Do You Just Discuss or Do You Solve? Meeting Analysis in a Software Project at Early Stages

Jil Klünder
Leibniz University Hannover,
Software Engineering Group
Hannover, Germany
jil.kluender@inf.uni-hannover.de

Nils Prenner
Leibniz University Hannover,
Software Engineering Group
Hannover, Germany
nils.prenner@inf.uni-hannover.de

Ann-Kathrin Windmann
TU Braunschweig, Dept. of Industrial,
Organizational and Social Psychology
Braunschweig, Germany
a.windmann@tu-braunschweig.de

Marek Stess
THI Investments GmbH
Stuttgart, Germany
m.stesss@thi-investments.de

Michael Nolting
Volkswagen Nutzfahrzeuge
Hannover, Germany
michael.nolting@volkswagen.de

Fabian Kortum
Leibniz University Hannover,
Software Engineering Group
Hannover, Germany
fabian.kortum@inf.uni-hannover.de

Lisa Handke
TU Braunschweig, Dept. of Industrial,
Organizational and Social Psychology
Braunschweig, Germany
l.handke@tu-braunschweig.de

Kurt Schneider
Leibniz University Hannover,
Software Engineering Group
Hannover, Germany
kurt.schneider@inf.uni-hannover.de

Simone Kauffeld
TU Braunschweig, Dept. of Industrial,
Organizational and Social Psychology
Braunschweig, Germany
s.kauffeld@tu-braunschweig.de

ABSTRACT

Software development is a very cooperative and communicative task. In most software projects, meetings are a very important medium to share information. However, these meetings are often not as effective as expected. One big issue hindering productive and satisfying meetings is inappropriate behavior such as complaining. In particular, talking about problems without at least trying to solve them decreases motivation and mood of the team.

Interaction analyses in meetings allow the assessment of appropriate and inappropriate behavior influencing the quality of a meeting. Derived from an established interaction analysis coding scheme in psychology, we present act4teams-SHORT which allows real-time coding of meetings in software projects. We apply act4teams-SHORT in an industrial case study at Volkswagen Commercial Vehicles, a large German company in the automotive domain. We analyze ten team-internal meetings at early project stages. Our results reveal difficulties due to missing project structure and the overall project goal. Furthermore, the team has an intrinsic interest in identifying problems and solving them, without any extrinsic input being required.

CCS CONCEPTS

• **Software and its engineering → Programming teams**; **Collaboration in software development**.

KEYWORDS

Communication, development team, meetings, software projects, human factors

ACM Reference Format:
Jil Klünder, Nils Prenner, Ann-Kathrin Windmann, Marek Stess, Michael Nolting, Fabian Kortum, Lisa Handke, Kurt Schneider, and Simone Kauffeld. 2020. Do You Just Discuss or Do You Solve? Meeting Analysis in a Software Project at Early Stages. In *IEEE/ACM 42nd International Conference on Software Engineering Workshops (ICSEW'20), Oct 5–11, 2020, Seoul, Republic of Korea.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3387940.3391468

1 INTRODUCTION

Meetings are an essential part of most software projects [9]. However, these meetings are often not as effective as expected [8]. This is often caused by inappropriate behavior such as complaining or an insufficient preparation. As effective team meetings are strong facilitators for a successful project [4], they should be productive and efficient in order to have motivated and satisfied developers [8]. Otherwise, the project will progress more slowly and the outcome will not be as good as possible [3]. Since there are a lot of projects struggling with difficult meetings, our objective is to *assess interactions in meetings in order to enable interventions from project leaders or the management.* These interventions help to keep the project on track. In addition, as the discussion on problems without providing solutions can lead to so-called complaining cycles [5]. They often lead to unsuccessful meetings which lead to a negative influence on project success. In our case study, we analyze *whether the team only identifies problems during the meeting or tries to solve them.*

The contribution of our paper is two-fold. (1) We present the coding scheme act4teams-SHORT which emerged from the established coding scheme act4teams [4]. Given the problems presented by Prenner et al. [8] with the first version of act4teams-SHORT, we

ICSEW'20, Oct 5–11, 2020, Seoul, Republic of Korea

extended the coding scheme leading to the version presented in this paper. (2) We apply the coding scheme in an industrial case study at Volkswagen Commercial Vehicles, a large German company in the automotive domain. Our results reveal difficulties due to missing project structure and the overall project goal. Furthermore, the observed team has an intrinsic interest in identifying problems and solving them, without any extrinsic input being required.

