted in order to emphasize the students whose information is complete. Thereby the persona "can at a glance know [how many students are] still missing".

Further, they added "an icon with a tooltip to the room column, to better explain what is that for". A different participant realized that the app needs "very responsive interfaces, because [the persona] could be using any platform (mobile, web, tablet)" and "[additionally], as [the persona] is not an advanced user, [the developer] should keep the interface clear, with descriptive icons about what means each option and for what is for".

Out of all tested methods Character Profiles were most likely to give a better understanding of end users (4.67) (see figure 6.7). A participant stated that "[he or she doesn't] usually put [himself or herself] in the shoes of the end-user, so this was a great exercise". Besides, it was mentioned in the group interview that the team now feels more connected with the end users. Another effect of this method that was claimed by a teammate was that it helps to "leave aside [personal] preferences". At the same time, "it was hard to put oneself in the shoes of another person" as a collaborator reported. The team "should print this personas and have it always near, so we never forget who is the user of the things we do", summarized a member of the team. In general, an attendee claimed that Character Profiles, among other methods, helped him "to change [his] point of view, and start thinking in [a human centered] way"

On the contrary to understanding end users, this method scored the last place in terms of understandability (3.33) (see figure 6.7). Four participants had severe problems understanding the method: one participant thought that the team has "to interview people", another misunderstood the idea of fictional profiles as he asked whether the team could "get real profiles", and two further colleagues misunderstood the method, but could grasp the idea after seeing examples of other teammates. Further, they explained that it was easier to understand as soon as they started to use the Character Profiles with a specific ticket. "It was hard in the beginning to understand for what the personas stand", thus the attendees wished to "have a little example about what was expected from us". A participant that "already knew this method", explained that "for the others it wasn't really clear how to use [the Character Profiles ... some teammates] thought they need to build the personas and then do the analysis". Because of these findings, the following methods include additional sections in the method descriptions. They contain information about what the method is about and why it should be used. On top of that, we have added more detailed examples of usage to the method sheets.

### 6.1.3. Customer Journey Map

The method was applied by dividing the participants in groups of up to four people and giving them a journey to describe. Depending on the project and the time that was available for the method, a scope was set for the journey to describe - either an overview of the project or a specific feature was chosen.


Figure 6.8.: Core Group Answers for *Customer Journey Map* Method Questionnaire

At Design-IT GmbH two groups have created a journey for two entire products[1]. The time constraint was set to four hours that were split in two meetings with a duration of two hours. Besides the team had the chance to ask questions that would be answered by a representative of the product, in both cases it was the customer. The groups consisted of members familiarized with the product, but also some that were totally new. Especially the ones that did not know the product beforehand, benefited of the detailed overview about the product. But also the ones that were already familiarized with the product, said that they never saw the complete chain of actions from the side of the end user and can now empathize much better.

At *Company II*, two groups of four people were analyzing the click-flow of their main product "B-Drive". Because of the limited time that was available during the workshop, the exercise was timed to 35 minutes. That is why for this method the group decided to create the journey for a smaller feature; a secure way of requesting files over different communication channels like e-mail. Both groups were able to create a basic Customer Journey Map in that short time and found multiple problems in their current solution, that was not apparent to them beforehand. They could imagine using this method to discuss new features, but also to document and improve the existing ones.

At *Company IV* the groups were not given a specific task, instead, each group chose a topic and set the scope. Some groups had problems to do the task on their own as one participant mentioned that "[i]t is quite difficult to use the method if there is nobody in the meeting who knows how it works". Participants mentioned that "you need to scope it right" and said in the group interview, that time was an issue. Some participants also mentioned in the group interview, that they tried to use the Customer Journey Map for internal processes, but that

[1]The Customer Journey Map created by one of the groups can be found in Appendix C.1.

did not lead to any success.

The method was used in two groups of four people at *Company III* - similarly as at *Company II*, the group had 35 minutes to complete the task. The teams were given the task to map the creation of a new user in their main product "*Company III* BCS". As there were no personas available for the product, the team has created one based on a profile that was familiar to the team: a human resources employee, that already has prior experience with the product. Participants have mentioned that this method "needs a thorough picture of the persona", which made it harder to be applied. In the observation we have noticed that it was not clear to the attendees how the chosen persona would navigate through the product, as there exist multiple ways to get to the user creation feature. Thus it was stated that it was "very difficult to present abstract facts, e.g. different solutions, consistently".

Most participants of all groups stated that they felt the method was time consuming - regardless of the time that the exercise took. A partaker of *Company II* argued though that "[i]t is [quite] a bit of work, but fun to do it. The results are completely worth the effort." On the other hand, not everyone agrees with this statement, e.g. a participant of the core group has a different opinion:"it's very time consuming and tiring to do so, so it might not be worth it in most cases". In the core group we noticed that the attendees e.g. had fun drawing the images.

The participants responded to the question how much they like the method with an average of 3.8 (see figure 6.8), thus the method is rather in the middle field, but it should be emphasized that participants at *Company II* liked the method much more (4.6) than at *Company IV* (3.25). This is likely related to the fact that *Company II* already has established personas and chosed a relatively small feature, where as *Company IV* had problems scoping the journey and there was no specified feature that the customer journey should build on. Nonetheless, a participant of *Company IV* said that "[i]t was helpful to use this method because now I am more aware that the team I did the activity with doesn't really know their customer and their 'journey'. So this is a good start for conversations about why we don't know our customer and how to fix it".

The Customer Journey Map was effective to find problems in the features considered (4.0) (see figure 6.8). A member from *Company III* stated that "in the short amount of time we identified [possible] drawbacks where the person had to leave our wizard in order to calculate something". At *Company II* "[huge] issues came up that were not apparent before". Participants at *Company II* proposed to add a new category to the template, that would also consider friction points and possible solutions. That being said, participants not only found weak spots in their solutions but also were able to find better solutions (3.73) (see figure 6.8).

Most partakers could agree that this method helped them to understand end users better (4.5) (see figure 6.8), as one participant of the core group claimed that the Customer Journey Map "is a great tool to put you in the place of what the user is feeling on each stage". At *Company IV* the new perspective that the method gives was highlighted: "It forces you to think about

another perspective than your own. Actually visualizing the steps a customer goes through [...] is something that you [don't] do very often as a developer." The attendees also liked the emotional lane of the customer journey to further empathize with the end user, as reported by an employee of *Company II* "[the] best part was the emotional journey, as that can really signal red flags for UX".

As expected the method was not very effective in understanding team members better (3.0) (see figure 6.8), also the team work was perceived differently by the collaborators. At *Company III* a colleague "really liked the cooperation" and at *Company IV* the group "had interesting conversations which" one of the participants considers to be "equally important as just having the journey map". On the contrary, a contributor of the core group claimed that they were "discussing with other teammates and sometimes [it] was difficult to reach a conclusion".

Additionally some participants noted that product understanding was increased, as e.g. in a solo interview it was claimed that "it [...] forces [one] to understand the complete trip/journey of the customer".Another colleague said in the solo interview that "the journey method was really helpful to break down step by step, that one functionality of BARNY [...]" He stated that he already knew the product, "but this was like a step by step breakdown so it helped [him] understand it better".Besides, it might be also a good tool for onboarding employees on a new project, as one core group attendee said that "it is a good way to be introduced into a project [... and it] would help a lot, to have that picture to start with".

The understandability was rated in average with 4.14 (see figure 6.8), which seems to be quite high but is still below average compared to the other methods. A participant at *Company IV* reported that "[it] is quite difficult to use the method if there is nobody in the meeting who knows how it works" and a co-worker added that "[more] examples would have been helpful because [the team] relied on the [only given example] a lot". In the observation of the core group it was further noticed that the attendees had problems understanding the difference between touchpoint and action. On the other hand, at *Company II* the group had no troubles at all and the result was also very good.

### 6.1.4. Five Whys

The participants were asked to perform the Five Whys in their daily work. It was suggested to use it for ticket descriptions, customer requirements, or any problem that the attendees would face. This method was performed by *Company IV* and *Company I* besides the core group. Most participants applied the method in groups of two and did not only perform the task for their daily work but also personal topics between the participants arose.

Voting for this method was very diverse, as it was a controversial method and was also used differently in some cases. With an average of 3.5 (see figure 6.9) it is the third least liked

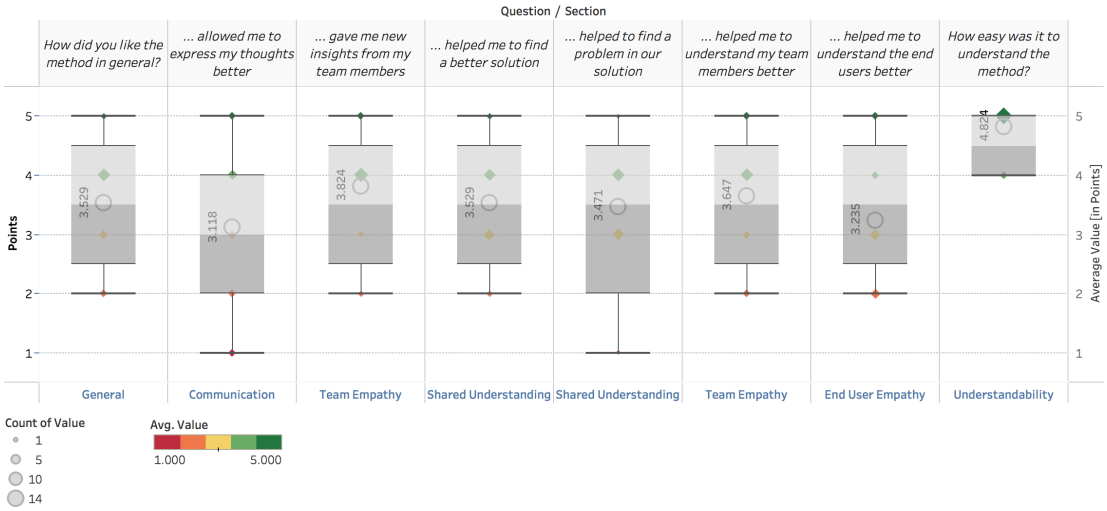



Figure 6.9.: Core Group Answers for *Five Whys* Method Questionnaire

method, but surprisingly it was entitled as "most liked" by four (50%) interviewees and "most useful" by further two (25%) in the solo interviews.

According to a core group participant the method "[is] quite easy to do and [one] can get answers that [one] wouldn't get", but in contrast to that opinion not all attendees believed that "digging to find the root cause [...] has to be a method".Still another participant of the core group is aware of the importance of root cause analysis: "we spend a lot of time doing things that could be made in a much simpler way or in a way that would be much more meaningful but we don't ask ourselves why we are doing them so we just kind of 'go with the flow'".

All groups liked that the method made one think and dig deeper. Participants at *Company IV* claim that Five Whys "forces everyone to really really think about the core problem [... and] also emphasises how a problem can be located in a completely different scope/area than initially thought". A second attendee of *Company IV* believes that the method "[makes] you think about the answers you give".

On the contrary, according to a member of the core group it "was not always evident when to [apply the method and some ...] askings stoped without any valuable insight to present."

On top of that, the attendees reported problems when handling issues that had several answers for a why question. As explained by an attendee of the core group: "Depending on the answer, you can have several paths to continue, and by following only one path, you leave out valuable information. Also, if you don't choose well in what to focus, you could end up in a path that doesn't have anything to do with the original question". Furthermore, collaborators at *Company IV* mentioned that choosing the right answer to continue is crucial, because not

all paths lead to a satisfactory answer.

Another reason for the overall bad ratings of the method, might be due to the feeling of participants that the method might be too invasive for the person that answers. It is a "license to annoy people" as stated by a teammate of *Company I*, and "[it] puts people under pressure" states a participant of *Company IV*. According to that attendee "[there] needs to be a friendly 'introduction' to the method in order to communicate consent."

Some attendees claimed that Five Whys was helpful to express their thoughts (3.12, standard deviation 1.41) (see figure 6.9), a participant of the core group reported that "it makes [one] feel that the other person needs more input and [one tries] to be more descriptive when answering", and a second participant believes that he "could describe deeply what [he] was doing". At *Company I*, a partaker agrees and adds that one could apply this method when "[trying] to deal with a communication problem among team members"

This method was rated above average for the question whether it gave new insights about team members (3.82) (see figure 6.9), it is worth to note that the ratings at *Company IV* were higher(4.25). Similarly, it also helped to understand team members better (3.65), again with higher ratings at *Company IV*(4.25). This unexpected discovery is likely related to the fact that some participants started to ask why questions about their team members. E.g. one teammate at *Company IV* asked their designers why they wear so much black or why their co-worker wears a watch. As seen in the example figure 6.10 using this method for team empathy resulted in interesting insights for the participants about their peers.

But also the core group reported that the Five Whys are beneficial for team empathy and reveal interesting insights as one contributor explained: "Why are you stressed? I asked him [...] and I didn't know that [my collegue] had the same problem [...] because he is always happy, it is like we couldn't imagine that he was getting stressed".

In addition, communication has also improved in the core group "because a lot of things that were like never talked about, started to be a common subject". Not only has communication benefited from the increased team empathy, an attendee believes that it is also "good for explaining maybe, because you are asking like a kid". On top of that, a partaker of the core group states that the method is "like a conversation in a bar, but about work".The same attendee likewise adds that "sometimes [they] do the five whys just joking, but in the end the joke ends up being useful".

In average the method did not seem to improve much the understanding of end users (3.26) (see figure 6.9), but some interviewees stated that this method could be useful for communication with the customer, especially to gather requirements. In an core group interview, a attendee claimed that Five Whys "could be useful to understand why the customer really wants something, or why they really don't like something". Furthermore, he adds that in some cases he would "forget [a constraint of the customer], because [he] didn't get the real reason".A second member of the group states that by applying the method he could identify

**Why do you wear a watch?** Out of convenience

**Why is convenience important to you?** When I am riding a bike or walking it takes longer to get my phone out to check the time than look at a watch

**Why is it important to check the time?** Because I have a busy schedule and I am trying to get somewhere on time

**Why is it important to get to places on time?** Because I made a commitment to a meeting with someone else and they expect me to be on time

**Why is it important to you to keep your commitments?** Because keeping commitments is a way of showing respect to other people and their time. I treat people the way I want to be treated and by respecting their time I hope they will respect mine

Figure 6.10.: Example usage of Five Whys at *Company IV*

the need for a comment functionality "to see why exactly, why do they want that now? and then tell them directly 'why isn't this a better solution?'".Obviously this method can only foster end user understanding, if it also applied with end users. In this study Five Whys was not primarily applied with end users, but inside the team.

Despite the mentioned problems while applying Five Whys, it was the method rated with the highest understandability (4.8) (see figure 6.9), as "[it] is very simple, easy to remember" and a "straight forward technique".

### 6.1.5. Letter to Grandma

Letter to Grandma was applied just by the core group. The group was given the task to write at least one letter and it was suggested to describe a feature that the participant is currently working on or a vision for a product.

Collaborators agreed that the method is good to simplify a project or feature, as one attendee stated in an interview that the project he is currently working on "seems like such a complicated thing [... but] to write it down in a very simple way, it really shown a light on how actually simple it is [...] and then it can be overcomplicated by details".It was also mentioned that Letter to Grandma is good for documentation, as one teammate liked "[the] thinking process and the result of it, which might be usable for documentation".

At the same time, most attendees agreed that the target level of simplicity was not ideal. One participant disliked that the "[level] grandma was [too] basic to provide empathy/insight of any kind", and another collaborator stated that it would be "[too] simple to be useful". It was


Figure 6.11.: Core Group Answers for *Letter to Grandma* Method Questionnaire

also stated that "the level of simplicity (level grandma), is something that could be adjusted at will", thus making the method more suitable for agile development.

In general, the method was perceived good for expressing thoughts (4.0) (see figure 6.11), as it "is a good exercise to really think carefully what needs to be said, when an uninformed person (most likely developer) is introduced to a new problem or feature", as explained by an attendee. It was also claimed that the method could by applied in "new projects to help categorize [ones] thoughts", and for "communicating with anyone that does not have [the] same technical profile/background."

For some participants, the understanding of end users has raised (3.63) (see figure 6.11). It would be useful to "[empathize] with someone that has very pure knowledge about things", reported a participant. Likewise, a co-worker believes that "[when] explaining something to a customer, [Letter to Grandma] could be a good way of setting the most basic points to hit and build from that the complexity of what [one wants] to say".

Most participants believed that this method was not very useful to understand team members (2.75, standard deviation 1.39) (see figure 6.11), and more, they felt that "[when] communicating with people that are already familiar with what is being described, [...] they feel patronized". Nevertheless, a participant mentions that "[working] every day on the same projects creates a bubble around team members and it is not easy to distinguish expert from common knowledge".

The method had a good understandability of 4.5 (see figure 6.11), as "[it] was like being on the first grade all over again". One participant was not sure "how much [...] depth" the letter

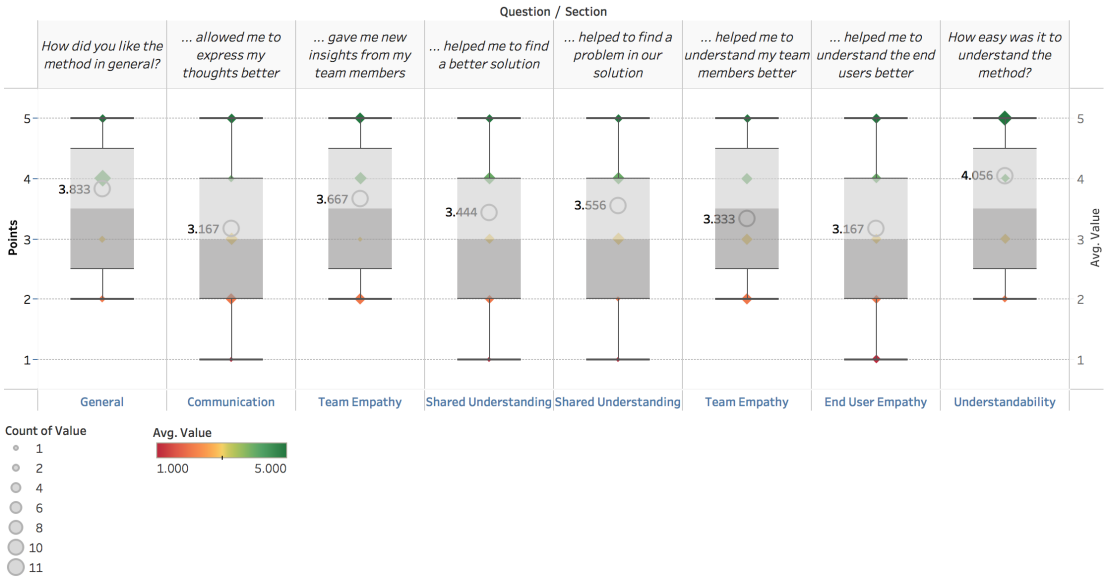



Figure 6.12.: Core Group Answers for *A Beginner's Mind* Method Questionnaire

should have though.

### 6.1.6. A Beginner's Mind

For A Beginner's Mind, the teams were given a very broad task to apply the method to a problem by trying to put oneself into a beginner's mind, but also by asking a real beginner.

At *Company I*, the method was conducted in a workshop setting in groups of two, having 15 minutes per topic. In the group interview, participants stated that it was hard to understand the method because it seemed very abstract and it was not clear in which cases it could be applied. Besides, it was mentioned that it is a known method that was repackaged and the "packaging is overkill".

At *Company II*, the method was similarly applied in teams of two colleagues having 15 minutes per topic. They have all used exclusively work related topics. Two groups were working with a whiteboard, one with post-its, and another one was just talking. The overall impression of the method was good in the group interview, one participant even stated that the results of the method "blew [his] mind".

In general the results were perceived as useful and a lot of potential solutions aroused. There have been different approaches to the method, e.g. an interesting one was: "I didn't state my assumptions in the first place, but I just explained my view and hoped that the other person

would identify some wrong assumptions, and it worked". The group agreed that the method challenges assumptions and by that, solved some problems. In one team there has been an exchange between a back- and frontend developer, which lead to knowledge sharing.

The method was liked mediocre (3.84) (see figure 6.12) compared to all methods.

As explained by a member of the core group, "[this] method is more like a way of approaching things, a way of thinking" and not a real method. Most attendees agreed that it was "hard to put [oneself] into a beginner's mind". A participant of the core group even said that it was impossible: "When you have to pretend you're a beginner on something, I don't think it works". Thus mostly the attendees agree that for this method to be useful a real beginner is needed, as one participant from *Company II* stated that A Beginner's Mind is "useful when you get the right person to question your assumptions". Another co-worker said in a group interview, that it is a "problem in finding the right person for feedback", nonetheless that attendee believes that "people with the same background but different experience (e.g. developed features, used protocols)" can serve as beginners.

The biggest benefit seen by the participants was that A Beginner's Mind "could truly bring forth new, easy and fast solutions that could get [you] unstuck from a problem" as a core group participant said. From another point of view a collaborator of the core group "could help [his team mates] with a small problem they were having". Likewise, a teammate of *Company II* claimed that the method would be useful for "any problem [one is] stuck with like product questions, implementation questions, process questions, communication questions, no matter".

The method has a rating of 3.17 (see figure 6.12) for expressing thoughts better, but still it was very useful for some participants by "[separating] problem and technical knowledge" as an employee of *Company I* said. Furthermore a co-worker said that this method teaches "to be able to explain a rather complex problem to someone who is not involved or even not an expert on the field of the problem". Finally, another attendee of *Company I* said that "[when he] can explain [something] for beginners [that means he understands] what [he is] explaining".

The method scored an average of 3.67 (see figure 6.12) for getting new insights from team members and could help to understand team members slightly better (3.33) (see figure 6.12). In that regard, one participant at *Company II* states that A Beginner's Mind "integrates other team members/stakeholders/whoever into the process and can serve as an opportunity to talk about more than just the problem".

The ratings suggest that this method did not help to understand end users a lot better (3.17) (see figure 6.12), still a core group member liked "[the] fact that [he or she understands] end user better now and [himself or herself] also". What is more important, a participant of *Company I* stated that it is good for "empathy with a beginner", which might explain why it also fosters understanding of end users.


Figure 6.13.: Core Group Answers for *30 Seconds Sketch* Method Questionnaire

Although the method has a rather high understandability as per the ratings (4.06) (see figure 6.12) some participants stated that it was not very clear how to apply this method at first, and it was also mentioned that more examples would be helpful. In accordance with a core group member, "a more clear example of what to deliver could be helpful" and a participant at *Company II* believes that the "[examples] of questions and situations are very helpful, but [he or she] still [doesn't] know if [he or she] really did the method right." In contrast, a second participant of *Company II* stated that "[the] method is not that hard, [there is] too much description".

### 6.1.7. 30 Seconds Sketch

For the 30 Seconds Sketch, the attendees were asked to create at least eight sketches. It was further proposed to use the variation of the method called "Crazy 8"[2], which combines eight sketches on one sheet of paper by folding it three times. The participants were encouraged to work on the same problem and to collaborate on it.

At *Company III*, the method was conducted by each participant for a specific topic that they could choose. Finding a topic was quite frustrating for the teammates, as it was hard for the team to imagine what should be sketched. Furthermore, the method took place in a workshop, which forced the attendants to find a topic in a short time span. The group liked that the method generated many ideas, especially it was useful to allow for ideas that are hard to

[2] `https://designsprintkit.withgoogle.com/methodology/phase3-sketch/crazy-eights` last accessed October 22, 2018

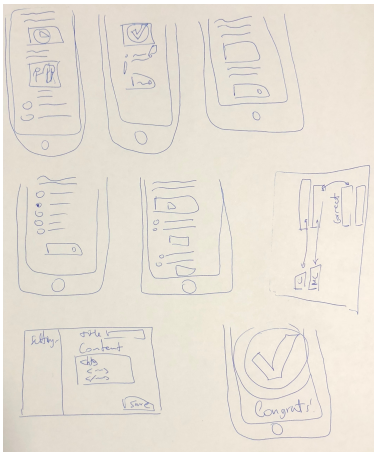



Figure 6.14.: Iterating Ideas for Creating and Attending Questionnaires on a Mobile Platform

implement or pointless. A participant of the group explained that these ideas foster creativity and eventually lead to better, realizable ideas.

At *Company IV*, the method was performed in their actual work time and in contrast to *Company III* the participants did not mention any problems finding a topic. Nevertheless, it was mentioned that "[before] starting [the participants] needed to have a clear problem statement, which helped understand the problem itself better and have a new perspective on it."

With an average of 3.94 (see figure 6.13), this method is the fifth most liked method of the toolbox. This is highlighted by an employee of *Company IV* who "liked the method very much [... as it] was really helpful for the challenge [he or she] had." Though the method description clearly states that good drawing skills are not necessary, it was mentioned by a few participants that this would be a barrier. An attendee of the core group claimed that it "is hard if you don't have good skills drawing, so the result doesn't help to much". Likewise, a participant of *Company III* agreed that he or she disliked the method because of "[his or her] drawing skills ;-)". An interviewee of the core group goes further and says that she did not like the method "because [she doesn't] draw well and then the others don't understand [the sketch]".

Furthermore, the time constraint was a controversial topic, as it was perceived as good and bad, even by the same participants. "It was a great pleasure to focus on different ideas and being restricted in terms of time did help a lot" reported an attendee of *Company III*. Besides, a participant from the core group stated that "[limiting] time [also] limits effort; and thereby reduces barriers to get started / apply [the method]". Nonetheless, for collaborators that "did the exercise for the first time [...] it was difficult [...] to change [to the next sketch] every 30 seconds". In the core group interview it was mentioned that by sticking to 30 seconds only non-sense was created, so sometimes the attendees cheated, because it was so hard to stay in the defined time span. In the same group interview, it was agreed by the attendees that the time constraint is important to get rid of details.

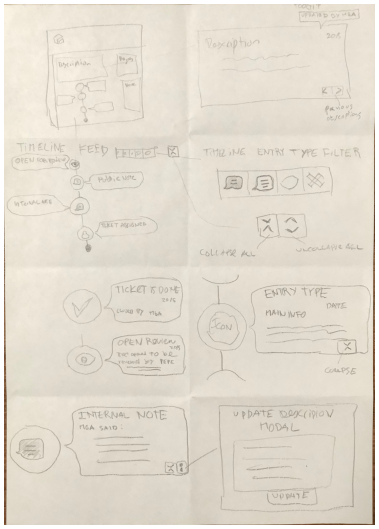



Figure 6.15.: Ideation to Implement a Feed in the Internal Ticket Management System of Design-IT

The method is rather good to get new insights from team members (3.56) (see figure 6.13) and it also slightly helped to understand team members (3.19) (see figure 6.13). This could be explained by the fact that "[everybody] gets the chance to speak up" as stated by a *Company III* employee and furthermore a core group member thinks that "its easier to communicate ones ideas and see them [drawn] makes [it] easier to [...] build upon [the idea]". On top of that in the core group interview it was claimed that the method provoked conversations between developers and designers.

Additionally, 30 Second Sketch helps to find a better solution (4.13) (see figure 6.13), as it helped to verify ideas as reported in the core group interview. Besides, finding a solution in the moment of applying the method, in the opinion of an employee of the core group the method has also long term effects: "when we had this 30 second sketch, I hadn't started with [my current project] yet, but afterwards I was constantly thinking about it - how the designs look like and if it's good that way".

What is more important is that this method fosters ideation, as mentioned by most collaborators in the questionnaires and also in the core group interview. "It is easy to apply and it makes you think about different solutions, so it is good for generating ideas" claimed a partaker of the core group. As can be seen in figure 6.14 the method was used to iterate over different ideas on how to organize creating and attending questionnaires on mobile platforms. Further, another attendee of the core group created various ideas for a feed in the internal ticket management system of Design-IT, as can be seen in figure 6.15. An employee of *Company III* explained that by "being forced to focus on the important aspects of the problem and not being able to rule ideas out simply by thinking about their feasibility [... one got] new ideas out in the wild making others think about it as well". Similarly, a core group participant

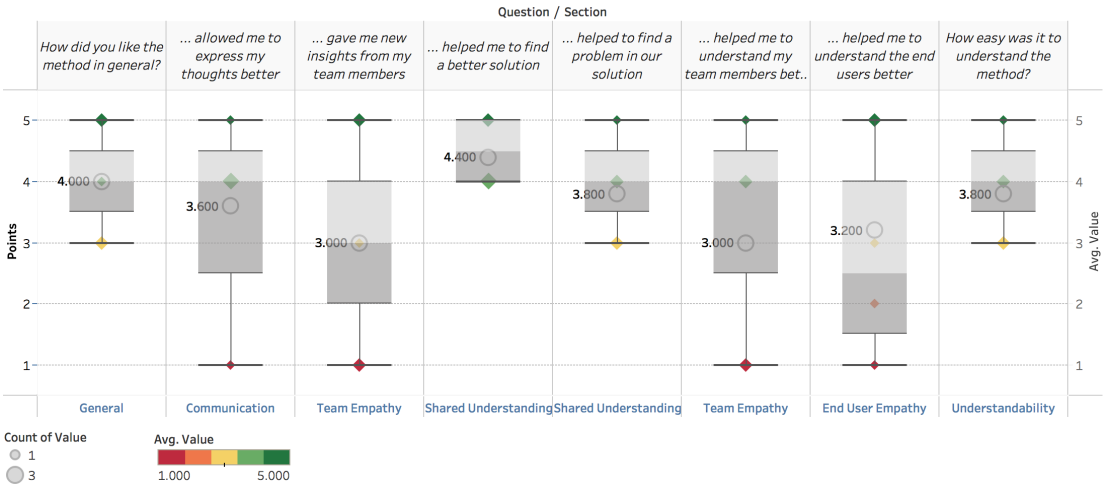



Figure 6.16.: Core Group Answers for *Powers of Ten* Method Questionnaire

thinks that "you end up doodling things up without paying too much attention to details and without noticing it, you come across some new ideas".

