Exploring Behavioral Driven Development
This exam work has been carried out at the School of Engineering in Jönköping in the subject area of information technology. The work is a part of the three-year Bachelor of Science in Engineering program. The authors take full responsibility for opinions, conclusions and findings presented.

Examiner: Ulf Seigerroth

Supervisor: Martin Lindh

Scope: 15 credits (first cycle)

Date: 6.6.2019
Abstract

Behavior Driven Development (BDD) is a modern agile software development approach that originates from Test Driven Development (TDD) and Acceptance Test Driven Development (ATDD). Other than TDD and ATDD, BDD introduces new methods and strategies that intend to discover the behavior of software in greater detail which is achieved through enhanced communication and cooperation between everyone involved in software projects. In this paper it is examined how BDD can be taken even further and be connected to the products of UX-strategy in order to explore the possibilities to improve internal communication within software project teams. The report guides the reader through the theoretical frameworks of BDD, UX-strategy and Communication and presents suggestions of how BDD and the products UX-strategy can be connected to improve communication and understanding.
Summary

The complexity of today’s software projects requires flawless communication of information that concern new software products. Especially large scale software projects can include complicated requirements and documentations which are hard to communicate and understand for everyone who is involved in the process.

The latest agile approach to software development is offered by BDD, Behavior Driven Development. Due to the youth of the agile framework the purpose of this research is to explore how BDD solves communication issues and explore how the framework can be further supported by integrating the products of UX-strategy into the process of software development, which led to following research questions:

*How can BDD and the products of UX-strategy be connected in order to improve internal communication within software project teams?*

*How can a combination of BDD and the products of UX-strategy improve understanding of features, user stories and scenarios?*

Since the research serves exploratory purposes a inductive research approach has been chosen to conduct a qualitative research design in form of semi structured interviews. The empirical data collection resulted in 6 interviews, that include opinions, insights and knowledge from professionals working at Munich RE in Germany. Through a thematic analysis of the empirical data, it could be identified that there are still major problems in sight of communication, which is addressed as a communication barricade between business and technical people. The findings also clearly indicate that there is a strong constrast in opinions between the different parties, the interviewees with a technical background and the interviewees from the business sections. Outgoing from the thematic analysis it could be concluded that there are still characteristics that imply problems in communication and understanding inside of software project teams. Accordingly to the prominent opinions of the intervieweed business people it, could be concluded that the deployment of a visual layer in form of coressponding UX prototypes, wireframes and user flow charts could support and clarify the complexities that the BDD frameworks entails, and further aid the generell communication inside software project teams as well as assist the creation of a common ground, that is necessary for successfull conversation between the parties. Lastly it was recognized that further research has to be conducted in order to explore the BDD framework further and answer all questions arround it.

**Keywords**

Software Development, Behavior Driven Development, UX-strategy, UX-strategy, Communication, Software Teams
<table>
<thead>
<tr>
<th>Contents</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Warm up</td>
<td>26</td>
</tr>
<tr>
<td>Body of the Interview</td>
<td>26</td>
</tr>
<tr>
<td>4.3 INTERVIEW WITH IT ARCHITECT / SCRUM MASTER</td>
<td>30</td>
</tr>
<tr>
<td>Warm up</td>
<td>30</td>
</tr>
<tr>
<td>Body of the Interview</td>
<td>30</td>
</tr>
<tr>
<td>4.4 INTERVIEW WITH TEST ENGINEER</td>
<td>33</td>
</tr>
<tr>
<td>Warm up</td>
<td>33</td>
</tr>
<tr>
<td>Body of the Interview</td>
<td>33</td>
</tr>
<tr>
<td>4.5 INTERVIEW WITH TECHNICAL PROGRAM MANAGER</td>
<td>36</td>
</tr>
<tr>
<td>Warm up</td>
<td>36</td>
</tr>
<tr>
<td>Body of the Interview</td>
<td>36</td>
</tr>
<tr>
<td>4.6 INTERVIEW WITH SENIOR SOFTWARE ENGINEER</td>
<td>39</td>
</tr>
<tr>
<td>Warm up</td>
<td>39</td>
</tr>
<tr>
<td>Body of the Interview</td>
<td>39</td>
</tr>
<tr>
<td>5 Analysis</td>
<td>42</td>
</tr>
<tr>
<td>5.1 HOW CAN BDD AND THE PRODUCTS OF UX-STRATEGY BE CONNECTED IN ORDER TO IMPROVE INTERNAL COMMUNICATION WITHIN SOFTWARE PROJECT TEAMS?</td>
<td>42</td>
</tr>
<tr>
<td>Communication between business and IT</td>
<td>42</td>
</tr>
<tr>
<td>Project Documentation</td>
<td>43</td>
</tr>
<tr>
<td>BDD Solutions</td>
<td>44</td>
</tr>
<tr>
<td>UX-strategy as tooling for communication</td>
<td>45</td>
</tr>
<tr>
<td>5.2 HOW CAN A COMBINATION OF BDD AND THE PRODUCTS OF UX-STRATEGY IMPROVE UNDERSTANDING OF FEATURES, USER STORIES AND SCENARIOS?</td>
<td>46</td>
</tr>
<tr>
<td>Views on formulation in BDD</td>
<td>46</td>
</tr>
<tr>
<td>Overview and understanding of features, user stories and scenarios</td>
<td>47</td>
</tr>
<tr>
<td>6 Discussion</td>
<td>48</td>
</tr>
<tr>
<td>6.1 DISCUSSION OF METHOD</td>
<td>48</td>
</tr>
<tr>
<td>Reliability</td>
<td>48</td>
</tr>
<tr>
<td>Validity</td>
<td>49</td>
</tr>
<tr>
<td>Generalizability</td>
<td>49</td>
</tr>
<tr>
<td>6.2 DISCUSSION OF FINDINGS</td>
<td>49</td>
</tr>
<tr>
<td>7 Conclusions</td>
<td>52</td>
</tr>
<tr>
<td>8 References</td>
<td>53</td>
</tr>
</tbody>
</table>
9 Appendices ................................................................. 55
  9.1 INTERVIEW GUIDE ...................................................... 55

Introduction ........................................................................... 55
Warm up ............................................................................... 55
Body of the Interview ............................................................. 55
Theoretical background

1 Introduction

As part of the bachelor program New Media Design at the Jönköping University this thesis researches the domain of Software Development. More precisely it is examined how the agile approach Behavior Driven Development and the products UX-strategy can be connected in order to improve internal communication within software projects teams. Furthermore, this research paper is part of a program at Munich Re in Munich, Germany, that explores the possibilities of Behavioral Driven Development.

Munich Re is a global reinsurance company. It’s inhouse IT department is responsible for the development of new and innovative software, which is developed, planned and executed by large projects teams, using Scrum. Scrum delivers various tools, workflows and concepts that are integrated into the project work at Munich Re, yet fail to satisfy the sophisticated needs of software development. Munich Re, therefore plans on integrating the BDD framework into its software projects, the motivation and expected outcome of integrating BDD, includes improved quality of communication and requirements, that define the outlines of a software project.

Following today’s trends in software development methods a strong shift from waterfall model methods to agile methods and practices can be noticed. As a result, agile approaches such as Scrum and Test Driven Development got introduced to the industry of software development (Egbreghts, n.d.). Test Driven Development (TDD), follows the logic to develop so called acceptance tests, before the development of software code or implementation of code. The written acceptance tests naturally fail, since no code to test is available at the first time the tests are ran. After the tests have failed the first time the applicable code to pass the tests is developed, which, in case of success, should let the tests pass. This way it should be prevented that any software is implemented that might endanger already existing implemented parts. However, TDD also entails new problems into the landscape of software development, one of said was the problem of finding a starting point of writing tests, since a clear hierarchy could mostly not be identified, furthermore, the language used for written tests with diverse class names and formulations caused many problems when correcting a failed test and general bug fixing. As a result, Behavioral Driven Development arose, with the main benefit of formulating acceptance test in natural language as ubiquitous communication for tests (Soeken, Wille, Drechsler, 2012). Behavioral Driven Development (BDD) evolved out of the just mentioned TDD and Acceptance Test Driven Development (ATDD), trying to solve the problems that occurred in either practice. In comparison to TDD, ATDD defines its acceptance test through the requirements set by project stakeholders. Both TDD and ATDD are widely spread due to their test automation which results in improved quality and productivity. Both attempts however, result in the same kind of problematic, since only the state, at a certain point of a system is tested and not the behavior. Furthermore, both practices have unstructured language to describe tests which results in confusion and errors on the development site (Solís, Wang, 2011). Dan North (2006) therefore introduced BDD to the landscape of software development, as an answer to the problems that occurred when using TDD and ATDD for software development.
Furthermore, Nagy and Rose (2018) discuss that BDD and Scrum go along well, which they explain by the layout the Scrum framework has to offer. Accordingly, to the authors for example, the “daily scrum”, which is a meeting, that evaluates the past 24 hours and plans the work tasks for the next 24 hours ahead, can be used for “requirement workshops” which are introduced as a part of the BDD framework. Why BDD and not any other method? As the BDD pioneers Nagy and Rose (2018) explain Behavior Driven Development is the last name to an agile approach, that has its origin in a combination of Test Driven Development (TDD) and Acceptance Tests Driven Development (ATTD). BDD’s main benefit is the attempt to establish an active conversation between the business side and the development side. BDD consists of three different aspects: discovery, formulation and testing, which are further discussed in chapter 2, Theoretical Background.

The main purpose of a finished product or software is not only to deliver the defined business goals, which ultimately will help achieving the vision. It is also about building software, that has the potential to grow and scale. Software that can not be maintained or updated, can not survive long nor satisfy the needs of the end user in the long run. To accomplish this, Smart (2015) explains, flawless communication and planing within project teams, especially between roles with different backgrounds and competences, is necessary. The end product can only deliver business value, if the entire team behind a project, has understood the business goals, requirements and scope of a project equally. Unfortunately this equal understanding is not given by nature, which creates unpredictable outcomes of projects and therefore inconsistency in quality between finished projects. Not only, can projects deliver the intended outcomes only partly, but are often more, time and money, consuming than planned. The general problem can therefore be addressed as a communication barricade between project team members, project owner and other stakeholders, which results in costly mistakes that influence the quality of the software and the time spent on a project. Which reveals the need of an approach that ensures that communication within project teams is functioning immaculately.

Accordingly to Smart (2015), this is where Behaviour Driven Development comes in play. BDD offers an agile software development approach, that tries to establish a common understanding between everyone within a software project team through specific tools, methods and strategies. In the BDD theory this can for example be achieved through strategic and collaborative meetings, including people from the business side, the delivery team and other individuals such as User Experience (UX) designers and Test-analysts. This construct of people is often referred as the three amigos. Through these collaborative meetings, that are held at several points within the cycle of software development, a clear and shared understanding of functionality, scope and requirements can be created. However, the intend of these meetings can serve different purposes; discovery, definition or evaluation, which all contribute to the main idea of a shared understanding. Meetings that are in the initial phase of a project can therefore result in the discovery of new functionalities that should be included in the application, in order to create business value. These functionalities, in BDD also introduced as features, are broken down into several user stories which help discovering edge cases of functionality, that are defined as concrete examples, called scenarios. Features, userstories and the corresponding scenarios are stored in the Backlog, which is a Scrum-tool to
organize, store and manage work tasks. In addition to being illustrative examples for user stories and requirements, scenarios are the base for formulating executable tests, that are later used for automated testing, which is also a part of behavior driven development (Smart, 2015).

