The Impact of Agile Practices on Team Interaction Quality – Insights into a Longitudinal Case Study

Full Paper

Christian Jentsch
University of Bamberg
christian.jentsch@uni-bamberg.de

Abstract

Since several years there is a trend of reorganizing the process of software development, by applying agile practices like scrum. These practices attempt to increase customer responsiveness and adaptability of the organization. In a longitudinal case study, I discuss the impact of commonly applied scrum practices on team interaction quality. I collected data in a software development project at three different times during the project. The findings indicate the potential longitudinal impact of scrum practices on team interaction quality and performance. First, the results indicate that daily meetings cannot fully compensate the benefits of colocation, which improves the information flow in a team. Second, the quality of the requirements development approach correlates with the degree of requirements fuzziness. Third, staff turnover does not necessarily impact team performance negatively. Fourth, scrum practices can even benefit in a highly regulated environment, in which agility is not an objective.

Keywords

Scrum practices, team interaction quality, longitudinal case study, software development

Introduction

Faster, better, agile. Many firms invest a lot of money and effort in restructuring the internal software development processes to react flexibly on changing user requirements. In the last few years agile practices (like Scrum) have been widely adopted, to achieve these goals (VersionOne 2016). In contrast to traditional software development approaches, agile approaches are defined by an iterative process in which the final system is planned, build and tested in small cycles (or software pieces) rather than in one sequential process of planning, building and testing (Anderson 2004). Previous research have found that agile (scrum) practices, like team autonomy or pair programming, increase the responsiveness of the software team (Lee and Xia 2010), enables a backup behavior and shared understanding in the engineering team (Schmidt et al. 2014) and increases the success rate of IT projects (Serrador and Pinto 2015). As critical success driver research highlights agile (scrum) practices as enablers to improve the quality of interactions and communication in the team. However, these results mainly show situational snapshots in companies by collecting data at one specific point in time. This kind of snapshot data collection makes it difficult to understand the development and reasons for changes of the team interaction quality in a project. By applying a process oriented analysis, the ups and downs of team interaction quality and performance can be determined. To interpret these ups and downs I state the research question as followed:

What is the impact of agile practices on the quality of team interaction in an IT project life cycle?

To answer this research question I collected mixed longitudinal data (polls, interviews and observation) in a software development project managed by scrum practices. After a brief introduction of the theoretical concept, I describe the respective project and provide insights into the data collection process followed by a presentation and discussion of the findings. Finally, I summarize the findings and highlight the contribution and limitations as well as further research based on this case study.
Theoretical Background

Team interaction quality

Team interaction quality refers to the social aspects in a team that defines the team culture and enable the performance of a software development team. Many studies focus on these social bindings in teams (e.g. He et al. 2007; Mathieu et al. 2000; Robert et al. 2008). A useful theory to explain these social aspects has been described by Nahapiet and Ghoshal (1998). The social capital theory has been applied to explain the development of business/IT mutual understanding (Wagner et al. 2014), the implementation of clan control in complex IT projects (Chua et al. 2012), or social integration in digitally enabled team (Robert et al. 2008). The theory conceptualizes structural capital, cognitive capital and relational capital.

Structural capital, which describes the “overall pattern of connections between actors” (Nahapiet and Ghoshal 1998, p. 244) has been analyzed by the frequency or intensity of interactions (e.g. Chua et al. 2012; Wagner et al. 2014) and can be influenced by mechanisms that enable communication. Cognitive capital represents the degree of similar interpretation of the external environment (Nahapiet and Ghoshal 1998). To specify the similarity of interpretation patterns I apply the concept of team mental models. Based on Klimoski and Mohammed (1994) team mental models “refer to what team members know about what their other team members know and share in common.” (p. 432). As suggested by Mathieu et al. (2000) I differ between task and team-based mental models. Task-based models focus on the understanding of the underlying task (like goals or work packages); team-based models analyze the understanding regarding teammates (Cannon-Bowers and Salas 2001). Last, relational capital focuses on the relations between persons, like trust or respect (Wagner et al. 2014). I apply the conceptualization of Robert et al. (2008), who described relational capital by the level of trust, team norms, team identification and obligation. In addition, I extend this conceptualization by quality variables focusing on conflict resolution (Goo et al. 2009) and information quality as suggested by Xu et al. (2013).

