Participation Balance in Pair Programming

Laura Plonka, Judith Segal, Helen Sharp, Janet van der Linden

Abstract

In this paper, we investigated the equity of participation in industrial pair programming (PP) sessions by quantifying two different types of contributions; verbal contributions and driving contributions. As a result, we found that two thirds of the sessions are not equitable.
Based on interviews with the developers, we analyse which factors influence the participation balance and discuss whether the concept of equity of participation could be used as metric to evaluate PP.

1 Introduction

Pair Programming (PP) is a software development practice where two programmers share mouse and keyboard while working together on one computer. [Williams 02, Kent 99].
The programmer who is typing is typically called the "driver" and the other is called the "navigator". These roles can be switched at any time by either programmer.
Over recent years, a wide range of studies [Begel et al. 2008, Layman et al. 2004, Vanhanen et al. 2007, Vanhanen et al. 2007a, Williams et al. 2000] have investigated PP; most of them report positive results indicating benefits of PP such as better quality software than solo programming, enhanced knowledge transfer among the developers, and improved communication within the development team.
However, a concern among some practitioners is that PP might be a waste of resources if developers have a very low equity of participation, for example, when one developer is doing all the work and the other is not contributing at all.
This paper focuses on the equity of participation of developers in PP in industrial settings.
It addresses the following questions:

- Do industrial pair programmers contribute equitably during a PP session?
- Which factors influence the participation balance in PP?
- What are the implications when developers do not contribute equitably and should we consider those PP sessions as a waste of resources?

In section 2, we explain the concept of equity of participation. In section 3, our study methodology including our data gathering and analysis approach is presented. The results of our analysis are presented in section 4 and the implications are discussed in section 5. Section 6 provides a conclusion and our future work.

2 Participation balance as a desirable state

Equity of participation provides a quantitative perspective on group collaboration by quantifying the contribution of each group member. Different types of equity can be investigated such as verbal equity, the equity of interactions or the perceived equity [Marshall et al. 2008].

Although the idea of equity of participation does not consider the quality of the contributions, equity itself is often described as a desirable state for different kind of tasks. These tasks can include collaborative learning, collaborative decision-making or tasks for which information sharing among the members of the group is crucial.

In the field of collaborative learning it seems that higher learning gains can be achieved by active participation in form of verbalisation of the group members. Therefore active participation "becomes a necessary, though not sufficient, predictor [...] of higher learning outcomes" [Bachour 2010].

In contrast, lack of participation can lead to suboptimal decision making due to a higher risk of potentially relevant information not being shared within the group [DiMocco et al. 2004]. Salomon and Globerson [Salomon et al. 89] describe two other undesirable effects ("sucker" and "free-rider" effect) of unbalanced participation that decrease group productivity; "sucker effect" means that under participation of group members can decrease the amount of participation of the more active group members because they try to avoid being taken advantage of, whereas the "free-rider" effect means that over participation of a group member might lead other members to put less effort in a common task.

3 Study Methodology

This paper is based on four one-week studies, each in different company. In this section, the study background, data gathering and analysis are described.

3.1 Data gathering

We used three types of data gathering (questionnaires, audio and video recordings, and interviews) during our study.

(1) Questionnaires were used to gather background information about the developers and the task, such as the type of task and experience of the developers in programming and PP. Developers were asked to fill in the questionnaires before they started to work on the task.

(2) Audio and video recordings were used to record the PP sessions.
The recordings of the session consist of the following three data sources:

- Audio recordings of the verbal communication between the participants,
- a video of the programmers capturing who is driver and who is navigator,
- and a full-resolution screen recording capturing all computer activities.

All three data sources are fully synchronized and stored in a single video file so that all information is available at once (see figure 1). The PP sessions were recorded in the developers' day-to-day work environment and while solving day-to-day tasks. Our recording setup was installed at one computer in each participating company (see figure 1).

(3) Interviews were conducted with the developers after their PP sessions to gather data about the developers' perception of their PP sessions. The interviews were conducted with both developers present.