Through collaborative meetings, that result in the integration of illustrative examples into the cycle of software development, BDD offers an attractive attempt to close the communication gap between individuals involved into the process of software development. However BDD forgets to address the importance of design, both as a tool, to lead decisions and as an artifact that supports the understanding of requirements, features, user stories and scenarios. The attempt of producing examples only in the form of text, that illustrate user stories which vice versa illustrate features is patchy. Project documentations at Munich Re are often long and especially specific parts are often hard to recall for individuals. Since features, user stories and scenarios are guiding the process behind BDD, as further discussed in the chapter theoretical background it becomes crucial, that all members can easily find, understand and recall each item in the correct context.

1.1 Purpose and research questions

Due to the youth of BDD, the available literature around BDD as well as conducted research concerning BDD is very limited up to this time, which is a general motivation for conducting further research on the agile software development approach BDD.

BDD is heavily dependet on cooperation between everyone involved in a software project, since the cooperation, communication and discussion between individuals, lead to the discovery, definition and formulation of requirements which controls any further process within in the framework (Smart, 2015). However, BDD does not introduce tooling that ensures the quality of communication, but instead introduces a list of available tooling, specifically for BDD, that can be used for automated testing and development. Thus it is unclear how the quality of communication, which is vital for the strategy and method behind BDD, is ensured.

The purpose therefore is to explore how BDD solves communication issues and explore how products of UX-strategy could further improve communication and understanding which leads to the following research questions:

How can BDD and the products of UX-strategy be connected in order to improve internal communication within software project teams?

How can a combination of BDD and the products of UX-strategy improve understanding of features, user stories and scenarios?
1.2 Delimitations

In order to create the margings of the conducted research it is necessary to define delimitations that narrow down the scope. Since this research is conducted as part of education at Jönköping University, within a timeframe of two month the used methods of the research had to be strategically choosen in order to be able to present findings (Jönköping University, 2019). Due to this time frame the BDD framework will not be tested in practice, instead interviews will be conducted in order to gain instights and produce findings that can later on be analyzed and act as the base for answering the research questions. Further the study will not asses the entire BDD framework in detail, but instead focus on the capabilites of BDD concerning the communication. Neither will this study measure how well BDD improves communication, or if it improves communication in comparison to other agile aproaches such as TDD and ATDD. Furthermore, it is not to be researched how the processes of BDD and UX-Strategy can be merged in order to improve the subject of study, but instead explore how the products of UX-strategy could be integrated as tooling to BDD that could improve communication within software time.

1.3 Outline

After the introductory chapter the theoretical background of BDD, UX strategy and Communication theory is introduced in order to establish an understanding of the different frameworks. Followed by the chapter Method and implementation which includes a description of the methods used to collect and analyze empirical data. In the chapter findings, a total number of 6 interviews is presented. In the chapter analysis, a categorization of the interviews is presented in order formulate an answer the research questions in the chapter analysis. In the following chapter discussion, the research methods and the findings are discussed. The last chapter, conclusions, summarizes the main insights gained from the research and introduces ideas for conducting further research.

2 Theoretical framework

2.1 Behavior Driven Development

”Building the software right. Building the right software.” (Smart, 2015) Are the guiding sentences and motivations for using BDD when developing new software. As Smart (2015) explains, that in BDD the development of new software all begins with the vision statement, it is this statement that gives a first definition to the purpose of a new software. This vision can only be achieved with clearly defined business goals. And these business goals should be motivation for all further process of software development, since it is the business goals that need to be fulfilled in order to create a value for the end user. These business goals can be achieved with different functionalities, also referred as features, which are segments of a software / application that provide core functionalities to the end user. These features give the end users the capability to fulfill the business goals, and therefore create business value. Accordingly, to Smart (2015), features are often defined in an initial workshop, called feature injection, at the beginning of each new software project. The
participants of this workshop should consist of people with different competences that are involved into the project, optimally these people include the Project Owner, a Business Analyst and a Developer/Tester, also introduced as the Three Amigos. This construct of professionals contains different points of view, understandings and concerns, which in the end results in the definition of features that are understood by all departments and meet the business goals. In addition to defining features concrete illustrative cases are created, called user stories. The purpose of a user story is to break down the complexity of a feature into smaller chunks. Smart (2015) suggests the usage of a template when expressing user stories in order to unify the process behind user stories as seen in Figure 1.

![Figure 1: Template for User Story.](image)

The next step in BDD, introduced by Nagy and Rose (2018) are so called Requirement Workshops, in which the Three Amigos meet, on a weekly or daily basis. The intend of those meetings is, to keep the conversation and collaboration between everyone involved going. This is important and leads to the discovery of important findings that might would stay hidden elsewise. In these Requirement Workshops, one user story at a time is analyzed. Nagy and Rose (2018) introduce example mapping as a suitable method to lead the workshop; a user stories, acceptance criteria and the found examples are written down on color coded cards, that later can be integrated into the project’s documentation. The purpose here is to have an open discussion with the whole team, that leads to the discovery of edge cases, exploring different possible directions a software can behave in. A user story is broken down into acceptance criteria, but those could easily be misunderstood. Therefore, each acceptance criteria gets one or more examples assigned, which work as an illustrative tool to understand acceptance criteria. These examples are called scenarios. This enables the team that participates in the meeting to explore the different interpretations of acceptance criteria, which elsewise would stay hidden until much later in the project, which could lead to development of software that can’t fulfill the acceptance criteria. These examples deliver the advantage that they describe the way a software can behave including context, event and outcome, which can later be expressed following the syntax template of Gherkin, as seen in Figure 2.

![Figure 2: Gherkin syntax.](image)

When all scenarios and the user story are discussed, the software project team evaluates again if the corresponding feature can deliver the intended business value. The next logical step is the formulation of examples into scenarios which is usually done by a test engineer and a developer. After the examples have been formulated into scenarios, the product owner and business analyst review, and
ideally approve the scenarios for further development. After formulation scenarios do not only work as illustrative examples of the user stories or acceptance criteria, but also form the base for automated testing, which is the next step in BDD as introduced by Smart (2015). Scenarios can now be integrated into automated testing, yet due to it’s structure, still allow business people, to understand the content. The process of BDD then keeps on with the writing of the automated tests, into step definitions, this happens before counter parts of software is even developed yet. The practice to first write automated tests, before developing software is taken over from Test Driven Development (TDD). The scenarios are then executed in a BDD tool called Spec Flow, specifically developed for Gherkin syntax and automated tests. Since no software is developed yet, the tests should naturally fail, the first time the test code is executed. After the first test round it is necessary to develop the corresponding segment of software to allow the test to pass. Since these tests are automated, they should be running daily. Which generally makes it easier to maintain the software, produces a living documentation, and ensures that all parts of the software are working before new software is deployed. If all tests that are corresponding to a feature have passed successfully, a feature is ready to deliver, since the automated tests act as acceptance criteria as defined earlier in this section. Due to the process of BDD it is now ensured that the finished segment of software can provide users with the capability to achieve business goals, and therefore create business value, which lastly will result into fulfilling the vision statement. Due to the automated testing other parts of the software can be developed using the same process yet ensuring that already deployed parts still function as intended, at any time during the implementation (Smart, 2015).

2.1.1 BDD characteristics

Accordingly to Solís and Wang (2011), who conducted research on BDD, that identifies the characteristics of BDD, there are six characterics of BDD which distinguish BDD from other frameworks such as TDD and ATDD, which are listed below:

- **Ubiquitous Language**
  Other than ATDD and TDD, BDD introduces ubiquitous language, language that follows certain templates, rules and vocabulary, so that, both the business domain and developers can communicate and understand the formulations and definitions of requirements and other project documentations.

- **Iterartive Decomposition Process**
  As identified before in the chapter Introduction, one of the problems that came with TDD and ATDD was that developers often had no starting point to write test and start development. In BDD it is analyzed which behaviours of a system are necessary to create business value. These behaviours are then formulated into features, in a workshop between business and developers, thus these features introduce a hierarchy, since they follow the hierarchy of the business goals that are to be achieved,
Theoretical background

thus it is those feature that are necessary to create value for the software. Since developers are involved in the definition of feature, a good starting point is created for the realization of the software. Furthermore features, which mirror the business goals, are broken down into user stories, which as before mentioned are examples of how users would use certain software functionalities to achieve their goal, thus create business value. Furthermore User stories are broken down into scenarios which lastly act as the acceptance criteria for a feature to pass the automated testing before implementation.

- **Plain Text Description with User Story and Scenario Templates**
  As part of the before mentioned ubiquitous language, BDD establishes certain templates for Features, User Stories and Scenarios. While user stories and features share the same template, the scenarios are following the syntax and logic of Gherkin, which as specified before is a executable language that enables to take the scenarios into automated testing.

- **Automated Acceptance Testing**
  Other than in ATDD, BDD uses the automated acceptance testing to test the behavior of a system rather than a state of a system. When preparing for the automated testing the decomposition process of BDD continues, and scenarios are further broken down into steps. A step can hold diverse values of the scenarios such as context, event and outcome. This implies that if event X occurs the outcome should be Y, but as an example if event Y occurs the outcome should be Z. Only if all steps pass the tests successfully, a scenario can be considered to be finished for implementation, and only if all scenarios of a user stories passed all tests a user story can be implement, and last if all user stories connected to a feature a feature can be implemented.

- **Readable Behaviour Oriented Specification Code**
  BDD introduces a living documentation, in a way that the code is part of the system's documentation. This is achieved through the integration of ubiquitous language, the code methods and class name should always be hold of descriptive text of their intention and align with the naming of features, user stories and scenarios, this way readable behaviour oriented code can be generated that can act as a living and self-updating documentation, due to the automated testing.

- **Behaviour Driven at Different Phases**
  The previously presented characteristics find it’s place at diverse phases within the cycle of software development, which means that the exploration of an application’s behavior is happening at all times during the project. At the Initial Phase, it is the business goals and values that define the behavior. At the analysing phase, the business values are decomposed into features. Furthermore the implementation phase follows, the naming of scenarios, which follows the introduces
vocabulary, and accordingly the names describe the intentions, thus the expected behavior.

2.2 User Experience Strategy Theory

At the core of UX strategy stands the vision, a solution to a problem, that is faced by users. This vision, and the fact that here is the need of a product that presents a solution has to be validated, to ensure that there is a need within the market to develop a product at all. This implies that the focus of User Experience Strategy is on researching the business domains and the market rather then just following general assumptions made by the business team. Levy (2015) describes User Experience Design as a process that already begins within the discovery phase of a project, accordingly to Levy (2015) the output of the discovery phase should already consist of empirical data, that is gained through input of the end user rather then going directly from idea to the first conceptional wireframing and development. This is especially important since it is the output of the discovery phase that can be the crucial factor of how a product will ultimately deliver value. Levy (2015) therefore introduces the four tenets that build the UX strategy framework, which consists of Business Strategy, Value Innovation, Valitated Research and “Killer” UX.

2.2.1 Tenet 1: Business strategy

The Business strategy is defining the margins of a product, its placement in the market towards competitors and its potential to grow. A business model canvas can be used as a tool to identify and gather all information that define these margins. It is not a tool that if once worked through is finished and moved to documentations, its rather used as a tool to identify how a company and a product can differentiate itself from others, which is one of keys to success and longevity of a finished product. The different parts of a business model canvas are the following:

- **Customer segments:**
  - Who can be included within the target group?
  - What are their behaviors?
  - What are their problems and needs?

- **Value proposition:**
  - In which way and what value can be delivered?
  - Is the delivered value qualitative or quantitative?