Participants rated this method with an average of 4.31 (see figure 6.13) in terms of understandability. In the questionnaires and group interviews there were no comments about understandability issues, but by looking at the results we have identified that some attendees did not use the method to iterate various sketches of the same screen rather they were drawing a series of sketches, which reminded a workflow of a feature. Though this is not how we imagined to use the method, it is a valid approach.

### 6.1.8. Powers of Ten

For the Powers of Ten method, participants should apply the method and then present it by describing the problem and providing the scale that was used. It was suggested to use the method for ideation, problem solving, or insight development. This method was only conducted in the core group.

With an average of 4.0 (see figure 6.16) this method is in the top three of the liked methods, in spite of that it was rarely mentioned in solo interviews. Many participants claimed that this method is "[something] that [they] already do, one way or the other" and further the attendee adds that it "was nice to have [the method] been applied formally". A second contributor of the team goes further and believes that he or she "[uses] the power of ten method every day". The familiarity of the method was further explained in a solo interview: "I think that it is very natural to think in that way, like when you are debugging a problem. [...] we just have to write what we always do. We didn't use the method to do anything different".

String Code -> QR-Code -> JavaScript Plugin -> PHP Application Programming Interface (API) -> Node Server

Figure 6.17.: *Powers of Ten* Example of a Scale

The attendees mostly liked the concept of the scale in Powers of Ten. In the group interview it became evident, that writing down the scale helped the team to see issues from a different point of view, to consider different aspects and to think about different levels of the problem. Additionally, it was mentioned that it is also possible to go back and forth on the scale, which makes it easier to tackle the complexity of an issue. A contributor further explained that "it's nice to stop and think that maybe the problem lies on a deeper level or perhaps is a more abstract and general error causing the problem". At the same time, not all members agreed and some reported that "[it] can be a bit tedious to write down the scale beforehand" or that it was "[hard] to find a personal scale for the problem [he or she] was focusing on". Likewise a teammate declared that Powers of Ten is "hard to apply consciously and in many cases won't help to solve the problem". The overall familiarity with the concept and problems with finding the scale might be the reason why the method did not receive much attention in the final solo interviews in spite of its good ratings in the questionnaires.

As can be seen in the ratings, the method was considered as the most helpful to find a better solution (4.4) (see figure 6.16), likewise, it also helped to find problems in existing solutions (3.8) (see figure 6.16). From the results of this method that were provided by the core group, it is evident that using Powers of Ten lead to satisfactory solutions of the tackled problems. One participant used the method to tackle a UX problem - she wanted to find out how much space had to be taken by a video or image in a specific screen layout, the initial design can be seen in figure 6.18. To solve the problem, the attendee used an existing design proposal of that screen and was changing perspectives by zooming between 80% and 200% of the image size. As can be seen in figure 6.19, after the exercise, the screen layout changed drastically by extending the size of the video to 80% of the screen layout, so it is big enough for the user. The remaining 20% are still enough to incorporate some text with the necessary details. Besides design issues, the method was also used for resolving technical problems, mostly by developers. A second teammate has used Powers of Ten to resolve an issue in an application that would not do the desired action when scanning a Quick Response (QR)-Code. He was looking for the issue first in the backend, that was in charge to register a student when his or her QR-Code was scanned. By using the Powers of Ten with the scale shown in figure 6.17, he finally managed to find the issue on the other side of the chain: the function that was responsible to provide the data to generate the QR-Code was returning 'undefined' instead of the proper code for the particular student.

A partaker of the group summarized that "when [he or she works] in something, then [he or she tries] to look the problem at different distances to find a better understanding". A different participant added that he liked about the method that it "allows you to think in each part of the problem from different perspectives, so the final result is a very good understanding of the problem".

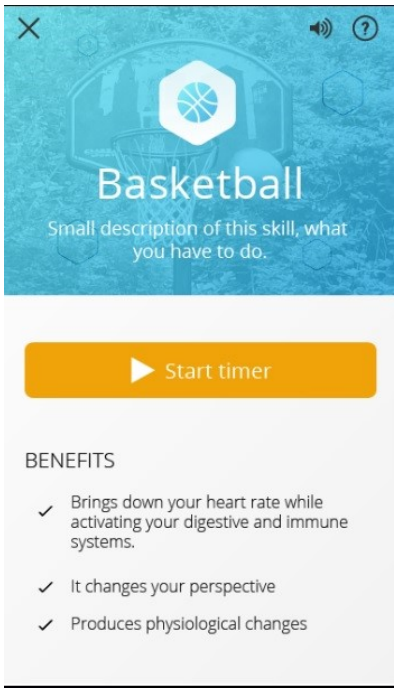



Figure 6.18.: *Powers of Ten* Example: Screen Layout Before Method Application

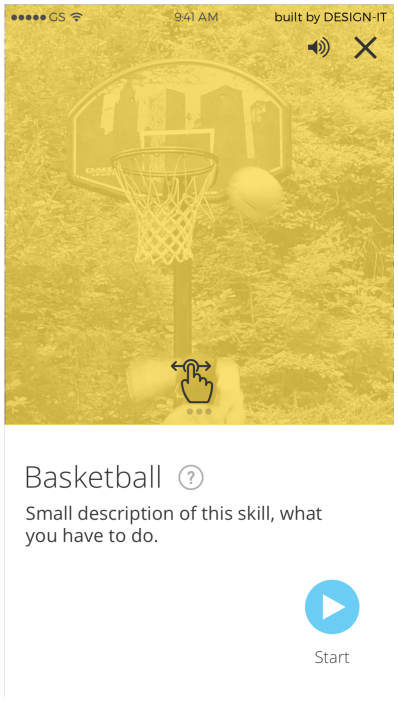



Figure 6.19.: *Powers of Ten* Example: Screen Layout After Method Application


Figure 6.20.: Core Group Answers for *Feedback Capture Grid* Method Questionnaire

Some participants had difficulties in understanding this method, or at least in how to apply it. This is emphasized by the relatively low voting on the understandability of the method (3.8) (see figure 6.16), which scored the penultimate place among all methods. Examples provided by their peers were helpful as an attendee stated that "[he or she] wasn't totally sure about how to apply this method until [he or she] saw some examples". However two interviewees stated in the solo interviews that the method is very abstract and they are still unsure when and how to apply this method, one employee said e.g. that "[the team was] just making something up". Nevertheless, as already stated in the beginning of this section, many participants were already familiar with this technique and appreciated to get to know a formal method for it.

### 6.1.9. Feedback Capture Grid

To apply this method, the participants were asked to collect feedback as a moderator and another time to provide feedback. Additionally, a sheet with general feedback rules, that can be found in Appendix B.13, was handed in.

Being the most liked method among all (4.14) (see figure 6.20), it was entitled by a attendee as a "[great] method, [that is] super useful". The contributor further added that he or she "will definitely be interested in applying it on the company". In the group interview, participants agreed that the method forces to think in all directions and that enforcing different types of feedback is beneficial. Furthermore, it was stated that the method gives a fresh look, similar to A Beginner's Mind. One teammate mentioned that a moderator is indispensable for a


Figure 6.21.: A Remote Feedback Capture Grid Session to Evaluate a New Feature of the Internal Ticket Management System of Design-IT

bigger group, to avoid repetition of topics.

> [Feedback Capture Grid] forces you to see everything from many angles, whether you like or dislike a functionality, you end up criticizing it even when you like it or praising it even when you hate it, which is something good I believe, so we don't settle for anything and we end up questioning everything all the time, which helps us reach better solutions.
>
> Core Group Participant

An attendee reported that the method was not very useful, because before collecting feedback the application had to be explained to the feedback givers, but "it was kind of hard for the rest [to] understand the structure". Nonetheless, that participant later put that statement into perspective: "maybe there was some issue with the screen design, because is still a bit difficult to use".

Not only was this method the most liked, but it also scored best for getting insights from team members (4.29) (see figure 6.20), and is on the second place to understand team members better (4.00) (see figure 6.20). In spite of the phenomenal ratings, we could not find any further insights in the questionnaires or interviews on increased team empathy. One possible reason might be that by facilitating feedback giving and reducing barriers by explicitly asking for negative and positive feedback, the attendees gained new insights of their co-workers. Thus by saying things, that they would not have said before, their teammates might understand them better. As one participant claimed the method "gave [him or her] some feedback from [his or her] team mates regarding attrack, and [he or she] could help them too with their [projects]".

Furthermore, the Feedback Capture Grid would help to find better solutions (3.57) (see figure 6.20) in some cases, as the attendees "end up questioning everything all the time, which helps [them to] reach better solutions". Secondly, the method was good for generating ideas according to the group interview. This statement was reinforced by another participant that explained that "normally you would move on after thinking about [the feedback topic], but seeing that u haven't posted an idea or a criticism yet could force the process". As seen in figure 6.21, two attendees have built upon a question of another, by providing ideas on how to improve the time logging experience.

On top of that, the method scored an average of 4.00 (see figure 6.20) in respect to finding problems in solutions. The process was facilitated by the fact that the method "enforces four types of feedback [so it] makes you think harder". The participant further explained that, "[otherwise] you might skip some aspects of the problem, just because you like or [dislike] it". Using the Feedback Capture Grid e.g. revealed issues on the search functionality of the internal ticket system of Design-IT and raised questions about UI elements those functionality was not clear to the participants (see figure 6.21). On the other hand, as already mentioned in this section, using the method further showed up usability problems of an application, as the moderator had problems explaining the program to the feedback givers.

The method was the second easiest to understand (4.71) (see figure 6.20), as per the average ratings of the method evaluation questionnaire.

### 6.1.10. Five Finger Feedback

Similarly as with the last method described in section 6.1.9, participants were asked to both collect and give feedback using the Five Finger Feedback method.

The method has been applied at *Company I* in a workshop setting - at first pairwise, and afterwards in a big group of six people. The participants had problems organizing the feedback experience, because the topic that was chosen for feedback was very broad. Additionally, the group got distracted rather easily. Some members had fear of giving direct feedback, as a participant explained in the group interview: "I notice we're a little shy". Besides, we could observe during the workshop that some collaborators had difficulties accepting the statements of their peers, if they did not coincide. An interesting fact is that this group has decided to use the Five Finger Feedback method to give us feedback about the method in the group interview.

*Company II* has applied the method in a similar setting, - first in groups of two and then in a bigger group of eight. The attendees already had experience with other feedback methods like Feedback Capture Grid and I like / I wish, which they use frequently. The feedback round in the big group was about "how is working at [*Company II*]" - similarly as at *Company I* the team chose a comprehensive feedback topic. It was criticized by the participants that the


Figure 6.22.: Core Group Answers for *Five Finger Feedback* Method Questionnaire

method is not suitable for a broad question - later it was agreed that wide questions are not good for feedback in general.

At *Company III*, the method has been only applied in groups of two people, but in two rounds by rotating the feedback giver and receiver. One group was giving feedback on a recent presentation that had been held by the feedback receiver and later that attendee claimed that the exercise "was not only about practicing the technique, but it resulted in really interesting feedback".

The Five Finger Feedback was the second most liked method (4.05) (see figure 6.22) voted by the participants. "It's easy, does not need material, [and is] easy to remember" according to a teammate of *Company III*. In the opinion of an employee at *Company II*, the method is "easy to remember" as one "always [has his or her] hand as the reminder". In general attendees liked that this method considered different angles and thus was good for structuring feedback. To give an example, a collaborator of *Company I* claimed that the method is "an easy 'rule' to not think about good and bad stuff, but also consider new perspectives".

In the group interview, almost all participants of the core group felt that this method is very similar to the Feedback Capture Grid. Some preferred this method, as it is not so detailed and thus "great to check main issues" as a participant stated. Others rather liked the feedback grid because they believed that this method has too many categories for feedback.

The five different categories lead to mixed responses: Some liked the variety it asked, and some disliked the complexity it adds. While some attendees preferred less categories, a member of *Company II* has a different opinion in this regard: "Maybe one could add the palm of

the hand as 6. point for the solution space". Some participants say that they liked a specific category, but disagreed with some other categories, e.g. one employee of *Company II* stated that he liked the additional category for what "came too short" - *the little finger* - but couldn't really agree that there is a benefit for having the additional index - *what was liked* - and ring - *what was emotionally interesting* - finger categories. On the other hand, another co-worker liked the emotional base that the ring finger gives to the feedback.

With an average of 3.95 (see figure 6.22) the method allowed to better express the thoughts of the participants. "It gives the feedback giver the opportunity to express not only good but also critical feedback and prepares the feedback receiver for it" claimed an employee of *Company III*.

Furthermore the method has revealed new insights about team members (4.10) (see figure 6.22) and also helped the team to understand their team members better (4.10) (see figure 6.22). The method has improved communication between remote teams in the core group, because the group "had to communicate between everyone" stated a collaborator. Further, he added that he "[feels] like [he doesn't] ever talk to [her remote co-worker], so [...] having a small conversation sometimes, it makes it easier for the next time". Another member of the core group reported that "when [her teammates] gave feedback about [her] document for START this also improved communication in the START project." She further added that "also in general [the team] now [knows] how to give structured feedback".

The method was rated 4.40 (see figure 6.22) for understandability, which is in the middle range compared to the other methods. Mostly it was criticized that it was not clear for every finger what type of feedback was expected. For "a couple of fingers [... he or she] wasn't sure what to say" as explained by a core group interviewee. Mostly participants had difficulties with the index - *something to point out* - and the ring finger - *what was emotionally interesting*. In agreement with an employee of *Company I* it is "difficult to find difference between index finger [- *something to point out* -] and middle [- *what was not so good* -] finger" - *what was not so good*. "Also the ring [- *what was emotionally interesting* -] finger and the thumb [- *what was good* -] seem to have some overlap" adds a collaborator of *Company II*. Another participant of *Company II* summarizes the problem: "the definitions are a little imprecise and hard to keep track of so you might end up falling back to 3 (like, dislike, noticing)".

### 6.1.11. Empathy Tools

The team at Design-IT was given the task to use Empathy Tools for five different inabilities of their choice.

With an average of 3.50 (see figure 6.23) Empathy Tools were the second least liked method in the toolbox. In the group interview, participants perceived the method as interesting as one is "moving from imagining to experimenting". It was further mentioned that the method is


Figure 6.23.: Core Group Answers for *Empathy Tools* Method Questionnaire

not only useful for designers, but can be applied by e.g. programmers, too. At the same time, the collaborators felt that the method is not useful for small target groups, and that without knowing the target group very well it is hard to know what to test. In a solo interview, an attendee explained why he did not perceive the method as very useful: "we don't take any actions regarding impair people, we don't even fill the 'alt's in the images [...] we should start working on more accessible systems first".

The method helped to find a better solution (3.38) (see figure 6.23) and to find a problem in a solution (3.88) (see figure 6.23). Compared to the other methods both rankings are mediocre. Nonetheless participants claimed that the method is a "[good] outlet for creative impulses" and "[makes] you think outside the box, [as] you need to be creative with your responses". In particular, a contributor assures that the method "forces you to use the product not in the way you usually do, so you end up discovering new issues".

> That was like an eye-opener, maybe [the method] is a very specific one, but on the global scale it makes you think outside of the box - to actually think of the customer, because sometimes maybe we don't?
>
> Core Group Participant

In contrast, this method was among the top three methods in understanding end users better (4.13) (see figure 6.23). In general, the attendees reported in the group interview that the method is good to empathize with the end user. An attendee especially liked to "be able to make small changes that mean a lot to others" and a second member of the team added that the method "[helps you to] understand better the end user and how would they feel by using


Figure 6.24.: Core Group Answers for *Paper Prototype* Method Questionnaire

the app". Furthermore a participant claimed that Empathy Tools "[give] insights that we normally never think of". Likewise, a teammate assured that he or she "started being aware of some the problems that many people experience" and besides, reported that he or she "didn't know that so many people are color blind or has some type [of] dyslexia". Empathy Tools are helpful to start thinking about how end users are different from the creators of a software claimed a participant in the group interview. Someone else agreed as "customers don't have this tech-knowledge, that [developers] have" he further added that Empathy Tools "in particular made [him] realize that". What is more important, the attendees started wondering whether their "customers are having problems because of this".

With an average understandability of 4.63 (see figure 6.23) another question of this method was rated top three among all methods. We did not notice that the participants would have any issues while applying this method.

### 6.1.12. Paper Prototype

The Paper Prototype was conducted in two groups, one with three and the other with five participants - they had two hours for the exercise. The groups were asked to create paper prototypes together and then test them with people outside of the group. Further, the participants were asked to iterate their prototypes at least two times and conduct a test after each iteration. Both groups were given the same topic: Create a platform for group trip agencies to be able to request and book hotel stays easily using an interface. Two participants of one group can be seen during the creation of the prototype in figure 6.28.

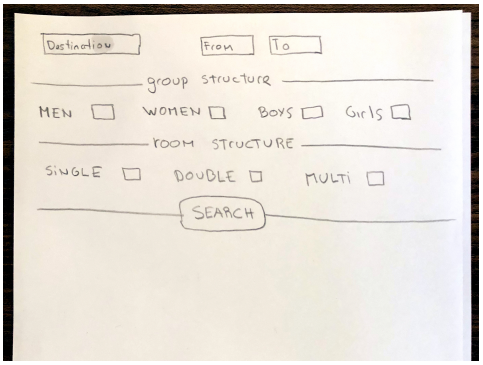



Figure 6.25.: Home Screen Layout of an Application to Book a Hotel for Grouptrips - First Iteration

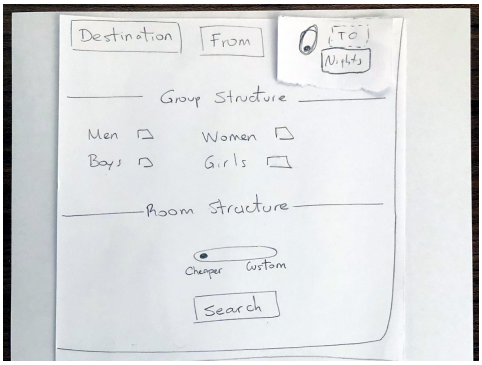



Figure 6.26.: Home Screen Layout of an Application to Book a Hotel for Grouptrips - Second Iteration

This method was not only the last method conducted by the core group, it was also the least liked (3.43) (see figure 6.24). In spite of that participants had fun applying the method and clearly enjoyed the group work experience. It was besides mentioned that it was great "how fast you can test stuff" and that "the iterations can be done super fast". At the same time, this exercise was considered as too intense to be done during day to day work. Further, a participant agreed and added that he did not like the time restriction, because he felt that the group needed more time to complete the prototypes and thus concluded that creating paper prototypes might not be the most efficient way. In the group interview, was additionally acknowledged that iterating the prototypes in digital wireframes would be much more complicated, and paper enabled the team to be very flexible. On top of that, another collaborator declared that using paper is better than using a whiteboard, because it is possible to touch it and in general it would be interesting to see the UI on paper.

On the other hand, an attendee claimed that "in order to be able to transmit how the screens and interactions will work, you need to be able to draw basic components, kind of aligned, and with the correct scale". According to another peer, "[it] was difficult for [him or her], because [...] you need some designing skills". Besides not being able to sketch, a participant "really [doesn't] like to draw".

Paper Prototypes were useful to express the participants' thoughts better (3.71) (see figure 6.24). It "[is] easier to interact with non designers" claimed a participant, and another one agreed and added that he or she liked "the interaction within the group and to understand other members ideas".

The method was also suitable to get better solutions (3.86) (see figure 6.24). In the group interview the attendees declared that each iteration of the prototype brought improvements and that they could create better solutions by working together. A teammate further ex-

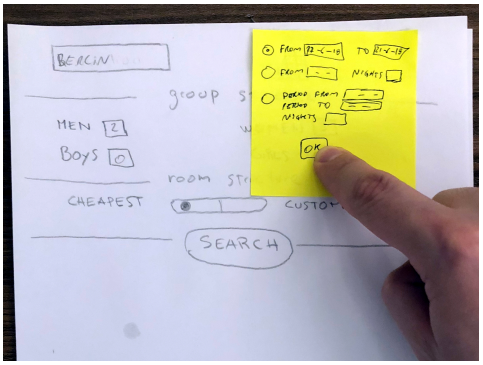



Figure 6.27.: Home Screen Layout of an Application to Book a Hotel for Grouptrips - Third Iteration

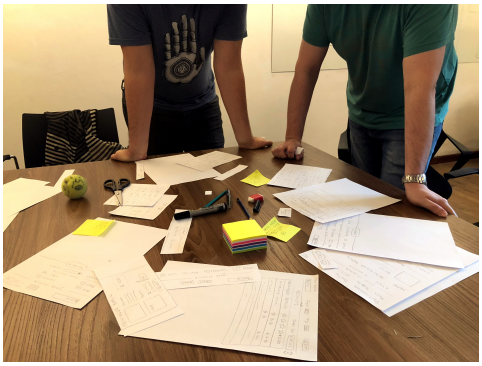


Figure 6.28.: Participants of the Core Group Creating the Paper Prototype

plained "[that] after each iteration you could [...] find something that could be not so nice and it can be changed discussing with the people". "[The] paper prototype could be something that would be nice to start doing - [...] a quick way to iterate and to present to somebody else".

The participants found this method also helpful to find a problem in their solution (4.14) (see figure 6.24). They reported in the group interviews that testing on paper has showed caveats in the planned UI. A participant explained in a solo interview that the method "was enlightening in the sense that it made [him or her] realize how a design decision that seemed tremendously trivial and obvious then proved to be inadequate when testing it on paper".

As can be seen in the figures 6.25, 6.26 and 6.27, three iterations of the same screen led to various improvements:

- Inputs for the number of participants have been vertically aligned, so that it is clear that not the amount of all male or female participants has to be filled in the fields *men* and *women*, but also *boys* and *girls*
- The room structure inputs were simplified by a slider, that would suggest the cheapest option by default
- The second date input has been replaced by a slider that makes it possible to enter the amount of nights instead of the end date
- In the third iteration the date selection has been further refined: a pop up modal gives the possibility to enter the date in three different manners

The method was not particularly successful for understanding end users (3.29) (see figure 6.24), nevertheless, a participant claims that "you start to think about the users". Further, another member believes that Paper Prototypes are a "[good] way to test and validate with the customer while keeping his expectations in check". "The user doesn't think that the paper is the final version" adds another participant and concludes that because of this it "[is] easier to interact with non designers".

Similarly, the method was slightly useful to understand team members (3.29) (see figure 6.24) and a bit more suitable to get insights from team members (3.57) (see figure 6.24). Being able to work in something different than normal work adds team spirit and allows to get new insights from team members by observing them during the exercise, as explained by an employee of Design-IT: "we never do physical tasks [...] but we had to cut these pieces of paper with scissors, [...] this kind of tasks [...] gives you a better understanding of how [team members] think, and of how perfectionist someone is". Likewise another participant assured that "[it] was fun to do something different than coding", besides, he or she liked "the interaction within the group and to understand other members ideas".

> He was like 'ahh where are the scissors?' and he is already up going to the place where the scissors are. And I think it is subtle, but it says something about him, I think. That he is always willing to do things and to take the first step, [...] it is nice to know how people think, to know what they think is important. If they care about the end result, how much they care about detail [...] their willingness to do stuff.
>
> Core Group Participant

## 6.2. Applicability in IT

Most of the methods presented in the toolbox are just slightly or not adjusted at all. We mostly tried to choose specific examples that lead the method in a specific direction. The *WHY* section of the description, which was introduced after the second method as described in 6.1.2, was targeted specifically towards IT problems. But as some methods were general communication, ideation or feedback methods, they do not need to be specifically linked with IT. At the same time, it seemed to be beneficial to have examples in that direction to ease the flow of imagination. Nevertheless, participants wished to have more examples and especially more concrete examples for some methods like Character Profiles, A Beginner's Mind or Powers of Ten.

Three of the twelve methods were adjusted. Primarily, the method Personal Inventory was adjusted to not only consider the personal inventory of an end user, but also of the own team. The definition of *inventory* has been slightly changed in order to show the working environment: Agile team members could show e.g. their desk and favorite object in the office, and

end users should demonstrate the space where they use the software - in most cases this is also their desk, but this could be also a small tablet in an industry hall used for machine maintenance. Additionally, the usual workflow of the teammates should be shown in front of the team, so others can grasp how their colleagues work. Character Profiles was not really changed, but just specifically restricted: It is not about creating the personas, which should be done with careful research beforehand, but about using them in daily agile work. The last adjusted method, Empathy Tools, was adjusted to the specific circumstances of software development by considering not only cognitive or sensory disabilities, but also technical disabilities and different device types.

*Personal Inventory*

The method Personal Inventory could be used "[in] the first stage of a project, to better understand the user and their needs, and build empathy", declared a member of the core group. Further, a second collaborator added that it could be applied for "project onboarding". Personal Inventory is likewise useful "[when] you want to empathize and make the people feel comfortable about the others users involved in the company" declared a participant. A second contributor explains that he is using this method regularly with his teammate since it was introduced, who "sends [him] a lot of pictures sometimes, about what [his colleague] is working on, or [...] shows [him] that he is not using another screen and [...] is sitting somewhere else". On top of that, a co-worker believes that this method is especially useful "[when] you have no much contact between your teammates and your customers", thus being helpful for remote teams.

Additionally, participants proposed to use this method to "[keep] track of used software and plugins in [their] private workspace". Another attendee further explains: "If I feel a co-worker maybe is having some issues with something, maybe we can compare how we work".

As already explained in section 6.1.1, "this method could be useful the other way around" to demonstrate the workflow of the agile team to the customer. This might make the customer more conscious about the different steps needed in software development and thus align expectations about time estimates for requested features.

At the same time, participants saw some limitations in the method. Essentially the success of the method depends on customer and end user participation and in most projects "the customer [didn't] send [their] desktops". "When there is no too much confidence with the customer, maybe is not nice for them to send images of their environment" explains a member of the team. Finally, an attendee declares that this method could not be applied "in bigger companies where firstly pictures are not allowed and software is prescribed".

On top of that it was hard for the participants to imagine how to apply this method more regularly. Despite this fact, some examples included to use it during the onboarding process of new team members or clients or to apply it in a monthly schedule by presenting just one team member per iteration, but more in depth. It could be also a useful intent of the team to

regularly obtain pictures from the working environment of new end users, e.g. in the case of the group trip booking application it could be possible to obtain pictures of the environment where the end users make use of the app.

*Character Profiles*

Character Profiles could be used for "finding ways to improve the system [... or] checking on current system to find pain points", but also "[when] you have to develop a new functionality [... or] when you have to come up with new ideas" the method improves "[thinking] and deciding on new functionalities" according to employees of Design-IT. Further, a participant explains that Character Profiles might be used to "get some insights about the customer that could increase his/her perception of satisfaction". On the other hand, some participants reported limitations of this method when using it with support tickets that "[focus] in fixing the very defined issue [to] provide its expected behavior". The participant further explained that "the bugs didn't gave much room to design think". In contrast, the method could be applied to "[find] solutions when a support tickets required a UI or process change" claimed another teammate.

*Customer Journey Map*

The Customer Journey Map is widely applicable, as a participant at *Company IV* stated that it is "applicable for each user story you're working on as a dev".

Most collaborators agreed that the method is useful to tackle UX issues or to find those in the first place. A participant of *Company II* believes that the method would be suitable for "[improving the] UX of an application" or "when figuring out what the user interaction flow of a use case should be" added a participant. "When trying to identify some weaknesses of [an] application and finding out new opportunities of user satisfaction" a Customer Journey Map could be created summarized a member of the core group - a second member of the team added that it is further useful to "find out missing stages".

Besides "[iterating] on old" screens the method might be effective "[when] adding new features" assured an employee of *Company II*. A member of the core group believes that the method should be applied for "every new project". In the same regard "[specifying] stories or features that are about to be implemented soon" could benefit from Customer Journey Maps. At *Company IV* a contributor reported that it is effective "[when] talking about bigger features as it makes it very clear in which stage a customer uses the feature".

An interesting comment as stated by an employee of *Company II* was that this method could facilitate "[documenting] acceptance tests of a system" and thus serving yet another purpose.