The rest of the paper is structured as follows: In Sec. 2, we present the background and related work. In Sec. 3, we present the research method, followed by the results in Sec. 4 which we discuss in Sec. 5. Section 6 concludes the paper.

2 BACKGROUND AND RELATED WORK

Interaction analyses are widely used in various domains of psychology, e.g., such as developmental, social, and organizational psychology. Accordingly, our study draws on psychological research, in particular on the well-established *act4teams* coding scheme [4, 5]. *act4teams* is a coding scheme designed for analyzing real team meetings in organizations. The *act4teams* coding scheme has been used in several studies. Kauffeld and Lehmann-Willenbrock [4] analyze the effect of team meetings on team and organizational success. They show that teams with more functional interactions, such as problem-solving and action planning, are more satisfied after the meeting. In addition, team productivity is also associated with functional interaction.

Using the act4teams coding scheme, Schneider et al. [9] investigate the behavior of 155 student software developers in 32 teams during the first project meeting. Schneider et al.'s [9] results indicate a significant positive influence of proactive statements on group affect, i.e. developers have been more satisfied after meetings with a lot of proactive statements. In case of supporting statements, this effect was even larger.

Oshri et al. [7] investigate the relevance of face-to-face meetings for socialization in globally distributed development teams. They found meetings to be rather short and with limited space for social informal exchanges [7]. Bless [1] presents an experience report outlining possibilities to have different kinds of meetings in distributed teams including retrospectives and planning pokers. The author clearly highlights the benefits of meetings in distributed teams, even if only video meetings are possible.

Already in 1992, Olson et al. [6] analyzed interactions in design meetings in development teams. They recorded ten design meetings in four projects on video and transcribed them afterwards. They analyzed the meetings using a coding scheme focusing i.a. on activities for problem-solving and for organizing the project. The authors developed the coding scheme during the analysis, leading to a total number of 22 categories including "issue", "project management" and "meeting management". Most of the categories used by Olson et al. [6] can also be found in the act4teams-SHORT coding scheme we use for our analysis.

In order to facilitate interaction analyses in meetings, Prenner et al. [8] present a simplification of the act4teams coding scheme. The authors compare the results of the neutral meeting analysis with the developers' satisfaction after the meeting and the perceived amount of shared information during the meeting. The results uncover the need for a more fine-grained coding scheme of the simplification of act4teams in order to draw adequate comparisons [8].

This paper presents, to some extend, the results of follow-up work by Prenner et al. [8] by solving the indicated problems and refining the coding scheme.

3 RESEARCH METHOD

The overall research design is visualized in Figure 1. In this section, we describe the coding scheme act4teams-SHORT and its application in an industrial case study at a project center at Volkswagen Commercial Vehicles.

3.1 Research Questions and Hypotheses

In order to meet our research goal, we want to answer the following research questions:

RQ1: *How do software project team members interact in meetings at early project phases?* With this research question, we gain an overview of the interactions in meetings. Furthermore, due to the progress of the project, we investigate the changes of meeting behavior over time. These changes indicate whether there is a need for interventions or whether the meetings get better over project duration automatically. In order to answer this research question, we analyze the meetings with respect to the number of statements about, i.a., problems, solutions, cooperation, as well as the amount of destructive and proactive behavior.

RQ2: *Does the team only talk about problems or does it try to solve them?* Problems are often subject of meetings. However, these problems should not only be named but also arranged, i.e., solved. Consequently, talking about problems should go along with talking about solutions in a meeting. We assume the following alternative hypothesis.