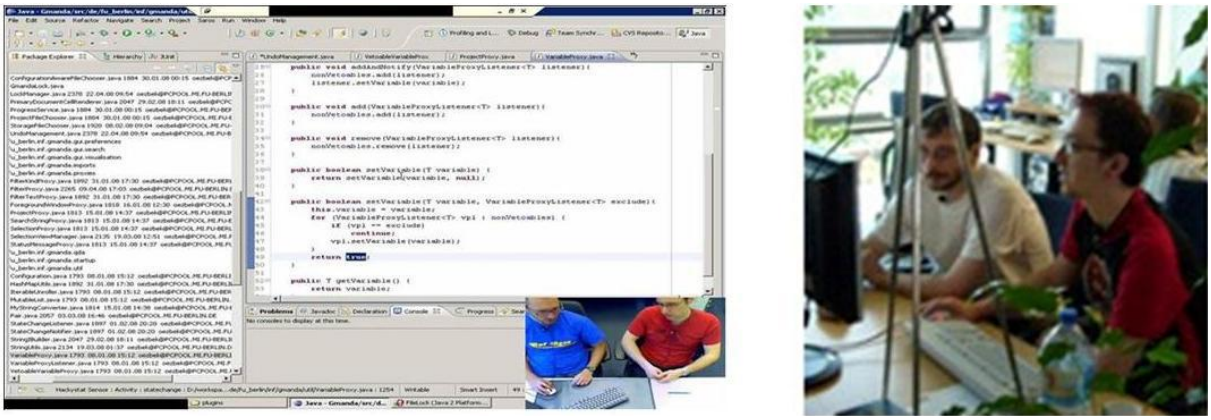

Fig. 1. On the left: Screenshot of a video of a PP session: This shows the developers' screen (Eclipse IDE) and, bottom right, the video recording of the developers. On the right: Recording setup in one of the companies.

3.2 Data Background

We recorded 21 PP sessions in four different companies. All companies work in different industries and all used agile approaches in the teams we observed. Two of the companies had just introduced PP and used Scrum, the other two had used PP for more than one year and used an adapted agile approach. In all four companies, PP was encouraged by the team leaders but the team leaders did not assign pairs or tasks. Instead the developers decided when to use PP and in which pair constellation. The forming of the pairs was done during their daily stand up meeting or spontaneously during the day.
Participation in the study was voluntary for all developers; in total 20 different pair constellations with 31 developers were recorded. Some developers were recorded more than once. The developers decided the physical layout of the pairing station; 17 pairs used one mouse/keyboard and four pairs used dual keyboards and mice. A PP session lasted between one and a half hours and three and a half hours; in total we videotaped about 37 hours of PP sessions. Table 1 provides an overview of the recordings.

Industry	**# of recorded sessions**	**# of participating developers**	**# of different pair constellations**	**Av. programming experience (in years)**	**Av. PP experience (in years)**
Geographic information systems	6	8	6	7.9	6.5
Transport and logistics	7	8	6	9.7	1
Email marketing	4	6	4	7.1	2.9
CRM systems estate	4	7	4	6.8	1.6

Table 1: Overview: Number of recordings and developers' backgrounds

3.3 Analysis method

For our analysis of equity of participation, we distinguish between “driving” contributions (the amount of time each developer is driver during a PP session) and verbal contributions (the amount of time each developer is talking).

Analysing the amount of driving for each developer
In the context of this study, we define "driving" as being in control of the mouse/keyboard and using one or both of them as an input device (this can include browsing, writing code, executing programs, etc.). This means that having a hand on mouse/keyboard without using it is not considered as driving. This definition evolved during the analysis based on the observation that some developers tend to place their hand on the keyboard or mouse although they are not typing; for example, when they are engaged in a discussion with their partner.

The driving times for each developer are coded in the videos and afterwards quantitatively analysed to investigate the percentage of driving time of each developer.
Certain events during the session can greatly influence the percentage of the driving times. For example, when one developer takes a break the other developer might continue or assume driving during this time. To reduce bias, episodes in which one or both of the developers were absent during the session were cut out and not included in the calculation of driving times.

Analysing the amount of verbal contributions for each developer
The amount of verbal contributions is analysed automatically using an analysis tool developed by the authors. The tool identifies when and for how long each developer is talking without considering any semantic information about the content.

To automatically detect the verbal contributions of each developer, each developer was wearing a noise cancelling, directed wireless microphone during the recordings. The audio data was recorded in a stereo file, each developer on a separate channel (see figure 2).

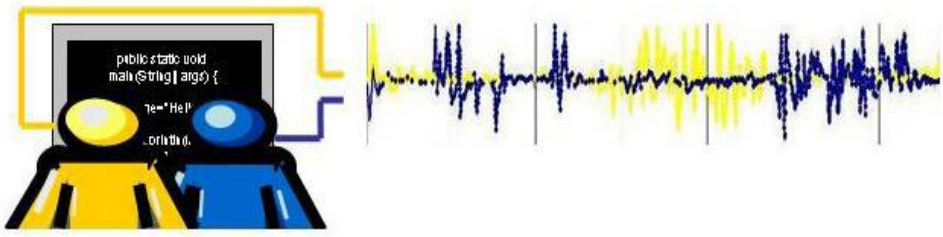

Fig. 2 Each developer was wearing a wireless microphone and the signal of each microphone was recorded on one channel of a stereo audio file.