In essence, the Customer Journey Map can be used for "[everything] during the lifecycle of a project", declared a member of the core group. Another participant summarized the uses of

the method: "in the design phase, looking for posible updates to the system, after introducing a new update to the system that might attract a different user profile to use it [...] and in any other phase really as it shows how to improve the customer experience at any time".

Despite its broad applicability there are also some limitations reported by the participants for this method.

As expected, the method was not perceived as useful for technical challenges that do not require user interaction. A member of the core group stated that "[for] functionalities in which there is no interaction between the customer and the software [... there are] no touch points". At *Company II* participants explained that the method "is very end user focused" and in "refactorings, technical tasks, [etc. there is] no direct user interaction".

On top of that, many participants thought that there are "[little] use cases because it takes too much time, in those cases it might not help but would take a lot of mental energy and time" as explained by a core group member. "In agile, this method would create too much work and output for describing or inventing features that will not be implemented for a long time [... as the] features will change anyways, so the work is lost once they do" declared an employee of *Company II*. A collaborator of the core group further believes that the method "requires the customer's investment".

*Five Whys*

According to some participants of the core group, Five Whys would be a useful method for "gathering new requirements from customers" and for a "better understanding of why the customer [requested] some change". A second member of the team further explained that applying the method "would be awesome when taking customers requirements, particularly if his/her needs are expressed as tied to a certain implementation (e.g. I would like a user list here or a button there)". The same attendee further adds that "when having a lot of uncertainty regarding the business logic/system's domain" the method could be helpful.

At *Company IV* it was stated that this method could be useful for "[bug] investigation" to "[dig] deeper and deeper to find the root cause of a bug" as a participant explained. The method would be useful to "[find] the core of a problem [...] in software development or other subjects" added an attendee at *Company I*. Further, a core group interviewee stated that "if [he is] stuck Five Whys" would help him. Similarly, at *Company I* a participant said that he can utilize the method "to understand a problem better".

Secondly, Five Whys are not only useful with problems but also it could be used for process optimization. As a core group member stated the team "[spends] a lot of time doing things that could be made in a much simpler way or in a way that would be much more meaningful but [the team doesn't] ask [themselves] why [they are] doing them".

At the same time, the method is also suitable "to deal with a communication problem among

team members / coworkers" believes a contributor at *Company I*. Another member of the core group further explains that "a root cause that you may not be aware its causing issues for other teammates, so sharing the exercises within the organization and visualizing the points of contact of the application of this method by everyone could bring forth more insights and team empathy". As already mentioned in section 6.1.4, the method was also used by some participants for team insights, e.g. in the core group "Five Whys was something that gave a lot of information about stress, yes that was a big one for [the team]".

On contrary, the teams had problems using the method for complex issues that have multiple paths. As a member of *Company IV* stated: "[The] first answer already led to an overwhelming amount of unspecified answers" and a core group member further explained that the method is not suitable for "[bigger] scale problems and challenges, whenever the topic is too complex for a 'one-directional' path of questioning". Further, at *Company IV* a participant believes that "you need trust for this method because the other part might feel easily interrogated". A core group member further explained that the method is not suitable "[when] the person answering is not in the mood for so many questions or when the relationship with the other person is not so good".

"When there it is a very specific issue or when you know you are going to fall in an obvious answer", Five Whys would not be helpful believes a core group member. A participant further declares that "[some] bugs have no [further] root cause than, sloppiness or having not taking into account/being aware of some aspect of the business".

Finally, a core group member summarizes that "[in] general you can always apply this method. In the worst case scenario, you will get to a place you already know, or to an answer that is obvious. But since it doesn't take much time, there's no problem".

*Letter to Grandma*

"Introducing bigger new features" could be facilitated by using Letter to Grandma states an employee of Design-IT. A participant further explains that the method is useful on "new projects to help categorize [his or her] thoughts". Furthermore, it is useful "[whenever the teammate doesn't] know what the project is about". Likewise, participants think that Letter to Grandma is utile for "[introducing] new team members to a project" or "when enters a new teammate in the company", thus facilitating onboarding. Another participant further adds that "when doing some kind of documentation about how a system works" Letter to Grandma could be helpful. At the same time, the method could be useful for communication. "When explaining something to a customer this could be a good way of setting the most basic points to hit and build from that the complexity of what you want to say" or "[in] general, when communicating with customers or team members, for explaining functionalities" the method would be helpful as two collaborators report. Another participant explains that Letter to Grandma could be used for "communicating with anyone that does not have my same technical profile/background".

From another point of view, the method also has some limitations - firstly the target level of simplicity was criticized because "[when] communicating with people that are already familiar with what is being described [...] they feel patronized" as a collaborator explained. Another colleague believes that "[to] get what you actually want to transmit to a customer, it's just too basic terms". In the same regard, Letter to Grandma was not perceived practical to describe "a specific or very basic functionality". Another participant "wouldn't use this method when there have been other design thinking methods applied that made it super clear what the purpose of the system is". Besides, a participant declares that the method "is too time intense".

*A Beginner's Mind*

Some participants have mentioned in the group interviews that they have already known the method before, but they appreciated that it was formalized so it can be used more consciously now. An attendee of the core group states that for "new functionality, this method is great", further a participant at *Company II* adds that it would be useful when "designing a new feature". "Choosing new frameworks/new technologies after researching them beforehand" might also be a possible use case assures a co-worker. Besides, it could be helpful "to test prototypes or use cases" believes a core group member. Another use case might be "[understanding] and solving complex/mathematical/algorithmic problems" declares an employee at *Company I*, likewise "[any] complex task that involves UI/UX or general architecture" could be solved using A Beginner's Mind believes a participant of *Company II*. A member of the core group goes further and claims that the method was used "mostly everyday in [his or her] work". A participant explains that "[he or she tries] to use it for every situation possible, and now that it has been formalized [... he or she tries] to use this even more". At *Company II*, an attendee believes that A Beginner's Mind is suitable for "any problem [he or she is] stuck with". Similarly, many participants believe that this method is a perfect fit "[when] stuck with a problem" as stated by an employee of *Company I*.

What is more important, some participants mentioned that they "cannot think of any" limitation for that method, as reported by a core group member. Nevertheless, collaborators need beginners with a "suitable experience level" claimed an employee of *Company II*. A participant of the core group further explains that "[when] you have to pretend you're a beginner on something, [he doesn't] think it works". Besides, it was mentioned at *Company II* that "[discussing] highly technical problems with this method may be impossible with true beginners e.g. non technical people".

*30 Second Sketch*

The 30 Second Sketch method was primarily considered for ideation, e.g. "when searching an 'out side of the box' idea", as reported by an attendee at *Company IV* or "[when] trying to find another solution for some functionality already developed" believes a core group member. At *Company III*, a collaborator further adds that this method is useful "[when] being in need of a new perspective on the problem". The method is also useful when "designing [a]

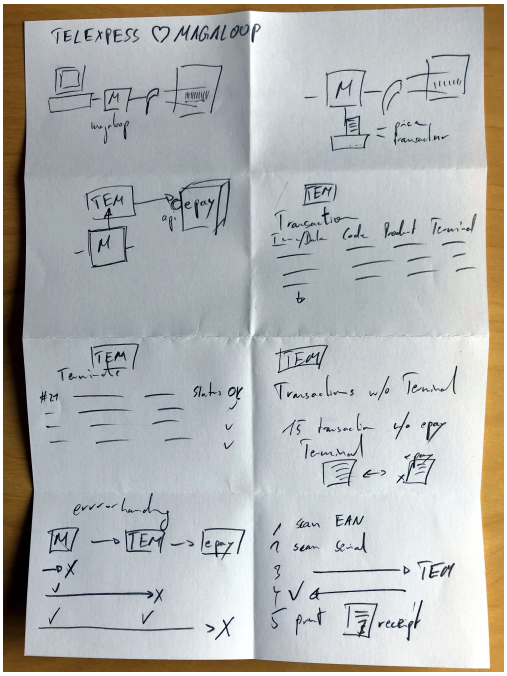



Figure 6.29.: 30 Seconds Sketch Example for Technical Architecture

new feature", claims a partaker of the core group, and a second member adds that "the best time to apply it is at the begining of a new feature, screen, etc.". Similarly, at *Company IV* a participant would use the method for "UI design" or when "[having] a GUI focused problem", as explained by an employee of *Company III*.

Participants agree that "[before] starting [, they] needed to have a clear problem statement" as a member at *Company IV* stated. Further, a participant at *Company III* reported that during the workshop "[the] task was not formulated precisely enough, which made the start of the method somewhat more difficult". Some programmers had difficulties in sketching or thought that they would not be able to sketch, a core group member would "probably never" use it as the participant believes he or she is "not good at it". Finally, the method is not suitable for "[technical] problems without a visual represantation" declared an employee of *Company IV*. A teammate of *Company III*, further adds that "[possibly] in non graphical environments like the backend" there would be no use cases for this method. Although the attendees claimed that 30 Seconds Sketch is not suitable for technical aspects, a participant of the core group has used 30 Seconds Sketch to iterate over a technical architecture of an IT system, as can be seen in figure 6.29. The pictured system is meant to extend traditional cashier machines, so they can be used to activate voucher cards.

*Powers of Ten*

Some programmers at Design-IT were already familiar with the method. A core group mem-

ber believes that "[when he or she doesn't] know how something works [,] it's nice to start understanding the general idea and then work your way down to the deeper levels". The same participant further adds that "it would be useful for debugging" - another partaker agrees and adds that the method is useful "[every] time [he or she debugs] an error". Further, it was mentioned that the method is also helpful for "design reviewing" and "[project] management".

Not many limitations were mentioned by the participants for the Powers of Ten method. One collaborator stated that the method would not be useful "when the problem is very simple". Further, another participant claimed that the method was not helpful "[when the teammates] need to code something that has already been designed", which is contradictory to what other attendees said, as described in the last paragraph programmers could use this method for debugging of code.

*Feedback Capture Grid*

Most participants considered this method to be suitable to gather feedback on a variety of tasks, e.g. "sketches, wireframes, designs, interactions or functionalities", as explained by a core group member, "but also other topics in the company", added another participant. Furthermore contributors could image using the method to test a "prototype in different iterations of the development process", likewise solutions could be tested as well. The core group has used this method to gather feedback for an application that was currently in development and additionally the time tracking feature of the company was evaluated.

Most attendees could not name a situation in which they could not use this method, at the same time, a participant believed that this method was not helpful for projects that are just starting. Another co-worker added that it is not practical "to develop a solution much further than an idea".

*Five Finger Feedback*

As with the Feedback Capture Grid, participants declared that this method was handy for gathering feedback on different topics: An employee of *Company III* explained that "[feedback] can be given in any situation".

Attendees mentioned that this method was especially useful for "spontaneous one-on-one feedback", as a participant of *Company II* stated, or to "[get] short feedback on tricky topics with colleagues" as a colleague from *Company III* declared.

Further, another participant from *Company III* stated that Five Finger Feedback is useful "[in] retrospectives" - an attendee of *Company II* further elaborated on that statement and claimed that the method is practical to "[express his or her] opinion in retrospectives". But also "in employee feedback talks with [...] executives" it would be effective a second member of *Company II* assured. A participant would further use this method "[after] longer meetings" to gather quick feedback. Additionally, a participant at *Company I* can imagine "[giving] feed-

back to team members about their projects" using Five Finger Feedback. A core team member agrees and adds that further it could be used if "there is a prototype to be analyzed".

In contrast, attendees would not use this method in "big rounds of feedback" as an employee of *Company II* stated. "Situations that require extensive written feedback may be handled better with a different feedback method because this method seems to be tailored rather for quick oral feedback" explained a co-worker.

*Empathy Tools*

Participants mainly saw Empathy Tools to be used in projects "targeting a large enough audience that guaranties there will be users with disabilities", explained a core group member. A co-worker further specified that the method is helpful "[while] developing a system that you know is going to be used by a significant amount of impaired people". Empathy Tools is useful for "improving a matured project like db-gruppen.de, trying to convince the [...] minorities among user groups", summarized an attendee. Some also stated that it might not be good for early stages of the product, because they see this method rather as an optimization "after [the team] polished the experience for the average user, prioritize". A collaborator has a different opinion in this regard, and believes that Empathy Tools is practical "[when] developing a new app and designing a possible solution for taking in account different aspects". One participant thinks that the method could be used "[whenever] a system is being designed or is already in production".

*Paper Prototype*

Most participants claimed that Paper Prototype could be used in new projects or features. E.g. "[when] deciding how the UI will behave, or after some crazy eight session, in order to bring more interaction and [to] test the designs" described an employee of Design-IT. Paper Prototype is on top of that a "[good] way to test and validate with the customer while keeping his expectations in check", assured a participant. But also, "[when] there are several ideas about how to design a interaction on an app", the method might be handy for decision making. Further, it was mentioned that it is also useful to find out "how to improve existing [features]".

At the same time, not all collaborators agree with that statement, as a participant explains "[when] the project is already designed, maybe is not that useful to try to reproduce the actual screens in paper". "When the design is pretty advanced, at that point the details need to be discussed over something already implemented maybe, on paper would be really hard to notice the subtleties of the design" another co-worker adds. Finally, it was also mentioned that the method takes too much time, and secondly, Paper Prototype could not be used for "[backend] stuff that needs no user interaction" declared a teammate.

| Method Name | Understandability |
|---|---|
| Five Whys | 4.824 |
| Feedback Capture Grid | 4.714 |
| Empathy Tools | 4.625 |
| Letter to Grandma | 4.500 |
| Paper Prototype | 4.429 |
| Five Finger Feedback | 4.400 |
| 30 Second Sketch | 4.313 |
| Personal Inventory | 4.222 |
| Customer Journey Map | 4.136 |
| A Beginner's Mind | 4.056 |
| Powers of Ten | 3.800 |
| Character Profiles | 3.333 |

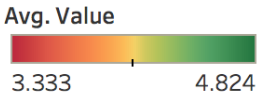



Figure 6.30.: Average Understandability of all Methods

## 6.3. Toolbox Evaluation

In general, all the methods were pretty well understood by most of the participants. In most situations, some details were not understood and only in a few situations there were severe problems with understanding the method or providing results that did not fulfill the task. Half of the participants (73/146) ranked the presentation of the methods with the highest possible grade - and if one also considers the second highest grade it is already above 80% (121/146). The majority of the participants that were not satisfied with the presentation, expected more examples or a better explanation on how to present the results. On the other hand, there were also people that believed that the description was too long. So clearly there exists a discrepancy between team members that can fully and intuitively understand the methods, and others that have difficulties. Thankfully, the team members were helping each other and thus were able to complete the exercises anyways.

As already described in section 6.1.2, after the method evaluation of the Character Profiles method, we have noticed that the method sheets had to be adjusted in order to ensure higher understandability. In the figure 6.30, it can be seen that no further method had an average for understandability as low as Character Profiles.

A listing of all methods with understandability problems can be found in table 6.2 - for each method that revealed issues in that regard the understandability was discussed in section 6.1.

| *Method* | *Section* |
|---|---|
| Character Profiles | 6.1.2 |
| Customer Journey Map | 6.1.3 |
| A Beginner's Mind | 6.1.6 |
| 30 Second Sketch | 6.1.7 |
| Powers of Ten | 6.1.8 |
| Five Finger Feedback | 6.1.10 |

Table 6.2.: Methods with Understandability Problems

*How often and when did the team use the methods?*

We can only answer this question for the core group, as we did not gather information about future method usage from the other workshop attendees. Participants have reported that methods have been used primarily to solve the given tasks during the method evaluation phase. As summarized in table 6.3, five attendees have used selected methods also for further daily tasks, but only three collaborators have done this consciously. The other two teammates have reported that they applied the methods, but as a participant explains "maybe you apply something but you don't notice [it ...] like A Beginner's Mind [...] you really don't know something and you are applying [the method] without noticing". On the contrary, three participants have reported that they have not been using the methods besides the evaluation phase. One employee explains that he had "[several] opportunities to use them [... but he] didn't use them" - he further adds that "[he doesn't] know exactly why, because [he likes] the methods and [he knows] they are useful - and [he knows] if you apply them the result is better".

| *Usage / Participant* | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | $\sum$ |
|---|---|---|---|---|---|---|---|---|---|
| *No Usage* | | | X | | | X | X | | 3 |
| *Unconscious Usage* | | | | X | X | | | | 2 |
| *Conscious Usage* | X | X | | | | | | X | 3 |

Table 6.3.: Method Usage After Evaluation Phase

In spite of that, all participants would like to use at least some of the methods in the future, as summarized in table 6.4. While just two collaborators stated that they would like to use the methods in general, six attendees claimed that they would rather use a selection of the methods, mostly the ones that they have claimed to be their favorite. There was no participant that did not want to keep on using the methods. A contributor stated that "when starting a project from scratch [...] write a letter, if [...] stuck five whys, [...] then ask the guys to draw

a hand on a sheet of paper [to] get feedback". Secondly, a participant would like to keep on applying the methods "Five Whys, A Beginner's Mind, Power of Ten, Feedback Grid and Paper Prototype".

| *Further Usage / Participant* | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | ∑ |
|---|---|---|---|---|---|---|---|---|---|
| *Yes* | | | | | X | X | | | 2 |
| *Yes, some Methods* | X | X | X | X | | | X | X | 6 |

Table 6.4.: Further Method Usage in the Future Reported by Participants

*What prevents the team from using the methods more often?*

A list of all problems that were mentioned by the participants can be found in table 6.5. Six attendees believed that they do not have enough time to apply the methods. It was mentioned that the team is "always in a rush, trying to deliver", and although the "the methods are kind of fast [...] anyway you loose some time". Another employee, further explained that "the end user wants to see something [...] customers [...] want to have the result just right now". A second member of the team agrees and adds that "it wasn't easy to always find some time [... as] you are trying to get stuff done". A co-worker further elaborates that it is "a feeling of missing or lacking time [... that] rather subconsciously puts pressure on oneself".

Additionally, three contributors mentioned that it was not always clear when the methods could be used. One of them believes that "it would be useful to make it extremely clear when [the team] should use [the methods]" and later summarized that "you have to realize, if you are not making progress, if you don't understand something: 'oh maybe I can use this'". It was further reported by two participants that the methods are not prominent enough to "have them present, [...] so that [the team is] aware of them". He further elaborates that "sometimes you don't [...] think about [the methods], and you have been stuck for a long time and using one of the methods would have been helpful". Both teammates had some ideas on how to improve that: One idea is "to have a printed set [with] a super small summary" and the other is to have "big signs in the wall or in the whiteboard [... with] the names of the methods".

Likewise, two participants claimed that a scheduled DT session could ease the usage of the methods. One said that the team has "to define certain times or moments in the week to use them" and his colleague added that "having [the methods] scheduled for a certain hour and all working together [...] on that, would greatly improve how we used the methods". Two collaborators agreed that the methods still need more practise as "this is something new and you know it's gonna take longer". His teammate further explains that "for the methods to be really useful, we should be [...] used to them".

| *Problem / Participant* | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | ∑ |
|---|---|---|---|---|---|---|---|---|---|
| *Time* | X | | X | X | | X | X | X | 6 |
| *Unclear When to Use* | | | X | | | | X | X | 3 |
| *Not Prominent Enough* | | | X | | | X | | | 2 |
| *Needs Scheduled DT Session* | | X | | | X | | | | 2 |
| *Needs More Practise* | | | | | | | X | X | 2 |
| *Needs Adaption of Work Style* | | | | X | | | | | 1 |
| *No Scope for Changes* | | | | | X | | | | 1 |
| *Management Restrictions* | | | | | | | | X | 1 |

Table 6.5.: Problems to Incorporate Methods into Daily Work Reported by Participants

## 6.4. Core Group General Results

The core group has additionally participated in further interviews and questionnaires that have been conducted before and after the method evaluation phase. Most participants of the core group have perceived the methods to be useful and important for software development. Nevertheless, not all members have improved equally in the domains team empathy, end user empathy and product understanding. Likewise, not for all collaborators the same factors lead to an improvement. Additionally during the interviews some participants have mentioned that the regular method calls had a huge impact on team empathy.

### 6.4.1. Empathy

The empathy has been measured using the adjusted Toronto Empathy Questionnaire for team and end user empathy, as described in section 4.3.2. The total score has been calculated as defined in [82] and then compared for each participant. The mean of the total empathy scores have been slightly below average (45) before the method evaluation phase, and have raised above average after application of the methods, as seen in table 6.6. The average of 45 was defined from the original authors to be the average empathy[82]. We use this definition to distinguish between lower than average empathy, so the total score is below 45, and higher the average for all scores higher than this. The maximum possible total score is 64, as it is possible to get up to four points per question[82]. We have used the Wilcoxon Signed Ranks Test to calculate the significance of our results, in the same way as described in a paper by Patterson et al.[67].

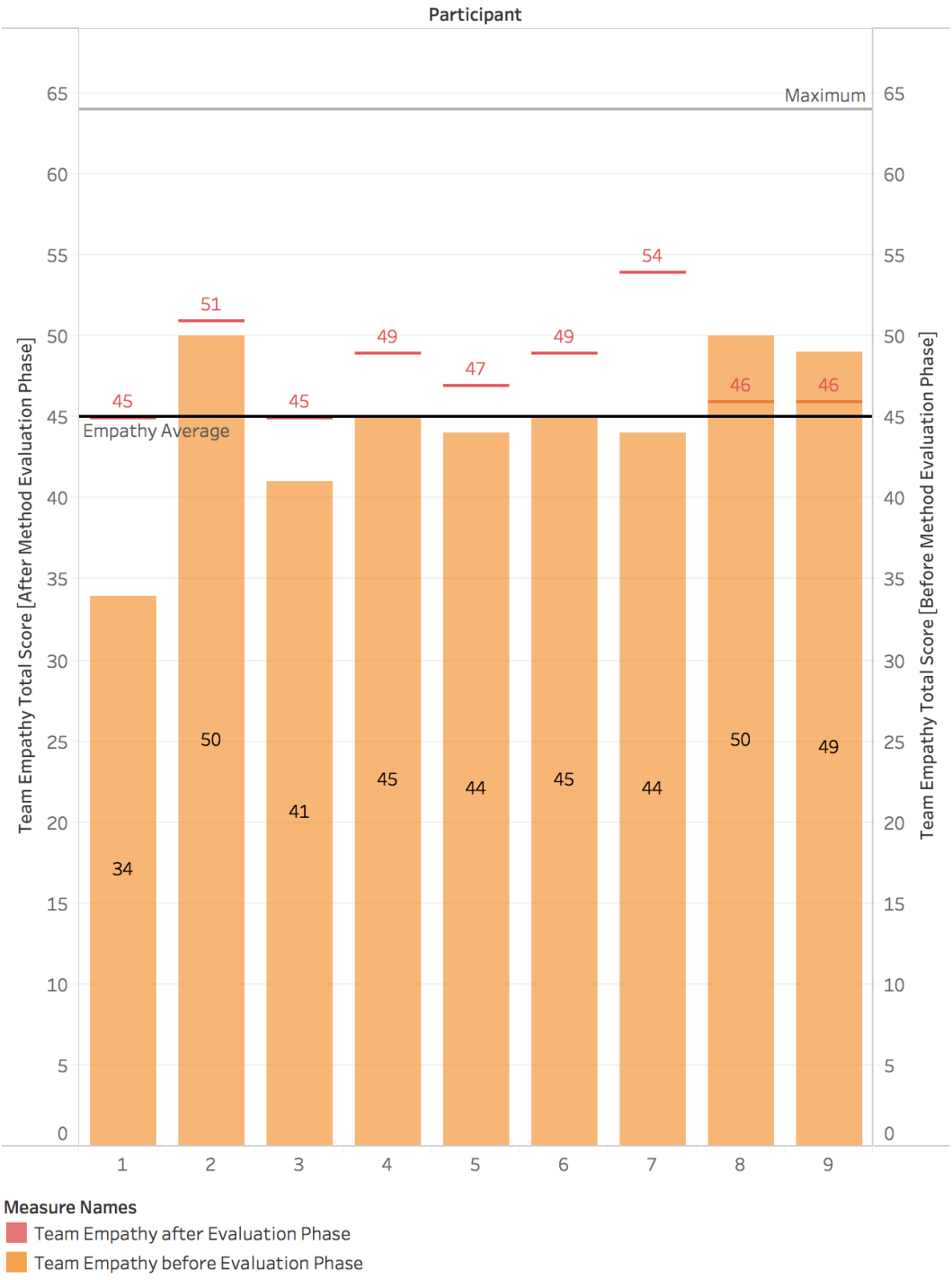



Figure 6.31.: Team Empathy Total Score before and after Method Evaluation Phase

| | *Mean Team Empathy* | *Standard Deviation Team Empathy* | *Mean End User Empathy* | *Standard Deviation End User Empathy* |
|---|---|---|---|---|
| *Before Method Evaluation Phase* | 44.67 | 5.05 | 44.00 | 7.21 |
| *After Method Evaluation Phase* | 48.00 | 3.04 | 46.56 | 4.80 |

Table 6.6.: Mean Empathy Scores Compared

#### 6.4.1.1. Team Empathy

As seen in figure 6.31, the empathy towards other team members has raised for most participants. Only in two cases there has been a drop from 50 and from 49 down to 46 points. In all other cases the empathy has raised. It should be pointed out, that all empathy scores are at least average after the method evaluation phase, where as before four participants had a lower than average empathy towards their team members.

There have been two extreme boosts of eleven (17,18%) and ten (15,63%) points for participant one and seven, both attendees had an empathy score below average before. Four moderate increases of four (6,25%) and three (4,96%) points were achieved by participants three, four, five and six. They all had an average or slightly below average score beofre the evaluation phase. Participant two gained one (1,56%) point on the scala, but this contributor already had an empathy score above average. Two participants had a moderate decrease of four (6,25%) and three (4,96%) points, nevertheless the score is still above average.

The Wilcoxon Signed Ranks Test has showed that the increase is significant at $p \leq 0.05$. This is because the W-value is 8 and the critical value of $W$ for $N = 9$ at $p \leq 0.05$ is 8.

The table 6.7 shows our coding categories, with subcategories and the mentions of the statements in the interviews. As can be seen in that table, participants reported that team empathy has improved in all cases, five out of eight interviewees said that it improved significantly and could provide examples. It has been reported that employees learned how other roles work - or at least are more conscious about the differences now. What is more important, half of the participants declared that there is now more cooperation between roles or at least it is planned to involve other roles more into their daily work. More than half of the attendees claim to hold back less comments now, and one reports that he or she stopped holding back comments completely. Secondly, the collaborators reported that they are more talkative and conscious about their remote co-workers. In general, half of the interviewees declared that their team members know better how to communicate and two participants started talking with more people in the company.

| *Category* | *Answer* | *Mentions* |
|---|---|---|
| Empathy | absolutely improved, with examples | 5 |
| | improved slightly, no real examples | 3 |
| Difference between roles | learned how other roles work | 5 |
| | more conscious | 2 |
| | no difference, no change | 1 |
| Cooperation between roles | no change | 4 |
| | much more cooperation | 2 |
| | more cooperation between roles | 1 |
| | more cooperation planned | 1 |
| Holding back comments | holding back less now | 4 |
| | still try not to hold back, no difference | 3 |
| | held back, now says everything | 1 |
| General communication | people know better how to communicate with examples | 4 |
| | started talking with more people | 2 |
| | no improvements | 1 |
| | slightly, no examples | 1 |
| Remote communication | no improvements | 4 |
| | more talkative, more conscious | 4 |

Table 6.7.: Coding Categories and Mentions for Team Empathy from the Interviews

#### 6.4.1.2. End User Empathy

On the other hand, the empathy towards end users has raised in all but two cases, as seen in figure 6.32. In one case, the drop is probably meaningless as it was just a drop of one point.