$H1_1$: *The number of statements about problems is related to the number of statements about solutions.*

However, it does not suffice to talk a lot about problems and solutions. Both should be connected in order to really solve a problem. Consequently, we expect a relationship between the number of problem-focused or solution-focused statements and the number of statements connecting solutions and problems. Consequently, we assume the following:

$H2_1$: *The number of statements that are either related to problems or to solutions is related to the number of statements connecting problems and solutions.*

3.2 Instrument Development

We base our research on the established coding scheme act4teams which was developed to analyze interactions in meetings [4]. However, applying this coding scheme is very time-consuming and requires a lot of knowledge and experience [8, 9]. Therefore, we developed a simplification of act4teams [8] where only events of interest are coded (selective coding). This coding scheme consisted of nine categories (result from stage 1 in Fig. 1), namely (1) naming problems, (2) linking problems, (3) naming solutions, (4) linking solutions, (5) linking and connecting, (6) counterproductivity, (7) proactivity, (8) structuring and (9) information sharing [8]. We applied this reduced coding scheme in a case study in three agile

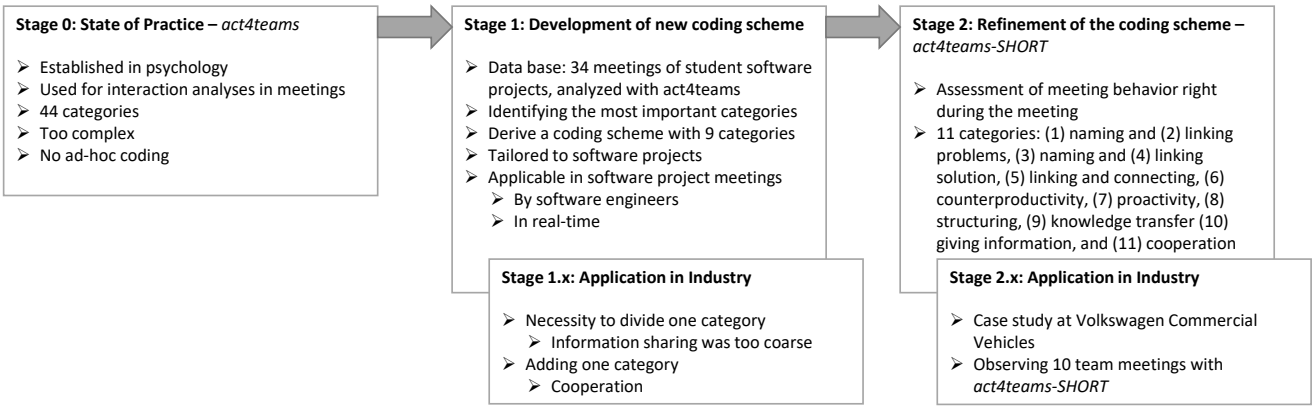

Figure 1: Overview of the research design

working teams at a large company in Germany working in the field of web applications [8] (stage 1.x in Fig. 1).

This case study revealed the need for further adjustments: Due to the nature of meetings in, e.g., the Scrum framework, which are mainly used for planning and information exchange, particularly the latter category occurred very often. Unfortunately, as information exchange is very neutral, we cannot draw a lot of conclusions when having counted this category at a high amount. Therefore, we decided to differentiate between information exchange and knowledge transfer, and distinguish between *giving information* and *knowledge transfer*. Furthermore, we noticed a high amount of cooperative behavior, namely by praise or approval [8]. This led to a completely new category called *cooperation*. This way, we obtained a coding scheme consisting of eleven categories (result of stage 2 in Fig. 1). This instrument was used in the present study. Table 1 summarizes and describes the categories.

3.3 Data Collection

The data collection using act4teams-SHORT was part of a case study we conducted at Volkswagen Commercial Vehicles, which is a large German company in the automotive domain. In the following, we describe the case company and the project under investigation as well as the meetings we observed.

3.3.1 Case Company. Volkswagen Commercial Vehicles has introduced a project center in August 2017 for their Mobile Online Services (MOD). The name MOD covers all activities related to online services depending on vehicles. The most relevant project is called *Connect Fleet*[1]. Connect Fleet is a mobile fleet management system for medium-sized enterprises with one to fifty vehicles. It enables users and companies to easily monitor their vehicles. For example, in Connect Fleet the system can record a driver's logbooks, fuel logs or statistical reports to present it to the driver itself as well as to the fleet manager. Furthermore, collecting this data enables Volkswagen to provide more complex but beneficial services to their customers. Hence the sub-project *Predictive Maintenance* started as a part of Connect Fleet. *Predictive Maintenance* aims at developing a method to determine the "health" of a car in real time by taking into account the driving behavior including speed, acceleration, braking behavior, motor revolutions, motor temperature and more. By determining the car's health beforehand, downtimes can be minimized by informing the fleet manager about potential problems with the car allowing him to intervene.