Agile software development

The basic idea of agile practices is to increase the flexibility of the development process by managing the team autonomy or continuous adaption capability (Lee and Xia 2010), which can be enabled through an iterative and incremental development. Based on this idea several agile practices have been emerged, like extreme programming, Scrum and Kanban. The agile methods comprise practices to increase the delivery capacity in the software development (Pikkarainen et al. 2008). This research focuses on scrum practices, since most (70%) of the current agile managed IT projects are managed by scrum or scrum-related practices (VersionOne 2016). Basically Scrum comprises five events (the sprint, sprint planning, daily scrum, sprint review, retrospective), three artefacts (product backlog, sprint backlog, increment) and three roles (product owner, development team, scrum master) (Schwaber and Sutherland 2016). Details about concrete scrum practices will be presented in the case description.

Research Methodology

This explorative study examines the linkage between agile scrum practices and the development of social capital in a software development team. To answer the research question, I collected mixed data (structured polls, semi-structured interviews and unstructured observations) in a software development project. The team applies agile scrum practices since several years. I had the opportunity to collect data in a project which started in August 2016 and was (officially) completed on the 1st of January 2017.

Case Description

The German company, called “doctorfit” in this paper, develops software for medical practices and hospitals. The software covers modules like calendar systems or work flow management systems. In total 800 people work for doctorfit. The market for software in medical practices in Germany is very specific due to its high level of governmental regulations, which in turn has led to a very small number of software providers in this sector. Furthermore, the medical practices have no or only very little influence on the features of the software. Either they use this system or they use no software system at all.
Because doctorfit cannot gain a competitive advantage on this market by innovative software features, doctorfit invests into the design of the features. To maximize the quality of the software design, the software development unit changed their development approach from a traditional waterfall approach to scrum-based approach around eight years ago.

The analyzed project started in August 2016. In a kick-off meeting, the project management office presented the governmental requirements. The goal was to develop an assisting system which guides the doctor through a prescription for a therapy. The problem, which the government attempts to address, is the large freedom for the doctors, when filling out the prescription. The doctor could write whatever he/she wants. The patient then went with the receipt to the therapist, who often could not understand the details of the prescription. Thus, the assisting system provides guidance and defines rules for the completion of a prescription. Based on the governmental requirements, the team defined their minimal system, which represents around 70% of the functionalities for the planned system. The other 30% are no governmental requirement, but features that add further value for the customer. The team had a very strict deadline. On the 1st of December the software had to be submitted for the governmental approval. If the software would not match the requirements, the software could not go live on the 1st of January 2017. Doctorfit would have to wait for the next round end of 2017. As already mentioned the team applied scrum practices to organize the development. The main practices are listed in Table 1.