Participants three and seven had a huge rise of seven (10,94%) and nine (14,06%) points originating both from below average. Further, there has been two moderate increases of 4 (6,25%) and 5 (7,81%) points for attendees one and nine from below average as well. Secondly, participants two, three, four and five had insignificant increases (1,56%) or decreases (-1,56%) of one point - one originated from below average, the others from above average. Finally, there has been one moderate decline of four (6,25%) points for participant eight, from above

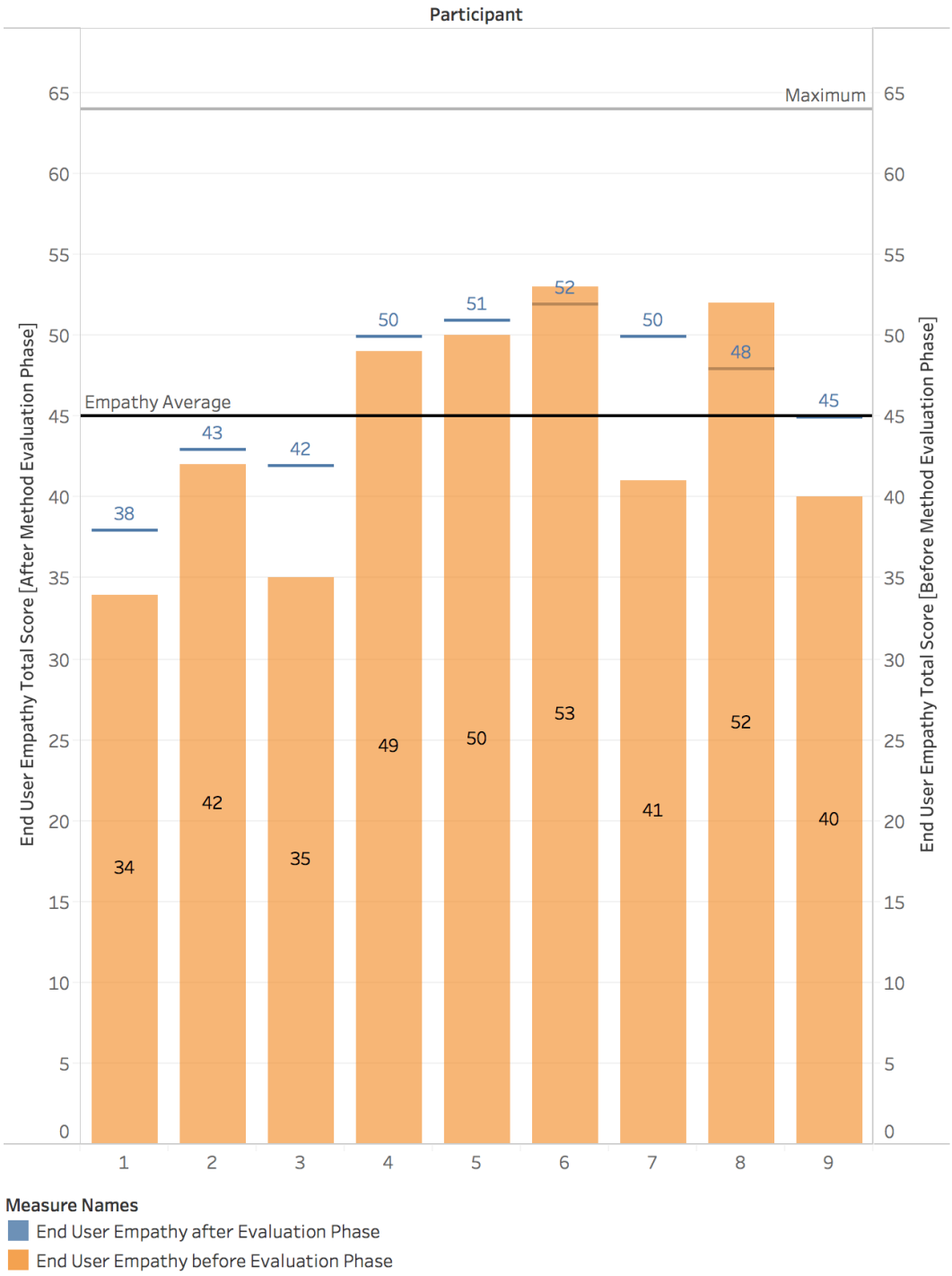



Figure 6.32.: End User Empathy Total Score before and after Method Evaluation Phase

average. Interestingly, attendee eight has also a decreased team empathy of four points, compared to before the evaluation phase. A possible explanation could be that this contributor had a bad day while filling out the post evaluation questionnaires. Another interesting fact, is that end user and team empathy do not always correlate among participants - e.g. participant two has a rather high team empathy but the end user empathy is below average.

The Wilcoxon Signed Ranks Test has showed that the increase is significant at $p \leq 0.05$. This is because the W-value is 8 and the critical value of $W$ for $N = 9$ at $p \leq 0.05$ is 8.

| *Category* | *Answer* | *Mentions* |
|---|---|---|
| Clear Picture of End User | has increased in general, with example | 5 |
| | no change | 1 |
| | improved for projects where methods were applied | 1 |
| | has increased in general, without example | 1 |
| Understand all Requirements | a bit better, no example | 4 |
| | no change | 2 |
| | more conscious | 1 |
| | yes, if not asking | 1 |
| Requirements don't fulfill need | yes, but no example | 4 |
| | now more aware, asking more | 2 |
| | no | 1 |
| | realized that his is happening, started to communicate with customer | 1 |
| Changed Thinking | no change | 4 |
| | Started thinking in a human centered way | 2 |
| | changed thinking, no examples | 1 |
| | Realized that problems understanding requirements were a general issue, not his/her fault | 1 |

Table 6.8.: Coding Categories and Mentions for End User Empathy from the Interviews

The findings from our solo interviews about end user empathy can be found in table 6.8. This table shows our coding categories, with subcategories and the mentions of the statements in the interviews. Almost all participants reported that they now have a clearer picture of the end user and five could name an example. Likewise, half of the interviewees reported to understand requirements at least a bit better than before. One participant has no realized

that there are requirements that do not fulfill the need of the end user and two further are now more aware of this problem. Two attendees claim to have started thinking in a human centered way and another participant realizes that problems understanding requirements of the customer were a general issue and not his or her fault.

### 6.4.2. Product Understanding

Each member of the core group filled out the product understanding questionnaire, as described in section 4.3.2, before and after the method phase for a specific project. For each question and participant we have counted the amount of new information that was written down about the product. If for a particular question and participant there was at least one new information, we marked it as seen in table 6.9.

| *Question / Participant* | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | ∑ of Mentions |
|---|---|---|---|---|---|---|---|---|---|
| Goal | - | X | X | X | - | X | - | - | 4 |
| User Base | - | - | X | X | - | - | - | - | 2 |
| Solves which Problem? | X | - | X | - | - | X | X | - | 4 |
| Special related to Product | - | - | - | - | - | - | - | - | 0 |
| Benefits | X | - | X | X | - | X | - | - | 4 |
| Competition | - | - | - | - | - | - | - | - | 0 |
| Life-Time | - | X | - | - | - | - | - | - | 1 |
| Key Functionalities / Benefits for End User | X | - | X | X | - | X | X | - | 5 |

Table 6.9.: New Information in Product Understanding Questionnaires after Method Evaluation Phase

Questions about what is special related to the product and about the product's competition have not yielded any new information. Those topics have not been successful before the evaluation phase either, e.g. the answers about the competition of the software ranges from "I don't really know", "I have no idea, but there are some competing softwares" to "Of course there is, I've never used any similar product. I can not do the comparation". On the contrary, especially for key functionalities and benefits for the end user, participants were revealing more knowledge in five cases. An example can be seen in figure 6.33, it demonstrates that the attendee has a better understanding of the product. In the first questionnaire, the participant just stated that users can get certifications and that further sites with well organized information exist. After the method evaluation phase, the participant was not referring to the general term *certification* anymore, but could specify that end users are able to collect Continuing Medical Education (CME) points on the platform. On top of that he mentioned

**Before** Straightforward interface for the users to get certifications, the site has few pages with well organized information and everything looks like it's high quality

**After** It's like a youtube with the ability to take a test for CME points, so the end user gets to learn something and can also see the videos later if they want to

Figure 6.33.: Key Functionalities and Benefits for the End User Described Before and After Method Evaluation Phase

that the platform hosts videos - instead of referring to it as *organized information* - and could name two benefits of the platform for end users: They can learn something and review the videos, if necessary. Likewise, the participants were able to put new information about the goal of the project, its benefits and the problem it solves. Two attendees as well revealed more information about the user base and one about the life-time.

A contributor has revealed that he "was asked to do a system with multiple choice questions and the possibility to select correct answers, and after some time working on it, [he] realized [he] didn't even know who would use the system, and what AIESEC was about as a company". In the solo interviews, attendants have mentioned that the DT methods helped to realize a lack of knowledge in the first place. A second member of the team clarifies that he had to switch to another project to apply Letter of Grandma because for the initial project "there have been other design thinking methods applied that made it super clear what the purpose of the system is and the client and [the team] are on the same page".

# 7. Summary

Our results show that team members liked the methods and are eager to keep on using them. We could further show, that an extensive DT phase using our method toolbox has improved team empathy, end user empathy and product understanding. In this chapter, we will first discuss our results, then present the future work for this thesis, and finally, conclude our contributions.

## 7.1. Discussion

### 7.1.1. Research Questions

In our first research question we have asked, which methods from DT can be used in an agile development process to support the work of developers in a human centered way and to support empathy for team members and end users.

We could support the statement of Pedersen et al.[68] that developers are open towards design methods, still, some participants were not happy to draw or believed that they are unable to draw. We could observe this for the methods 30 Second Sketch and Paper Prototype. Nevertheless the methods were positively received by the participants - after the method evaluation phase Design-IT has even added the methods to their blog, showing a general interest in that regard.

On the contrary, we could not support the statement of Domschke et al. that homogeneous teams resist DT. While interdisciplinary teams did provide better outcomes, our results do not show that homogeneous teams had difficulties accepting DT methods. The teams at *Company I* and *Company III* consisted only of developers and they have accepted the methods as good as the other teams that were mixed. On the other hand, we found out that teams that already got in touch with DT, had an easier start and provided better results. During our observation the participants of *Company II* were already familiar working with post-its and whiteboards and spent less time before starting to apply the methods.

To answer our first research question, we have created a table with all the methods and their benefits according to our results. From the observations that we made during the workshops

and in the method evaluation phase, we come to the conclusion that the ability to solve and find problems in day to day work, as well as a better communication, support the work of developers in a human centered way. These findings, and the empathy towards the team and end users, are represented in the following table 7.1.

| *Method* | *Solving and Finding Problems* | *Communication* | *End User Empathy* | *Team Empathy* |
|---|---|---|---|---|
| Personal Inventory | | | X | X |
| Character Profiles | X | | X | |
| Customer Journey Map | X | | X | |
| Five Whys | | X | | X |
| Letter to Grandma | | X | | |
| A Beginner's Mind | X | X | | |
| 30 Second Sketch | X | X | | |
| Powers of Ten | X | | | |
| Feedback Capture Grid | X | X | | X |
| Five Finger Feedback | | X | | X |
| Empathy Tools | | | X | |
| Paper Prototype | X | X | X | |

Table 7.1.: Methods with their Benefits According to our Results

Further, we want to answer our second research question: "How to adjust the methods to suite the needs of the engineering profession?".

As reported by de Paula et al.[23], undergraduate students with no experience in design methods can successfully use DT for software development. We can support this statement, as no real adjustment for the methods was necessary and the participants found various application fields for every method. We have only adjusted the method Personal Inventory, to not only reflect the end users but also the team. Further, we can extend the statement by de Paula et al.[23], as not only undergraduate students are able to apply DT methods in a software development context, but as in our case, also employees of software development companies with experience were able to use and benefit from DT in that context.

Pedersen et al. modified their selected UX methods for the IT context, and although not necessary, we could imagine that some methods could indeed benefit from adjustments or refinements. We have observed that for some methods, e.g. the Customer Journey Map, groups

that were given a tighter time schedule still produced good results, but complained less about spending too much time. A possible adjustment to the Customer Journey Map mentioned during a workshop, was to include pain points and ideas directly in the map, so these ideas do not get lost during the creation of the map. Likewise, for methods that require drawing, in our case 30 Second Sketch and Paper Prototyping, we could imagine to include facilitators so that the participants are more comfortable with the task. It would be possible to include warm-up exercises or provide a sketch cheat sheet, in order to provide easy ways to sketch user interfaces. E.g. OpenSAP has created an online course with similar content for the IT context[1]. Additionally, both methods, 30 Second Sketch and Paper Prototype, could be combined, so that first rough sketches are created in a fast way and later these are used to create a Paper Prototype. For the method Empathy Tools it would be useful to add an example to not only reflect the inabilities of a user, but also the context in which the software is used, e.g. an app directed towards cyclists might be used while riding a bike or while wearing with gloves.

Further, our results support that DT outbreaks, as defined in processes like DT@Scrum or Converge, can be used in case of blockers. Our findings also show that the methods Five Whys and A Beginner's Mind can be used when a team member is stuck with a problem. Various participants have mentioned that explaining a problem to a beginner in some situations already solved the problem they faced. In another example a participant could help a co-worker to find a solution for an issue by asking questions that were challenging his assumptions.

While the participants can imagine many possible applications for the methods, they still lack the ability to spot these moments in daily work, and would like to be supported in this regard. Mostly, we have only given very broad tasks to the teams, but in the cases where more specific tasks were handed out, better results could be observed. We observed this e.g. comparing the Customer Journey Map experience at *Company II* and *Company IV*: The groups at *Company II* were given a specific task and delivered a good map with various pain points and ideas how to solve these. At *Company IV* the groups were given no specific tasks and multiple attendees had problems to get started with the method and thus provided less useful results.

For every method we have picked the main applications mentioned by the participants:

**Personal Inventory** Onboarding a project, team or employee

**Character Profiles** Find pain points in a system / Verify and get ideas for new functionalities

**Customer Journey Maps** Find pain points in a system or solution and resolve these issues to improve UX

[1] "Be Visual! Sketching Basics for IT Business" `https://open.sap.com/courses/bvis1` last accessed October 22, 2018

**Five Whys** When stuck / Gathering customer requirements

**Letter to Grandma** Communicating new features and projects

**A Beginner's Mind** When stuck / Discuss new functionalities

**30 Second Sketch** UI design / Ideation

**Powers of Ten** Understand a problem

**Feedback Capture Grid** Gather feedback in any situation / Test prototypes and solutions

**Five Finger Feedback** Gather feedback in any situation / Spontaneous one-on-one feedback / Retrospectives

**Empathy Tools** Find UX problems in big systems, or systems targeting impaired people

**Paper Prototype** Designing new projects or features

Most methods can be used regularly, as participants mentioned, e.g. they can be used for every new feature, or when stuck. This was not the case with Personal Inventory. Participants had doubts whether this method could be applied repeatedly. The method could be e.g. used regularly for onboarding new team members, by presenting the workflow of one employee every month or by providing regularly pictures of different environments where end users are using the app.

In the following, we want to answer our third research question: "How to gather these methods in a toolbox, so that software developers can be easily trained to effectively apply them in appropriate situations?".

Our results show that most methods were very understandable for the participants and even though in some cases the participants were not sure how to perform a certain method, members of a team shared their knowledge and thus the team could perform the task on their own. We observed this e.g. for the Character Profiles method: Though the participants were not sure how to apply the method at first, they have started exchanging examples and ideas about the application of the method and finally were able to provide useful results.

In contrary to Pedersen et al.[68], we did not perform any additional training, but just handed out method sheets that were read by the participants. Our results show that this was enough to provide the attendees with a good understanding, as more than 80% of the participants have rated the method with the highest and second highest understandability in the questionnaires. Further Pedersen et al. reports that software developers were able to perform their work without the need of UX specialists, we can support that statement, as we did not involve any external specialists to facilitate the usage of the methods. Though there was no

UX or DT specialist involved, in three of our evaluation groups a designer was part of the team.

According to Frye and Inge[30], smaller teams are better to avoid long discussions when applying DT. Most of our participants applied the methods in small teams and we did not observe any problems related to long discussions. There were two exceptions though. In one case we observed that in a bigger group of eight people the application of the Five Finger Feedback method was considered to take too much time. On the other hand, we observed difficulties in coming to an agreement in a group of three participants while doing the Customer Journey Map.

Although general understanding of the methods was good, more examples could be beneficial. Our findings show that participants expect detailed examples. As an example, though the method sheet for A Beginner's Mind provided three examples, participants expected these to be more detailed. On the other hand, it might be good to include short summaries for more experienced participants, that otherwise might feel overwhelmed by the long descriptions. This was evident for the A Beginner's Mind method, that as already mentioned would benefit from more detailed examples, but on the other hand there were participants that disliked the long description for a method, that was very easy to understand from their perspective.

Character Profiles was the least understandable method in our study. This method has only one unspecific example, that just contains questions that one could ask - a detailed example with an exemplary persona and a task to be solved could help to get a better understanding of how to use Character Profiles. Likewise, it might be helpful to change the example of the Customer Journey Map to map an IT feature or project. Additionally, the difference between action and touchpoint should be explained more in depth. For Powers of Ten our method sheet already provided three examples, in this case it could be helpful to explain the similarity to debugging a software problem. Although most participants had no trouble understanding A Beginner's Mind, a useful addition might be to extend the provided explanations in the existing examples. Attendees mentioned issues when following multiple paths with the Five Whys method, in this case it would be beneficial to include a better description on how to handle this situation. For the Five Finger Feedback, it could be helpful to add a better description of the meanings of each finger, e.g. it could be easier to understand "what I take with me" than "what I find emotionally interesting" as a definition for the ring finger.

Even though most methods were perceived as useful and the participants had no problems understanding them, the pickup rate was rather low after the method evaluation phase. Our results show that the most prominent reason for this was lack of time. Gurusamy et al.[39] describe that, compared to user centered approaches, agile is reducing most upfront research and analysis. Thus in IT development, people are concerned that too much time is spent before the development starts. As DT was used during development in our approach, we can extend that statement as our results show that the feeling of spending too much time in DT is still an issue when in the development phase. A similar statement is described by Frye and Inge[30], especially mentioning time spent on infeasible ideas. We cannot support that com-

pletely, as our results show that participants welcomed ideas that are hard to realize or pointless in the 30 Seconds Sketch method. On the other hand, according to Frye and Inge[30] DT can eliminate much more waste than it creates, if it is used correctly. Although our findings do not show this tendency directly, we observed that various methods have found issues in solutions that were not developed yet. In this way, it was possible to effectively reduce development effort by avoiding expensive redesigns after the implementation of the feature. E.g. applying Powers of Ten has changed the layout of a screen to be more understandable before the development started.

The perceived lack of time is thereby short sighted and should be tackled by reducing the feeling that DT activities are additional measures to take. This could be done e.g. by explicitly allocating time for DT activities in daily work.

Further, our results show that participants did not use the methods, because it was not easy for the team members to realize in the particular moment that a DT method could be helpful. Additionally, attendees stated that it could help to make the methods more prominent. This could be done, e.g. in a scheduled DT session that would be done regularly. This would tackle another issue mentioned by the participants, the fact that the methods need more practise. At the same time, we can imagine that DT methods could be integrated into agile meetings, e.g.:

**Backlog Grooming** 30 Second Sketch, Paper Prototype, Character Profiles, A Beginner's Mind, Powers of Ten

**Sprint Planning** Customer Journey Map, Letter to Grandma, Character Profiles

**Sprint Review** Feedback Methods, Empathy Tools

**Retrospective** Feedback Methods, Five Whys

**Interdepartmental Meetings** Personal Inventory

On the contrary to Gurusamy et al.[39], our findings do not show that DT is perceived as a risk by employees that report to a higher hierarchy, but the results do not refute it either.

### 7.1.2. Hypothesis

In our first hypothesis, we assume, that establishing Design Thinking practices for giving feedback and exploring and sharing ideas, fosters empathy for team members, and thereby leads to a better shared understanding and less miscommunication.

Jurca et al.[51] recommend artifacts and practises to improve communication and collabora-

tion between agile developers and UX designers. Our results show that not only the methods themselves have proved to be useful for communication and collaboration, but also the regular method calls done in the core group. These calls were not a method itself, but were conducted in a DT setting - every participant had to present their results and later the team has evaluated these in a group setting. Thus we can support the statement by Jurca et al.[51]. E.g. a designer of the core group stated that she started to talk with more of the developers because of the DT methods. On the other hand a developer that had problems speaking up in company meetings started to be very talkative.

Further, according to Ungar and White[85], collaboration between developers and designers is increased by using DT in a design studio setting. Our findings show that collaboration did increase, but not in all cases. On the other hand, Ungar and White[85] showed that knowledge transfer is also facilitated in a design studio setting. Our observations support this statement, as e.g. participants were exchanging information to understand the methods. Likewise, we observed that by using A Beginner's Mind, a core group member could share knowledge that he had obtained from his colleague with two other co-workers.

Also, for distributed teams incorporating DT methods has an advantage. Due to the Personal Inventory participants have a better imagination of the workplace of their remote peers, and working together in mixed teams leads to more openness towards some remote co-workers that one normally does not work with. In this way we can support and extend the statement by Bravo and Adler[10] that a rise in interdepartmental communication could be observed.

Further, it could be shown that empathy in the team has significantly increased. Our results show that it was not necessarily the methods that had supported team empathy but also the act of doing things together - and as already said especially the method calls have proven to be beneficial. Incorporating DT methods into daily work also leads to a better communication in the team and less holding back of comments. Participants could learn how other roles work or are at least more conscious about the differences, e.g. programmers learned about the workflow of designers using the Personal Inventory method. Additionally, we could observe more cooperation between the roles, as e.g. a lot of developers from the core group stated after the method evaluation phase that a specific designer would be their primary source of information about the products that they develop. Before the method evaluation phase this was mostly the head of the company.

Our second hypothesis states that making use of Design Thinking techniques, that bring developers into contact with the user, together with prototyping methods, help developers to gain empathy for the user and bring a human centered perspective to the development team.

Ximenes et al.[86] used DT at the beginning of a project to empathize with the end user. According to our findings, this is not only possible in the beginning but throughout the whole project. Though some methods were mentioned to be more beneficial in the beginning of a project, in general, participants have used the methods in any state of the project. At *Com-*

*pany II* e.g. the Customer Journey Map was used to map an existing feature and an attendee could empathize with the end users using the emotional journey of the map. De Paula et al.[22] on the other hand, report that DT should be used in the entire process to consider the users' needs. Our findings show that participants are more aware of users' needs and have a clearer picture of the end user, thus supporting the statement of de Paula et al.[22]. To give an example, a participant of the core group reported that he would now consider limitations of the end users' workplace of the end users when providing instructions on how to do something.

As with the empathy towards team members, our findings show that participants also have a significantly higher empathy towards end users. While not all participants have a better understanding of requirements, our results show that participants started to change their mindset to think whether requirements make sense and do not just accept requirements as given. This ensures, that in general, there is a better understanding of requirements or in case of a lack of understanding, participants would ask for more information. In some cases our results could also prove a changed mindset that lead to a more human centered thinking. An interviewee of the core group e.g. stated that after she got introduced to the 30 Second Sketch method, she started to think about designs and screen layouts and would not just accept them as given.

### 7.1.3. Limitations

Though we have conducted our research to the best of our knowledge, there are some limitations. Our results show an increase in team empathy, end user empathy and product understanding, but due to the nature of our study we cannot exclude external factors that might have had an impact. Likewise, some findings are only based on the core group, due to limitations of time and availability of the other evaluation groups. Thereby, our results are based on a small data sample. The participation of the groups is further described in section 4.3 and additionally, for each method the groups that attended are mentioned in section 6.1.

## 7.2. Future Work

Our findings have shown that DT methods are useful in agile software development, however, we could only test a limited set of methods. For future work, we suggest to analyze further methods. Including observation methods, e.g. Shadowing[46] or Fly on the Wall[46] could be valid extensions of the toolbox, as our results show that participants would like to see the software that they develop being used by the end users.

Further, we have shown that though participants liked the methods, there are still open questions regarding the usage of the methods in daily work. While we suggested some counter-

measures to facilitate method usage, we were not able to test them. An extensive analysis of the problems why software development teams are perceiving DT as too time consuming, could also be investigated in further research.

In the same matter, future work could be conducted to experiment with time restrictions of the methods. Our results show that e.g. creating a Customer Journey Map was possible in four hours, but also in 30 minutes. In future work, it could be further explored whether teams accept DT methods better, when they are conducted in a shorter time frame.

In our work we have showed possible applications for each method, but future work could further investigate how useful these applications are and provide recommendations or specific instructions when a method can be used. This could be done e.g. by creating a process that combines DT and agile software development by using our toolbox.

Additionally, we have mentioned a few possible adaptations to our methods, that could further optimize them for the usage in IT. Our results show that participants could apply the methods without any modifications, but there is room for improvement.

Finally, a good addition would be to conduct a study with a quantitative test sample involving more participants.

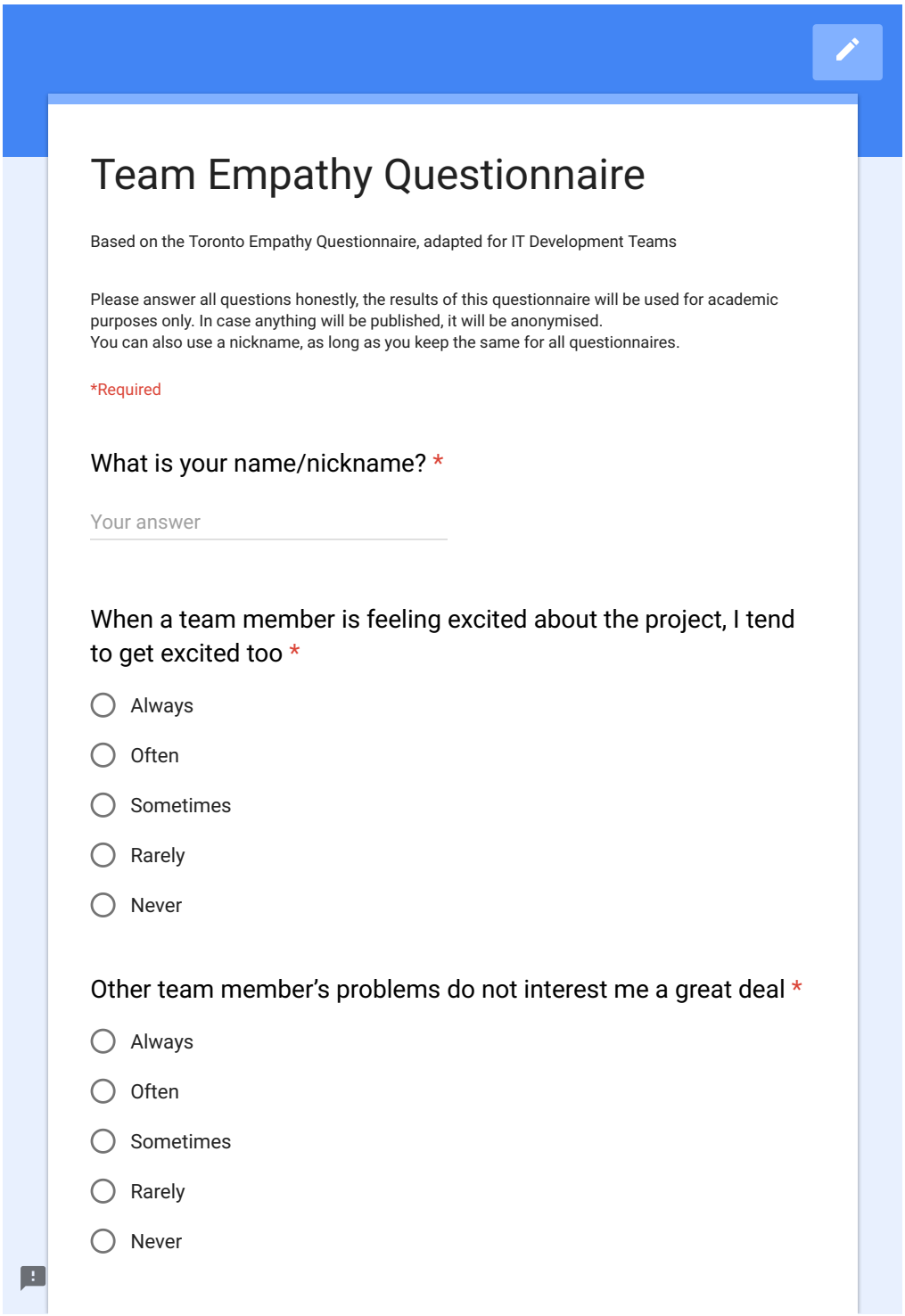
## 7.3. Conclusion

Our work extends existing research on integrating DT with agile software development methodologies, as this is the first DT toolbox for agile software development teams. This toolbox can be used in existing processes that combine DT with agile methodologies - or can be used to create new processes.

We could show that applying DT practices in agile day-to-day work leads to higher team empathy, end user empathy and product understanding. Further, we have shown that DT methods are useful to support the work of developers in a human-centered way by finding and resolving UX issues in existing or planned solutions. Despite these findings, our research shows that after being introduced to the DT methods, members of software development teams did not start using them on their own. Further research needs to be conducted on how to overcome these barriers.

# A. Questionnaires and Interviews

## A.1. Team Empathy Questionnaire



# Team Empathy Questionnaire

Based on the Toronto Empathy Questionnaire, adapted for IT Development Teams

Please answer all questions honestly, the results of this questionnaire will be used for academic purposes only. In case anything will be published, it will be anonymised.
You can also use a nickname, as long as you keep the same for all questionnaires.