3.3.2 Data Collection in Meetings. During August and December 2018, we collected data in ten team meetings of the project *Predictive Maintenance*. In the meetings, the whole team comes together to exchange information and to discuss the next steps as well as different topics and problems. The regular participants are the product owner (also moderating the meeting), a data solution architect, two UX designer, a quality assurance representative, and a hardware specialist. This meeting is scheduled for one hour and the team is determined to keep this time limit. The observed meetings had an average duration of 55 minutes (min: 32min, max: 66min, SD: 10 min). The first two authors of this paper coded the interactions in these meetings using a software tool [8]. They counted the statements in the meeting per category. The workload of the coding process were equally divided between the first and second author.

3.4 Data Analysis

Our data analysis consists of a descriptive analysis by looking at the occurrences of the different categories and a quantitative analysis by analyzing specific correlations between problem- and solution-focused statements.

Since the length of the meetings and the number of codes vary, we perform our analysis using the relative amount of codes, unless otherwise stated.

3.4.1 Data Aggregation. In order to analyze the data quantitatively, we aggregated some of the collected data and defined the following variables. The number of problem-related statements is defined to be the sum of statements related to category (1) naming problems and (2) connecting problems. Analogue, the number of solution-related statements is defined to be the sum of statements related to category (3) naming solutions and (4) connecting solutions. The number of statements connecting problems and solutions is given by the number of statements related to category (5) linking and interconnections.

[1]For further information, look at https://connectfleet.io/home.

Table 1: Overview of the categories to assess interactions in meetings.

Category	Description
(1) Naming problems	Identifying/explaining a problem
(2) Linking problems	Analyzing problem causes and consequences
(3) Naming solutions	Gathering/elaborating solutions
(4) Linking solutions	Analyzing requirements for solutions, comparing solutions
(5) Linking and connecting	Showing links between problems and solutions
(6) Counterproductivity	Backbiting, putting others down, mean remarks, complaining, blaming others, ...
(7) Proactivity	Showing interest in ideas, engagement, taking responsibility for ideas/plans
(8) Structuring	Prioritizing, procedural suggestions, allocating tasks/roles, summarizing
(9) Giving information	Passing on information, giving information with reference to external sources
(10) Knowledge transfer	Applying own knowledge to the discussion, explaining information
(11) Cooperation	Praising others, thanking others, making an effort to be nice, ...

3.4.2 Descriptive Analysis. For analyzing changes in the occurrence of categories, we visualized the collected data as a bar chart and as a pie chart. This way, we were also able to uncover variations in the sequence of each category. We interpreted the results using Kauffeld et al.'s [4] results on the influence of each act4teams category and adjusted them to the context of act4teams-SHORT. However, this step is part of future work and, at the moment, basically bases on experience.

3.4.3 Hypotheses Testing. We tested the hypotheses formulated in Sec. 3.1. In order to identify the appropriate way of investigating on the hypotheses, we first applied the Shapiro-Wilk test for normal distribution [10]. In the case of having normally distributed data, we used Pearson's correlation coefficient r to measure the relation between the two variables. In the case of not normally distributed data, we calculated Spearman's ρ.

4 RESULTS

To answer RQ1, we performed descriptive analysis as described in Sec. 3.4.2. RQ2 was answered by testing the hypotheses as presented in Sec. 3.4.3.

4.1 Research Question 1

Figure 2 summarizes the results of all team meetings. On first sight, we see a huge amount of *giving information* (violet parts in Fig. 2). After a couple of weeks, this amount increases (meetings on Sep, 05 and Sep, 10). This was a very intense time for the team which is why all team members did not only meet once a week but twice. Furthermore, owed by the little communication outside the meetings, the meetings are meant to be used for information sharing with the team. As the team does not often communicate outside the meetings, they share the results of a whole week during the meeting. As a lot of the results are new information (next to faced problems), this category is highly present in the meetings.

In particular at the beginning of the project, there is only little *knowledge transfer* (dark orange parts in Fig. 2). With an ongoing project, the amount of this category increases.

Furthermore, we only observe a small amount of *structuring behavior* in the meetings (dark blue parts in Fig. 2). Structuring behavior occurs when somebody leads to meeting back to the intended structure or the focus. However, in the observed meetings, there was no structure to lead back to. This complicates having a good and time-boxed meeting.