- **Channels:**
  - Through which marketing channel can the target audience best be reached?
  - Are those online or offline channels?

- **Customer relationships:**
  - How can customers be gained and retained?

- **Revenue Streams**
  - How can the business generate financial income from the value proposition?
  - Are the customer segments required to pay for the product?
  - Are there other options to ensure positive financial health?

- **Key resources**
Theoretical background

- What are the USPs and ESPs?
- Which assets are necessary to publish the product?

- Key activities
  - Which unique strategies and activities are necessary to achieve the value propositions?

- Key partnerships
  - Are there strategic partnership that would help achieving the value proposition?

- Cost structure
  - What are the major costs that need to be covered in order to bring the product to realization?

The business model canvas can be seen as a collection of hypotheses that are to be proven and revised during the discovery phase of a project. Through constant testing and validated research, as further discussed later in this chapter, the different segments within the model, are adjusted accordingly, in order to have proof of concept rather than following assumptions, which will partly ensure and influence the outcome of further steps (Levy, 2015).

2.2.2 Tenet 2: Value Innovation

Products generally need to fulfill two high level requirements. The first requirement is the necessity of some kind of value that the target audience can draw from using the product, which makes it an attractive choice that they can integrate into their life. The users choice of using the product, enables a company to draw financial wealth from the product, which enables it to keep the product alive, and offer its services beyond the development. Value innovation therefore is the result of perfect coordination of originality, utility and price, or as Chan Kim and Mauborgne (2015), define it: “the simultaneous pursuit of differentiation and low cost, creating a leap in value for both buyers and the company.” Chan Kim and Mauborgne (2015) also illustrate the software market in form of two different oceans, the red ocean and the blue ocean. In this metaphor the red ocean is filled with competition that tries to attract potential customers with all their strength. Vise versa the blue ocean describes the space that has not been occupied yet. Which means that there are no set boundaries nor competition that could hold a product back from reaching its full potential. Levy (2015) therefore concludes that, to enter the blue ocean, it is therefore necessary to create a user experience and business model that go hand in hand and deliver innovation together.

A good example of a company that successfully entered the blue ocean, with it’s innovative business model and unique user experience is Spotify. Before the release of Spotify, Apple controlled the market with it’s innovative mp3 device, the iPod and its corresponding music library, iTunes. The problem with iTunes at that time was, that it was only possible to browse through music at the own local pc with installed software, or on apples products that were able to connect via WiFi. Another problem was that it bound the users to either purchase a song directly in iTunes or upload a song via iTunes from a local pc or mac to the iPod. The fact that: users had to pay for each and every song, that the music service
Theoretical background

did not enable users to be spontaneous and the need to reinvent the market of online music with a new business model, was identified by Spotify. Which lead them to the discovery of an innovative new product, that offers music streaming for a monthly flatrate. With spotify users could listen to music online and offline, browse and discover, be spontaneous without spending huge amounts of money for their personal music library.

Accordingly to Levy (2015), this value innovation can be found in the originality and assemblment of features that a product or software has to offer. Furthermore he introduce 4 patterns of feature alignment that can bring value innovation:

- A product that includes key features that are already used by different competition and relevant UX influencers, but assembles them in a new creative way in order to achieve a goal. (Meetup + a payment system = Eventbrite)

- A product that offers an innovative addon to a already existing platform (Google Maps + Crowd-sourcing = Waze)

- A product that unifies different user experiences into one simple and crucial solution. It therefore becomes the go-to platform for fullfilling a goal, such as Instagram in regards of sharing and browsing online mobile content.

- A product that enables a conversation between two different user segments in order to fullfill a deal, such as Uber which brings travellers and providers together.

2.2.3 Tenet 3: Validated Research

As described in the beginning of this chapter, software or products that can only identify it’s value through assumptions made by stakeholders are often an instance of failure. Research of the target audience is therefore a make-or-break factor. User research can not only result in an estimate that describes the target audience, but can also validate the value proposition that is definend in the buisness canvas plan. In this context validation implies the researched audience can or can’t draw value from the product. Ries (2011) motivates to move away from traditional user research methods such as; the development of personas, card sorting and user oberservation. Ries (2011) instead focuses on a concept that provides direct feedback from cooperation with end users.

Lean (2015) introduces the MVP, a Minimum Viable Product, in form of a prototypes that only contains the core features. Testing the MVP against end users with a positiv result would therefore proof that the core concept and value is accepted by the users or vice versa an idicator that the created value is not sufficient to fullfill the users needs. This comes with the advantage that there is no time wasted on a complex Ux prototype that includes all functionalities, before their necessity has even been validated. Testing the MVP against many individuals ensures that an entire segement of users, accept the value proposition not only a single persona, in addition the in the repition of testing the MVP new features can be added up and validated, which is also called the
feedback loop of build-measure-learn. This ultimately would confirm that the product vision is aligned with the target audience’s needs.

### 2.2.4 Tenet 4: "Killer" UX

Accordingly to Levy (2015) a general definition to UX is; User Experience is the way a person feels while operating through the UI of a digital product in order to fulfill a task or achieve a personal goal. UX often makes use of conventions in parts of information architecture and interaction design, but innovation can not be achieved by sticking to conventions. Thus Innovation demands the experimentation of some conventions to reveal new and perhaps better ways. This habit of classical UX-strategy often results in prototypes that only adress the issue of user engagement and design, and forget to consider the value proposition, which includes the main motivations why a user is using a service or product. Another common problem is, that it is often underestimated how important a coherent user experience really is. The best marketing strategies in the world, will fail to gain new customers, if in the moment a users starts using the product the UX-strategy fails do convince.

As Levy (2015) further explains; "Killer" UX-strategy can be differentiated from ordinary UX-strategy in way that it coordiantes the value innovation with different practices as listed below:

- Work is conducted in collaboration of stakeholders, project team members from the beginning of a project. After the value proposition has found its form in a first draft a UX-strategyer is involved in the process of designing the first MVP, which is then used for validating and redefinition of the value propositions. The results can act as crucial indicator for further design decisions.

- Since the UX-strategyer is involved in the process of developing the MVP, the UX-strategyer is also impacting the definition of key features that are vital to delivering value to the user. Through Techniques such as storyboarding, which is a way to gather all features at one place with illustrative examples and cherry picking the best features from competion with the intend to improve them UX-strategyer help discover innovation.

- UX-strategyers know everything about the market of digital software and UX-strategy. Which enables them to create and use User Experience to its full potentials.

- Through MVP testing UX-strategyers are in constant contact with the end users wich enables them to evaluate their feedback, adapt solutions accordingly and lastly validate features.

- They create consistent User Experience in all places a user interacts with the software product, online and offline.

### 2.2.5 Summary

UX strategy is much more than just executing a thought through plan, its first and foremost about collaboration between everyone involved in a project. Analyzing the market and taking the findings into consideration, is only the beginning of a process that requires constant adaption and improvements.
Until validation can ensure that the software’s vision clearly addresses a need that people have, instead of being a wild idea. The outputs or products of UX-strategy are therefore validated and researched UX, prototypes, user flow charts, wireframes, user journeys, site maps and information architecture.

### 2.3 BDD and UX

Accordingly to Chris Parsons (2011), the workflows and process behind UX-strategy are not the be compared with the concepts that an agile approach includes. As further explained, most UX practitioeers and designers, create hundreds of wireframes upfront, before all possibilities and factors concerning a product have been discovered.

The main motivation behind using an agile approach originates from the thought, that a project team should have the flexibility to change and discover a product during the process of developing it, and not, vice versa, fulfilling all requirements and expectations that have been defined upfront in a big workshop. Outgoing of the disadvantages concerning BDD’s communication methods and the limits that UX-strategy as a standalone is depended on, it can be motivated that a combination of BDD and UX-strategy is a promising attempt to improve communication in software projects.

### 2.4 Communication Theory

As elaborated before in this chapter, the success of BDD is highly dependent on successful conversations and communication between project teams and individuals, which creates the need to introduce some general communication theories concerning communication, to late form an answer to the research question. Accordingly to Clark and Brennan (1991) both parties of a conversation must share an equal understanding about the conversation matter in order to run a successful conversation. A conversation can be divided into two distinguish segments; a presentation phase where one party communicates information, which is followed by the acceptance phase, where the opposite party confirms the understanding of these information. To conduct a successful conversation it is therefore necessary to share or create a common ground.

Recent Research conducted by Dewan (2015), examines the influence of visual communication as tool to understanding. The importance of information and how these are communicated between individuals, is the essential condition to successful communication. In the context of software development, this implies that all information given, discovered and communicated are vital to the success of a software project. Dewan (2015) further explains, that due to education, the written word has always been, and still is, the obvious choice when expressing, saving and sharing knowledge. It can be argued that the complexity of the written word is cause to interpretation and can easily be misunderstood.

Furthermore, Dewan (2015) elaborates the advantages of visual tools to support most of text driven communication. Pictures can effortlessly be recognized, processed and recalled by the human brain, this is reason to the fact that pictures are stored in two independent ways, one visually and one verbally. Words, contrarily, are only stored verbally, which decreases the chances of
remembering. Dewan (2015) also concludes that the combination of text and illustrations increases the understanding by 98 percent in comparison to only text. It is further explained that pictures as an aiding tool, ease the process of learning and understanding by providing examples and context.

Which implies that the combination of visual tooling and text are most suitable for recalling and understanding information, which is necessary to reach a common ground for a conversation between different individuals. This leads to the insight that, a common ground can best be achieved by using both verbal and visual communication, which will ultimately lead to successful communication (Dewan 2015). In relation to software development, this entails that, BDD can deliver methods and tooling that supports verbal communication, complementary to UX-strategy which could add a visual layer to the communication within software projects.
3 Method and implementation

3.1 Scientific Research Approach

In contemplation of the complexity and youth of Behavioral Driven Development, the most suitable research approach for this study is an exploratory inductive approach. Since research around Behavioral Driven Development is very limited at this time, and previous research has mostly been comparing the benefits of Behavioral Driven Development against other agile software development frameworks, the scope of this research was narrowed down, to the branch that the research questions focus on. Hence, the inductive approach is suitable, since it is not the theoretical framework that is being used to generate empirical data but the research questions that give definition to the collection of empirical data. The empirical data can therefore contribute to utilization and redefine understanding of the research domain, whilst the theoretical framework, in an inductive approach, enables, and is used, to understand and examine the empirical data (Blomkvist & Hallin, 2015).

3.2 Research Design

Accordingly, to Blomkvist and Hallin (2015) the most suitable choice, when conducting inductive research, is a qualitative research design since the gained empirical data aid to explain the complexity of the study and enable exploratory. In the context of Behavioral Driven Development and the narrowed down scope that this study defines, this implies that the collection of qualitative data is necessary in order to formulate an answer to the research questions. The research of this study therefore consists of two independent parts: a pre-study and semis-structured interviews, which are individually explained in greater detail in the sections below.

3.2.1 Pre-Study

Prior to the collection of empirical data a literature study has been conducted, in order to become familiar with the theoretical frameworks of Behavioral Driven Development. Which enabled to identify the scope of the study and the formulation of the research question. Due to the young age of Behavioral Driven Development the only available books, at the time this study was conducted, that concern the topic are: Smart (2015), Nagy and Rose(2018). These books have been used to create a deep understanding of Behavioral Driven Development and form the content for the theoretical framework described in the chapter Theoretical Background. Ultimately the review of the mentioned BDD books lead to the formulation of the research questions.
3.2.2 Interviews

3.2.2.1 Empirical Data Collection

Since the purpose of this study is to explore the BDD framework, semi structured interviews have been chosen as method for the collection of empirical data. Semi structured interviews are suitable since they enable the researcher to establish a certain structure in the conversation with the interviewed person yet leaving room for questions that may arise depending on the answers received, which serves the purpose of exploratory in this research (Blomkvist & Hallin, 2015).