<table>
<thead>
<tr>
<th>Scrum practice</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incremental development</td>
<td>The team applies the concept of user stories, to break down the software into smaller pieces. The whole team collaboratively agrees on a broad categorization for the user stories. Next, the product owner starts compiling the single user cases and forwards each case to the testers. After the tester defined the test cases, the user story is being developed and forwarded to the testers, who test the software piece. If the tests are positive, the software piece can be evaluated and approved by the product owner.</td>
</tr>
<tr>
<td>Story Slicing</td>
<td>The user stories are ‘sliced’ into small functionalities. A specific set of functionalities represents one user story.</td>
</tr>
<tr>
<td>Test-driven Development</td>
<td>The product owner describes the functions and basic requirements in form of story slices. Next the testers expand the story slice by several functional test cases and forward the story slice and test cases to the software developer. The software is being developed based on the story slices and the corresponding list of test cases.</td>
</tr>
<tr>
<td>Backlogs</td>
<td>All user stories and slices are collected and visualized digitally on a story board. The backlog or story board provides information about the single user stories and slices which are uploaded to the system by the product owner. The information cover a description of the feature, test cases and status in the development cycle.</td>
</tr>
<tr>
<td>Active Stakeholder Participation</td>
<td>The product owner and the product manager represent the customers, who define and provide feedback on the software features. In addition, three functional testers work for the team. All testers have been employed for several years in medical practices or hospitals and know the daily business very well. In addition, doctorfit has several beta customers, who test the software in their practice before the release day.</td>
</tr>
<tr>
<td>Restrospective</td>
<td>The team has a box in their office, in which they can post social issues (and conflicts) between the team members. Every two weeks the team comes together, to discuss the issues which has been posted.</td>
</tr>
<tr>
<td>Daily Scrum</td>
<td>Every day the team discusses current tasks within a 15 minute session. The objective of this session is to provide an update of the current tasks.</td>
</tr>
<tr>
<td>Colocation</td>
<td>All team members work together in one office room.</td>
</tr>
<tr>
<td>Scrum Master</td>
<td>The responsibility of the scrum master is to manage the daily scrum and retrospectives. The scrum master only intervenes if the rules of the meetings are not met.</td>
</tr>
</tbody>
</table>

Table 1. Scrum practices applied in the IT project

Table 2 lists the team members of the scrum team. All testers are functional testers. I collected poll data at three different times during the project. The column ‘Poll Data’ indicates the poll data, I received from each team member. The data collection process will be explained in detail in the following section.
### Data Collection Approach

To analyze the impact of agile practices on the development of interaction quality, I applied a mixed method approach in a longitudinal case study. The data collection approach is visualized in Figure 1.

The data collection approach covers three phases. In the first phase, I collected data to understand the context of doctorfit's situation and work practices. In one week, I conducted short interviews with each team member and the manager of the division. In addition, I had the chance to observe the daily routines of the team during this week. Each team member explained in very detail the tasks and their respective role and responsibilities in the software development process.

The second phase started after the kick-off meeting with a series of three online polls. The first poll has been sent out at the same day of the kick-off meeting. The second poll has been sent out on the 1st of November 2016. I picked this date because the manager of the division explained that it usually starts getting interesting around one month before the governmental deadline. The third poll has been sent out on the 13th of December – one week after the software has been released to the beta users. All team members have filled out the first online poll. The second poll has been filled out from 5 out of 8 team members and the third poll has been completed by 7 out of 8 team members.

In the third data collection phase I conducted two interviews. The first interview was conducted with the manager of the division, who has been regularly involved and updated in the progress of the project. As a second interviewee tester T1 has been picked because of her longtime work experience at doctorfit. She works for doctorfit for around 15 years and knows the governmental restrictions as well as the needs and daily business of the medical practices very well. In addition, she has been working for doctorfit when the unit changed their development approach from waterfall to scrum. The interviews were structured in three topics. First, each respondent has been asked about his/her general impression of the project. Next, I presented the visualized findings of the three polls and asked the respondent for his/her interpretation of the findings. Last, the respondent described the perceived effects of the applied scrum practices (as listed in Table 1) on the performance in the project.