The amount of *counterproductive and proactive behavior* is visualized in Fig. 2 (dark green resp. yellow parts) and in Fig. 3. In each of the meetings, we only observe a small amount of counterproductive behavior. In particular in early meetings, having little or no counterproductive behavior facilitates the project start. However, in the first two observed meetings, we observe a noticeable amount of counterproductive behavior. At the beginning of a project, this amount should be reduced as destructive behavior reduces the trust between the team members [4]. Fortunately, there are some meetings with a lot of proactive behavior which can – to some extent – balance counterproductive behavior. This amount increases during the project indicating that team members take responsibility for the project and its progress. At the beginning of the project, we have a rather small amount of proactive behavior as the tasks remain unclear during the meetings: Content-related and process-related discussions got mixed up during these meetings, resulting in tasks that remain unclear and vague so that team members cannot take responsibility for them since they do not know what to do and the tasks remain undone. Therefore, having clear tasks and a to-do-list for each team member at the end of a meeting facilitates project progress. It also supports the next meeting since it can be structured along the task list.

As visualized in Fig. 2 (red and light orange parts), problems make up a notable amount of time in most of the meetings. Figure 4 concretises the statements from Fig. 2 by comparing the number of problems and solutions as well as the connection of both. While solutions indicate the progress of the project, just naming problems does not. Expect a few meetings, there is little variance in the number of statements concerning problems over time. Furthermore, we often have a rather small amount of solutions indicating that the problems are discussed and explained, but remain unsolved.

The evolution of problems can be explained by the still early project phase. A lot of unexpected problems occur that need to be discussed in order to continue working. At this project stage, talking about problems is not necessarily bad. However, talking a

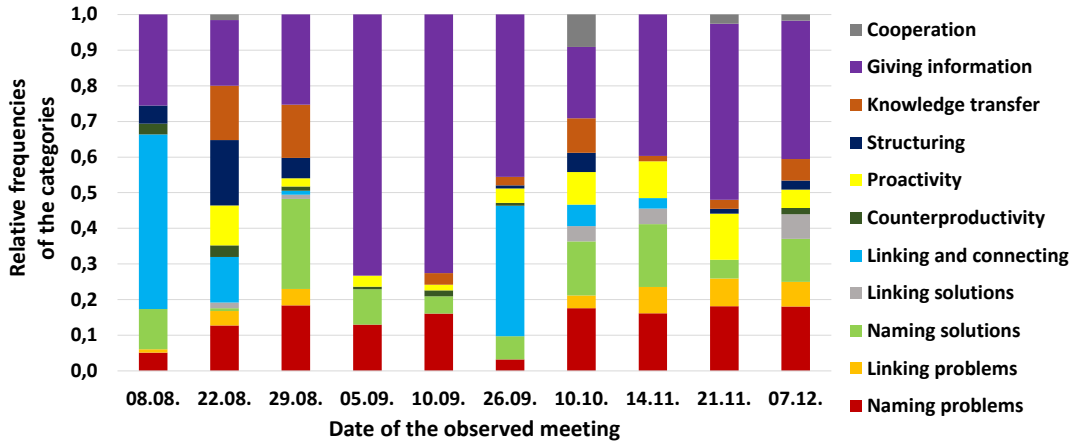

Figure 2: Overview of interactions in all team meetings

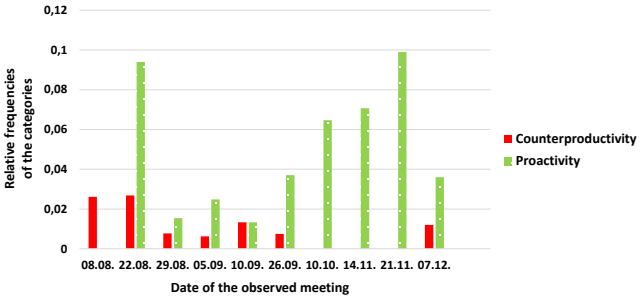

Figure 3: Frequencies of counterproductive and proactive statements in the observed meetings