*Required

What is your name/nickname? *

Your answer

When a team member is feeling excited about the project, I tend to get excited too *

- Always
- Often
- Sometimes
- Rarely
- Never

Other team member's problems do not interest me a great deal *

- Always
- Often
- Sometimes
- Rarely
- Never

It upsets me to see a team member being treated disrespectfully *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I remain unaffected when a team member is happy about our project *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I enjoy helping out other team members *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I have tender, concerned feelings for team members that have more problems at work than me *

- Always
- Often
- Sometimes
- Rarely
- Never

When a team member starts to talk about his/her problems, I try to find an easy way out by delegating the issue to somebody else or by proposing a quick-fix *

- Always
- Often
- Sometimes
- Rarely
- Never

I can tell when a team member is unsatisfied with something in the project even when they do not say anything *

- Always
- Often
- Sometimes
- Rarely
- Never

I find that my own mood about a project reflects the mood of my team members *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I do not feel sympathy for team members who cause their own problems *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I become irritated when a team member gets furious *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I am not really interested in how other team members feel about the project and their work *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I get a strong urge to help a team member, when I see them struggling with the project *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

When I notice a team member getting wrongly criticised for their work, I do not feel very much pity for them *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I find it silly for team members to be very frustrated about a project *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

When I notice a team member getting too much workload, I feel the need to take over some work *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

**Thank you for your time**

Don't forget, as this is an academic work, I rely on honest answers. Thank you :)

SUBMIT

Never submit passwords through Google Forms.

## A.2. End User Empathy Questionnaire

# End User Empathy Questionnaire

Based on the Toronto Empathy Questionnaire, adapted for targeting end users.

Please answer all questions honestly, the results of this questionnaire will be used for academic purposes only. In case anything will be published, it will be anonymised.
You can also use a nickname, as long as you keep the same for all questionnaires.

*Required

What is your name/nickname? *

Your answer

When an end user is feeling excited about the software, I tend to get excited too *

- Always
- Often
- Sometimes
- Rarely
- Never

End users' problems do not interest me a great deal *

- Always
- Often
- Sometimes
- Rarely
- Never

It upsets me to see an end user being treated disrespectfully *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I remain unaffected when an end user is happy about our software *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I enjoy helping out end users *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I have tender, concerned feelings for end users that have more problems with the software than me *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

When an end user starts to talk about his\her problems with the software, I try to steer the conversation towards something else, e.g. I believe that the problem lies with the user not the software *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I can tell when an end user is unsatisfied with something in the software even when they do not say anything *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I find that my own mood for the software often reflects the mood of the end users *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I do not feel sympathy for end users who cause their own problems in the software *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I become irritated when an end user gets furious *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I am not really interested in how end users feel about the software and our work *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I get a strong urge to help an end user, when I see them struggling with our software *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

When I notice an end user getting wrongly criticized for their work or interaction with the software, I do not feel pity for them *

- ◯ Always
- ◯ Often
- ◯ Sometimes
- ◯ Rarely
- ◯ Never

I find it silly for end users to get very frustrated about the software *

- Always
- Often
- Sometimes
- Rarely
- Never

When I see an end user being exploited in his/her rights, I feel kind of protective towards him/her *

- Always
- Often
- Sometimes
- Rarely
- Never

**Thank you for your time**

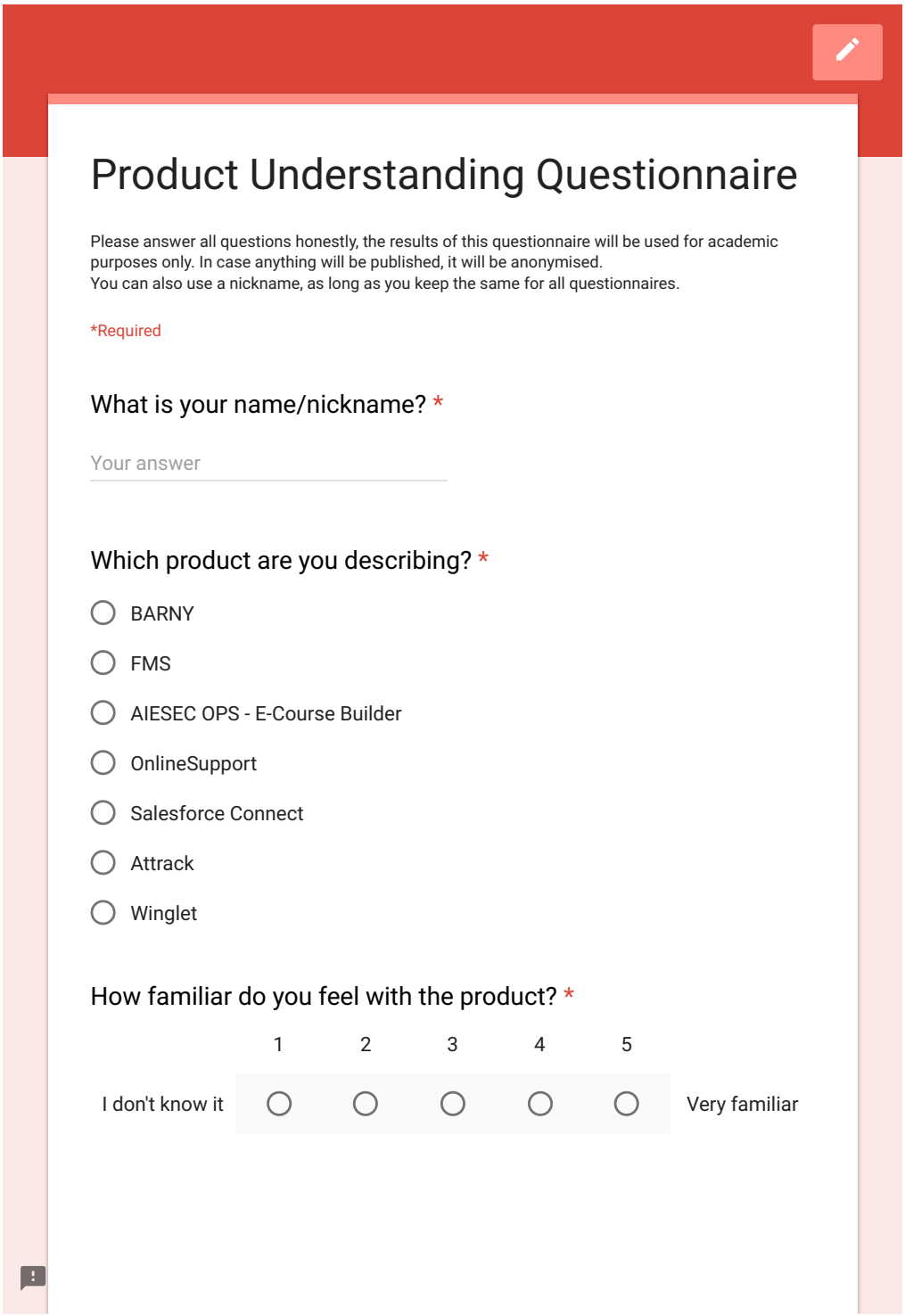
Don't forget, as this is an academic work, I rely on honest answers. Thank you :)

SUBMIT

Never submit passwords through Google Forms.



## A.3. Product Understanding Questionnaire



# Product Understanding Questionnaire

Please answer all questions honestly, the results of this questionnaire will be used for academic purposes only. In case anything will be published, it will be anonymised.
You can also use a nickname, as long as you keep the same for all questionnaires.

*Required

What is your name/nickname? *

Your answer

Which product are you describing? *

- BARNY
- FMS
- AIESEC OPS - E-Course Builder
- OnlineSupport
- Salesforce Connect
- Attrack
- Winglet

How familiar do you feel with the product? *

| | 1 | 2 | 3 | 4 | 5 | |
|---|---|---|---|---|---|---|
| I don't know it | ○ | ○ | ○ | ○ | ○ | Very familiar |

Do you think you should be more familiar with the product? *

- ○ Yes
- ○ No
- ○ Other: _____

What is the goal of the product? *

Your answer

Who is using the product? / Who is the user base? *

Your answer

What problem does the product solve? *

Your answer

What is special related to this product? *

Your answer

What is the benefit of using this product? *

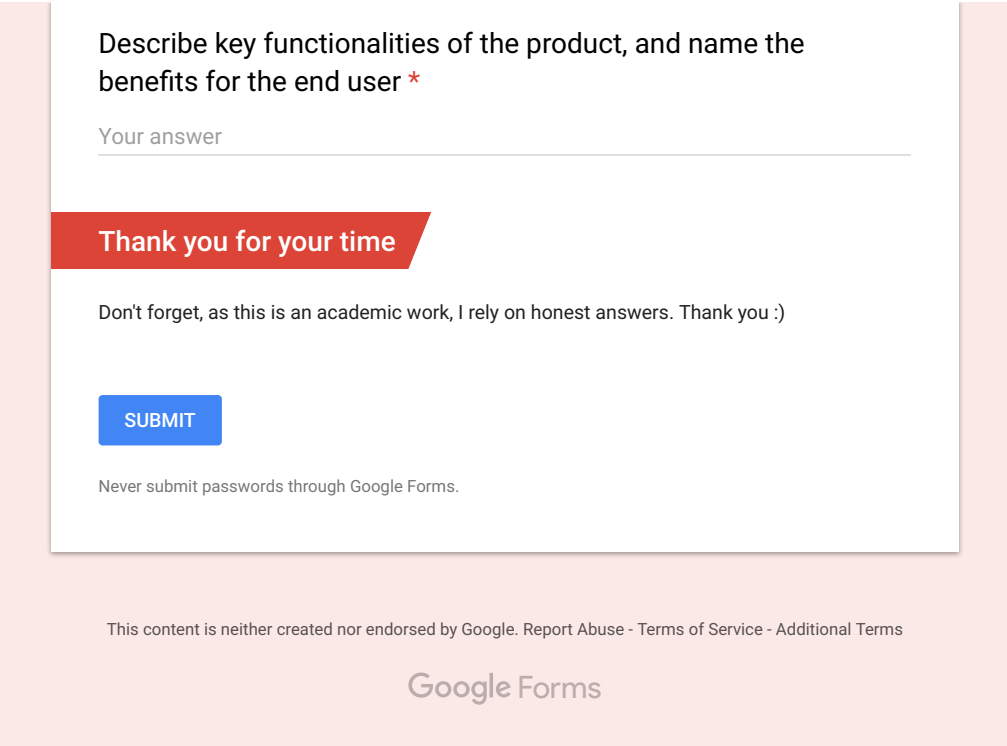
Your answer

Is there any competition for this product, and if, how does it compare to our product? *

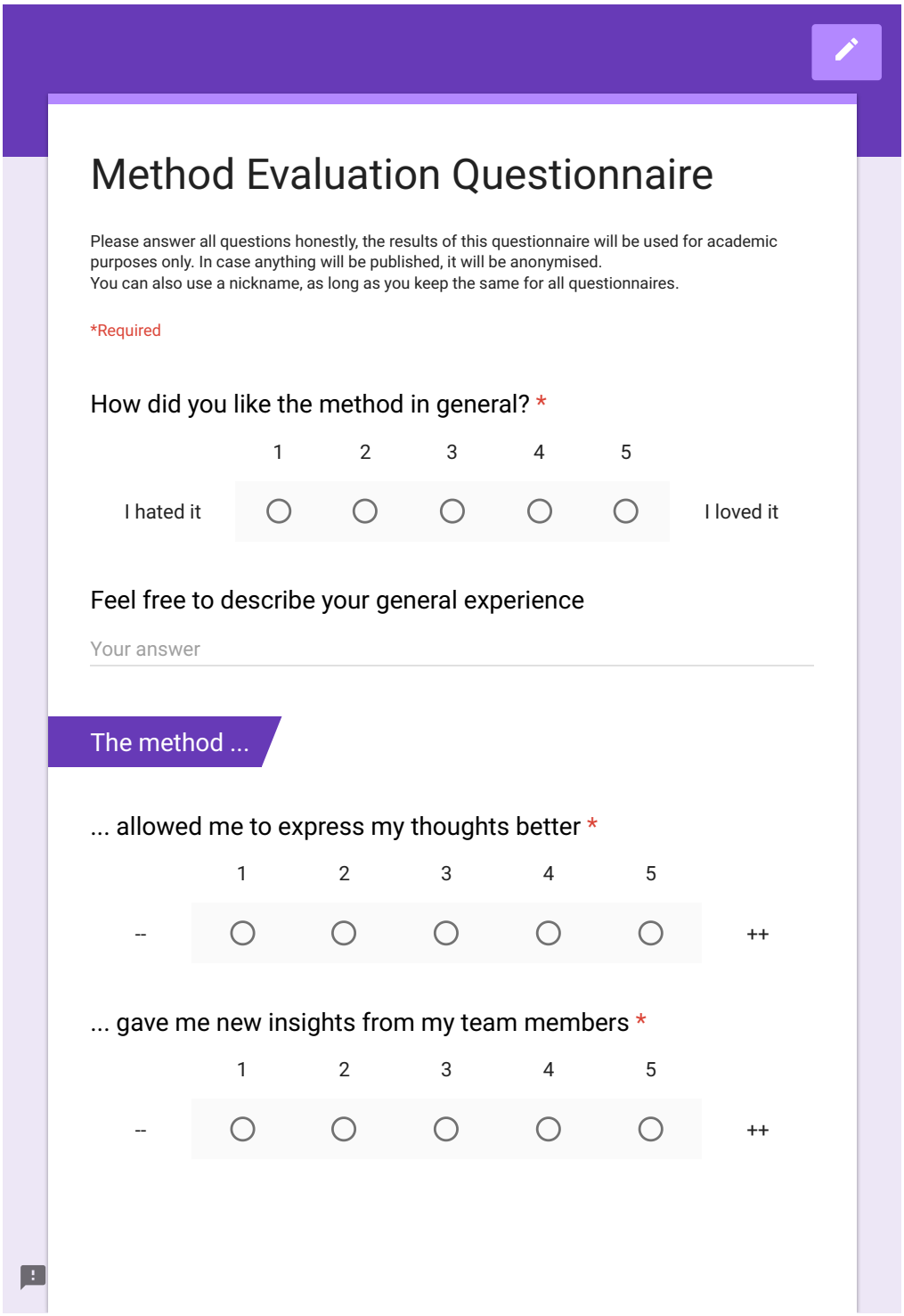
Your answer

What is the expected life-time of the product, is it bound to special conditions? *

Your answer


Describe key functionalities of the product, and name the benefits for the end user *
Your answer
Thank you for your time
Don't forget, as this is an academic work, I rely on honest answers. Thank you :)
SUBMIT
Never submit passwords through Google Forms.
This content is neither created nor endorsed by Google. Report Abuse - Terms of Service - Additional Terms
Google Forms

## A.4. Method Evaluation Questionnaire



# Method Evaluation Questionnaire

Please answer all questions honestly, the results of this questionnaire will be used for academic purposes only. In case anything will be published, it will be anonymised.
You can also use a nickname, as long as you keep the same for all questionnaires.

*Required

How did you like the method in general? *

| | 1 | 2 | 3 | 4 | 5 | |
|---|---|---|---|---|---|---|
| I hated it | ○ | ○ | ○ | ○ | ○ | I loved it |

Feel free to describe your general experience

Your answer

## The method ...

... allowed me to express my thoughts better *

| | 1 | 2 | 3 | 4 | 5 | |
|---|---|---|---|---|---|---|
| -- | ○ | ○ | ○ | ○ | ○ | ++ |

... gave me new insights from my team members *

| | 1 | 2 | 3 | 4 | 5 | |
|---|---|---|---|---|---|---|
| -- | ○ | ○ | ○ | ○ | ○ | ++ |

... helped me to find a better solution *

| | 1 | 2 | 3 | 4 | 5 | |
|---|---|---|---|---|---|---|
| -- | ○ | ○ | ○ | ○ | ○ | ++ |

... helped to find a problem in our solution *

| | 1 | 2 | 3 | 4 | 5 | |
|---|---|---|---|---|---|---|
| -- | ○ | ○ | ○ | ○ | ○ | ++ |

... helped me to understand my team members better *

| | 1 | 2 | 3 | 4 | 5 | |
|---|---|---|---|---|---|---|
| -- | ○ | ○ | ○ | ○ | ○ | ++ |

... helped me to understand the end users better *

| | 1 | 2 | 3 | 4 | 5 | |
|---|---|---|---|---|---|---|
| -- | ○ | ○ | ○ | ○ | ○ | ++ |

I liked about the method... *

Your answer

I disliked about the method... *

Your answer

I could apply this method in the following situations: *

Your answer

I could not apply this method in the following situations ... because ... *

Your answer

## Method Presentation

How easy was it to understand the method? *

| | 1 | 2 | 3 | 4 | 5 | |
|---|---|---|---|---|---|---|
| Hard | ○ | ○ | ○ | ○ | ○ | Easy |

How did you like the way of presenting the method? *

Your answer

## Thank you for your time

Don't forget, as this is an academic work, I rely on honest answers. Thank you :)

SUBMIT

Never submit passwords through Google Forms.

## A.5. Core Group Interview Guideline

**What do you think about the empathy for your team members?**

Do you always know what your team members *think*, *feel* or *want*?
- Is there any difference between programmers, designers and "*managers*"?

Do you sometimes wonder how your colleagues *do* things?
- What makes you wonder? Why?

Do you sometimes *hold back* a comment or feedback in discussions?
- Why?

Would you like to have something *improved* about the communication in your team?

**What do you think about the empathy for the end users?**

Do you have a *clear picture* of who is the end user of the software that you are working on?

Can you understand *all* the requirements of the end user for the software?
- If not, *which* requirements are the ones you have most *issues* with?

Do you sometimes feel that a requirement of the customer does not fulfil their *actual need*?

**Do you think you understand the products that you are working on well enough?**

What is the *primary source* of information for you about the product?
- Do you wish to have a more *direct contact* with the end users or customers?

Would you like to be more involved in specific *discussions*, *decisions* or *presentations*?
- Which ones?
- Why?

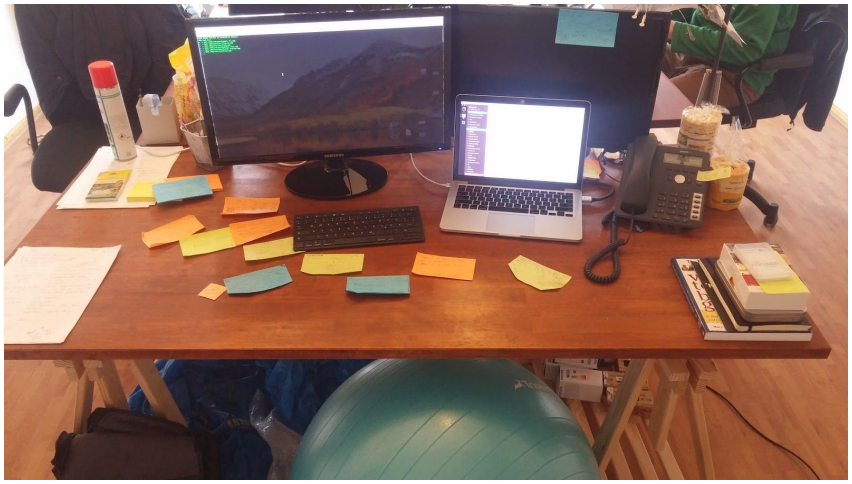
Do you think that you sometimes *lack information* about the product for resolving your tasks?

# B. Toolbox

## B.1. Personal Inventory

# PERSONAL INVENTORY

*Foster **Empathy** for the **Team** and **End User***



**PREREQUISITES**

- Camera, Smartphone is enough
- Screen Recording Tool, or any way of sharing your screen to the group

**TIME**

30-60 minutes

**HOW TO**

Take a photo of your **working space** - this is most probably your desk but also things like whiteboard and kitchen (e.g. coffee machine) count. The point is to show your daily working routine including any hardware that you use for working. Annotate the photo as good as you can and present it later to the group.

Describe your **working stack** - take screenshots of software and tools that you use and create a short screen recording of your typical workflow. This step can also be done via a live presentation. Whatever you choose - the result should be presented to the group.

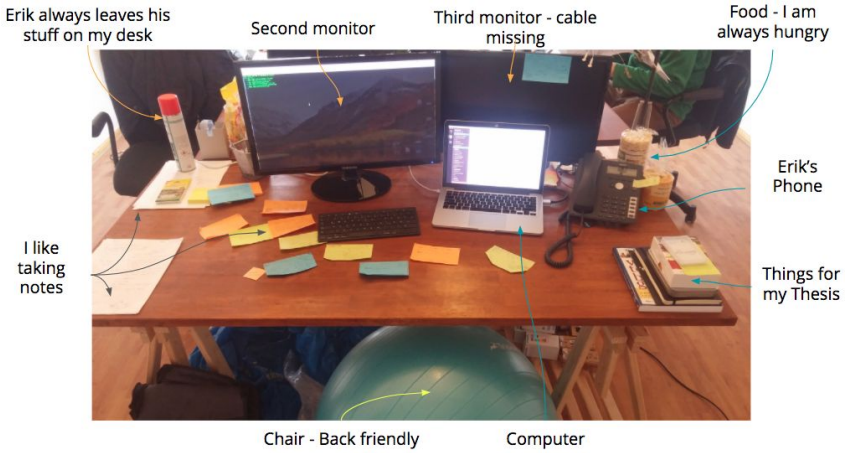
Ask your **customer** to send you photos of their **working space** - this might be a desk, but also a production hall or a store. If possible, let them describe it to get hidden insights.

**EXAMPLE**

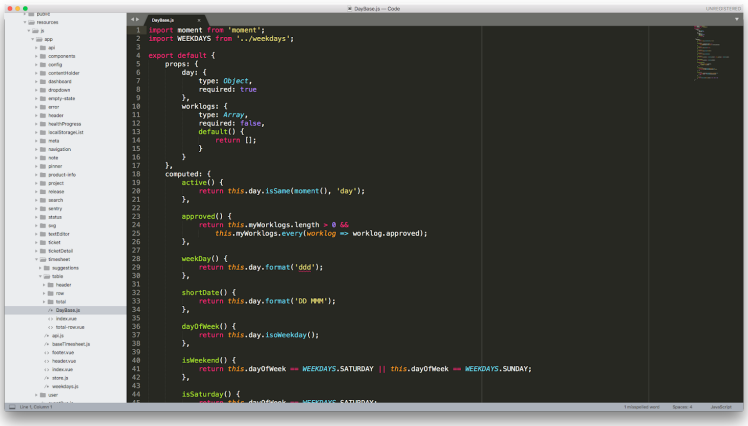
On the next three pages you will find my working space and my working stack described in pictures with a bit of text. Try to show the things that are special to your workflow.

## PERSONAL INVENTORY

*Foster **Empathy** for the **Team** and **End User***



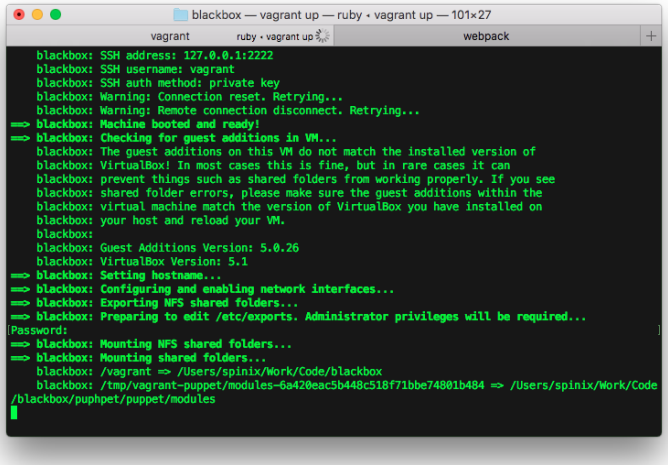

This is my desk with some annotations of the things you can see.



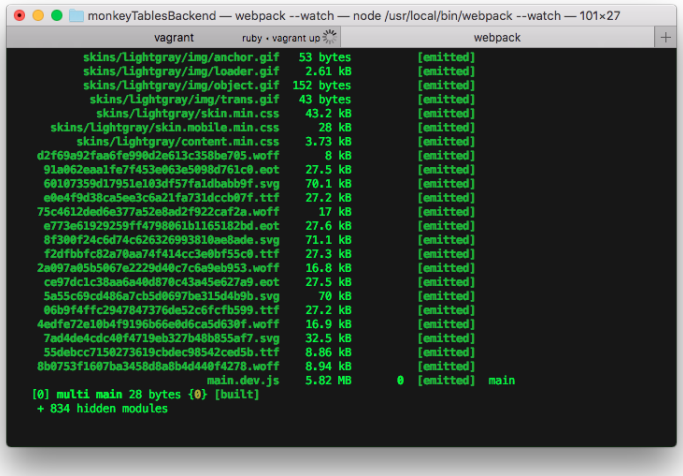
My primary code editor is sublime, with some useful plugins like code style formatter, auto-complete and ruler.

## PERSONAL INVENTORY

*Foster **Empathy** for the **Team** and **End User***



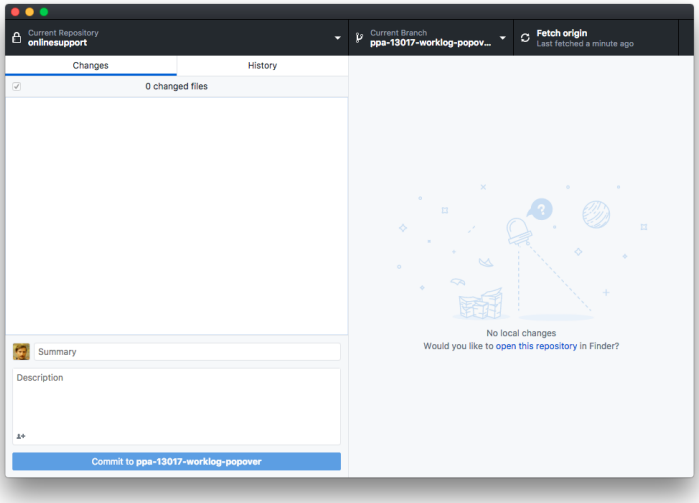

To run my development code I use a virtual machine that automatically sets up my projects and imports databases if needed.



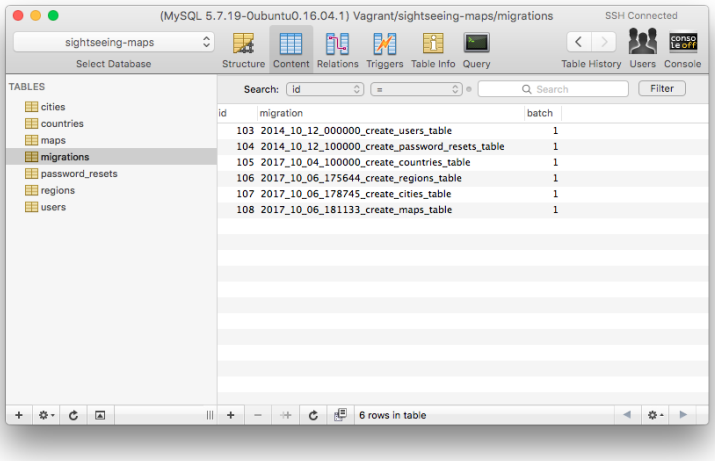

Some projects need a task runner like webpack to constantly compile my code

## PERSONAL INVENTORY

*Foster **Empathy** for the **Team** and **End User***




To commit my code I use the GitHub Desktop application which allows me to do all the necessary steps and can even open a new pull request by using a shortcut.

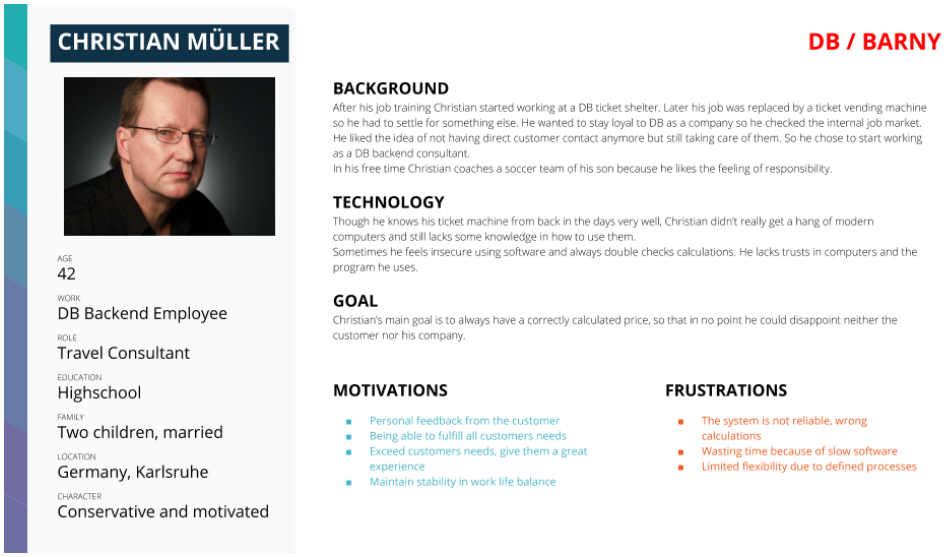



In case any direct access to the database is needed, I use Sequel Pro for it. Favorites allow me to access my local systems and by using ssh certificates also for test and production systems.

## B.2. Character Profiles

# CHARACTER PROFILES

*Foster **Empathy** for the **End User***




**PREREQUISITES**

- Character Profiles for your End Users

**TIME**

5-20 minutes

**HOW TO**

Character Profiles, also called Personas, are fictional Characters, that are based on real data and research about your end users. They are created by observing and interviewing your target group and then bringing to life a representative of an end user. A Character Profile consists of demographic information about the persona and a background roughly describing the context of the person. Furthermore there are four further sections that describe the persona in relation to the product or software that is being developed.