Therefore, in beforehand of conducting the interviews, a semi structured interview guide had been designed which questions serve the purpose of collecting relevant data that enable the formulation of the research questions. Baxter, Courage and Caine (2015) suggest that it is important to make participants as comfortable as possible during an interview in order to generate qualitative content, and receive honest answers, which influences the results. The interviews therefore have been conducted in person. In order to make the participants feel comfortable, it has been chosen to record each individual interview, which all participants have been asked for permission and agreed upon. This way the focus of the interviewer has always been with the participants during each individual interview session. Each individual session lasted a time interval of 30-40 minutes. The full Interview guide can be found under Appendix 1.

Since this research is conducted at Munich Re, the selection of participants does not range several companies. The framework of Behavioral Driven Development includes many diverse roles; therefore, the selection of participants has been made under consideration of these roles. This enables the research method to generate empirical data that cover more than just one view of a specific role but instead consists of several roles ranging different departments that are enrolled into software projects at Munich Re. Table 1 below displays the interview participants and displays each’s participants role.

<table>
<thead>
<tr>
<th>Name of participant</th>
<th>Role at Munich Re</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zahira Adriana Ioana</td>
<td>Requirement Engineer</td>
</tr>
<tr>
<td>Norhoff Björn</td>
<td>IT architect, Scrum Master, Agile Couch</td>
</tr>
<tr>
<td>Gebhardt Evelyn Elisabeth</td>
<td>Test engineer</td>
</tr>
<tr>
<td>Radu Razvan</td>
<td>Technical project manager</td>
</tr>
<tr>
<td>Moosavinezhad Parisa</td>
<td>Senior Software engineer</td>
</tr>
<tr>
<td>Andreas Schuster</td>
<td>Client Manager</td>
</tr>
</tbody>
</table>
3.2.2.2 Analysis of Findings

Since the raw data collection resulted in audio files it was necessary to transcribe such files in order to present them in this research paper. Accordingly to Baxter et. al. (2015) there are three different forms of transcribing audio recordings; verbatim, edited and summarized. For the presentation of the empirical data the edited transcribing form has been chosen. In comparison to verbatim, the edited form does not contain misstatements or word crutches. Additionality it is to be stated that only 3 interviews have been conducted in English. With the motivation to make the participants feel as comfortable as possible in order to generate qualitative results three interviews have been conducted in German. These interviews have then been translated into English after transcription which might have created a contrast in the result.

Baxter et. al. (2015) suggest conducting a thematic analysis when processing textual qualitative data generated by an interview. The analysis of the content in this study will therefore be done manually, without the aid of software analyzing tools. When conducting a thematic analysis, the researcher must examine the interviews in order to develop and identify categories which the answer can be assigned to. The categorization of the answers can then systematically be assigned to the corresponding research question in order answer these.

3.3 Quality of Research

3.3.1 Reliability

3.3.1.1 Interrater Reliability

Interrater reliability measures the degree to which two independent researchers, also referred as coders by Baxter et. al. (2015), can agree when independently analyzing empirical data. Since this research was conducted by one author only, it was not possible to have diverse people conduct the analysis, and therefore the results of the analysis can not be measured with tools such as Cohen’s kappa for interrater reliability, in which the agreement rate of found categorizaton could be measured.

3.3.1.2 General Reliability

Blomkvist and Hallin (2015) argue that the reliability on qualitative methods and data gathering such as semistructured interviews is rather low, since the results can be influenced by different factors. The interviewees mood, knowledge and concentrating are variables that can not be predicted nor duplicated for further studies which influences the results. The evaluation, the interpretation, the manual transcription and the translation of interviews can also influence the final results presented in this study, which is why the reliability can be classified rather low in this sight. Furthermore this study was conducted only at Munich Re with the companies employees. Since BDD is a very young topic to Munich Re aswell, and the company integrates parts slowly into its processes the interviewed employees can only give answers from their knowledge base about the framework, and the newest practical experiences with the framework.
3.3.2 Validity

As motivated before in this section Research Design, choosing a qualitative method such as semi-structured interviews is suitable when conducting inductive research to serve the purpose of discovery and exploratory. The validity of the study can therefore be rated as rather high. Furthermore, the careful choice of interviewees in hindsight of the BDD framework can be seen as positive, since when selecting the participants of the interviews it has been taking into consideration that the BDD framework includes different roles with different responsibilities. Moreover, the conducted semi-structured interview guide has been designed to always steer the conversation in a direction that focuses around the BDD framework and the research question.

3.3.3 Generalizability

The generalizability of this study can be rated as very low. Since the collecting of empirical data originates from only 6 interviews the findings are not generalizable nor redefine the examined BDD framework and its influence on communication. Furthermore, the study has only been conducted at one company, Munich RE, which works in an agile software project manner, in order for it to be generalizable the study had to be conducted with more participants and more companies that work in the field and are relevant for the study.
4 Empirical Data

4.1 Interview with Requirement Engineer

Warm up

1. This call will be recorded are you okay with that?
   Interviewee: "Yes."

2. Could you quickly introduce yourself? Including your Job title?
   Interviewee: "My name is Zahira Adriana Ioana and I am a requirement engineer."

3. For how long have you been working in this field?
   Interviewee: "Here at MunichRe for 2 month."

4. Are you familiar with the concepts of BDD?
   Interviewee: "Yes since it is a popular topic here at Munich RE"

Body of the Interview

1. Software Development at Munich Re includes many different individuals with different roles and responsibilities. Can you identify any kind of issues concerning the communication between different roles when conducting a software project?

   Interviewee: “There are some issues within team communication throughout software projects at Munich Re, it often happens, especially in multi teams, that only one person or team is informed about a change or other decision making, while the other participants have no idea about that change in a project. In depends a lot on the team. Also, there is a language barrier between the German and English-speaking teams.”

   ➔ How could BDD solve some of these Issues?

   Interviewee: “Well that depends, in general the BDD framework establishes much more collaboration between individuals, as the requirement workshops for example. Other than in TDD, BDD uses ubiquitous language to define requirements and acceptance criteria which should be understood by everyone, it also solves the common problem of writing acceptance tests in TDD since code and class names can get very confusing in TDD. Concerning the language barrier, that’s nothing that BDD or any other framework could change.”

2. One strategic aspect of BDD are the so-called requirement workshops, in which the Product owner, a Business analyst and a developer / tester meet in order to discover illustrative examples to a corresponding user story. What is your view on conducting meetings like this?
Discussion and conclusions

Interviewee: “Outgoing from my previous answer, this is going to be of great help. If the participants of such meetings would consist of the team leaders of the different teams involved, it could be a great way to keep everyone involved updated about the process, decisions and other changes of the project. Let alone the opportunity to create a dialogue between business and technical people were important questions can be raised and discussed.”

➔ Since you just mentioned, the business and technical people, how can those two very different groups create a common ground for their conversation?

Interviewee: “The common ground should just be created naturally since they are in an open conversation. It could be case specific, since everyone has a different personality. It could also relay on the available tools for such meetings, it’s really different how meetings are structured, sometimes team leaders have a short presentation, sometimes we use whiteboards or go through a feature file.”

➔ In BDD theory the product of a requirement workshop are user stories and scenarios to a specific feature, could you imagine that UX prototypes of the discussed feature as a tool to create a common understanding would benefit such meetings?

Interviewee: “As I just explained, the availability of tools differs from meeting to meeting, perhaps you could make it a company standard to prepare such in beforehand of a meeting, generally I think that the business folks would benefit the most from that, since they lacking the technical knowledge.”

3. In BDD, features, user stories and scenarios are saved as plain text files. Can you identify any issues with this structure?

Interviewee: “Sometimes it can be hard to understand some requirements, the quality is often directly connected to the individual performing the formulation.”

➔ Could you imagine that the documentation in BDD style could be further supported by connecting textual descriptions with imagery such as the corresponding UX wireframes or prototypes?

Interviewee: “Well, visuals available for each and every feature, user story and scenario sounds like a lot of work to organize apart from the mandatory things, I am not sure how the benefits this would bring weight out against the work that has to be done in order to use them.”

4. Most software projects teams at Munich Re are using a Backlog to organize work task and save features, user stories and scenarios, can you identify any issues with the organization of the Backlog?
Discussion and conclusions

Interviewee: “I don’t see any issues because the features files contain the user stories, so it’s all easy to find and organize. You can also add tags to organize and link things together. Maybe it could be a good idea to not save the feature files in plain text in the backlog but link them to the GIT repository.”

5. In BDD, scenarios are translated into Gherkin, and act as executable specifications, can you identify any issues concerning the understanding of scenarios in Gherkin?

Interviewee: “No technical context is necessary; Gherkin is super clear and easy for everybody. There is a tool called pickle that shows the corresponding feature files.”

6. BDD focuses on breaking down features into smaller chunks (user stories / scenarios), In your opinion how could a high-level view of a feature from a user-story / scenario be drawn?

No answer available due to technical issues with the recording.

7. In your opinion, how could communication within software development projects be improved?

No answer available due to technical issues with the recording.

8. What is your opinion about introducing a visual layer to software documentations?

Interviewee: “In our projects working close with the designs is important. However, it is not yet company standard to link the designs to the backlog items in Azur Dev Ops, maybe that could improve some things.”
4.2 Interview with Client-Manager

Warm up

1. This call will be recorded are you okay with that?
   Interviewee: "Yes."

2. Could you quickly introduce yourself? Including your Job title?
   Interviewee: "My name is Andreas Schuster, I am a client manager for Munich RE"

3. For how long have you been working in this field?
   Interviewee: "11 years, here a Munich RE"

4. Are you familiar with the concepts of BDD?
   Interviewee: "I am somewhat familiar with the theory."

Body of the Interview

1. Software Development at Munich Re includes many different individuals with different roles and responsibilities. Can you identify any kind of issues concerning the communication between different roles when conducting a software project?

   Interviewee: “Problems concerning the communication in software projects at Munich Re do exist. From my point of view, the major problem consists of the requirements, as a re-insurance company, to create a bridge between the demands and wishes that our customers have, that lastly must be translated into requirements that can be understood by our IT teams.

   Another big issue is that I noticed that the business site only has a vague idea how the IT department can help us to realize a new product. When the IT then receives the requirements created by the business we do not know, which parts are easy and which parts are complicated, time and money intensive for the IT people. The complexity of many software projects is us, as the businesspeople, often not known.

   On the positive site are the daily standups that we are currently conducting. Every morning we take 15 minutes to talk about what has happened the day before, what is planned for the current day and the general overview of the project we are currently working on. Since these meetings are seated with central roles that are involved in the project it results in the clarification of many questions."

   ➔ How could BDD solve some of these Issues?

   Interviewee: “BDD makes a lot of use of illustrative examples, the user stories and scenarios I mean. So as the name of BDD suggests, illustrating the diverse behaviors of a software application probably aids the business side a lot. I’m just not so sure how much the development site benefit from this, or if it even complicates the implementation for them. Other than that, I think it is great that BDD
2. One strategic aspect of BDD are the so-called requirement workshops, in which the Product owner, a Business analyst and a developer / tester meet in order to discover illustrative examples to a corresponding user story. What is your view on conducting meetings like this?