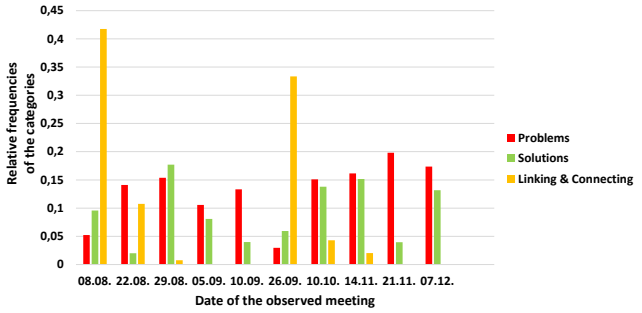

Figure 4: Frequency of statements about problems, solutions and connections between both in the meetings

lot about problems without solving them usually has a negative influence on the team members' satisfaction and mood.

4.2 Research Question 2

Before testing hypothesis H1, we ensured having normally distributed data. Applying the Shapiro-Wilk test for normal distribution to both data sets of problem-focused and solution-focused statements justified the use of Pearson's R ($W = 0.90382$, $p = .24120$ for problem-focused statements and $W = 0.94206$, $p = .57616$ for solution-focused statements)[2] . Calculating the Pearson correlation between both variables shows a strong positive relationship between problems and solutions which is significant at a significance level of $p \leq 0.05$ ($R = 0.7862$, $p = .006999$), leading to a rejection of $H1_0$. Consequently, **there is a correlation between the number of problem-related and solution-related statements**.

To test hypothesis H2, we ensured that the variable for linking and interconnections is also normally distributed. Shapiro-Wilk refutes the assumption of having normally distributed data ($W = 0.67102$, $p = .00039$).

As the variable given by the number of links and connections between problems and solutions is not normally distributed, we calculated Spearman's ρ to analyze the relationship between all statements on solutions and problems and the number of statements connecting these statements (H2). A value of $\rho = 0.51702$ supports the assumption of having at least a weak relationship between these kinds of statements. This result is significant (p(2-tailed) = 0.01958). Thus, **the team does not only talk about problems and solutions but also interrelates them with each other.**

5 DISCUSSION

We discuss our findings with respect to the research questions, the threats to validity and derive some lessons learned to support software projects at early stages.

5.1 Answers to the Research Questions

In order to assess interactions in team meetings of software projects at early stages, we formulated two research questions. These can be answered as follows:

RQ1: Most of the time in the observed team meetings at Volkswagen Commercial Vehicles was used for information exchange. The identification of problems, discussing solutions and finally solving problems also takes a lot of time in the meetings. Knowledge transfer, structuring, and counterproductivity only take a little amount

[2]We verified these results using two other tests for normal distribution (Kolmogorow-Smirnow test and Anderson-Darling test) as well as the graphical visualisization of the data, all supporting the statement of having normally distributed data.

of time in the meetings. Proactive behavior is also rather absent during the meetings.

RQ2: Our results reveal a positive correlation between the number of statements about problems and solutions as well as a (weak) correlation with the connection and linking with each other. Consequently, the team does not only talk about problems but also talks about how these problems can be solved: The more problems are mentioned during the meeting, the more the team also talks about solutions. The team started to work on solutions right from the start of the project and did not just pile problems. The implication is that other teams at an early project phase should show a similar behavior. Manager who observe that their team is just piling problems without discussing their solutions, should fast intervene because it indicates that something is going wrong in the team.

5.2 Limitations and Threats to Validity

The results of our case study are subject to some limitations that threaten their validity and their generalizability.

The database for the analysis consists of 10 team meetings from project start. This limits the statistical power of our findings. However, since we wanted just to observe the team during the early project phase, we consider this amount as sufficient. Further, errors in the coding process (e.g., due to misinterpreted categories) may lead to wrong results. We addressed this threat by choosing two experienced researcher with the use of act4teams-SHORT.

Two researchers themselves have been present during data collection. This can affect the results due to the Hawthorne effect. In order to reduce the influence on the meeting behavior of the participants, we integrated the analyses in a long-term cooperation between the Leibniz University Hannover and Volkswagen Commercial Vehicles. The researchers visited the project center various times before the data collection. Furthermore, the participants were assured that we collect the data completely anonymized.