### Table 2. Team members

<table>
<thead>
<tr>
<th>ID</th>
<th>Role</th>
<th>Experience (years in team)</th>
<th>Experience (years in medical industry)</th>
<th>Poll Data</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>PM</td>
<td>Product manager</td>
<td>&lt;1 year</td>
<td>&lt;1 year</td>
<td>t1, t3</td>
<td></td>
</tr>
<tr>
<td>PO</td>
<td>Product owner</td>
<td>&lt;1 year</td>
<td>&lt;1 year</td>
<td>t1, t2, t3</td>
<td></td>
</tr>
<tr>
<td>SD1</td>
<td>Software developer</td>
<td>&lt;1 year</td>
<td>&lt;1 year</td>
<td>t1</td>
<td></td>
</tr>
<tr>
<td>SD2</td>
<td>Software developer</td>
<td>3-4 years</td>
<td>3-4 years</td>
<td>t1, t2, t3</td>
<td></td>
</tr>
<tr>
<td>SD3</td>
<td>Software developer</td>
<td>-</td>
<td>-</td>
<td>t3</td>
<td>Joined team after t1</td>
</tr>
<tr>
<td>T1</td>
<td>Tester</td>
<td>&lt;1 year</td>
<td>&gt;10 years</td>
<td>t1, t2, t3</td>
<td>Left doctorfit after t1</td>
</tr>
<tr>
<td>T2</td>
<td>Tester</td>
<td>&lt;1 year</td>
<td>&lt;1 year</td>
<td>t1, t2, t3</td>
<td>Left doctorfit after t1</td>
</tr>
<tr>
<td>T3</td>
<td>Tester</td>
<td>3-4 years</td>
<td>5-6 years</td>
<td>t1</td>
<td>Left doctorfit after t1</td>
</tr>
<tr>
<td>T4</td>
<td>Tester</td>
<td>&lt;1 year</td>
<td>&gt;10 years</td>
<td>t1</td>
<td>Left doctorfit after t1</td>
</tr>
<tr>
<td>T5</td>
<td>Tester</td>
<td>&lt;1 year</td>
<td>&lt;1 year</td>
<td>t2, t3</td>
<td>Joined team after t1</td>
</tr>
</tbody>
</table>

### Figure 1. Data collection approach

The data collection approach covers three phases. In the first phase, I collected data to understand the context of doctorfit’s situation and work practices. In one week, I conducted short interviews with each team member and the manager of the division. In addition, I had the chance to observe the daily routines of the team during this week. Each team member explained in very detail the tasks and their respective role and responsibilities in the software development process.

The second phase started after the kick-off meeting with a series of three online polls. The first poll has been sent out at the same day of the kick-off meeting. The second poll has been sent out on the 1st of November 2016. I picked this date because the manager of the division explained that it usually starts getting interesting around one month before the governmental deadline. The third poll has been sent out on the 13th of December – one week after the software has been released to the beta users. All team members have filled out the first online poll. The second poll has been filled out from 5 out of 8 team members and the third poll has been completed by 7 out of 8 team members.

In the third data collection phase I conducted two interviews. The first interview was conducted with the manager of the division, who has been regularly involved and updated in the progress of the project. As a second interviewee tester T1 has been picked because of her longtime work experience at doctorfit. She works for doctorfit for around 15 years and knows the governmental restrictions as well as the needs and daily business of the medical practices very well. In addition, she has been working for doctorfit when the unit changed their development approach from waterfall to scrum. The interviews were structured in three topics. First, each respondent has been asked about his/her general impression of the project. Next, I presented the visualized findings of the three polls and asked the respondent for his/her interpretation of the findings. Last, the respondent described the perceived effects of the applied scrum practices (as listed in Table 1) on the performance in the project.
Measurements

Whenever possible, I adopted existing measures in the development of the poll. The items of the poll cover project success, team performance, relational capital and cognitive capital. Project success has been operationalized by the construct suggested by Lee and Xia (2010) measuring system quality. The items to measure team performance have been adopted from He et al. (2007), who measured the time, cost and quality of the team’s processes. Furthermore information quality (four items) had been captured by adapting the survey items from Wixom and Todd (2005): (“Information distributed in the team is always (1) very up-to-date; (2) very complete; (3) well-structured and well-formatted (4) accurate”). To measure relational capital I applied the four items suggested by Robert et al. (2008) measuring trust, team norms, team identification and obligation. In addition, I developed one item to measure the degree of conflicts (“In our team there are no conflicts.”) and the quality of conflict resolution (“In case of a conflict we openly discuss different opinions in very detail.”).

To evaluate the cognitive capital, I applied a measurement approach, analyzing the perceptual distance of the team members. The respondents have to evaluate the own understanding on the respective aspect (“Do you understand the aspect?”) and then rate his/her perception of the teammates understanding on the same aspect (“Do your partners understand the aspect?”). Based on the literature on shared mental models, the aspects cover task-based and team-based aspects, like system requirements, work packages, project approach and knowledge distribution (Cannon-Bowers and Salas 2001; Mathieu et al. 2000).