Interviewee: “That would be very good, I’m not so sure about the difference to the just mentioned daily standups, but it would be a great concept for cross functional communication with the project owner involved. I noticed throughout my career that it is vital for a project that the project owner knows exactly how a project is going and knows every specific detail of a software project. It is often hard to the product owner to illustrate an idea for the rest of the project team so it is understandable, creating a room in the meetings where everyone can get the chance to ask the important questions sounds very interesting to me. The technical people can also educate the product owner about possible complication that could come with the implementation which would prevent time loss even before it happens!”

➤ Since you just mentioned, the business and technical people, how can those two very different groups create a common ground for their conversation?

Interviewee: “I think that is up to the individuals involved, it should be in everyone’s interest and mindset as a professional to be curious and ask about things that are unclear or not even defined, since these meetings offer a room to do so, that should be the way to achieve a shared level of understanding.”

➤ In BDD theory the product of a requirement workshop are user stories and scenarios to a specific feature, could you imagine that UX prototypes of the discussed feature, as a tool to create a common understanding, would benefit such meetings?

Interviewee: ”Feature descriptions are often complex, especially features that reach the complexity of an epic, since we as humans naturally think in pictures I would say that it generally productive to involve pictures, white boards or anything that can visualize the different connections and ease the complexity.”

3. In BDD, features, user stories and scenarios are saved as plain text files. Can you identify any issues with this structure?

Interviewee: “It is often not very comfortable if only text documents are available to understand what requirements are about. Text is always cause to interpretation and often raises a lot of question marks. Describing features textually can often be very confusing. We sometimes are closely working together with the designs, depends on the projects, and in some cases screens that visualize features are available, which
eases the understanding quiet a lot, and they are a nice addon to the textually described feature.

Also, it is hard to even figure out which feature belongs to which step in the project process. It sometimes would be very helpful for the organization and overview to create graphics that clearly show the hierarchy of specific features and how everything is connected, this is close to impossible to read out of pure text files, and yet really important for everyone involved.”

Could you imagine that the documentation in BDD style could be further supported by connecting textual descriptions with imagery such as the corresponding UX wireframes or prototypes?

Answered in previous question.

4. Most software projects teams at Munich Re are using a Backlog to organize work task and save features, user stories and scenarios, can you identify any issues with the organization of the Backlog?

Interviewee: “Something that I really appreciate about organization in the backlog is that there are many small work packages which can be managed within one or two sprints. But that requires of course that the content of the backlog and the logic behind every item within is understood, if there would be a system that would enables us to add illustration into the organization of the backlog that would solve all the issues I just mentioned. Nevertheless, the backlog is a strong tool for organizing work in agile project teams, yet heavily depend on how well it is managed.”

5. In BDD, scenarios are translated into Gherkin, and act as executable specifications, can you identify any issues concerning the understanding of scenarios in Gherkin?

Interviewee: “I haven’t had the opportunity yet to work with Gherkin directly, but I strongly believe that a not IT affine product owner has very little interest, knowledge nor time to understand the entire logic behind gherkin. I therefore believe that a product owner will always be dependent on the help of a business analyst that can translate the Gherkin syntax. I think this also creates the need to identify the business analyst as the responsible person to translate requirements into Gherkin and prepare those statements in a manner that they contain all important information of the requirement’s content yet are as easy as possible to understand. And making things easy in general is very complicated, I could imagine that this will result in a very time-consuming task. It is then important to question the value that is outgoing from using Gherkin, automated testing can and has been done without Gherkin.
In addition, the business owner often does not even get that much into the details, he or she is often busy understanding the developed concept and the product that originated his initial idea with all its complexity.”
6. BDD focuses on breaking down features into smaller chunks (user stories/scenarios), In your opinion how could a high-level view of a feature from a user-story/scenario be drawn?

Interviewee: “No as I explained in a question before, without a good visualization of the hierarchy tree in which features, user stories and scenarios connect with each other, understanding these connections out of listed items in the backlog is usually very hard.”

7. In your opinion, how could communication within software development projects be improved?

Interviewee: “I think that the prerequisite for functioning communication is often the consideration of knowledge of everyone that is participating in a project. For me personal it was hard in the beginning to get used to the agile process of a project, and even more what the IT people are talking about. The aim of communication should always be to find a level where everyone can understand each other, while yet communicating the right things. It would be interesting to find out how this level can always be given when doing software projects.”

8. What is your opinion about introducing a visual layer to software documentations?

No answer, since the interviewee had to leave due to an upcoming meeting.
4.3 Interview with IT Architect / Scrum Master

Warm up

1. This call will be recorded are you okay with that?
   Interviewee: “Yes.”

2. Could you quickly introduce yourself? Including your Job title?
   Interviewee: “My name is Björn Norhoff, Scrum master, Agile Couch and IT architect”

3. For how long have you been working in this field?
   Interviewee: “14 years at Munich RE”

4. Are you familiar with the concepts of BDD?
   Interviewee: “Yes.”

Body of the Interview

1. Software Development at Munich Re includes many different individuals with different roles and responsibilities. Can you identify any kind of issues concerning the communication between different roles when conducting a software project?

   Interviewee: “In general, I would say that communication is the biggest challenge in all projects. The interpersonally and emotional factors as well as documentation of all facts, often aggravate good communication within software projects. It all really matters how technical affine someone is, customers often are not, which complicates communication further, since there are many internal and external influences. Product owners often only have an idea, but very limited knowledge and understanding what it takes to bring this idea to realization. Besides all that, the highest level of problems concerning communication can be reached when the technical world and the business worlds collide.”

   ➔ How could BDD solve some of these Issues?

   Interviewee: “The major reason why BDD differs from other agile software development methods and frameworks is the collaboration it integrates into the development of software, so between the business and technical world there suddenly is a bridge, where the business world is suddenly challenged to take, in a way at least, part in the formulation of acceptance criteria, which in BDD also act as the ground for automated testing, so I would say that BDD offers the possibility that the business side gets forced to get familiar with technical aspects, as well as the technical people who need to demonstrate a deep understanding of the business side.”


2. One strategic aspect of BDD are the so-called requirement workshops, in which the Product owner, a Business analyst and a developer / tester meet in order to discover illustrative examples to a corresponding user story. What is your view on conducting meetings like this?

Interviewee: “It would be nice to conduct such meetings, for me personally that would probably mean that I would get very nicely defined backlog items, backlog items can differ a lot in quality, so the more discussion went into the detail and structure of such the better. So, in general it would be very nice to join all different parties together in a workshop like that, where results and findings are produced in cooperation, that are then documented, in our projects it could get hard since we often miss the role of a tester.”

⇒ Since you just mentioned, the business and technical people, how can those two very different groups create a common ground for their conversation?

Interviewee: “I think that it all comes down to the preparations in order to design the margins in which a meeting can happen, if BDD for example would become company standard everyone conducting such meetings should be educated about BDD and its methods. Furthermore, the available tooling and structure of such meetings plays a big role, that can differ from meeting to meeting.”

⇒ In BDD theory the product of a requirement workshop are user stories and scenarios to a specific feature, could you imagine that UX prototypes of the discussed feature as a tool to create a common understanding would benefit such meetings?

Interviewee: “I think that tooling in general is very important in meetings, it does not have to be prototypes, it could really be anything, a whiteboard, if available UX wireframes or prototypes as just said, anything that can ease the complexity and aid to understand.”

3. In BDD, features, user stories and scenarios are saved as plain text files. Can you identify any issues with this structure?

Interviewee: “I personally really like detailed descriptions, if there is a good architecture in the descriptions and I know that it is not always the case. It can be very helpful to have some posters or similar for orientation for that reason, user flows as big posters in the office can be of great help to gain an overview of the great whole.”

⇒ Could you imagine that the documentation in BDD style could be further supported by connecting textual descriptions with imagery such as the corresponding UX wireframes or prototypes?

Interviewee: “If I may add to my previous answer, user flow charts are always a help, basically everything that can visualize the
Discussion and conclusions

architecture of software in one way or another. So yes, I think that BDD could benefit from that.”

4. Most software projects teams at Munich Re are using a Backlog to organize work task and save features, user stories and scenarios, can you identify any issues with the organization of the Backlog?

Interviewee: “Yes and no. It all comes down to habits and knowledge in that case, businesspeople are generally not working a lot with visual studio for example, so the organization looks a lot like that. Structure is always very important; the responsible persons often have a hard time themselves to navigate through some messes they created. Refining backlogs is a lot of work, not conducting refinements therefore often results in unorganized backlogs that are hard to handle.”

5. In BDD, scenarios are translated into Gherkin, and act as executable specifications, can you identify any issues concerning the understanding of scenarios in Gherkin?

Interviewee: “We took part in a BDD education, with diverse teams from Munich Re. There were some obvious complications depending on the background of the individuals. Gherkin certainly was something that not everyone understood and had to first learn and read about. Other teams even had a lot of troubles to formulate the acceptance criteria in free text. The quality of said highly depends on the individual and their knowledge.”

6. BDD focuses on breaking down features into smaller chunks (user stories / scenarios), In your opinion how could a high-level view of a feature from a user-story / scenario be drawn?

Interviewee: “If you do not have an exceptional product owner who has the overview the whole time, I do not think that it is possible, no.”

7. In your opinion, how could communication within software development projects be improved?

Interviewee: “A lot of feedback always during the projects, generally planned and structured meetings. I think that BDD eases a lot, but it is hard to integrate the framework, so it is necessary to have someone involved that has gathered experience with the framework itself. People in this business are often very afraid of learning new things since they must get out of their comfort zone, and BDD requires that everyone involved is on the same side.”

8. What is your opinion about introducing a visual layer to software documentations?

Interviewee: “Yes I think that would be good, humans think a lot in pictures, and that would certainly make complexities more understandable. It would generate a clear overview.”
4.4 Interview with Test Engineer

Warm up

1. This call will be recorded are you okay with that?
   Interviewee: "Yes."

2. Could you quickly introduce yourself? Including your Job title?
   Interviewee: "My name is Gebhard Evelyn, Test engineer."

3. For how long have you been working in this field?
   Interviewee: "7 years at Munich RE"

4. Are you familiar with the concepts of BDD?
   Interviewee: "Yes but only vaguely."

Body of the Interview

1. Software Development at Munich Re includes many different individuals with different roles and responsibilities. Can you identify any kind of issues concerning the communication between different roles when conducting a software project?

   Interviewee: "Something that can complicate the communication a lot is the communication between the product owner and developers, since both parties, very often, talk about completely different things when discussing a feature or a piece of functionality for example. So to illustrate that, if the product owner tells a developer "I would like to have feature x in my application", the developer acknowledges that, but most likely will have a lot of question when developing and implementing the feature, developers nevertheless often implement the feature anyhow without communicating with the product owner again, which leads to errors.

   This also works the other way around, if a developer explains to a product owner how the technical requirements are and the product owner cannot really follow, he most likely will approve it anyhow, since the developer is the professional in his work, or even worse the product owner believes that he understood everything a developer said, that can lead to a lot of trouble when developing new software."

   ➔ How could BDD solve some of these Issues?

   Interviewee: "I do not think that BDD addresses this issue directly, it is more of a problem that occurs between individuals, there are cases where collaboration works out better and some cases where it does not."

2. One strategic aspect of BDD are the so-called requirement workshops, in which the Product owner, a Business analyst and a developer / tester meet in order to discover illustrative examples to a corresponding user story. What is your view on conducting meetings like this?
Discussion and conclusions

Interviewee: “Yes definitely, if the requirements are discussed in detail, that would certainly improve the communication, and a room is created in which all involved parties can remove all uncertainties and questions. Of course, that would not solve all problems between the different parties, but it would certainly improve a lot, the barricade in terms of understanding and background knowledge between diverse roles still exists.”