The **technology** section describes the general affinity towards technical devices that might be used to control the software, their abilities in handling them and also personal preferences.
The Persona's **goal** describes what she or he would like to accomplish by using the software or how it could help to reach it. This is not necessarily limited to the actual process but can also involve feelings while using a product.

# CHARACTER PROFILES

*Foster **Empathy** for the **End User***

To achieve the goal **motivations** are listed, that describe not what the persona wants to achieve, but why.
**Frustrations** on the other hand can be related to existing attempts to reach the goal or more general fears.

Use Character Profiles in **any situation**, to ensure that the product you are building is having enough empathy towards your end users. Especially use it when in doubt or when stuck - it will help you to **formulate the right questions** and **answer these with your end users in mind**.

**EXAMPLE**
A Character Profile should be treated like a personified example of your end users. With this in mind there are endless possibilities on how to use this. Some examples include:

**ANALYSE** Find out how a persona would use your software
*"How would Christian experience, react and behave in relation to feature X?"*

**DESCRIBE** Spice up your user stories with personas
*"As a student like Julia I want to see an overview of my courses."*

**REJECT** Realize early if you are developing the wrong feature
*"Does feature Y help Dr. Rath to reach his goal?"*

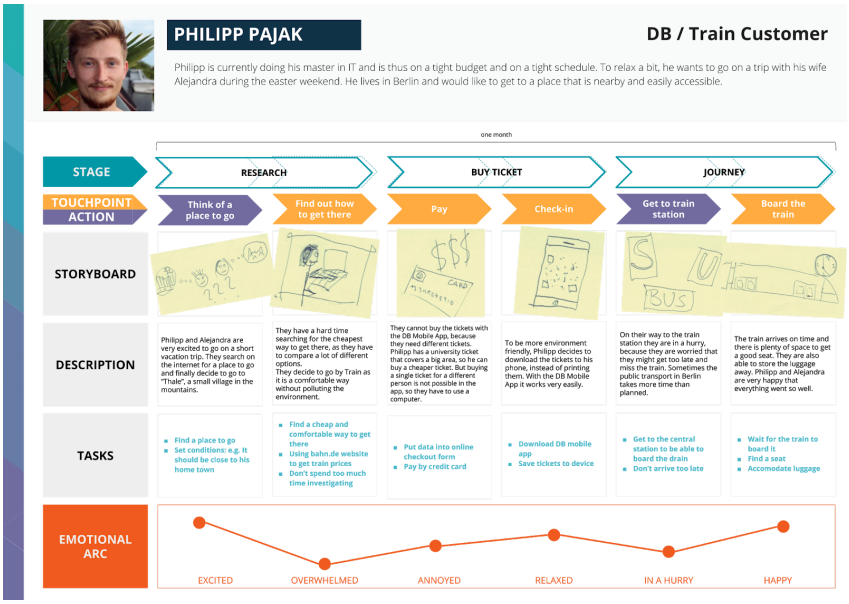
**IDEATE** Come up with new features that can fulfill your users needs
*"How could we help Frau Hansen with her frustrations?"*

**TWEAK** Adjust your features so they have more value for your user
*"How should we design feature Z so that Dr. Martin Cent can use it with his technological background?"*

## B.3. Customer Journey Map

# CUSTOMER JOURNEY MAP

*Foster **Product Understanding** and **Empathy** for the **End User***




**PREREQUESITES**

- Diverse team of 3-5 people
- Post-Its, pens, tape, coffee, ...
- Knowledge about a project
- Customer and end user contact is very useful
- At least one Character Profile
- *Optional* to get started: Customer Journey Map template

**TIME**

2-4 hours

**WHAT**

A Customer Journey Map basically describes the journey that your customer takes in order to consume your product or service on a **timeline**. You define all the steps that the customer takes - even the ones that do not have direct touchpoints with you. E.g. the journey could start by having a need, over finding a product that would fulfill this need and ending up in consuming your product or service and finally recommending it. The gathered data is then **visualized** and propagated by **storytelling**.

# CUSTOMER JOURNEY MAP

*Foster **Product Understanding** and **Empathy** for the **End User***

**WHY**

The Customer Journey Map can show you the experiences of the customer with the **point of view** of the customer. Additionally it allows you to connect this customer perspective with all the touchpoints that exist with your product and service. It can be used to find gaps in customer experience and explore potential solutions, too.
For internal communication it can be used to visualize data about the entire customer experience and foster product understanding within the whole team.

**HOW TO**

There is more than one correct way of creating a Customer Journey Map - it is a creative and complex process and as such creating it will be always different. Still one can be guided through it by taking the following actions.

**RESEARCH** Choose a Character Profile as a basis that will represent the persona whose journey you would like to explore. Identify the needs, pains and goals in the whole life-cycle and validate your assumptions by conducting interviews or doing internet research on social media and search engines.

**STAGES AND TOUCHPOINTS** Identify **stages** as groupings of touchpoints and other actions that your customer would take. A stage could be for example “planning” or “sharing”. For each stage start with the main **touchpoint** and try to refine it by adding steps before and after. A touchpoint is anything that connects your service and your customer: advertisement, an e-mail, your product, etc. Sometimes you also have to add **actions** that are not directly associated with your service.

**COMPLETE STEPS** Make sure that you have included all necessary touchpoints - feel free to break up or merge touchpoints and actions, so that you have a consistent and detailed experience. Always keep in mind to take the **point of view** of the customer and don’t force any established processes of your service.

**JOURNEY** Add life to your map, by adding a **storyboard** to each touchpoint or action. A storyboard is a visual representation of your step by using sketches, photos or any other material available. A short **description** should make clear what is happening during that step and additionally **tasks or goals** can be listed.
As a last step an **emotional journey** graph is used to represent the mood of our persona. For each touchpoint estimate the emotions of the customer and give it a grade from 1 to 10 - a higher number means the customer felt better. Additionally write down a word that would describe that emotion and finally connect all emotional journey grades throughout all steps to create a graph.

As with Design Thinking in general, this is an iterative process and you can obviously jump from step to step forth and back. Feel free to use the attached template or use any other type of presentation if you feel like it.

## CUSTOMER JOURNEY MAP

*Foster **Product Understanding** and **Empathy** for the **End User***

**EXAMPLE**

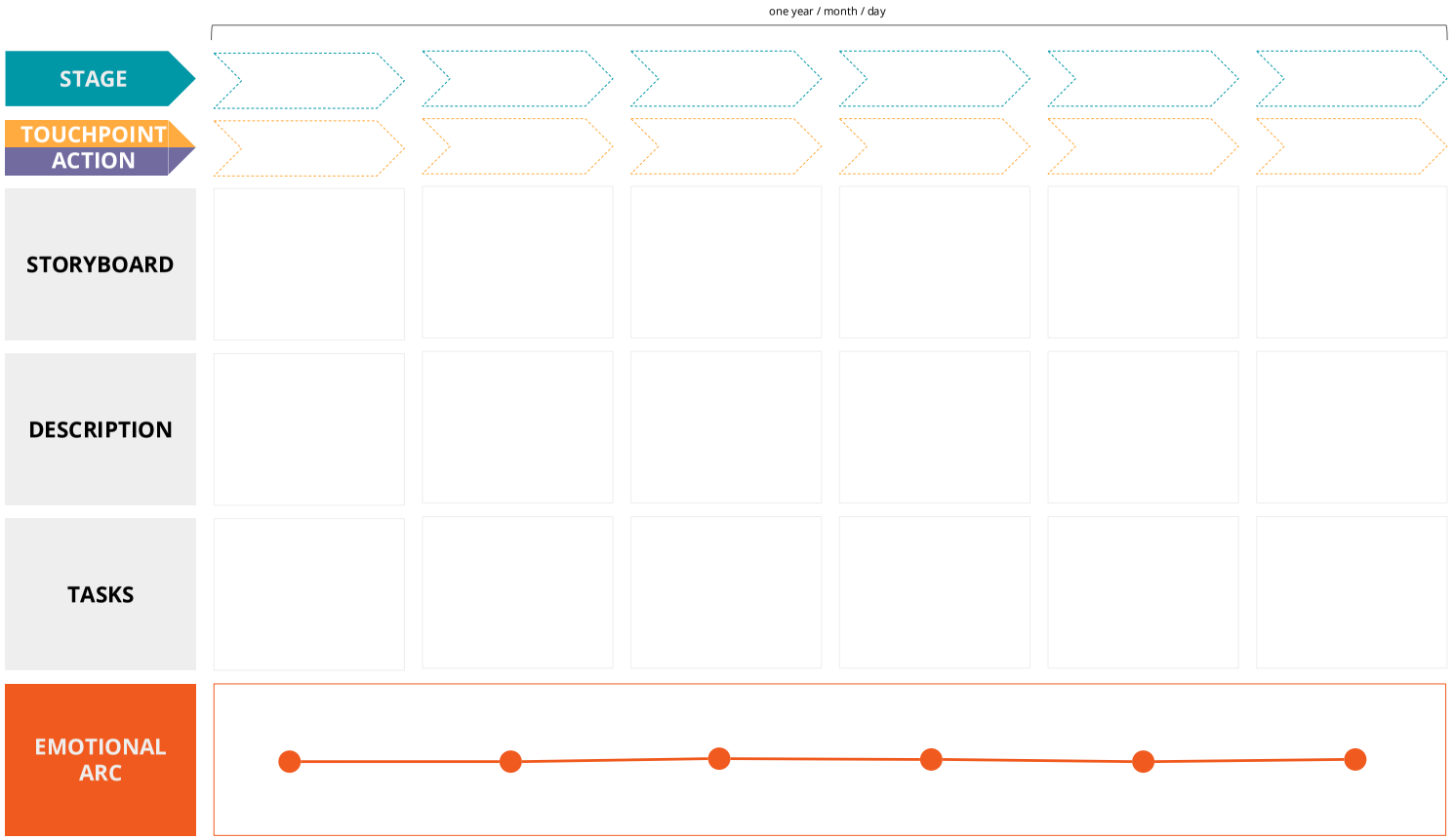
Please see the attached Customer Journey Map. It describes a customer experience from the point of view of a customer. Depending on the scope of the customer journey map to be made, the grade of detail can be adjusted compared to the example.

### B.3.1. Template and Example

## CHRISTIAN MÜLLER

**DB / BARNY**

After his job training Christian started working at a DB ticket shelter. Later his job was replaced by a ticket vending machine so he had to settle for something else. He wanted to stay loyal to DB as a company so he checked the internal job market. He liked the idea of not having direct customer contact anymore but still taking care of them. So he chose to start working as a DB backend consultant.



## PHILIPP PAJAK

**DB / Train Customer**

Philipp is currently doing his master in IT and is thus on a tight budget and on a tight schedule. To relax a bit, he wants to go on a trip with his wife Alejandra during the easter weekend. He lives in Berlin and would like to get to a place that is nearby and easily accessible.

one month

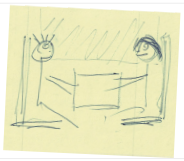
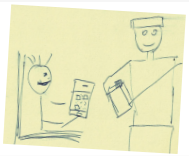
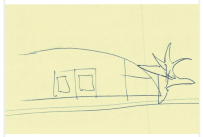
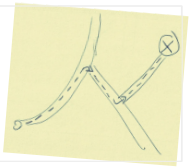
| | | | | | | |
|---|---|---|---|---|---|---|
| **STAGE** | RESEARCH | | BUY TICKET | | JOURNEY | |
| **TOUCHPOINT ACTION** | Think of a place to go | Find out how to get there | Pay | Check-in | Get to train station | Board the train |
| **STORYBOARD** |  | | | | | |
| **DESCRIPTION** | Philipp and Alejandra are very excited to go on a short vacation trip. They search on the internet for a place to go and finally decide to go to "Thale", a small village in the mountains. | They have a hard time searching for the cheapest way to get there, as they have to compare a lot of different options. They decide to go by Train as it is a comfortable way without polluting the environment. | They cannot buy the tickets with the DB Mobile App, because they need different tickets. Philipp has a university ticket that covers a big area, so he can buy a cheaper ticket. But buying a single ticket for a different person is not possible in the app, so they have to use a computer. | To be more environment friendly, Philipp decides to download the tickets to his phone, instead of printing them. With the DB Mobile App it works very easily. | On their way to the train station they are in a hurry, because they are worried that they might get too late and miss the train. Sometimes the public transport in Berlin takes more time than planned. | The train arrives on time and there is plenty of space to get a good seat. They are also able to store the luggage away. Philipp and Alejandra are very happy that everything went so well. |
| **TASKS** | ▪ Find a place to go<br>▪ Set conditions: e.g. It should be close to his home town | ▪ Find a cheap and comfortable way to get there<br>▪ Using bahn.de website to get train prices<br>▪ Don't spend too much time investigating | ▪ Put data into online checkout form<br>▪ Pay by credit card | ▪ Download DB mobile app<br>▪ Save tickets to device | ▪ Get to the central station to be able to board the drain<br>▪ Don't arrive too late | ▪ Wait for the train to board it<br>▪ Find a seat<br>▪ Accomodate luggage |
| **EMOTIONAL ARC** | EXCITED | OVERWHELMED | ANNOYED | RELAXED | IN A HURRY | HAPPY |

## PHILIPP PAJAK

**DB / Train Customer**

Philipp is currently doing his master in IT and is thus on a tight budget and on a tight schedule. To relax a bit, he wants to go on a trip with his wife during the easter weekend. He lives in Berlin and would like to get to a place that is nearby and easily accessible.

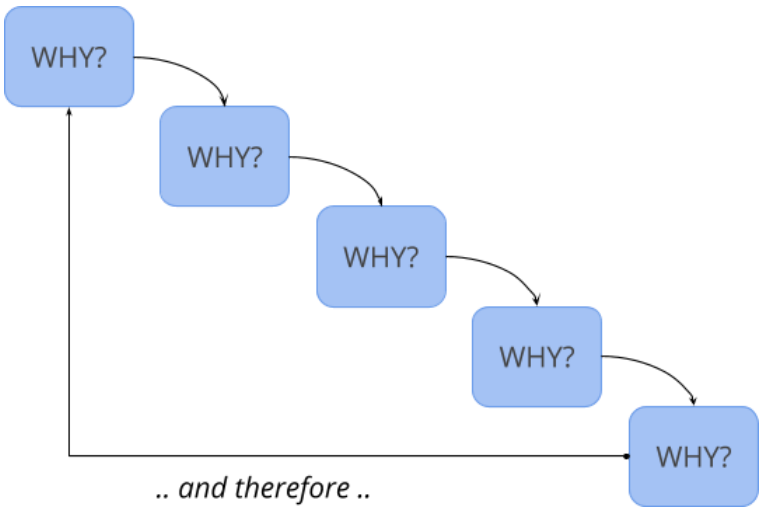
one month

| | | | | | | |
|---|---|---|---|---|---|---|
| **STAGE** | | | JOURNEY | | | RECOMMEND |
| **TOUCHPOINT ACTION** | Ride the train | Ticket Control | Wait because of an accident | Arrive to destination | Get to hotel | Tell friends about journey |
| **STORYBOARD** |  |  |  |   |  |  |
| **DESCRIPTION** | The train ride is just fine. Philipp is a bit bored, because there is no entertainment system on the train. But he likes the view out of the window. | Philipp is annoyed a bit by the train conductor because he needs to search for the tickets on his phone to prove that he has a valid ticket. | The train has to stop because another train had an accident. They have to make space for those passengers, as they join the train. The train is now full of people. After one hour the train can continue with the journey. | Finally arrived at the destination, Philipp and his wife need to pack their stuff and get out of the train. They are relieved to finally reach their destination. | To get to the hotel Philipp needs to find out where it is and finally they walk 500 meters with the luggage. They are both tired of the trip. | After arriving and drinking a cup of tea, Philipp tells his friends and family about the curious accident that happened during the train ride. As he could relax in the meantime he is now very excited for his vacation. |
| **TASKS** | ▪ Be patient for three and a half hours | ▪ Hold ticket ready to show it to the train conductor | ▪ Make space for rescued passengers | ▪ Gather all luggage<br>▪ Leave train at correct station | ▪ Find hotel on a internet base map service<br>▪ Walk 500 meters with the luggage to the hotel | ▪ Write message to parents<br>▪ Tell friends what happened on the journey |
| **EMOTIONAL ARC** | A BIT BORED | A BIT BOTHERED | ANXIOUS | RELIEVED | TIRED | EXCITED |

## B.4. Five Whys

# FIVE WHYS?

*Foster **Communication***




**PREREQUESITES**

- A problem or an unclear statement
- At least one conversation partner
- Enough “Whys” :)

**TIME**

5-15min

**WHAT**

Five Whys is a simple tool, to find out the **root cause** of a problem, to dig deep into your end-users or to **uncover** useful **insights**.
It was invented by Toyoda Sakichi to explore the cause-and-effect relationships in industrial manufacturing, but in general it is applicable to any field.

**WHY**

In most cases root causes are **hidden** behind symptoms, and the first reaction is to discover those symptoms. As symptoms are caused by further causes, fighting the symptoms can only give you short-lived results, as the root cause still exists.
Besides, also with information based problems you need to take care of finding out the root cause: For example a specific customer desire might be based on a **root desire** that could be solved in a much more efficient way.

## FIVE WHYS?

*Foster **Communication***

A lot of times when asking something the answers are not clear, or not complete. So to know that your work is based on correct assumptions, asking and digging deeper will allow you to better understand your conversation partner.

### HOW TO

In its simplest form, you would repeatedly ask "**Why?**" until you are satisfied with the answer. Still there are some rules to ensure that you get the most out of the method.

**FIND THE ROOT CAUSE** Don't stop, until you determine the root cause - the number of iterations is flexible and doesn't necessarily have to be five, as it depends on how complex the actual topic is. Sometimes asking three times might be enough, but in other cases you might need to dig much deeper. But be sure to take it step by step and don't be tempted to jump directly into conclusions.

**ENSURE YOU FOUND THE ROOT CAUSE** Take great care to distinguish symptoms and root causes - the latter can be verified by turning around the answer and saying: "[***root cause***] and therefore [***symptom***]".

**DON'T BLAME** This exercise should be used to find out causes and insights - not trying to blame anybody. People will give you less information once they know that you are looking for somebody to blame.

**DON'T FEEL STUPID** At the first sight it might look strange to repeatedly ask "Why?". You should incorporate a part of the last answer in your next question, so you don't lose track of information and show your conversation partner that you are picking up the conversation. You can also warn your conversation partner about the technique you are using, to make sure that nobody feels interrogated.

### EXAMPLE

Five Whys can be used in one-to-one conversation as well as in big groups and in many different situations. These two simple examples should give you an idea on how you could use this method.

**PROBLEM**
"The production server is down."
"**Why** is the server down?"
"There was a new commit that broke the system."
"**Why** did it break the system?"
"Because the code wasn't checked for a condition that affected another part of the system."
"**Why** was the check for that part of the software omitted?"
"The system is very big and ***there is no testing automation available yet."***

## FIVE WHYS?

*Foster **Communication***

In this example, instead of blaming a specific developer, you can find out faults in your processes to optimize the operations in your team.

**INSIGHT**
"Please add a new button here."
"**Why**?"
"The customer needs a new button here."
"**Why** does the customer need a new button?"
"The end-users cannot see the other button, that is on the bottom."
"**Why** can't they see the button?"
"On iPads the button is not in the visible scrollarea."
"**Why** is that button not in the visible scrollarea on iPads?"
*"**This part of our web page is not responsive yet.**"*

Here the insight allows you to figure out a solution that is more human centered or technically viable.

## B.5. Letter to Grandma

# LETTER TO GRANDMA

*Foster **Communication** and **Product Understanding***


**PREREQUESITES**

- A user story, epic user story, release plan or anything else you would like to communicate and describe
- Pen and paper

**TIME**

15-45min

**WHAT**

"Letter to Grandma" is a method to improve **sharing** and understanding of requirements and tasks. It allows a **knowledge transfer** that focuses on the **essential aspects** without omitting **key information** that seems to be obvious.

**WHY**

When sharing information the **knowledge provider** holds the required information and transfers this knowledge to one or various **knowledge recipients**. The task of **knowledge sharing** requires the knowledge provider to translate the knowledge in a format that allows to encode the information in a way that can be interpreted by the knowledge recipient. The most common format of knowledge transfer is to translate the knowledge into words. A big problem of knowledge transfer lies in this translation, as **tacit knowledge** has no exchange format that would fit all cases. Thus information is lost while encoding and also while decoding information by the recipient.
Writing a letter to grandma helps you to minimize these problems when dealing with requirements or tasks to be shared. The letter could also be addressed to somebody

## LETTER TO GRANDMA

*Foster **Communication** and **Product Understanding***

that has never dealt with your project - or to give an extreme example: an alien. All these recipients have in common, that they have no **prior knowledge** and thus need a description of the essentials enriched with further information that is only known to the team. This last part is actually a very important one: While the knowledge provider might think that some piece of information is already known to the team - or not worth to mention - the recipients may be missing that specific part to be able to complete the knowledge transfer. Concentrating on the essentials makes sure that one sticks to the important parts and does not overwhelm the recipient with unnecessary details, but gives a good overview of the requirements.

**HOW TO**
Write a letter that describes requirements or tasks in a way that you would write to your grandmother.

**EMPATHISE** No matter if you chose to write your grandmother or an alien, make sure that you fully empathise with your recipient. Imagine your recipient reading the letter, would they be able to fully understand it?

**ESSENTIALS** Your recipient has no prior knowledge about the things you want to describe, but as you are writing only a letter, you are also not able to explain everything very detailed. Try to concentrate on the essentials.

**UNDERSTAND** Before starting the letter, make sure that you indeed understand the topic that you are going to write about. If something is still unclear to yourself, solve that gap first.

**PAPER** As with a real letter you should use paper to write down your letter. It forces you to create a concept before starting to write, as you cannot just swap words around as you would on a computer.

**EXAMPLE**
Use this method to describe a product vision, bigger features or sprints. Feel free to use it in other situations, if you think that it will help you to communicate your ideas and product understanding.

Imagine that you are integrating a bitcoin payment system for an online shop.

Dear Grandma,
In my new workplace we are in charge of an online shop, you know those web pages where you can buy things. Until now you can pay in this shop by issuing a bank transfer or using a credit card. But there is now a new trend that is called bitcoin. Did you hear about that? It is like money, but everything is virtual. So you don’t have real coins and no real wallet. All your money is only in the computer and you can access it with your wallet

## LETTER TO GRANDMA

*Foster **Communication** and **Product Understanding***

number, this is like a bank account number, and a long password. So now the shop that I am working with, wants us to add the possibility to pay with bitcoins. Luckily we can use a service called EasyBitcoin that will do all the complicated things for us: The service will take care of receiving the money and later transfer it to our bank account. When the customer decides to pay on our web page, then they will have a third option available. After choosing the bitcoin payment the customer will see the page of the EasyBitcoin service. On that page they will have to enter their wallet number and password. If everything is correct our online shop is displayed again with a message that the payment succeeded. Of course if something goes wrong, they can go back and choose a different way to pay.

Love,
Your Grandson

## B.6. A Beginner's Mind

# A BEGINNER'S MIND

*Foster **Ideation***




**PREREQUESITES**

- A topic to ideate or discuss
- Ideally somebody who is not involved in the project

**TIME**

15-45min

**WHAT**

The general idea for this method comes from **Zen Buddhism** and is called "shoshin", which means "beginner's mind" in English. This state of mind provides an attitude of **openness** and a **lack of preconceptions**. In contrary there also exists the "**expert's mind**" that describes the state when the constraints of the subject are **obvious** and there is only a few, very **clear solutions**.

**WHY**

When getting an expert in a specific field, our way of thinking changes. Mostly we benefit from this change as we can identify problems and find solutions quickly. We help ourselves by developing habits and assumptions that eventually push us to quick conclusions. Humans are good at adapting and eventually everything normalises

## A BEGINNER'S MIND

*Foster **Ideation***

around us - we quickly lose the ability to see the details. Our experiences with the topic include assumptions, perceptions and stereotypes that we take for granted.
By applying a beginner's mind we open ourselves to **unexpected discovery** and innovation because we are not blinded by our prior assumptions and have a **better view of the details** and the bigger picture. In most cases there is more than one way of achieving a specific result - and our way is not necessarily the best. Humans are resistant to change and thus we have a tendency to repeat behaviour, but one of the **alternative paths** might be a better solution, or at least just a better solution for our specific use case. In general uncertainty gives us the courage to **try things out** anyway, regardless whether they will work or not - because we don't know better yet.
Apart from the discussed benefits, with a mind of a beginner you are able to empathise better with other beginners. By listening with an open mind you create trust and a deeper personal connection.

**HOW TO**
To apply this method you can either try to adopt a beginner's mind yourself or ask a real beginner - either somebody outside of your team that ideally has not much knowledge about the project, or you can even go out on the streets and ask somebody - e.g. at lunch. There are some general rules that apply to both cases, as we are always both a student and a teacher.

**QUESTION EVERYTHING** Use the strength of your imagination and question everything, especially those aspects, that you assume to fully understand. Think of a five year old asking tons questions - be truly curious.

**CHALLENGE ASSUMPTIONS** Pretty sure that something is given and can only work this way? Make sure to test it out!

**DEFER JUDGEMENT** Even though your current knowledge, constraints or best practices might conclude that something cannot be done a specific way, still you should try to focus on the possibilities. Any idea, no matter how crazy, can lead to a useful solution.

**EMBRACE FAILURE** Be open to discover something new and step outside of your comfort zone. Don't hesitate to ask "dumb" questions and learn from your failures.

**EXAMPLE**
This technique is mainly used for ideation on a given problem or topic. These can range from implementation details over system architecture to feature reframing. Additionally it can be also used for validation of ideas or prototypes.

#1
You are fixing a bug in your code. You are already three hours on it, and you are completely stuck.

## A BEGINNER'S MIND

*Foster **Ideation***

You could show your code to somebody not familiar with your project, to get ideas on what to check. Maybe something that you assume is working, is actually broken - or you could get ideas on how to better debug your issue.

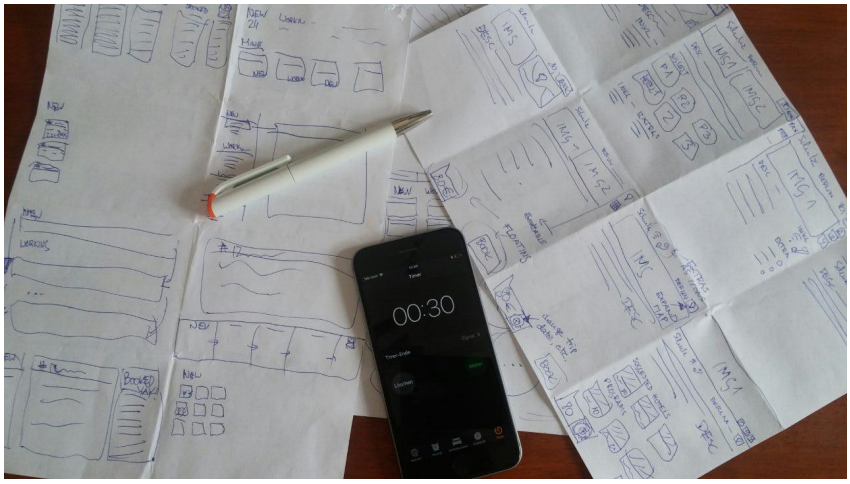
#2
You are developing a new UI feature and your first idea is to copy the layout of a similar feature already existing in the project.
Beginners would tackle the issue differently and propose a different layout because they are not biased with your project knowledge.

#3
You are analyzing a specific data set that you already know for some time. You need to find a new correlation between two attributes.
A beginner might ask some questions that you would take for granted and see details in your data that you already masked out.

## B.7. 30 Seconds Sketch

## 30 SECONDS SKETCH

*Foster **Ideation***




**PREREQUESITES**

- Pen and a piece of paper
- A topic for ideation
- A timer, e.g. from the smartphone
- Optionally a group of people

**TIME**

5-30min

**WHAT**

In the early ideation stage this method is used similarly like brainstorming to generate a lot of ideas in short time, but giving it additionally a visualization. A sketch can be seen as a simple and low fidelity prototype - it is quick and easily disposable.

**WHY**

Sketches in general are a good tool to **express**, **develop** and **communicate ideas**. By **visualizing** your idea you make it much more tangible for your peers which enables you to gather better critique and feedback on them. It is easy to abandon an idea from a sketch because of the tiny effort one does not get too attached to it. This is important to be able to **explore different concepts** and thus seek for alternatives by thinking through your ideas. In the end a sketch can suggest new ideas, but should not limit them.