To ensure the reliability of the collected data (i.e., independence of the researchers), we also calculated the interrater reliability, namely the intraclass correlation coefficient (ICC), of the two researchers in one meeting. Obtaining an ICC of 0.86, which is almost perfect according to Cicchetti [2] reduces the risk of this threat.

We analyzed 10 meetings to reduce the *mono-operation bias*. However, only applying a single method to analyze the meetings, reduces the reliability of the results due to the *mono-method bias*. Unfortunately, it was not possible to compare the results of act4teams-SHORT with the results of, e.g., act4teams, because we were not allowed to video-record the meetings and act4teams is not applicable in a live setting.

In this case study, we only observed meetings in one specific team over a project period of four months. This decreases the generalizability of our results. In order to obtain more generalizable results, this study should be repeated in other contexts. Further research is required to support our results or to concretize them. Even though other teams at early project stages may behave differently, we observed a team that shows a possible behavior during the project. We expect other teams at early project stages to show a similar behavior.

6 CONCLUSION

Meetings are a very important part of software projects. In particular at early project stages, most communication takes part in meetings. However, these meetings are often not as effective and as productive as expected, leading to dissatisfied software project teams and demotivation. Assessing interactions in meetings can help to reduce inappropriate behavior in meetings such as complaining or losing the train of thoughts. act4teams-SHORT is a coding scheme enabling software engineers to assess this kind of interactions in meeting. This way, the team gets an overview of its behavior in meetings.

In a case study at Volkswagen Commercial Vehicles, we analyzed ten team meetings at early stages in one project. Our results indicate that the meetings are a very important medium to transport information and to talk about problems. In addition, the number of statements about problems is positively linked to the number of statements about solutions and that both of them are connected.

In future work, we plan to extend our results and to increase their reliability by conducting more case studies in other project teams, at different project stages, and in other companies. In addition, we plan to develop patterns that show which amount of behavior of which category should occur during a meetings in dependence from the intended outcome of the meeting.

ACKNOWLEDGMENTS

This work was funded by the German Research Foundation (DFG) under grant number *263807701* (Project TeamDynamics, 2018-2020). We also thank the team at Volkswagen Nutzfahrzeuge for participating in our case study.

REFERENCES

[1] Marc Bless. 2010. Distributed meetings in distributed teams. In *International Conference on Agile Software Development*. Springer, 251–260.
[2] Domenic V Cicchetti. 1994. Guidelines, criteria, and rules of thumb for evaluating normed and standardized assessment instruments in psychology. *Psychological assessment* 6, 4 (1994), 284.
[3] Daniel Graziotin, Fabian Fagerholm, Xiaofeng Wang, and Pekka Abrahamsson. 2017. Unhappy developers: Bad for themselves, bad for process, and bad for software product. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*. IEEE, 362–364.
[4] Simone Kauffeld and Nale Lehmann-Willenbrock. 2012. Meetings matter: Effects of team meetings on team and organizational success. *Small Group Research* 43, 2 (2012), 130–158.
[5] Simone Kauffeld and Renee A Meyers. 2009. Complaint and solution-oriented circles: Interaction patterns in work group discussions. *European Journal of Work and Organizational Psychology* 18, 3 (2009), 267–294.
[6] Gary M Olson, Judith S Olson, Mark R Carter, and Marianne Storrosten. 1992. Small group design meetings: An analysis of collaboration. *Human–Computer Interaction* 7, 4 (1992), 347–374.
[7] Ilan Oshri, Julia Kotlarsky, and Leslie P Willcocks. 2007. Global software development: Exploring socialization and face-to-face meetings in distributed strategic projects. *The Journal of Strategic Information Systems* 16, 1 (2007), 25–49.
[8] Nils Prenner, Jil Klünder, and Kurt Schneider. 2018. Making meeting success measurable by participants' feedback. In *Proceedings of the 3rd International Workshop on Emotion Awareness in Software Engineering*. ACM.
[9] Kurt Schneider, Jil Klünder, Fabian Kortum, Lisa Handke, Julia Straube, and Simone Kauffeld. 2018. Positive affect through interactions in meetings: The role of proactive and supportive statements. *Journal of Systems and Software* 143 (2018), 59–70.
[10] Samuel Sanford Shapiro and Martin B Wilk. 1965. An analysis of variance test for normality (complete samples). *Biometrika* 52, 3/4 (1965), 591–611.