Last, I applied the measurement suggested by (Lee and Xia 2010) to evaluate the autonomy and diversity of the team. I extended the measure by a bipolar scaling to specify the left and right pole (e.g. when measuring autonomy, I asked if the approaches are given from an external entity (right pole) or if the team can choose freely (left pole)).

Findings and Discussion

In the final interview with the manager of the division, he said: “When you started [collecting poll data] in this project, I thought ‘Oh well, you won’t find anything interesting. The team is great, the requirements are pretty precise and we have sufficient time to deliver the software’. Well, I was wrong.”

Even though the team was able to release the software in time, it was not a success as expected at the beginning of the project. As mentioned previously around 70% of the functionalities are governmental requirements. It was out of the question that the team would not succeed to implement the ‘must-have’ requirements in time. Doctorfit would have been in serious trouble. However, due to some critical problems in the team, doctorfit released only a minimal system on the 1st of January. The remaining 30% ‘nice-to-have’ functionalities will be released in a later updated version.

Indications for these insufficiencies can be found in the quality of the team performance. In t=3 the team has been asked to rate the overall team performance (see Figure 2).

![Figure 2. Satisfaction with team performance](image)

This negative perception of the performance can be explained by two major problems in the operational realization of the software (finding 1 and finding 2). In addition, I discovered an interesting explanation, why staff turnover does not necessarily influence the performance (finding 3) and explain why scrum practices do not automatically increase the agility of a team (finding 4).
Finding 1. Daily meetings cannot compensate the benefits of colocation.

The concept of scrum is based on intense communications between testers, developers and customers. To enable intense and frequent communication, practices like daily meetings or colocation shall help. At the beginning of the project the team applied both mechanisms: Every morning they met in front of the story board for 15 minutes to discuss open tasks (Daily Scrum). Additionally, they all had their workplace in the same office room (Colocation). However, after t=1 the product owner decided that he cannot concentrate in the room, when everybody is speaking and asking questions. He decided to work in another room, where it was more quiet. Even though, the whole team still met every morning for the daily meeting, the lack of colocation had an enormous impact on the information flow. As shown in Figure 3 the information currency (which is one of the four dimensions of information quality) dropped sharply. The team had the impression, that the information they shared in the team had not been up-to-date anymore. The tester explained, that this situation was quite difficult. Before the product owner left the room, it was easy to solve problems straight away. “You could just immediately ask: ‘Hey, how did you mean this requirement in the user story xy?’”. After t=1 the team had to wait to the next daily meeting to solve the problem.

This caused another issue. The team members have not been certain about the distribution of knowledge (or ‘who-knows-what’) in the team anymore. As highlighted in Figure 4, the team members respective understanding (own understanding) dropped sharply after the product owner left the room. Simultaneously, (but not that sharply) the team members rated their partners’ understanding for the knowledge distribution in the team lower as well. In t=1 and t=3 a higher level of perceptive similarity, than in t=2 has been found, which might indicate an uncertainty regarding the new situation. By referring to the social capital theory, this finding proves the positive linkage between colocation (or structural capital) and cognitive capital.

In addition, this new situation affected the relational capital – namely the conflict resolution. For the team, it was now difficult to solve conflicts straight away because they got more (physically) distant. Figure 3 highlights the development of conflict resolution. However, the tester also mentioned daily meetings were very important to solve the postponed issues and “to inform the product owner and get feedback for current problems.” (T1). Thus, the daily meeting was a necessary condition to cushion the drop of information currency, understanding of knowledge distribution and quality of conflict resolution caused by the dislocation of the product owner.

Finding 2. Being too detailed in the user stories can cause confusion.