➔ Since you just mentioned, the business and technical people, how can those two very different groups create a common ground for their conversation?

Interviewee: “The first step before any other action would be that the involved parties both identify that they are not talking about the same thing in some cases. I would say that this also depends on the individuals taking a step forward and discussing a certain topic or feature they did not understand to clarify that.”

➔ In BDD theory the product of a requirement workshop are user stories and scenarios to a specific feature, could you imagine that UX prototypes of the discussed feature as a tool to create a common understanding would benefit such meetings?

Interviewee: “I don’t think that it would solve all issues I mentioned before, but it might be a good starting point, or thinking in the right direction to use tooling like that.”

3. In BDD, features, user stories and scenarios are saved as plain text files. Can you identify any issues with this structure?

Interviewee: “Generally, I am only partly found of documentation in form of text files in software development and in the backlog, the backlog as well as excel sheets are very often cause to interpretation of the developers who need to implement the textual descriptions of features”

➔ Could you imagine that the documentation in BDD style could be further supported by connecting textual descriptions with imagery such as the corresponding UX wireframes or prototypes?

Interviewee: “It would certainly improve a lot, but I can imagine that the organization of the documentation could quickly turn a lot harder.”

4. Most software projects teams at Munich Re are using a Backlog to organize work task and save features, user stories and scenarios, can you identify any issues with the organization of the Backlog?

Interviewee: “I believe it always works as good as the responsible persons who are working with it. A backlog is a very powerful tool, if the backlog is not maintained on a regular basis it turns into a nightmare of long lists that no one could draw a clear overview from.”
5. In BDD, scenarios are translated into Gherkin, and act as executable specifications, can you identify any issues concerning the understanding of scenarios in Gherkin?

Interviewee: “I think that no matter what language or formulation is used, the more details a textual description contains the less room for misinterpretation is given. I personally do not like templates, they cannot always provide the complexity of software descriptions that is often necessary.”

6. BDD focuses on breaking down features into smaller chunks (user stories / scenarios), In your opinion how could a high-level view of a feature from a user-story / scenario be drawn?

Interviewee: “The connection between features, user stories and scenarios must be clear for the product owner. The product owner should have a clear understanding of what is to be implemented and how it relates to other parts and influences of the software. Else he or she cannot really judge how well a scenario is fulfilling its purpose.”

7. In your opinion, how could communication within software development projects be improved?

Interviewee: “Something that is very helpful are the before mentioned requirement workshops or the dailyss from scrum, even though these meetings only stretch a short timeframe of 15-30minutes, it is vital to meet up every day to discuss all done, pending and future work task and bring the entire time on one side with that. It also offers a moment for questions and discussion which is very important.”

8. What is your opinion about introducing a visual layer to software documentations?

Interviewee: “I do not really know how I would benefit from that in my position as a test engineer.”
4.5 Interview with Technical Program Manager

Warm up

1. This call will be recorded are you okay with that?
   Interviewee: "Yes."

2. Could you quickly introduce yourself? Including your Job title?
   Interviewee: "My name is Radu Razwan, Technical project manager."

3. For how long have you been working in this field?
   Interviewee: "1 year at Munich RE" 

4. Are you familiar with the concepts of BDD?
   Interviewee: "Yes quiet a lot."

Body of the Interview

1. Software Development at Munich Re includes many different individuals with different roles and responsibilities. Can you identify any kind of issues concerning the communication between different roles when conducting a software project?

   Interviewee: “About communication issues, probably one thing would be the interaction between businesspeople and technical people. Because most of the time when we start a project, we discuss with the business to understand the project needs, and when you get to more technical details, then you might have some issues when connecting the dots, due to their technical complexity.”

   ➔ How could BDD solve some of these Issues?

   Interviewee: “I think the ways of BDD install a lot more cooperation between people, so that.”

2. One strategic aspect of BDD are the so-called requirement workshops, in which the Product owner, a Business analyst and a developer / tester meet in order to discover illustrative examples to a corresponding user story. What is your view on conducting meetings like this?

   Interviewee: “Maybe a good starting point would be to compare this against the previous approach, in the past we worked like this; we had design workshops with designers and business guys. Most of the time after design workshops we had a design sprint, but for two weeks there was no communication between the designer and the business or the technical team and the requirement engineer. The parties joined as soon as someone organized it. If you compare those two, BDD puts those people together faster and more controlled. I would expect that everyone who shares their ideas and concerns can express themselves pretty good, so it should be understandable for everyone.”
Since you just mentioned, the business and technical people, how can those two very different groups create a common ground for their conversation?

**Interviewee:** “As I just said, I would expect everyone can express themselves, so there shouldn’t be that much of a problem, the main issues in my opinion is that how we did before we had no control over the process, with BDD we do.”

In BDD theory the product of a requirement workshop are user stories and scenarios to a specific feature, could you imagine that UX prototypes of the discussed feature as a tool to create a common understanding would benefit such meetings?

**Interviewee:** “I think it depends highly on what is being discussed, if its stuff that is barely visible on the frontend there is not really anything to show. Maybe some information architecture or something.”

3. In BDD, features, user stories and scenarios are saved as plain text files. Can you identify any issues with this structure?

**Interviewee:** “No not really in the end we are talking about content produced by humans so it should be easy to understand.”

4. Most software projects teams at Munich Re are using a Backlog to organize work task and save features, user stories and scenarios, can you identify any issues with the organization of the Backlog?

**Interviewee:** “No not really, I don’t see any issues if we can group everything accordingly everything is fine. And of course, there must be a connection between different files, as long as we can keep the structure it’s all good.”

5. In BDD, scenarios are translated into Gherkin, and act as executable specifications, can you identify any issues concerning the understanding of scenarios in Gherkin?

**Interviewee:** “Gherkin can be understood by businesspeople as well; I mean I can understand Gherkin.”

6. BDD focuses on breaking down features into smaller chunks (user stories / scenarios), In your opinion how could a high-level view of a feature from a user-story / scenario be drawn?

**Interviewee:** “Yeah it shouldn’t be complicated to see the big picture, but maybe I see this a bit different cause I’m a more technical person.”
7. In your opinion, how could communication within software development projects be improved?
   Interviewee: “One thing that can be improved could be the response time and feedback, communication could be much faster.”

8. What is your opinion about introducing a visual layer to software documentations?
   Interviewee: “I don’t really see the necessity for that.”
4.6 Interview with Senior Software Engineer

Warm up

1. This call will be recorded are you okay with that?
   Interviewee: "Yes."

2. Could you quickly introduce yourself? Including your Job titel?
   Interviewee: "My name is Parisa Moosavinezhad, Senior Software Engineer."

3. For how long have you been working in this field?
   Interviewee: "4 years at Munich RE"

4. Are you familiar with the concepts of BDD?
   Interviewee: "Yes of course!"

Body of the Interview

1. Software Development at Munich Re includes many different individuals with different roles and responsibilities. Can you identify any kind of issues concerning the communication between different roles when conducting a software project?

   Interviewee: “Of course there are always cases where communication is not perfect or flawless, especially if you are developing in multi teams with business, technical and customer, lawyer and other sides involved. If all team members are not staying synchronized about their work tasks and progresses, that’s when most mistakes happen.”

   ➔ How could BDD solve some of these Issues?

   Interviewee: “I think that first it is necessary to distinguish between IT and business. Requirements for example that are to be implemented to the software, technical requirements, can be documented very well with BDD. But the business requirements which do not influence the software in any way are not very good described in a BDD manner. And that was the case in a recent project, the businesspeople did not have the motivation to read through the BDD structured text files, the interest is simply missing. On the opposite site formulations on the business side are very simple, that is because businesspeople are used to that. So as a conclusion I learned from that, if all requirements are globally written in BDD style, there has to be a responsible person that has to translate the business requirements into BDD style, that could for example be a business analyst or requirement engineer that has high competences in the technical world as well.”

2. One strategic aspect of BDD are the so-called requirement workshops, in which the Product owner, a Business analyst and a developer / tester meet in order to discover illustrative examples to a corresponding user story. What is your view on conducting meetings like this?
Interviewee: “That would be very good, I really like that part about BDD.”

Since you just mentioned, the business and technical people, how can those two very different groups create a common ground for their conversation?

Left out due to shortage of previous answer.

In BDD theory the product of a requirement workshop are user stories and scenarios to a specific feature, could you imagine that UX prototypes of the discussed feature as a tool to create a common understanding would benefit such meetings?

Left out due to shortage of previous answer.

3. In BDD, features, user stories and scenarios are saved as plain text files. Can you identify any issues with this structure?

Interviewee: “I think it should be fine as long as the person who has to read the formulations can understand them. Of course, it can be hard to guarantee that everything is understood. Normally the people involved with BDD have good technical background knowledge, if that is not given it could get complicated.”

4. Most software projects teams at Munich Re are using a Backlog to organize work task and save features, user stories and scenarios, can you identify any issues with the organization of the Backlog?

Interviewee: “It would be nice if the backlog could somehow be automated to ease the organization of it, but unfortunately it is all hand work until this day. In some projects however we were able to link the GitHub repositories to the backlog which was very useful. The content of the individual PBIs differs a lot in quality however and is hard to be assured to be good.”

5. In BDD, scenarios are translated into Gherkin, and act as executable specifications, can you identify any issues concerning the understanding of scenarios in Gherkin?

Interviewee: “That can differ a lot. If I personally formulate business requirements in BDD, in the Gherkin syntax, I do not have a problem with understanding it. But if the businesspeople read those specifications it could happen that a lot of details are not very clear to him or her, in that case technical background knowledge is most likely required to understand the logic and syntax wording, just as the IF statements in script languages.”

6. BDD focuses on breaking down features into smaller chunks (user stories / scenarios), In your opinion how could a high-level view of a feature from a user-story / scenario be drawn?
Interviewee: “Scenarios can be tagged at any times, in order to connect them with user stories and features, which is then visible in the backlog. I think though that might only work on Visual Studio in connection with Agile DevOps, also it depends on very good organization.”

7. In your opinion, how could communication within software development projects be improved?

Interviewee: “I have an example for that. We currently are having issues within a software project. The main issue are the differences. Differences between teams and differences between roles. The question that comes up for is, how BDD can cover the wide variety of diverse meetings and concepts for themes and so on. In a development team for example the issues are completely different to the ones a business team might have. And I am not sure how BDD closes this gap completely for both sides, I do not think that BDD can cover the gap completely.

If BDD can lead to better communication therefore is still to be answered in my opinion. If two software teams are developing a new product it could very well be that BDD can improve a lot, because both teams are in the same area. But if multi teams, with lawyer teams, business teams and an IT team are developing a product together, then it can get difficult.

To explore the possibilities of BDD in an academic context is ok in my opinion but implementing the concepts into praxis to a company like Munich RE would require a long period of testing. Testing if it is accepted by all diverse roles, testing if processes are really improving the end product and so on.

Why would you keep on with something if roughly 70% of the working people don’t want to deal with it? Researching is ok, and even though I like the Idea of BDD a lot, I still see it as a grey area with many question marks.

8. What is your opinion about introducing a visual layer to software documentations?

Left out due to an upcoming meeting of the interviewee.
5 Analysis

This chapter includes the thematic analysis, in which the interviewee's answers, found in the previous chapter, are analyzed. The categorization of the answers is done by identifying trends and equalities between the different answer of each interview. The found categories will then be presented and analyzed. The chapter will be structured with the use of the research question as headings. The related analysis of the categories is then presented underneath the research question in order to answer the research question. Each category further more, if available, will hold a representative quote from the interview.