If we take it further now, and limit our sketching exercise to 30 seconds per sketch, we automatically would enforce some of the important rules of sketching and loosen the

## 30 SECONDS SKETCH

*Foster **Ideation***

leash of our creativity even further. By limiting the time, you just cannot worry about quality and are able to generate tons of solution attempts in a short time. By focusing on **quantity** and **iteratively** producing more and more sketches, one is able to explore the **problem** and **solution space**. Exploring in this case means to experience what would be basically possible - without really caring whether the solution is viable or useful.
Time pressure on the other hand also allows you to easier identify **critical details**, which are the ones that you need to take special care of, as they are crucial for the feature that is currently worked on.

**HOW TO**
Grab your timer, set it to 30 seconds and start sketching your idea on a piece of paper. Depending on the phase of development, you would need to adjust how deep you want to dive into the solution space. You can perform this exercise with multiple people and also iteratively draw multiple sketches, until you gather enough ideas.

**EXPLORE** You shouldn't care whether the idea is realizable or even worth sketching - sketches should be seen as "mini ideas" that are "half baked" to explore the solution space.

**NO NEED FOR PICASSO** Sketches are not meant to win any beauty contest, but they should allow you to record ideas. Don't be afraid of your drawing skills, simple and easy drawings are completely OK.

**FOCUS ON THE ESSENCE** Try to focus on the essence of your idea and don't get distracted too much by details. Only including minimal detail helps you to concentrate on the core idea and problem.

**INDEPENDENT** This exercise should be performed silently and independently, so that one is able to express his creativity without relying too much on your team mates ideas.

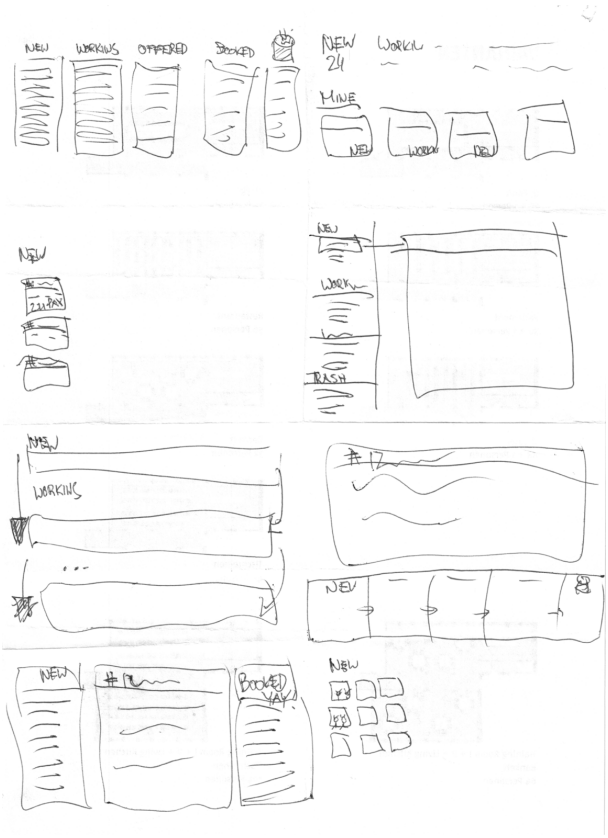
An interesting variation of this method is called **Crazy 8**: You have to draw 8 sketches in five minutes. Why eight sketches? You can simply take a sheet of paper and fold it three times in half so that you have eight fields to fill out.

**EXAMPLE**
In most cases the idea that you want to sketch would be a user interface, but also other things like user experience touchpoints, logos or even hardware designs can be sketched.

The first example is a sketch of a new dashboard for a process oriented application that handles all the details of booking a group trip.

## 30 SECONDS SKETCH

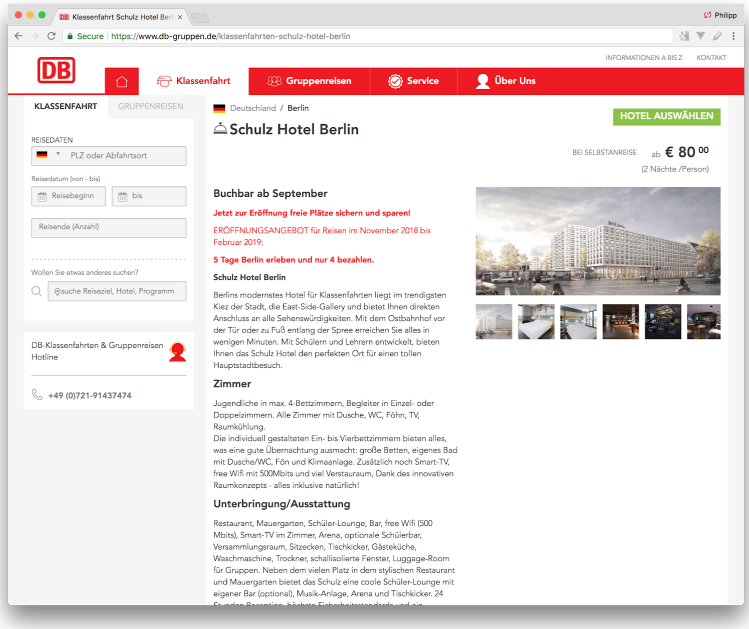
*Foster **Ideation***



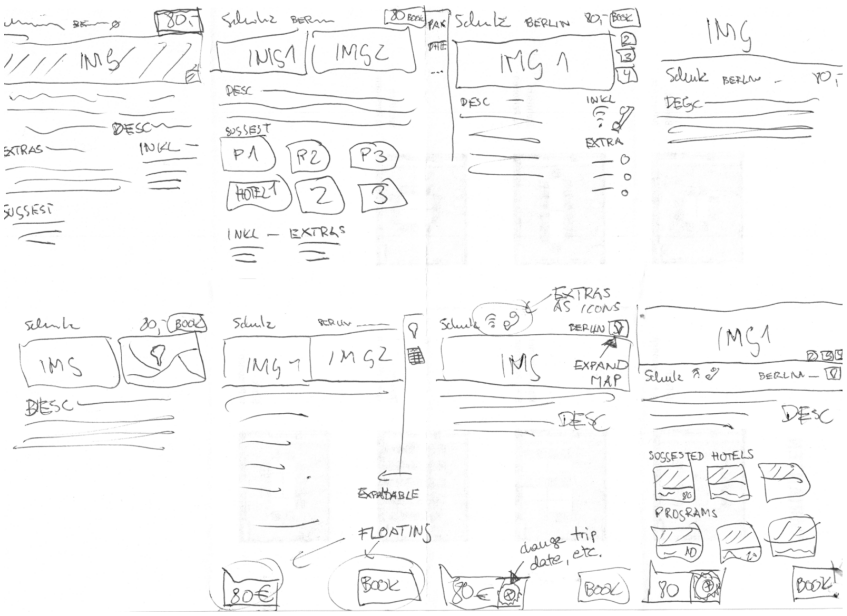

In the second example an existing web page will be taken as a base to find out how a mobile version might look like.

## 30 SECONDS SKETCH

*Foster **Ideation***




https://www.db-gruppen.de/klassenfahrten-schulz-hotel-berlin



## 30 SECONDS SKETCH

*Foster **Ideation***

You can clearly see in both examples how ideas are evolving during the sketches. While the first sketch of a group is very simple, subsequent sketches involve new elements. Later those elements are then slightly changed or mixed with other new elements.

The second example already contains quite a few interesting ideas, but the first example could take more iterations to explore more potential solutions. Sharing those sketches among your team members can help to develop even more ideas.

## B.8. Power's of Ten

## POWERS OF TEN

*Foster **Ideation***


**PREREQUESITES**

- Mathematical knowledge about powers

**TIME**

5-50min

**WHAT**

The method Powers of Ten is a reframing technique based on the short film of the same name from the year 1977. The movie shows a picnic scene in Chicago with different magnitudes of distance from 1 meter up to $10^{24}$ meters and then down to $10^{-16}$ meters. It takes you up into the deep universe and then back into the hand of the man that is at the picnic until you only see atoms.

**WHY**

The first interesting thing about Powers of Ten is that you don't have linear, but **exponential growth**. For humans it is hard to think exponentially because our brains are trained to think linearly. But a lot of things actually grow exponentially.
The second thing is changing the **point of view** - you can look at a problem from very different magnitudes of a certain dimension. Varying the point of view allows you to find the **right framing** for different issues of this problem - basically by zooming in and out the current scene.
When working on a virtual or design problem one needs to construct the whole model in one's mind to be able to work on it. This is a very complex process where Powers of

## POWERS OF TEN

*Foster **Ideation***

Ten can help by first discovering the whole problem space with different magnitudes and then finding and **developing patterns**.

**HOW TO**
Powers of Ten is an exercise that allows to change the point of view by varying magnitudes of context. You can start at $10^0$ and then work your way up and down.

**FIND THE RIGHT SCALE** The original short film is working with magnitudes of distance, but feel free to use any other scale that might fit in. Apart from the obvious ones like 'distance', 'size' and 'time' you could use others like 'cost', 'complexity' or even something abstract like 'reward points'.

**MOVE IN AND OUT** Step back and try to really understand how the problem that you are working on would fit into its context. Remember to move in Powers of Ten, so always add or remove a zero at the end.

**DETECT PATTERNS** By moving through the magnitudes of your scale try to find patterns that repeat at different magnitudes and try to group the information properly.

**EXAMPLE**
You can apply powers of ten in problem solving, insight development and ideation.

#1
In problem solving you can take the problem and reframe it in other magnitudes of context - try to define it in as many magnitudes as you can to get a better overview of your problem.

Imagine you get assigned a new ticket to fix a specific bug. It can help a lot to view the different issues of the problem by putting them into the defined scale.

Example scale to use:

> Product Experience > Product > Sprint > Programming Language > Framework > Software Architecture > Module > Class > Method > Block > Line of Code > Expression > Character > Bit > Electricity > Atom

Now while fixing the problem you can try to see it from different point of views, e.g.:

- How does the error appear in the product?
- In which sprint was the feature developed that is now broken?
- Does the framework help us to bypass the error?
- In which class is the bug?
- In which function?
- Is there maybe a 'hidden character' that breaks the code?

## POWERS OF TEN

*Foster **Ideation***

#2
In insight development you can empathise with the user through different magnitudes of context.

Imagine you are designing a checkout experience. For developing new insights, you can try to think what happens if the user buys products of different values.

Example scale to use:

House (100000$) > Car (10000$) > Laptop (1000$) > Shoes (100$) > Pizza (10$) > Soft Drink (1$)

#3
In an ideation session creativity sometimes gets stuck and you are out of ideas. You can try to view the context from different magnitudes to explore the solution space or restrict your ideas in a certain magnitude of context.

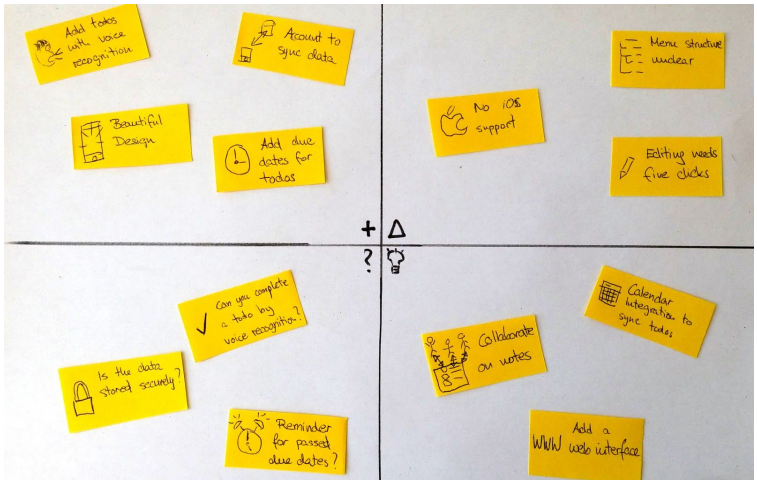
Imagine you are a developing a new feature for a software. You could put specific constraints on your ideation to facilitate the flow of new ideas, e.g.:

- Endless time for development / Must be ready in one hour
- It can run on a supercomputer / It has to run on a low power computer
- The user has a whole day to complete the task / The user just has 10 milliseconds
- The UI is a big wall / There is no visible UI at all

## B.9. Feedback Capture Grid

# FEEDBACK CAPTURE GRID

*Foster **Feedback***




**PREREQUESITES**

- A surface to draw on, e.g. paper, whiteboard, etc.
- Pens
- Post-it notes

**TIME**

5-25 min

**WHAT**

The Feedback Capture Grid is a method to gather feedback in a **structured grid**, that is documented along the way. It is based on **four quadrants** labeled with easy to understand **symbols**, that are used as categories for the feedback.

**WHY**

This method helps you to be systematic about feedback and forces you to think about **different aspects**. It is especially useful for **written**, **prepared** or **formal** feedback. Because this method is performed as a written exercise, it **facilitates documentation** by capturing the results e.g. in a photo.

For more general information on feedback, please check the attached "***General Feedback Tips***".

# FEEDBACK CAPTURE GRID

*Foster **Feedback***

**HOW TO**
In general there are two ways to use this method, for either gathering on-demand feedback or unpacking feedback internally after an external testing session. The procedure is basically the same, just that in the latter case the feedback givers don't present their own feedback but rather the feedback from the attendees of the external testing session.
One could add a third option when collecting feedback with a bigger group: The Feedback Capture Grid is done as a poster - Post-its notes and pens are given to the audience and at the end the audience is then asked to leave the feedback inside the grid.

**PREPARE** The method itself is very simple - just take a rather big piece of paper, or any other surface that you can write on, e.g. a whiteboard. Then divide it into four quadrants and label them with simple symbols:

| | |
|---|---|
| [ + ] **LIKES**<br>*positive feedback* | [ Δ ] **CRITICISMS**<br>*constructive negative feedback* |
| [ ? ] **QUESTIONS**<br>*questions about the experience* | [ 💡 ] **IDEAS**<br>*ideas that sparked during the presentation* |

**USE** As a moderator you can explain your team the basics of this method, if needed. The actual feedback giving is divided into two parts:

1. A quick solo brainstorming session to fill Post-it notes with feedback
    a. Every feedback point is written on one Post-it note
    b. Add a simple drawing to highlight your point
    c. Put your Post-it notes into the right quadrant
2. Presentation of the Post-it notes
    a. Either while putting them on the board
    b. Or at the end, everybody presents their Post-it notes

As a feedback giver, always try to give feedback for every quadrant - and as a moderator make sure that every quadrant receives at least a few notes. The moderator can try to steer the conversation into a certain direction if more input is needed.

During the presentation of the Post-it notes a discussion might be useful, to

a. Get more details about the feedback point, e.g. if unclear
b. Build upon ideas of others to further enrich the feedback, e.g. add solution ideas for mentioned critique

Of course these new points can be written on Post-it notes and added to the grid.

## FEEDBACK CAPTURE GRID

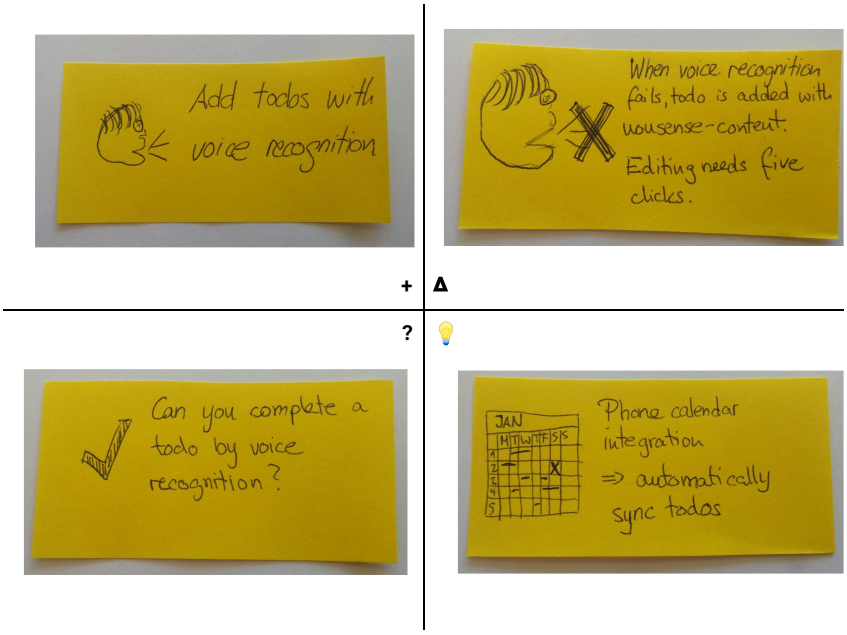
*Foster **Feedback***

**FOLLOW-UP** After gathering the feedback make sure to share it with your team members, e.g. by taking a picture of the feedback and sending it to your peers. Take care of open questions and critique that was raised, but on the other hand also consider positive points and ideas for future development.
The next step then would be to iterate and create the next prototype for gathering more feedback.

You can do this exercise live on a whiteboard or pinboard with Post-it notes or for distributed teams virtually on drawing tools like “Invision Freehand” or “Awwapp”.

**EXAMPLE**
This feedback method can be used anytime to provide feedback on any topic.

Imagine issuing feedback on a todo application, one person might add the following feedback notes:



## B.10. Five Finger Feedback

# FIVE FINGER FEEDBACK

*Foster **Feedback***

**PREREQUESITES**

- A topic that needs feedback
- A group of people that is willing to provide feedback

**TIME**

5-25min

**WHAT**

Gathering feedback should be simplified by the Five Finger Feedback method. The method is assigning each finger a **specific meaning** and those associations are then used to give **structured** feedback.

**WHY**

To make the feedback experience more pleasant and useful, different methods exist to gather feedback. Five Finger Feedback is especially useful for **fast**, **ondemand** or **informal** feedback. The gathered feedback is structured and the exercise can be performed written or oral.

For more general information on feedback, please check the attached "***General Feedback Tips***"

# FIVE FINGER FEEDBACK

*Foster **Feedback***

**HOW TO**
Every finger is associated with a type of feedback that should be given. Please be aware that in some cultures the meanings of the gestures could be different.

**THUMB** As showing somebody a thumbs up, you should tell what you really liked.

**INDEX FINGER** Tell what you have noticed or what you would like to point out. Maybe you are worried about a specific point that you would like to mention.

**MIDDLE FINGER** While this finger is meant to give the negative feedback, describing what was not so good or what could be done better, it is important to stay objective to not hurt anybody.

**RING FINGER** Show your connection or emotional part of the feedback. Tell what you would like to take away and what you would like to keep. Maybe you saw something interesting during the feedback session that you would like to take away.

**LITTLE FINGER** The shortest finger should remind us of what did not get enough attention - or simply something that was missing from your perspective.

There are multiple ways in which you can gather the five finger feedback. You can draw one big hand and let your users write something for each finger, you could tell them to get a sheet of paper and draw the hand themselves and fill it out, or you could just ask for the feedback verbally. The important thing is that every participant provides feedback for all fingers.
This method is particularly good for small groups, but it can be also used well in bigger groups if the feedback is written down.

**EXAMPLE**
This feedback method can be used anytime to provide feedback on any topic.

Imagine feedback on a todo application:
**THUMB** *I liked the possibility to add todos without typing, just by using voice recognition.*
**INDEX FINGER** *I wanted to point out, that the todo date is only recognized when it is spoken at the end of the new todo note.*
**MIDDLE FINGER** *When the voice recognition fails, the todo is added with nonsense-content anyways. Editing the note is only possible by typing and you need five clicks to get there.*
**RING FINGER** *I didn't know that adding support for voice recognition can be that easy, I have noted down the name of the library so I can use it in my projects.*
**LITTLE FINGER** *I was missing an integration with the phone calendar, so the created todos can be automatically synced.*

## B.11. Empathy Tools

# EMPATHY TOOLS

*Foster **Prototyping***

**PREREQUESITES**

- Know your end users
- Something to test
- Creativity

**TIME**

5-25 min

**WHAT**

Empathy Tools is a term used to describe **physical objects** and **cognitive or social techniques** that provide designers with a feeling and deeper understanding of users with **various abilities**. With **inclusive design** it is possible to gain more empathy and understand better other user's capabilities, by experiencing a product in a way that an end user with certain disabilities would.
It can be used to better understand target user's needs and context in the research phase, but also in the prototyping phase it can help to ensure that the product meets the needs of users with various abilities while testing.

**WHY**

Empathy Tools are used to develop an "**informed intuition**" for users with special conditions. Instead of just imagining the differing conditions it helps a lot to actually live them by using tools that allow you to experience a similar condition, as it is difficult to empathise with users that have different abilities or live in a different personal or social situation.

# EMPATHY TOOLS

*Foster **Prototyping***

Especially in the field of software development it is crucial to see the differing abilities in using technical devices: While the team developing the software has mostly very advanced and from most non-developers differing abilities, their end-users mostly just have basic abilities. And people with minimal or almost no technical abilities are still common. Here empathy tools can serve as a good countermeasure to develop more human centered software by gaining insights from these extreme users.
Finally when thinking about how well something can be used we should also focus on how inappropriate design can even further disable some people from using your product.

**HOW TO**
Empathy Tools is an informal method that has no defined process to be taken. In general first one has to think about the abilities and special conditions that the end users might have and then try to find a good way to simulate those.

**RESEARCH** Do some user research to find out which special conditions you need to take into account.

**DOMAIN** Do some further research on the domain to understand the specific situation of your users - e.g. there are different levels of poor eyesight like seeing a bit blurry, not seeing colors or not seeing at all.

**FOREKNOWLEDGE** The team has developed the product so it has specific knowledge that the real end user will not have - e.g. a blind user never had seen your product before.

**SIMULATION** Think of how you could simulate the special condition. The possibilities are nearly endless from simple physical objects (blindfold), specialized physical objects (that can simulate a specific eye weakness) to cognitive and social techniques (count backwards from 100 in steps of seven to simulate a bad mental condition).

**DOCUMENTATION** Have an observer taking notes or take a video of your experience. After finishing the simulation take some time for a period of reflection and taking your own notes.

Remember, although this exercise is a good way to empathise with extreme users, it does not replace speaking to real users!

**EXAMPLE**
Empathy Tools are simple, quick and inexpensive ways of empathetic testing and should be used in any situation where applicable.

# EMPATHY TOOLS

*Foster **Prototyping***

In software development Empathy Tools can be used in very different ways depending on the target device and target user group. This list has some possible topics and examples and should help to find cases for using Empathy Tools. Simulating the experience in most cases is very straightforward, in others some more creativity might be needed.

**TECHNICAL ABILITIES**

- Ten vs. two vs. one finger typing
- Scrolling using the scrollbar, not the mouse wheel - or even not knowing it is possible to scroll
- Advanced operations
    - Software asking you to: “empty your local storage”, “install driver xy”, “format your disk”, etc.
- Understand more complex UI
    - Open on hover or uncollapse by clicking on an icon
- Technical knowledge
    - Software asking you for: type of internet connection, ip address or some specific software internal identifiers

**MOTORIC AND SENSORY ABILITIES**

- Shaking hands
    - Difficulties using mouse, keyboard or touchscreen
- “Big fingers”
    - Difficulties using touchscreen or touchscreen keyboard
- Hearing problems
    - Difficulties understanding voice output
- Bad eyes
    - Difficulties seeing output on screen
    - Not able to distinguish UI elements because they are too close
- Language difficulties
    - Difficulties understanding voice output
    - Language dialect not recognized by voice recognition

**DEVICE ABILITIES**

- Device specs
    - Slower/faster CPU, higher/lower RAM, disk space, ...
- Screen
    - Small/big screen
- Type of device
    - Desktop, laptop, tablet, mobile, watch, ...
- Internet speed
- Device manufacturer
- Operating system

## EMPATHY TOOLS

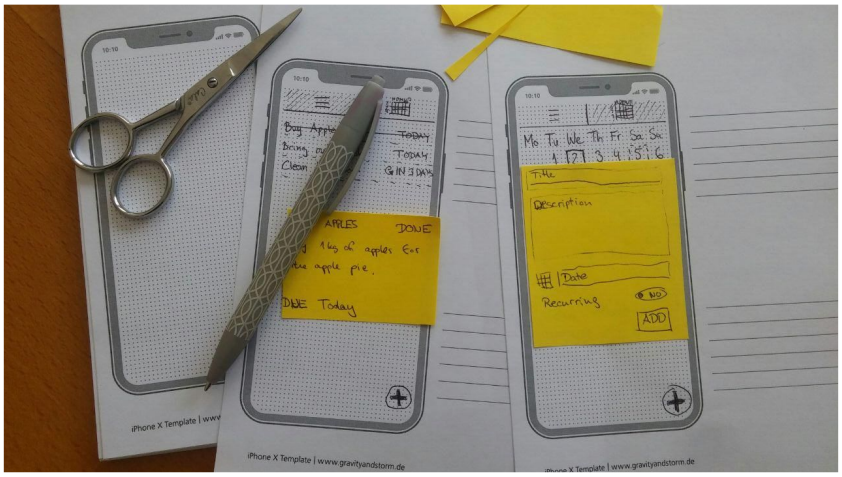
*Foster **Prototyping***

Some environments already help you with simulating different device abilities, like for example choosing different devices in mobile simulators, throttling internet speed or changing screen size in a browser.

## B.12. Paper Prototype

# PAPER PROTOTYPE

*Foster **Prototyping***




**PREREQUESITES**

- Paper, pen
- Scissors, glue, post-its

**TIME**

30-90 min

**WHAT**

Paper Prototyping is a method to create **low fidelity prototypes** based on sketches made on **paper** or a whiteboard. It should represent the screens and flow of the application in a fast, easy and fun way to get a **first impression** of your **concept**.

**WHY**

Many programmers have been in the situation that the designer is telling them to change the click-flow again - this is not only tiring for the programmer but also very inefficient. Low fidelity prototypes, like Paper Prototyping, could have helped beforehands. This method is very **inexpensive** in time and effort - on paper you can explore a lot of different versions without spending too much time, and if necessary also discard any idea that in the end wouldn't fit. Compared to digital prototypes paper gives you higher **flexibility** in a faster and cheaper way. You can make changes and annotations on the fly - even while testing - because there are **no technical barriers** that would distract you from the actual task. You can sketch anything on your page and you are only limited by the borders of the paper and your creativity. Prototyping on

## PAPER PROTOTYPE

*Foster **Prototyping***

paper also gives the chance to non-designers to participate in the process and thus facilitate **communication of ideas** and taking decisions in the team.
Another important point is the quality of the feedback that can be gathered with paper prototypes. As it is very obvious that in paper prototypes you are not asking the user whether the final product should use a different font or color scheme - feedback will concentrate on the **general usability**. And not to forget: Paper prototyping can be a lot of fun and strengthen the **team spirit**.

Of course paper prototyping is not suitable for all kinds of products, and may be not ideal for visually complex or highly interactive designs. Some say that it does not look professional enough to show it to clients or end users - this is a matter of taste and should be weighed up against the benefits of this method, especially because people tend to give more **honest feedback** on less polished prototypes.

**HOW TO**
A very basic Paper Prototype consists of sketches of each screen. While the name suggests to do this on paper, it is also possible to use the whiteboard with some limitations. There also exist prototyping kits and templates, which you can use to create basic UI elements quickly. Use Paper Prototypes in early testing or in brainstorming meetings and sessions.

**CONTEXT** Before starting quickly think where and how the app should be used. You could be designing an app for somebody running and using a smartwatch, driving and following the smartphone or sitting at desktop computer.

**GET YOUR TOOLS** Gather all required utensils like paper and pens before starting. Don't forget scissors, glue, post-its and any other tools that can give you more flexibility.

**PREPARE THE INTERFACE** Depending on the type of device you are designing for, cut or fold the paper to the right size. On bigger devices the paper should be exactly the size of the screen - on smaller devices the size can be even three to four times of the original, but keep the ratio.

**FOCUS ON USABILITY** Don't be tempted to prioritize graphical issues in this stage of development, rather try to find out how to make the product more usable.

**START** "*The way to get started is to quit talking and begin doing.*" - Walt Disney
To start it is enough to sketch roughly the user interface and cut out the parts. Depending on your use-case you can start mobile-first, as it allows you to concentrate on the important things. Be sure to have one sketch per screen.

# PAPER PROTOTYPE

*Foster **Prototyping***

**UI ELEMENTS** Help yourself to recreate common elements quickly, e.g. by using stencils or copying already sketched elements and cutting them out.

**INTERACTIVE ELEMENTS** Use Post-it stickers to simulate elements like pop-ups, dialogs and messages.

**TEST** One person is playing the “computer” while the tester is “clicking” around on the prototype. Be sure not to give any usability hints or explanations while testing. Take notes by directly writing on the prototype or by using post-it notes.

**ITERATE** After testing, incorporate any learnings into the prototype, either by adding them on the existing prototype or recreating the necessary parts. Make sure to iterate multiple times - you can also explore multiple versions if you have different ideas.

**DIGITALIZE** At the end of your iterations it makes sense to take pictures of your paper prototype and use a prototyping tool to add digital interactions and create a Digital Prototype.