As mentioned previously, the team has a lot of experience with scrum practices and has built solid routines in the iterative process of the development. However, in this project the team decided to add one practice to their established set of practices, which they have tested in a smaller previous project. In the scrum practice called **story slicing** the team ‘slices’ the user story into smaller functions. The product owner specifies the functions of clickable boxes, typing fields and transitions in the frontend screen instead of (only) describing the frontend screen as user stories. “This method is about splitting the user’s interaction into smaller pieces. [...] You cut the very large components, into small pieces. The team applied this approach in a previous project and everything went fine.” (division manager).
But not in this project. As highlighted in Figure 6 and Figure 7 the perceived information completeness and information format dropped sharply from \( t=1 \) to \( t=2 \). In the final interviews the respondents mentioned that all team members got lost in the details of the functions. “We all lost the overview. The profound degree of details made it very difficult to solve the problems. Sometimes I had questions for the developer, when I was testing the functions. For the developer, it was difficult to refocus on the respective function, because his concentration was already on a completely different function.” (T1).

Until \( t=2 \) everybody in the team was confused but could not exactly point out the problem. The members had the impression that it was only their own lack of understanding for the detailed features. Figure 9 and especially Figure 10 highlight this obscure situation. The own understanding of the project approach and work packages dropped from \( t=1 \) to \( t=2 \). In addition, we can see – especially in \( t=2 \) in Figure 10 – a large gap between the perception of the own understanding and the perception of the partners’ understanding. We can see that until \( t=2 \) the team members only rated their own understanding on a lower level.

After \( t=2 \) the team came together for an ‘emergency meeting’. Everybody noticed that something went wrong, but they needed this meeting to figure out the problem which confused everybody in the team. This meeting might be the explanation why the perceptions of the own and the partners’ understanding converged in \( t=3 \) (Figure 9 and Figure 10). The team members understood that it was not only their own understanding, but that everybody in the team had difficulties with this new practice of slicing the stories. Thus, the team decided to go back to the well-known abstraction level of the user stories. The respondents in the interviews are convinced that only this turning back saved the (minimal) success of the project. Figure 6, Figure 7 and Figure 8 indicate the success of this meeting after \( t=2 \).

The division manager has the following explanation for the failure of the story slicing practice: “From today’s perspective I can see the differences between the previous project and this project. The previous project was a statistical analysis tool – more mathematical. It was much easier to point out the different functions than in this project, which was much fuzzier”.

To interpret this finding by using the social capital theory, we can assume that the team was overstrained by the story slicing approach in this specific project context. The low level of cognitive capital in this context impacted the level of information quality (namely information completeness and format).
However, the ‘emergency meeting’ helped to turn things around. Based on the interview with T1, it seems likely that the meeting has been a great success due to the high level of relational capital in the team. Because the team members trusted each other and can speak openly with each other, the team was able to detect the problem and find a solution to get back on track.

**Finding 3. Staff turnover does not necessarily influence the performance.**

The project is characterized by a high degree of staff turnover (see Table 2). Two testers, who have worked for doctorfit for several years, left the company. During the project, I received the information from the team members, that it was a great loss for the team. To compensate this capacity gap two new employees (one tester and one developer) have been assigned to the team. By referring to Tuckman’s four phases of group development we would expect an increase of the number of conflicts and a suffering team performance. Tuckman (1965) argues, that every new team experiences four different phases in the group development: forming, storming, norming, performing.

However, in this project it was not the case. The findings highlight a steady (and even a slightly decreasing) number of conflicts from $t=1$ to $t=3$ (Figure 11) and an increasing level of team performance from $t=2$ to $t=3$ (Figure 12). In the final interview the tester stated “Surprisingly, we had no problem at all [regarding the staff turnover]. The two new members quickly got settled in the team.” (T1). Based on the social capital theory we would assume that a change of structural capital would impact the team’s performance in some way.

A possible explanation might be found in the teams’ overall cognitive capital. Both persons were testers, who received the software requirements from the product owner and compiled a list of test cases. Many of these test cases are standardized and the testers often build routines in compiling these test cases. Thus, the findings might be explained by the degree of standardization of the tasks in the job. It can be hypothesized that a certain degree of task standardization, minimizes or even eliminates the linkage between the cognitive impact of a certain individual in a team and the teams’ overall cognitive capital. At least in this project, it seems like staff turnover in a more standardized task field, does not have an impact on team performance.