5.1 How can BDD and the products of UX-strategy be connected in order to improve internal communication within software project teams?

The following categories could be found when analyzing the conducted interviews. The categories range from the identification of common problems that occur within software projects, over solutions offered by BDD up to the possibilities that UX-strategy can offer to support BDD to improve communication.

1. Communication between business and IT
2. Project documentation
3. BDD solutions
4. UX-strategy as tooling for communication

Communication between business and IT

Throughout all interviews it became clear that all participants of the interview from both sites, the business and the technical side could identify the major issue, concerning communication, to be the knowledge and competence gap between the business and technical side when conducting software projects. Accordingly to the interviewees it is very hard to document and communicate requirements, decision and the process between the diverse sites, since there is a huge gap from both parties that influences the understanding of what the other party expects and does. Which is represented in the following quote.

"...the highest level of problems concerning communication can be reached when the technical world and the business worlds collide."

-Björn Norhoff

It could furthermore be identified that throughout the interviews the dependency of individuals knowledge and personality can play a big role for the overall communication. The formulation of requirements, the background knowledge about technical aspects and the motivation of getting familiar with the product are all factors that influence the outcome of communication between diverse roles.
Discussion and conclusions

Project Documentation
In sight of the project documentation it could be identified the opinions concerning the documentation strongly differ between interviewees with a technical background and interviewees from the business side. In detail it was identified that the parties shared opinions about the plain format of documentation in plain text, the opinions however scatter apart concerning the the backlog organization and gherkin. Concerning the plain text there can be found representative quotes that showcase the contrast between the parties.

"Text is always cause to interpretation and often raises a lot of question marks.”
-Andreas Schuster

“Sometimes it can be hard to understand some requirements, the quality is often directly connected to the individual performing the formulation.”
-Zahira Adriana Ioana

Both quotes above can showcase that the documentation in form of plain text files is cause to interpretation. Furthermore, it can be identified that the formulation of descriptive tests is depended on the skills and abilities of the individuals. The different opinion about gherkin can clearly be identified to be depended upon a technical or non-technical background of the interviewees. The interviewees with a technical background agree upon that Gherkin is a very easy syntax that should be understood by everyone, as seen in the following quote for example:

“Gherkin can be understood by businesspeople as well; I mean I can understand Gherkin.”
-Radu Razwan

In contrary to the representative quote above, people from the business side stated that even after education with the formulation and structure of gherkin it can be hard to understand the executable language.

“There were some obvious complications depending on the background of the individuals. Gherkin certainly was something that not everyone understood and had to first learn and read about.”
-Björn Norhoff

Concerning the backlog, it is mostly agreed upon to be a strong tool In many occasions it is mentioned that the quality differs a lot in quality depending on if people from the business side or the technical side have been conducting the formulation of feature definition and requirements, organization and maintenance. As a summary it can be identified as a strong tool, that is hard to maintain and organize which is also summarized in the following quote:

“I believe it always works as good as the responsible persons who are working with it. A backlog is a very powerful tool, if the backlog is not maintained on a regular basis it turns into a nightmare of long lists that no one could draw a clear overview from.”
-Gebhardt Evelyn
BDD Solutions
Throughout all interviews it is obvious that all participants agree upon the solutions that BDD introduce to solve communication issues within software project teams. The most common problem, mentioned in Communication between Business and IT above, is the diversity in knowledge and competences between the business and IT side which leads to a lot of communication issues. Accordingly to the interviewees BDD solves these problems with the integration of requirement workshops, which involves representatives from all parties involved in a software project. In some occasions it is also mentioned how BDD eases the process of software projects with the decomposition process, the way that features are broken down into smaller chunks; user stories, which are further broken down into scenario, as a way to illustrate behavior, plan work task accordingly, and ease understanding and communication. It is furthermore mentioned, that through this integrated collaboration a deeper understanding for requirements from the business side is established, and a room is created where everyone can discuss and ask open question daily. In some occasions it is also mentioned that BDD integrates a ubiquitous language into the development of software which creates a shared vocabulary that should ease the understanding of descriptions.

“Well that depends, in general the BDD framework establishes much more collaboration between individuals, as the requirement workshops for example. Other than in TDD, BDD uses ubiquitous language to define requirements and acceptance criteria which should be understood by everyone, ...”

-Zahira Adriana Ioana

Between all interviews, one specific answer, by Parisa Moosavinezhad was especially noticeable, since the interviewee questioned the capabilities and solutions of BDD mentioned above, both the ubiquitous language and the general way of expressing requirements in a BDD manner with the use of templates and Gherkin. Which is presented by the following quote:

“I think that first it is necessary to distinguish between IT and business. Requirements for example that are to be implemented to the software, technical requirements, can be documented very well with BDD. But the business requirements which do not influence the software in any way are not very good described in a BDD manner...”

-Parisa Moosavinezhad

Later in the interview Moosavinezhad, also questions the, through BDD integrated workshops, in her opinion is not yet clear how the framework of BDD and its practices; the requirement workshops and form of documentation can cover all the diverse needs of multi software teams:

“The question that comes up for is, how BDD can cover the wide variety of diverse meetings and concepts for themes and so on. In a development team for example the issues are completely different to the ones a business team might have. And I am not sure how BDD closes this gap completely for both sides, I do not think that BDD can cover the gap completely.”

-Parisa Moosavinezhad
**UX-strategy as tooling for communication**

As a last category that could be identified, which is also crucial for a formulation of an answer to the first research question, it was noticeable, that in addition to the solutions offered by BDD, UX products could be integrated into the process, practices, and documentation as an aiding tool to support communication. The interviewees could identify that there is a need for visualizations that can ease the understanding of complex connections and descriptive text. It could also be identified that a common ground between the business and the IT side is not always given and must be created, which could, among other, be achieved by presenting and integrating prototypes and user flow charts at the requirement workshops. Which can be summarized by the following quotes:

> “The aim of communication should always be to find a level where everyone can understand each other, while yet communicating the right things. It would be interesting to find out how this level can always be given when doing software projects.”  
> - Andreas Schuster

> “... humans think a lot in pictures, and that would certainly make complexities more understandable. It would generate a clear overview.”  
> - Björn Norhoff

> “I think that tooling in general is very important in meetings, it does not have to be prototypes, it could really be anything, a whiteboard, if available UX wireframes or prototypes as just said, anything that can ease the complexity and aid to understand.”  
> - Björn Norhoff

However, it is noticeable that the need for visuals is more desired and recognized by the interviewees with a context from the business side, the answers given by participants with a technical background, only seem to acknowledge the need but do not necessarily see it as a big improvement, but instead identify the problems that are connected to the documentation of UX products in the project documentation. As seen in the following quotes:

> ”As I just explained, the availability of tools differs from meeting to meeting, perhaps you could make it a company standard to prepare such in beforehand of a meeting, generally I think that the business folks would benefit the most from that, since they lacking the technical knowledge.”  
> - Zahira Adriana Ioana

> “I do not really know how I would benefit from that in my position as a test engineer.”  
> - Gebhardt Evelyn
5.2 How can a combination of BDD and the products of UX-strategy improve understanding of features, user stories and scenarios?

In sight of the understanding of features, user stories and scenarios it can be recognized that the opinions differ a lot between the involved parties. People from the business side of a project seem to prioritize the visualizations of complexity more, contrary people on the business site can not always recognize the benefits of such but instead recognize the problems that could occur when combining UX-strategy and BDD. Outgoing of the interviews the following categories have been found:

1. Views on formulation in BDD
2. Overview and understanding of features, user stories and scenarios

Views on formulation in BDD

When analyzing the interviews in becomes clear that there are diverse opinions concerning the formulations in BDD. There are several opinions and views on the formulations. According to the interviewees expressing the documentation in form of text, which’s quality is dependent on the performance of an individual conducting the formulation, will always be cause to interpretation. It becomes clear that the complexity of software products and their functionalities can not be captured without excessive and detailed descriptions, however, the way that BDD expresses these formulations do, according to the interviewees, often require technical background knowledge or a certain education. Especially the syntax of the executable language Gherkin is critically viewed at, from participants at the business side and in some occasions from participants with a technical background as well. Even though BDD defines a language that can be understood by everyone, the interviewees doubt that Gherkin can be understood without education or technical background knowledge, as the quotes below demonstrate.

“... if the businesspeople read those specifications it could happen that a lot of details are not very clear to him or her, in that case technical background knowledge is most likely required to understand the logic and syntax wording, just as the IF statements in script languages.”
-Parisa Moosavinezhad

“Generally, I am only partly found of documentation in form of text files in software development and in the backlog, the backlog as well as excel sheets are very often cause to interpretation of the developers who need to implement the textual descriptions of features”
-Gebhardt Evelyn
Discussion and conclusions

Overview and understanding of features, user stories and scenarios

“...without a good visualization of the hierarchy tree in which features, user stories and scenarios connect with each other, understanding these connections out of listed items in the backlog is usually very hard.”

- Andreas Schuster

“Feature descriptions are often complex, especially features that reach the complexity of an epic, since we as humans naturally think in pictures I would say that it generally productive to involve pictures, white boards or anything that can visualize the different connections and ease the complexity.”

- Andreas Schuster

The excerpts of the interviews above, summarize the diverse answers concerning the understanding of features, user stories and scenarios. As in the previous parts of the analysis it is strongly noticeable that the business side openly communicates the motivations behind integrating visualizations to support the understanding, whilst the participants from the IT side mostly argue about problems that occur when integrating said into the documentations of a software projects. Which can be led back to the dependency on individuals performing maintenance of the backlog. Participants from the business sector often recognize the need for visualization. Due to complexity of features and the decomposition process behind it, it can be hard to see the bigger picture behind certain sets of user stories or scenarios. It was therefore suggested to integrate anything that can visualize the complexity, the diverse connections and display a hierarchy in which all the single features, user stories and scenarios stand, as the quote below displays:

“It can be very helpful to have some posters or similar for orientation for that reason, user flows as big posters in the office can be of great help to gain an overview of the great whole.”

- Andreas Schuster

At this point it can be assumed that, the integration of UX products, that visualize the description of features, user stories and scenarios can ease the complexity and aid the understanding these, but the necessity of doing so is mostly recognized by the interviewees from the business side who lack the technical background. It could therfore be argued that the very specific formulations and templates originating from the BDD framework lack to create a overview of a software project, where products of UX, such as prototypes and user flow charts could support the understanding of a high level view. It is however argued that the amount of work necessary to integrate the visualization in the backlog could not weight out the value it would generate for the product, which is motivated in the following quote:

“Well, visuals available for each and every feature, user story and scenario sounds like a lot of work to organize apart from the mandatory things, I am not sure how the benefits this would bring weight out against the work that has to be done in order to use them.”

-Zahira Adriana Ioana
6 Discussion

6.1 Discussion of method

Since the motivation behind this study was of exploratory nature and inductive approach has been a suitable choice to reach the objective of exploration. It can be further motivated that, the limited literature and research concerning BDD gives further reasoning for conducting exploratory research in an inductive approach. Due to the given timeframe of the bachelor course of two month, conducting a qualitative method for the collection of empirical data has been suitable, which resulted in the design of a semi structured interview. Since the study serves exploratory purposes the Interview guide has been designed to explore BDD, in detail how BDD improves communication and furthermore where it lacks to improve communication. Therefore the interview guide aims to unveil the issues concerning the communication and understanding that experts in the industry have to face when using the BDD framework, either from practical, related or theoretical knowledge. It could be argued that the design of the interview guide in this manner is risky, since it does not guarantee that an answer to the research question will be possible after conducting the interview, since the findings, and therefore the data which lead to an answer of the research question are very dependent on the specific answer the interviewees give in an interview. It can still be argued that in order to serve the exploratory purpose and expand the knowledge around BDD, it would have been much more suitable to conduct long term experiments, that could showcase the weaknesses in the BDD framework concerning the communication, and how these weaknesses could be filled or improved by connecting BDD with UX. In comparison to the conducted interviews much stronger insights and deeper knowledge could have been gained.