The tool calculates the speech units for each channel. A speech unit is period of time in which one of the developers is speaking. A speech unit starts when a pre-defined number of consecutive samples of a specific channel are above a manually defined threshold, and it ends when a pre-defined number of consecutive samples are below the same threshold.
The threshold was adapted manually to the volumes of the developer's voices.

4 Results

4.1 Participation balance

Driving equity: Is the driving distribution balanced in industrial PP sessions?

We analysed the percentage of driving time for each developer with respect to the total driving time of each session; Times when neither of the developers are driving are not included. We say the driver distribution is equitable if the driving times of each developer are between 40% and 60%. This is the grey coloured area in figure 3. The figure shows that the driver distribution is equitable in 7 out of 21 sessions whereas the distribution is extremely inequitable in four sessions (session 1, 7, 9 and 11).

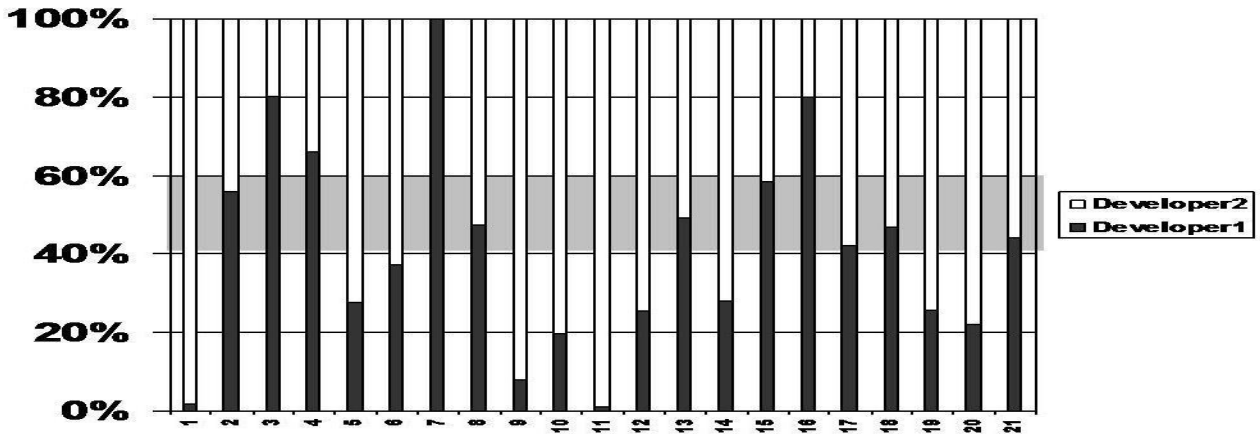

Fig. 3. Driver distribution among the developers
This figure shows the driving times (in percent) of each developer with respect to the total driving time of both developers. Driver distribution is considered to be equitable if both developers have between 40% and 60% of the total driving time. The equitable area is marked in grey.

Verbal equity: Do both developers contribute equitably on a verbal level?

We analysed the percentage of verbal contributions for each developer with respect to the total speaking time of each session; Times when neither of the developers are speaking are not included. We say the verbal distribution is equitable if the driving times of each developer are between 40% and 60%. This is the grey coloured area in figure 4. The figure shows that the verbal contributions are equitable in 8 out of 21 sessions.