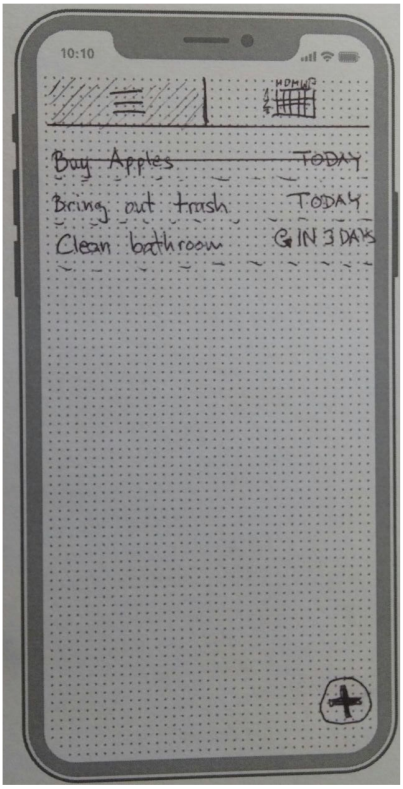
A side note on sketches vs. Paper Prototypes:
Sketches are only showing one screen in a static state, prototypes are meant to show an interactive flow that should resemble the interaction of the final product.

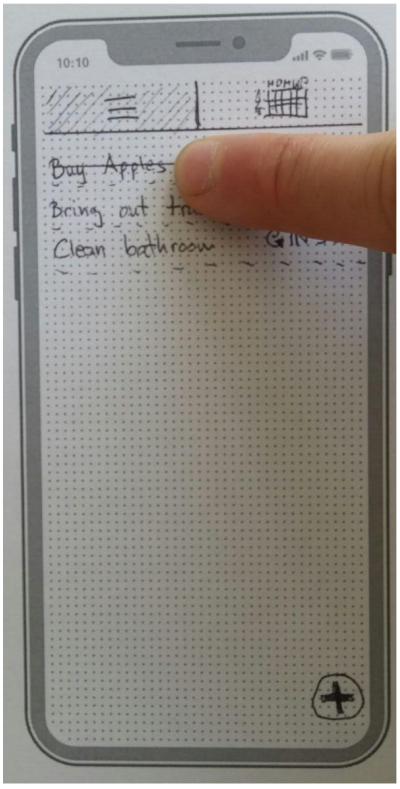
**EXAMPLE**
This method is mostly used for the early stages of a product or feature. Use it to test your ideas, no matter how small or big. You can also use it to test out something quickly before actually implementing it if unsure about the usability.

The next pages contain pictures of a prototype for a todo application.

## PAPER PROTOTYPE

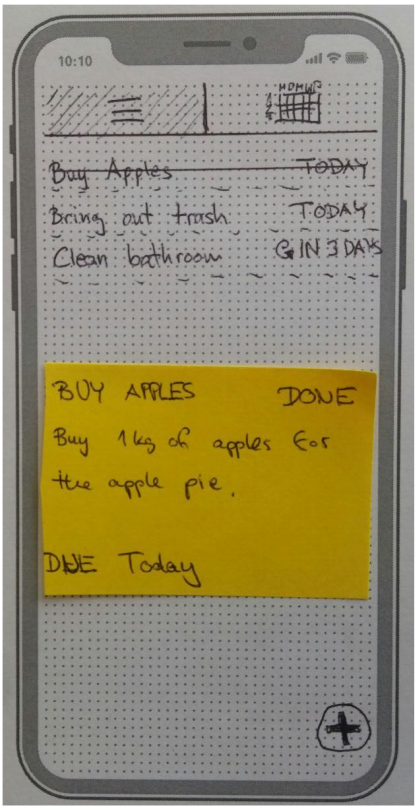
*Foster **Prototyping***

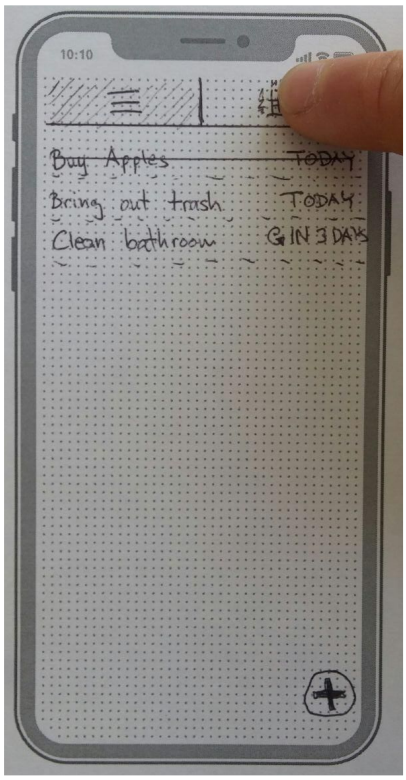


## PAPER PROTOTYPE

*Foster **Prototyping***



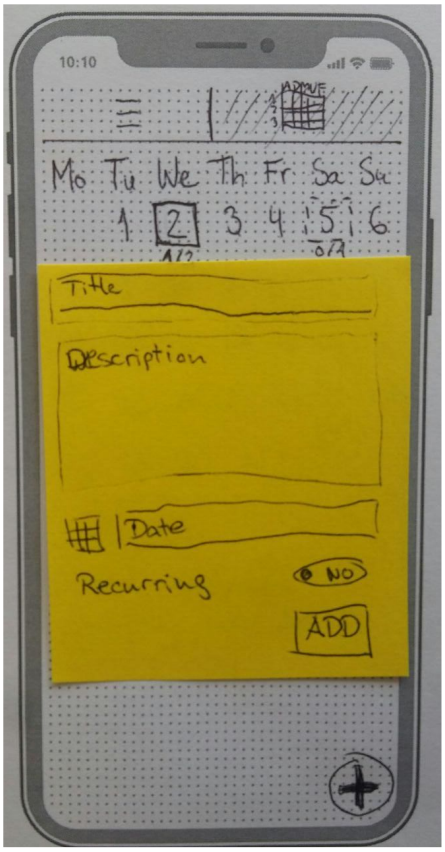



## PAPER PROTOTYPE

*Foster **Prototyping***


## PAPER PROTOTYPE

*Foster* ***Prototyping***



## B.13. General Feedback Tips

# GENERAL FEEDBACK TIPS

*Foster **Feedback***

Gathering feedback is not only a crucial element in the Design Thinking process but a substantial part of daily work nowadays. Giving feedback means to state your opinion or evaluate a specific prototype. Our own estimation and perception is complemented by an objective opinion from another person.
Getting honest feedback from your testers is hard, because it involves hearing and telling negative things about something that you built, but it is one of the most powerful tools to validate and improve your solution.

**WHAT TO TEST?** Before gathering feedback you first have to know what exactly you want to test out. Have a few core questions ready but still be open to receive feedback on other topics.

**PREPARE THE TEST** If possible you should test out different versions of your prototype and find a way to be able to freely observe the interaction with the prototype.

**WHO TO ASK?** Asking anybody is better than nobody, but especially in the more advanced stages of your development it is crucial to also test with real end users.

**BE OPEN** No matter how much work you put into your prototype, be open for feedback. At any point it might be necessary to dismantle, change or even abandon your idea.

**DON'T GET EMOTIONAL** As a feedback giver avoid generalized statements and minimize emotions in favor of fruitful, concrete suggestions for improvement.

**DON'T DEFEND** No matter how convinced you are about your prototype, take a step back to hear and accept feedback, criticism and suggestions. Instead of defending your existing ideas, try to dig deeper and find out what is really wrong with your prototype.

**BE OBJECTIVE** You want to gather feedback for improvement, so don't try to sell your idea. Highlight positive but also negative aspects of your solution.

**OBSERVE** Apart from what people tell you, observe the reactions on your prototype and also note down any non-verbal observations.

**DOCUMENT** When receiving feedback make sure to record all that information for later usage. You could have a written form of the feedback, or even take photos and videos of the feedback session.

# C. Method Usage Example

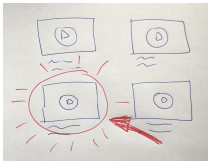
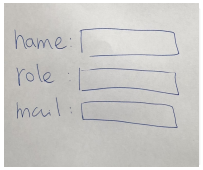
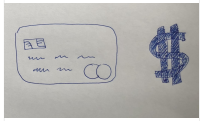
## C.1. Customer Journey Map Result Core Group

## DR. MAR 10 CENTS

**WINGLET**

Dr. Cents is an Orthopedist that has a private practice, is married, has two kids and lives in Augsburg. He likes to always buy the newest tech products. He is active in social media and likes to connect various sport gadgets with his smartphone. For his medical career Dr. Cents is obligated to collect 250 CME points in the next five years. Apart from the legal obligation, he always wants to be up to date on the latest trends in his medical field.

One week

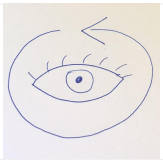
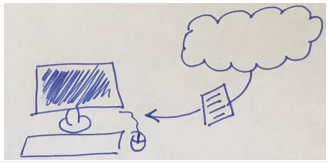
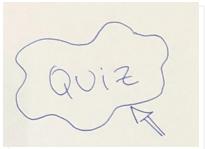
| STAGE | RESEARCH | | BUY TICKET | | WATCH EVENT | |
|---|---|---|---|---|---|---|
| TOUCHPOINT ACTION | Search for a source of CME certification | Find event to watch | Register user | Pay ticket for event | Access the event | Participate in the event |
| STORYBOARD |  |  |  |  |  |  |
| DESCRIPTION | Mar10 searches Google and Facebook for interesting events or media that award CME points. After reviewing some sites he decides to attend a winglet live event since it beats attending an event in person that requires him to do some travel. | Mar10 feels happy with the look and feel of the winglet site. After browsing for some time Martin decides to attend the next live event available in the site. | Mar10 registers in the site so he is able to proceed with the payment of the event | Mar10 adds the event to his cart and proceeds to checkout. He fills all his billing information like Address and Telephone number. He then confirms the payment and gets access to the event page. | After waiting until the day of the event he accesses the live center of the event and waits for the event to start. | Mar10 watches the event, feels a little shy to ask questions at first but once someone kicks off the questions he dares to ask and upvote others he finds interesting. |
| TASKS | ▪ Browse google for 1hr<br>▪ Ask friend<br>▪ Search on Facebook<br>▪ Get to winglet | ▪ Browse Winglet events<br>▪ Select one event | ▪ Fill out sign up form | ▪ Fill out billing info<br>▪ Schedule event on my calendar | ▪ Log in<br>▪ Select event | ▪ Watch event<br>▪ Ask questions in slido |
| EMOTIONAL ARC |  | | | | | |

## DR. MAR 10 CENTS

**WINGLET**

Dr. Cents is an Orthopedist that has a private practice, is married, has two kids and lives in Augsburg. He likes to always buy the newest tech products. He is active in social media and likes to connect various sport gadgets with his smartphone. For his medical career Dr. Cents is obligated to collect 250 CME points in the next five years. Apart from the legal obligation, he always wants to be up to date on the latest trends in his medical field.

One week

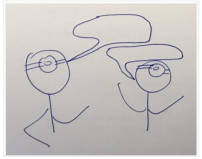
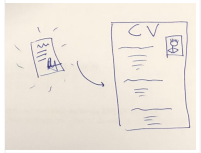
| | | | | | |
|---|---|---|---|---|---|
| **STAGE** | POST EVENT | | | GET CERTIFIED | |
| **TOUCHPOINT ACTION** | Discuss | Rewatch | Download Content | Access Quiz | Take Quiz |
| **STORYBOARD** |  |  |  |  |  |
| **DESCRIPTION** | After the event finishes Mar10 still has a lot of unanswered questions and would like to continue the discussion both with the event experts and with people that watched the event. He feels a little frustrated that no post event discussion is available. | In the hopes of getting those answers he tries to rewatch the event but with no luck since post live video won't be available for a while, could take days even. | Very useful information was shared during the event, interesting studies and facts. Being as detail oriented as he is, Mar10 is very keen on reviewing that data but is disappointed that winglet doesn't attach enough files to the event such as studies or papers to the event, such a shame since it sounds simple enough. To make things worse he can't even rewatch it to get the names of the studies and look for them by himself. | After the event finishes he enters the page for the Quiz and reads the disclaimer and the quiz' description. | Mar10 reads the questions and answers them. |
| **TASKS** | ▪ Find where to access the quiz<br>▪ Answer questions<br>▪ Get results | ▪ Suck it up | ▪ Download event's attached files such as papers and referenced studies | ▪ Post comments<br>▪ Answer comments | ▪ Upload attachments |
| **EMOTIONAL ARC** | Disappointed | Anxious | Disenchanted | A little nervous | Comfortable |

## DR. MAR 10 CENTS

**WINGLET**

Dr. Cents is an Orthopedist that has a private practice, is married, has two kids and lives in Augsburg. He likes to always buy the newest tech products. He is active in social media and likes to connect various sport gadgets with his smartphone. For his medical career Dr. Cents is obligated to collect 250 CME points in the next five years. Apart from the legal obligation, he always wants to be up to date on the latest trends in his medical field.

One week

| | | | | | | |
|---|---|---|---|---|---|---|
| **STAGE** | GET CERTIFIED | | | SHARE | | |
| **TOUCHPOINT ACTION** | Rewatch | Retake Quiz | Get Certificate | Recommend to colleague | Brag about certificate | Add to CV |
| **STORYBOARD** |  |  |  |  |  |   |
| **DESCRIPTION** | Having failed the first attempt, he tries to rewatch the event to make sure he answers correctly next time. He realizes again that is still not an option. | Martin reads the questions again but this time he is extra careful before answering. | After acing the quiz (With a stunning 73% mark) he visits his profile and downloads the certificate he was granted. | In spite of a few setbacks Martin is happy with the quality of the event's content and for having obtained CME points from the comfort of his own home. This motivates him to recommend the platform with his colleagues. He will do this face to face, as well as via social networking. | Having finally passed the quiz and being somewhat vain Martin would like to brag about his newly obtained certification in a casual manner. Would be nice to have a link in the quiz success page to easily post to facebook a certification obtained entry. | The Dr is quick to add the certification to his CV |
| **TASKS** | ▪ Suck it up | ▪ Answer quiz again | ▪ Visits his profile page<br>▪ Download certificate PDF | ▪ Share event in facebook<br>▪ Tweet | ▪ Share CME certificate in Facebook and other social media apps | ▪ Update CV |
| **EMOTIONAL ARC** | Annoyed | Uneasy | Proud | Excited | Proud | Happy |

# Bibliography


[1] Aerssen, B., Buchholz, C., Burkhardt, N., Ernst, A., Rings, J., Rings, S., Schobloch, A., Spicker, M., Wigge, K., Wirth, D., and et al. (2018). *Das große Handbuch Innovation: 555 Methoden und Instrumente für mehr Kreativität und Innovation im Unternehmen*. Vahlen.

[2] Alliance, A. (2018). Agile 101 (2018). `https://www.agilealliance.org/agile101/`. [Online; accessed October 22, 2018].

[3] Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., et al. (2001). Manifesto for agile software development.

[4] Belshee, A. (2005). Promiscuous pairing and beginner's mind: embrace inexperience [agile programming]. In *Agile Conference, 2005. Proceedings*, pages 125–131. IEEE.

[5] Beyer, H., Holtzblatt, K., and Baker, L. (2004). An agile customer-centered method: rapid contextual design. In *Conference on Extreme Programming and Agile Methods*, pages 50–59. Springer.

[6] Bias, R. G. and Karat, C.-M. (2005). Justifying cost-justifying usability. In *Cost-Justifying Usability (Second edition)*. Elsevier.

[7] Böddicker, N., Hauch, H., Hinzer, A., Hofer, M., Karsten, N., Khan, A., Rubens-Laarmann, A., and Wilhelm, S. (2016). Methodensammlung der heinrich heine universität, düsseldorf. `https://www.uni-duesseldorf.de/home/fileadmin/redaktion/Lehre/Hochschuldidaktik/Downloads/Methodenbuch_Stand151216.pdf`. [Online; accessed October 22, 2018].

[8] bosshart consulting gmbh. 5-finger-feedback. `https://www.bosshart-consulting.ch/resources/5FingerFeedback.pdf`. [Online; accessed October 22, 2018].

[9] Brauneck, P., Urbanek, R., and Zimmermann, F. (1995). *Methodensammlung: Anregungen und Beispiele für die Moderation*. Verlag für Schule und Weiterbildung, Dr.-Verlag Kettler.

[10] BRAVO, C. and ADLER, I. K. (2015). Design thinking & lean: An assertive and human

approach to software development. In *4TH PARTICIPATORY INNOVATION CONFERENCE 2015*, pages 162–167.

[11] Brischke, L.-A., Leuser, L., Duscha, M., Thomas, S., Thema, J., Spitzner, M., Kopatz, M., Baedeker, C., Lahusen, M., Ekardt, F., et al. (2016). Energiesuffizienz: Strategien und instrumente für eine technische, systemische und kulturelle transformation zur nachhaltigen begrenzung des energiebedarfs im konsumfeld bauen/wohnen: Endbericht.

[12] Brown, T. (2009). Change by design.

[13] Cajander, Å., Larusdottir, M., and Gulliksen, J. (2013). Existing but not explicit-the user perspective in scrum projects in practice. In *IFIP Conference on Human-Computer Interaction*, pages 762–779. Springer.

[14] Commission, E. (2014). Topic : Platform for ict for learning and inclusion. `http://ec.europa.eu/research/participants/portal/desktop/en/opportunities/h2020/topics/inso-6-2014.html`. [Online; accessed October 22, 2018].

[15] Constantine, L. L. and Lockwood, L. (2002). Process agility and software usability: Toward lightweight usage-centered design. *Information Age*, 8(8):1–10.

[16] Creswell, J. and Clark, V. (2011). *Designing and Conducting Mixed Methods Research*. SAGE Publications.

[17] Creswell, J. W. and Creswell, J. D. (2017). *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage publications.

[18] Cross, N. (2003). Engineering design methods: strategies for product design. 2000. *Milton Keynes, UK: John Wiley & Sons Ltd*.

[19] Dam, R. and Siang, T. (2018). What is design thinking and why is it so popular? `https://www.interaction-design.org/literature/article/what-is-design-thinking-and-why-is-it-so-popular`. [Online; accessed October 22, 2018].

[20] De Langhe, B., Puntoni, S., and Larrick, R. P. (2017). Linear thinking in a nonlinear world. *Harvard Business Review*, 2017(May-June):11.

[21] De Lille, C., Roscam Abbing, E., and Kleinsmann, M. (2012). A designerly approach to enable organizations to deliver product-service systems. In *International DMI Education Conference: Design Thinking: Challenges for Designers, managers and Organizations, 14-15 April 2008, Cergy-Pointoise, France*. DMI.

[22] de Paula, D. F. O. and 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*, volume 617, page 30–34. Springer International Publishing.

[23] de Paula, D. F. O., Menezes, B. H. X. M., and Araújo, C. C. (2014). *Building a Quality Mobile Application: A User-Centered Study Focusing on Design Thinking, User Experience and Usability*, volume 8518, page 313–322. Springer International Publishing.

[24] Dobrigkeit, F. and de Paula, D. (2017). The best of three worlds-the creation of innodev a software development approach that integrates design thinking, scrum and lean startup. In *DS 87-8 Proceedings of the 21st International Conference on Engineering Design (ICED 17) Vol 8: Human Behaviour in Design, Vancouver, Canada, 21-25.08. 2017*.

[25] Domschke, M., Bog, A., and Zeier, A. (2009). Teaching design thinking to software engineers: Two future-oriented curriculum case studies. In *26th ICSID World Design Congress, Design Education Conference*, pages 22–25.

[26] Donahue, G. M., Weinschenk, S., and Nowicki, J. (1999). Usability is good business. *Compuware Corp., julio*.

[27] Eames, C. and Eames, R. (1977). Powers of ten (film). *Santa Monica: Eames Office*.

[28] Ferré, X., Juristo, N., Windl, H., and Constantine, L. (2001). Usability basics for software developers. *IEEE software*, 18(1):22–29.

[29] Flick, U., Kardorff, E. v., and Steinke, I. (2008). Qualitative forschung. *Ein Handbuch*, 6:14.

[30] Frye, U. and Inge, T. (2013). The integration of design thinking and lean software development from the perspective of product owners and scrum masters. *Recuperado el*, 1:64.

[31] Gayatri, V. and Pammi, K. (2013). Agile user stories.

[32] Gloger, B. and Margetich, J. (2014). Das scrum-prinzip. agile organisationen aufbauen und gestalten. *Projektmanagement, Innovationsmanagement*, page 7.

[33] Glomann, L. (2018). *Introducing 'Human-Centered Agile Workflow' (HCAW) – An Agile Conception and Development Process Model*, volume 607, page 646–655. Springer International Publishing.

[34] Greenberg, S., Carpendale, S., Marquardt, N., and Buxton, B. (2011). *Sketching user experiences: The workbook*. Elsevier.

[35] Griffith, E. (2014). Why startups fail, according to their founders. *Retrieved September*, 20:2016.

[36] GROSSMAN-KAHN, B. and ROSENSWEIG, R. (2012). Skip the silver bullet: driving innovation through small bets and diverse practices. *Leading Through Design*, page 815.

[37] Grots, A. and Pratschke, M. (2009). Design thinking—kreativität als methode. *Marketing Review St. Gallen*, 26(2):18–23.

[38] Grudin, J. and Pruitt, J. (2002). Personas, participatory design and product development: An infrastructure for engagement. In *PDC*, pages 144–152.

[39] Gurusamy, K., Srinivasaraghavan, N., and Adikari, S. (2016). *An Integrated Framework for Design Thinking and Agile Methods for Digital Transformation*, volume 9746, page 34–42. Springer International Publishing.

[40] Häger, F., Kowark, T., Krüger, J., Vetterli, C., Übernickel, F., and Uflacker, M. (2015). Dt@ scrum: integrating design thinking with software development processes. In *Design Thinking Research*, pages 263–289. Springer.

[41] Hanington, B. and Martin, B. (2012). *Universal methods of design: 100 ways to research complex problems, develop innovative ideas, and design effective solutions*. Rockport Publishers.

[42] Hicks, A. (2015). *Empathy and Experiential Learning: How a Community-Based Project Enhances the College Experience*. PhD thesis, University Honors College, Middle Tennessee State University.

[43] Hirschfeld, R., Steinert, B., and Lincke, J. (2011). Agile software development in virtual collaboration environments. In *Design Thinking*, pages 197–218. Springer.

[44] Hitchcock, D. and Taylor, A. (2003). Simulation for inclusion... true user-centred design? pages 105–110.

[45] IDEO (2003a). Method cards. `https://www.ideo.com/post/method-cards`. [Online; accessed October 22, 2018].

[46] IDEO, I. (2003b). Method cards: 51 ways to inspire design. *Palo Alto*.

[47] J2C. Design thinking a short introduction. `http://j2c.de/content/backpack/1-overview-of-subpages/3-design-thinking-brochure/1-download/j2c-design-thining-brochure.pdf`. [Online; accessed October 22, 2018].

[48] Jakob, M., Larbig, C., Minder, B., and Berger, F. (2014). Handbook sociallab. `https://issuu.com/nilsloeffel/docs/method_sheets_buch_issuu`. [Online; accessed October 22, 2018].

[49] Jalleh, S. (2013). Co-designing an ec135 air ambulance cabin. page 8.

[50] Jobst, B. and PI, C. M. How to give orientation and inspiration to design thinking novices regarding prototyping methods?

[51] Jurca, G., Hellmann, T. D., and Maurer, F. (2014). Integrating agile and user-centered design: a systematic mapping and review of evaluation and validation studies of agile-ux. In *2014 Agile Conference (AGILE)*, pages 24–32. IEEE.

[52] Kumar, V. (2012). *101 design methods: A structured approach for driving innovation in your organization*. John Wiley & Sons.

[53] Lárusdóttir, M. K., Cajander, Å., and Gulliksen, J. (2012). The big picture of ux is missing in scrum projects. In *Proceedings of the International Workshop on the Interplay between User Experience (UX) Evaluation and System Development (I-UxSED)*, pages 49–54.

[54] Lemon, K. N. and Verhoef, P. C. (2016). Understanding customer experience throughout the customer journey. *Journal of Marketing*, 80(6):69–96.

[55] Leung, W.-C. (2001). How to design a questionnaire. *student BMJ*, 9(11):187–189.

[56] Lewrick, M., Link, P., and Leifer, L. (2017). Das design thinking playbook. *Munich, Germany: Vahlen*.

[57] Lindberg, T., Rauth, I., Köppen, E., and Meinel, C. (2001). Design thinking in the it industry: Exploring language games on understanding, implementation and adoption. Published in the proceedings of Design Thinking Research Symposium.

[58] Liyanage, C., Elhag, T., Ballal, T., and Li, Q. (2009). Knowledge communication and translation–a knowledge transfer model. *Journal of Knowledge management*, 13(3):118–131.

[59] Maedche, A., Botzenhardt, A., and Neer, L. (2012). *Software for people: fundamentals, trends and best practices*. Springer Science & Business Media.

[60] Marcus, A. (2002). Return on investment for usable user-interface design: Examples and statistics. *User Experience Magazine*, 1(3):25–31.

[61] Meinel, C., Rhinow, H., and Köppen, E. (2013). Design thinking prototyping cardset. *Hasso Plattner Institut für Softwaresystemtechnik*.

[62] Miaskiewicz, T. and Kozar, K. A. (2011). Personas and user-centered design: How can personas benefit product design processes? *Design Studies*, 32(5):417–430.

[63] Moore, P. and Conn, C. (1985). *Disguised!* Word Books.

[64] Moser, C. (2012). *User Experience Design: Mit erlebniszentrierter Softwareentwicklung zu Produkten, die begeistern*. X.media.press. Springer Berlin Heidelberg.

[65] Norrmalm, T. (2011). Achieving lean software development: implementation of agile and lean practices in a manufacturing-oriented organization.

[66] Ohno, T. (1988). *Toyota production system: beyond large-scale production*. crc Press.

[67] Patterson, L. E., Boone, A. E., and Engsberg, J. R. (2017). The creation and testing of a simulation-based workshop to increase level of understanding and empathy of eating disorders. *Journal of Psychiatry and Psychiatric Disorders*, 1(5):270–289.

[68] Pedersen, T. Ø. (2016). *UX Toolbox for Software Developers: Methods and Training*. PhD thesis, Aalborg Universitetsforlag.

[69] Pérez, J. and Murray, M. C. (2010). Generativity: The new frontier for information and communication technology literacy.

[70] Plattner, H. (2010). Bootcamp bootleg. *Design School Stanford, Palo Alto*.

[71] Plattner, H., Meinel, C., and Weinberg, U. (2009). „design thinking–innovation lernen–ideenwelten öffnen “, mi-wirtschaftsbuch. *München Google Scholar*.

[72] Pojasek, R. B. (2000). Asking "why?" five times. *Environmental Quality Management*, 10(1):79–84.

[73] Rising, L. and Janoff, N. S. (2000). The scrum software development process for small teams. *IEEE software*, 17(4):26–32.

[74] Ritchie, J., Tinker, A., and Power, J. (2016). The effectiveness of design thinking techniques to enhance undergraduate student learning. In *Futurescan 3: Intersecting Identities*. Association of Fashion & Textiles (FTC), Loughborough University.

[75] Rosenbaum, M. S., Otalora, M. L., and Ramírez, G. C. (2017). How to create a realistic customer journey map. *Business Horizons*, 60(1):143–150.

[76] Sauvola, T., Rontti, S., Laivamaa, L., Oivo, M., and Kuvaja, P. (2016). Integrating service design prototyping into software development. *ICSEA 2016*, pages 325–332.

[77] Schwaber, K. (1997). Scrum development process. In *Business object design and implementation*, pages 117–134. Springer.

[78] Seffah, A. and Metzker, E. (2004). The obstacles and myths of usability and software engineering. *Communications of the ACM*, 47(12):71–76.

[79] Serrat, O. (2017). The five whys technique. In *Knowledge solutions*, pages 307–310. Springer.

[80] Snyder, C. (2003). *Paper prototyping: The fast and easy way to design and refine user interfaces*. Morgan Kaufmann.

[81] Sohaib, O., Solanki, H., Dhaliwa, N., Hussain, W., and Asif, M. (2018). Integrating design thinking into extreme programming. *Journal of Ambient Intelligence and Humanized Computing*.

[82] Spreng*, R. N., McKinnon*, M. C., Mar, R. A., and Levine, B. (2009). The toronto empathy questionnaire: Scale development and initial validation of a factor-analytic solution to multiple empathy measures. *Journal of personality assessment*, 91(1):62–71.

[83] Suzuki, S. (1970). Zen mind. *Beginner's Mind. New York: Weatherhill*.

[84] Tam, M. (2001). Using paper prototyping as a usability testing methodology for web application development. page 94.

[85] Ungar, J. and White, J. (2008). Agile user centered design: enter the design studio-a case study. In *CHI'08 Extended Abstracts on Human Factors in Computing Systems*, pages 2167–2178. ACM.

[86] Ximenes, B. H., Alves, I. N., and Araújo, C. C. (2015). *Software Project Management Combining Agile, Lean Startup and Design Thinking*, volume 9186, page 356–367. Springer International Publishing.