![Figure 11. Number of conflicts](image1)

![Figure 12. Team performance](image2)

**Finding 4. Scrum practices even benefit in a highly regulated industry.**

“Why do you use scrum, if you cannot be agile?” This question has been asked to every respondent in the interview. The very strict governmental requirements leave no possibility for explorative development. The government states clearly what they want. In addition, the government has no interest in getting involved in the development process – either doctorfit meets the requirements on a specific due day or the software will not be released. These conditions do not seem to leave much space for the potentials of agile software development. Figure 13 and Figure 14 confirm this situation. The level of understanding for the functional requirements is constantly on the same high level. The team knows very well, what to achieve from the beginning of the project. Regarding the perceived team autonomy (Figure 14) the team is undecided whether or not they can fully design their software development autonomous (autonomy) or the objectives and rules are defined from the outside (heteronomy).
However, the employees of doctorfit perceive great benefits of the scrum approach. First, even though the team cannot be ‘agile’ regarding the functional requirements, they still can be agile with the technical implementation of the requirements. The main objective of the division manager is “to build nice software”. By nice software he especially points out the way of technical implementation and “the beauty of the programming code”. From his experience, he stated that scrum improves the quality of code due to the iterative process which enhances the structure of the software iteration by iteration.

Second, as the head of the department stated in the very first interview, the products of doctorfit differentiates by a high level of usability and additional ‘nice-to-have’ functionalities. It does not matter if a scrum team explores an optimal set of functionalities for a system, or if they (only) search for the optimal presentation of a given set of functionalities. In both cases, scrum practices can help to improve the quality of the outcome.

Third, the tester T1 pointed out the benefits of the iterative process. Earlier, when they used the waterfall approach, they had problems with the personnel capacity. The software development cycles are very precise in the industry of doctorfit. Smaller updates are released only at the beginning of a quarter and larger changes are only implemented once a year. Due to this restriction, the testers had only a little work at the beginning of the quarter but too much work at the end of the quarter – the days before the release. Now, with the iterative approach the employees’ capacity is being used more constantly over the whole years. Respondent T1 stated (joking), that she is not anymore only stressed out at the end of the quarter, but now she is stressed out on a more constant level over the whole year.

**Contribution, limitations and further research**

In this longitudinal case study, I analyzed the impact of scrum practices on the development of team interaction quality in a software development team. The contribution in this paper lies especially in the lessons learned of the case study. First, the findings indicate that daily meetings are a necessary condition to ensure a sufficient level of information currency, mutual understanding of the distribution of knowledge and the quality of conflict resolution in a team. However daily meeting cannot fully compensate the benefits of colocation to achieve and sustain a high level of information flow. Second, it is important to choose an appropriate method to describe the requirements. The findings in this case indicates that more ‘mathematical’ requirements favor for slicing the user stories into single functionalities (Story Slicing). Fuzzier requirements on the other hand should be organized by more abstract user stories, to not get lost in the details. Third, staff turnover does not necessarily influence team performance negatively. There seem to be aspects – like job characteristics – that eliminate the impact of staff turnover. Fourth, agility does not have to be the only goal of scrum practices. Even in highly regulated industries, in which the software functionalities are strictly defined, scrum practices can help to improve the coding structure, optimize the software design or apportion the utilization of the workforce.

I also want to point out the restrictions of this findings. The findings should be understood only as indications to initiate further discussions and raise the necessity for more research. I am aware that the findings of this explorative and qualitative case study imply no statistical or generalizable evidence. The polls were only used to expose the trends and tendencies of the team’s climate, but not to validate any
correlations. Furthermore, I cannot guarantee that I have not overlooked any alternative explanations for the impact of the scrum practices. The findings are based on the explanations in the interviews and I cannot exclude the possibility that the respondents misinterpreted the impact of the practices. To respond to these limitations further research is needed. A confirmatory analysis and statistical validation of these findings can address this limitation. The research should focus in different agile software development project in different industries.

Nevertheless, this research did enlighten some critical connections between agile (Scrum) practices and the development of team interaction quality and I hope, I could provide a foundation for further discussions about the longitudinal and relative impact of agile practices.

References