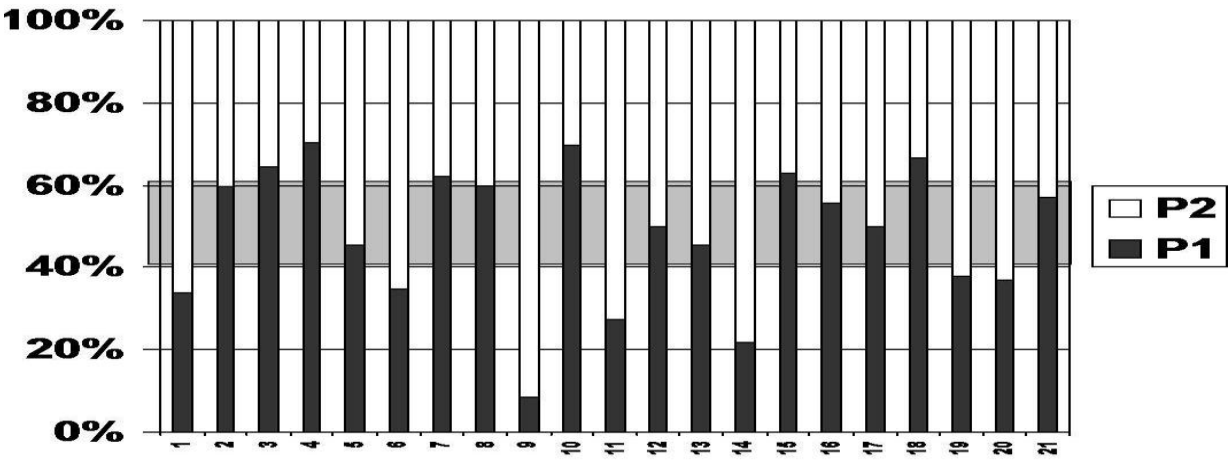

Fig. 4. Verbal contributions among the developers
This figure shows the verbal contributions (in percent) of each developer with respect to the total speaking time of both developers. Verbal contributions are considered to be equitable if both developers speak between 40% and 60% of the total speaking time. The equitable area is marked in grey.

4.2 Which factors influence the participation balance in PP?

Based on the interviews, we found four main factors that influence the equity of driving of developers during PP sessions. The analysis of factors that influence the verbal equity is still ongoing.

Workstation
In 17 of the 21 sessions in our study, developer pairs were working on a computer that belonged to one of the two developers. This implies that in those cases, one developer is working on an unfamiliar computer. According to the developers, this seems to affect the driving distribution during a PP session.
Developers stated that the developer who "owns" the computer is more likely to be the driver. Furthermore, developers reported that they feel more efficient and comfortable when driving their own computers and that an unfamiliar computer slows them down.
This is due to customisations of the workstations that can range from the adaption of the mouse and keyboard settings, different organisation of the file system, different Plug-Ins for the IDE, or even different operating systems.

Quote from one of the interviews:
"I rather sit on my own computer when I'm driving. I'd say I'm 20 percent more efficient on my own computer because I know where everything is and that [not knowing where everything is] slows me down when I'm working on another computer."

We analysed the driving distribution with respect to who "owns" the workstation.
We found that the owner of the computer dominated the driving in 7 out of 17 sessions; the driving distribution was equitable in 7 sessions and only in 3 sessions the developer who did not own the computer dominated the driving.

Personal preferences and PP experience
Some developers stated that they prefer being the driver rather than being the navigator. They described that they get easily impatient when navigating, especially when their partner is typing slowly.

Quote from one of the interviews:
"I'm really impatient while coding, even when I work on my own. […] So, I definitely prefer to be driver when I'm pairing.

In contrast, the "navigator types" report that they do not feel very comfortable when someone else is watching them while typing.

Quote from one of the interviews:
"I just don't like if someone watches me typing. But maybe I just have to get used to that.

This quote of a "navigator types" indicates that the developer might not be an experienced pair programmer and therefore might prefer to be navigator.
We checked the backgrounds of the developers and found that developers who stated that they prefer being navigators are developers with less than 6 months of PP experience.
Hence, PP experience might be crucial for the role preferences.

Work style
There are different approaches to solve a development task. Some developers prefer to follow a very systematic approach while others work in a "trial and error" mode. Another aspect is the efficiency of certain work styles; some developers type very fast and use short-cuts while other developers need more time for typing or use menus instead of short-cuts. According to the developers, the efficiency of the partner's work style seems to influence the driver-navigator distribution. Developers reported that it is difficult being a navigator if they perceive the partner's work style as being inefficient, error prone or unorganized.

Skill differences
PP is often used to enhance knowledge transfer among the developers. Therefore, developers with different skill levels work together in order to exchange knowledge during the PP session. In most sessions knowledge transfer is bi-directional. However, there are pair constellations in which the knowledge transfer is mostly directional.
Based on the questionnaires and the interview data, we identified seven PP sessions (out of the 21 recorded sessions) in which the developers used PP mostly for directional knowledge transfer.
In those cases both developers stated that they wanted to use PP in order to exchange knowledge and both developers agreed that one developer is more knowledgeable (in terms of how to solve the task) than the other one.
We call those sessions expert-novice sessions. Expertise refers to knowledge that is necessary to solve the current task rather than programming or PP expertise in general.