Concerning the choice of analysing method there is no doubt that a thematic analysis is suitable when evaluating qualitative collected empirical data, and can therefore be seen as a suitable method for the analysis part in general. It can however be argued that the recognized themes partly have strong familiarities with the questions set in the interview guide, which lowers the quality of the thematic analysis.

Reliability

Since the chosen method has been interviews the reliability of the study can be considered to be rather low, since interviews are often cause to diverse influences that could influence the results. Considering the selection of participants there are several aspects to mention that could have influenced the results. Firstly the interviews, have been conducted during normal working hours at Munich Re, which means that some participants might have been under stress, when conducting the interview and therefore rushed answers, or gave short answers. Furthermore, the interviews have partly been conducted in english and partly in german, during the translation process relevant data could have gotten lost and therefore could have influenced the presentation of the findings in this paper. Since the interviews have been recorded the process of manual transcription could also have influenced the presentation of the findings in this research paper. Something that could increase, the reliability in sight of the participants is that every participant is a experienced professional in the field.
and could give educated answers based on theoretical knowledge and practical working experience within the industry and with work of the BDD framework. Since the BDD framework generally includes many diverse roles from the business and technical side, thus it can be motivated that the choice of participants who originate from diverse roles within software project teams has positive influence on the reliability of the study. Furthermore, the fact that the same interview template has been used throughout all interviews indicates that the opportunity to answer questions for each participant has been equal, which could also improve the reliability.

**Validity**
Because of the exploratory purpose of the study and the choice of a qualitative data collection method which aligns with the purpose, the validity of the study can be rated to be rather high. Through the design of the semi structured interview and the selection of diversity in roles of the participants it was possible to collect qualitative and diverse empirical data.

**Generalizability**
The collection of the empirical data solely are gained from six conducted interviews with employees of Munich RE in Germany. The results and findings can therefore not be generalizd, to the industry, the BDD framework or any other way. The study can therefore only deliver an impression of the study objective and its results that is applicable for the internal structures and opinions at Munich RE. To increase the generalizability the study would have to be conducted in several diverse company within the industry in order to generate findings that are applicable to the industry and the BDD framework in general.

### 6.2 Discussion of findings

The purpose of the study was to explore how BDD solves communication issues and explore how products of UX-strategy could further improve communication and understanding, which lead to the following research questions:

*How can BDD and the products of UX-strategy be connected in order to improve internal communication within software project teams?*

*How can a combination of BDD and the products of UX-strategy improve understanding of features, user stories and scenarios?*

When analyzing the empirical data, it became quiet clear that there is a obvious difference between the opinions of technical affine participants and people from the operative side of the business. However the findings could display that there are also some shared opinions. All participants could identify that a major problem as a communication barrikade between the diverse teams involved in software projects, which is due to the diversity of knowledge, and more precesicly most often the lack of technical background knowledge. The ways that BDD introduces into agile software development includes strategic
workshops that install more cooperation between parties, a unified way of expressing formulations, documenting, communicating functionalities, objectives and requirements of a software project. One of the participants, Parisa Moosavinezhad, however could stand out with an unique opinion, that BDD is a method developed for people with a technical context and understanding, and that it is unclear to which extent the business side can use BDD. Which in the motivations behind BDD implies a huge gap, since the decomposition process, which starts of with the business objectives which, are composed into features all the way to the before in this report mentioned step definitions, yet, later in the process BDD, fail to take the capabilities of the business side into consideration. When analyzing the findings it became clear that the formulation, the documentation and the possibilities to use the framework in multi teams are very limited, and that without further education the unified language and templates used are not accessible or usable for anyone involved. The interviewees with a technical background could not really identify the need to integrate further support that aids the understanding and communication, but instead showed to acknowledge that the framework itself is not perfectly suitable for the teams on the business side. In comparison, the interviewees from the business side, and partly from a technical background as well, could recognize the problems, that the BDD framework creates in sight of communication, in which points it can improve it and in which points it could need further support. The main issues for the business side, apart from the before stated general problems, were the unified language that BDD uses, especially Gherkin. It was argued that without further education, there are no advantages or motivations for using this syntax to express scenarios. Another problem that was specifically highlighted was the possibilities to get a clear overview after decomposition process behind BDD. The complexity, the connection between features, user stories and scenarios and the hierarchy of such therefore would be very hard to identify by the business side. The only possibilities that would be available to do so would be the interpretation of the backlog, which’s quality is highly dependent on maintenance. These problems however lead the interviewees to the expression of wishful tools and ways that could improve the communication and understanding for people without a technical background.

As the interview had been designed to explore and identify, where the main issues of communication are currently, how BDD solves them and how BDD fails to solve them, it can be motivated that the logic is capable of producing findings that can answer the research questions. The presented findings and conducted analysis however was only capable to present the ideas and opinions about how UX could be integrated into the processes of BDD. The interviewees therefore could mostly agree upon that it is necessary to create a common ground for a successful conversation and that it would be very helpful to integrate UX products such as prototypes, user flow charts and wireframes into the documentation and use them as a tool to ease communication, create a shared understanding and ease all the complexities that the BDD framework entails. It could be argued that the findings and the analysis do only satisfy an answer to the research questions on a very vague level and do not present very concrete or precisely defined rules, processes or methods of how BDD and UX-strategy can be connected. It however can be motivated that the answer to the research
Discussion and conclusions

questions is highly depended on the quality of data received from the answers of the interviewees. And since the purpose of the study was of exploratory nature, and therefore a qualitative method is suitable, it could be concluded that the presented answers to the research questions satisfy the purpose of the study to a full extent. The analysis of the findings clearly displays the gaps, where accordingly to the interviews, issues can be recognized and furthermore hint possible solutions for closing such gaps with the products of UX-strategy and offer other suggestive ways in which UX-strategy could support BDD. As a summary it is to be argued if or if not the findings can satisfy an answer to the research questions, the found data and presented answers however, can align with the purpose of the study and answer the research questions to that extend.
7 Conclusions

The conducted research examines the principles and theories of BDD and UX-strategy in order form a basic understanding for the theoretical frameworks. Which leads to an understanding of the subject and objective of the conducted research.

Concerning the choice of method, it can be concluded that the method has been suitable to serve the exploratory purpose in an inductive research approach and has been enabling to present empirical data that could be analyzed in order to offer answers to the research questions. Furthermore, it can be concluded that the used method has been suitable for the given timeframe, but as an insight gained from conducting the used method and analyzing the results, it can be concluded that extensive field studies, testing and experiments with the framework could unveil more in depth knowledge about study objectives.

As a conclusion from the findings can be drawn that; BDD is a suitable framework, that in comparison to it’s predecessors improves and solves a lot of known problems. The findings however lead to the insight there are still problems concerning communication within software development teams. While the communication barricade between the diverse teams involed in the development of new software is the most prominent problem found in the empirical data, there is also a obvious diversity in opinions concerning other communication issues. The business side within software project is mostly facing issues that occur due to their lack of technical background knowledge, while the IT teams seem to have very complications with the principles, methods and processes of the BDD framework. It can be concluded that apart from the practices introduced by BDD, communication and understanding within software project teams could further be improved through the integration of UX-strategy, that clarify hierachys, complexities and connections between the diverse functionalities of a software. Through the deployment of visual elements a common ground between parties, and understanding for requirements and project specifications could be more easily reached, yet create new challenges in the documentation of software projects that are yet to be unveiled. Since it has been explored and presented that it could be valuable to include a visual layer to the project documentation and integrate such as a tooling for workshops and other, the following questions arise.

How could generalizable rules be defined, that can be used to decide upon when a visualization is necessary, that aids the purpose to support communicating and understanding?

How can the processes of BDD and UX-strategy be merged together, to solve the issues concerning software project documentation?

As a conclusion that can be drawn from this study it is to be said that the framework of BDD is yet to be fully explored, and that this study can only contribute new knowledge to a small extend. Further studies that are connected to this study could elaborate and research upon the questions above.
8 References


Blomkvist Pär, Hallin Anette (2015) *Methods for engineering students – Degree projects using the 4-phase Model*

Kathy Baxter, Catherine Courage, Kelly Caine (2015), *Understanding your Users – a practical guide to user research methods*.


https://www.belatrixsf.com/whitepapers/project-behavior-driven-development/
Retrieved 9 March 2019

Retrieved 20.4.2019
http://damiantgordon.com/Methodologies/Papers/A%20Study%20of %20the%20Characteristics%20of%20Behaviour%20Driven%20Develo pment.pdf

Retrieved 21.4.2019
https://pdfs.semanticscholar.org/4f03/eco675d08cfd1ecdbaac3361a29 d756ce656.pdf

Mathias Soeken, Robert Wille and Rolf Drechsler (2012). *Assisted Behavior Driven Development Using Natural Language Processing*
Retrieved 18.4.2019
https://link.springer.com/chapter/10.1007/978-3-642-30561-0_19#citeas

Dan North (2006). *Introducing BDD*
Retrieved 21.4.2019
https://dannorth.net/introducing-bdd/

Jönköping University (2019). Course: *15 Credits Final Project Work in Informatics*
Retrieved 30.4.2019
9 Appendices

9.1 Interview Guide

Introduction
1. Introduction of the conducted work and field of study and research questions.
2. How can BDD and the products of UX-strategy be connected in order to improve internal communication within software project teams?
3. How can a combination of BDD and the products of UX-strategy improve understanding of features, user stories and scenarios?

Warm up
1. This call will be recorded are you okay with that?
2. Could you quickly introduce yourself? Including your Job title?
3. For how long have you been working in this field?
4. Are you familiar with the concepts of BDD?

Body of the Interview
1. Software Development at Munich Re includes many different individuals with different roles and responsibilities. Can you identify any kind of issues concerning the communication between different roles when conducting a software project?

⇒ How could BDD solve some of these Issues?

2. One strategic aspect of BDD are the so-called requirement workshops, in which the Product owner, a Business analyst and a developer / tester meet in order to discover illustrative examples to a corresponding user story. What is your view on conducting meetings like this?

⇒ Since you just mentioned, the business and technical people, how can those two very different groups create a common ground for their conversation?

3. In BDD theory the product of a requirement workshop are user stories and scenarios to a specific feature, could you imagine that UX prototypes of the discussed feature as a tool to create a common understanding would benefit such meetings?
4. In BDD, features, user stories and scenarios are saved as plain text files. Can you identify any issues with this structure?

5. Could you imagine that the documentation in BDD style could be further supported by connecting textual descriptions with imagery such as the corresponding UX wireframes or prototypes?

6. Most software projects teams at Munich Re are using a Backlog to organize work task and save features, user stories and scenarios, can you identify any issues with the organization of the Backlog?

7. In BDD, scenarios are translated into Gherkin, and act as executable specifications, can you identify any issues concerning the understanding of scenarios in Gherkin?

8. BDD focuses on breaking down features into smaller chunks (user stories / scenarios), In your opinion how could a high-level view of a feature from a user-story / scenario be drawn?

9. In your opinion, how could communication within software development projects be improved?

10. What is your opinion about introducing a visual layer to software documentations?