In those constellations, the novice were aware of the fact that solving the task will take longer when they are driving since they are slower and they will require a lot of explanations from the expert. According to the novices, they therefore prefer being navigator because they do not want to slow their partner down and are worried that they might seem unknowledgeable while driving. On the other hand, they stated that it is easier for them to learn when they are doing the actual typing.
The expert developers had a similar perspective. They were aware that their partners achieve higher learning gains when driving but they also stated that it is more effective if they do the typing themselves.

Experts as well as novices stated that time pressure increases the likelihood that the expert is driving during the session.

We checked the equity of driving for those constellations and found that one out of the seven sessions was equitable.

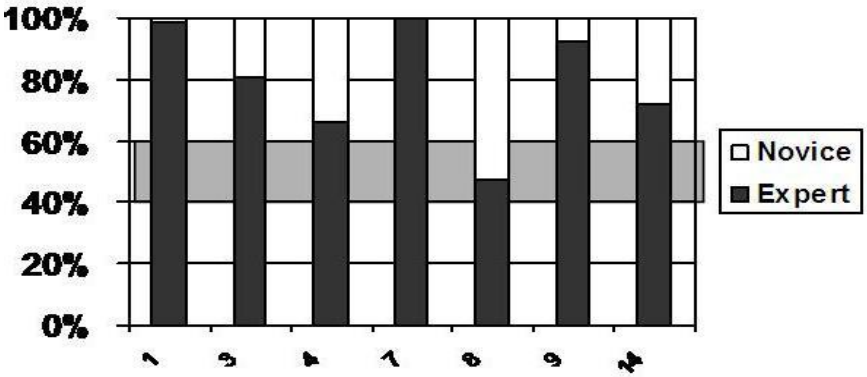

Fig 5. Driving distribution of expert-novice constellations

5 Discussion: What are the implications of inequitably PP sessions?

As discussed in section 2, the literature suggests that equity of participation is a desirable state for different types of tasks. We investigated the equity of participation in industrial PP sessions by quantifying two different types of contributions; verbal contributions and driving contributions. These are the two types of contributions in PP that are most visible for everyone in an office (co-workers and managers) and they are easily measurable. Therefore, practitioners might refer to those types of contribution when they express their concerns that PP might be a waste of resources if one developer is doing all the work and the other is not contributing at all.

Based on our analysis, we found that about two thirds of PP sessions are not equitable in terms of driving and verbal contributions.
Those sessions would be a waste of resources if the concept of equity of participation would be used as a metric to evaluate PP sessions. Therefore in this section, we will discuss whether equity of participation a desirable state in PP and whether the concept of equity of participation could be used as a metric to evaluate PP sessions.

We argue that the concept of equity of participation can't be applied without considering the following aspects:

Perception of the developers
During the interviews, developers were asked to rate their PP sessions. Based on the ratings of the developers, we couldn't find any indicator that there is a relationship between the ratings and the participation balance during the session. This implies that a balanced participation is not necessary for a positive perception of the session by the developers.

Types of task
Based on the questionnaires, we found that developers use PP for different reasons; for example to work more effectively, to solve a task that is too complex to be solved by one developer, and for knowledge transfer.
According to the literature, equity of participation might be important for the learning aspect, but it might not be crucial for the goal of effective problem solving.

Therefore, the reason why PP is be used should be considered before using the concept of equity of participation.

Types of contribution
Verbal and driving contributions are probably the most visible contributions during a PP session. However, since developers are collocated and work together on an artefact (code), they can contribute and communicate on other levels as well. Developers might, for example use gestures to communicate, such as nodding to reassure the partner while typing or pointing to something on the screen to give directions.

Additionally, developers use the code itself or external representations to communicate; we found that some developers prefer to show their ideas to their partners rather than explaining them, for example, by writing a method or by using paper based external representations.

Quality of contributions
Equity of participation focuses on the quantity of contributions and does not consider the quality and the content of contributions. We argue that the quality and content of contributions might be more important than the amount of contributions, especially if developers discuss different approaches to a task.
In terms of the amount of contribution, it seems that even very short verbalisations can be very important to the developers. During the interviews, developers stated that getting feedback from their partners – by nodding or just confirming their actions by saying "yes"- is important to them as it shows that their partner is aware of what they are working on and agreeing with it.

The aspects discussed above show that a balanced participation (in terms of driving and verbal contributions) might not be necessary for a successful PP session. However, a certain degree of participation and the opportunity to contribute seem to be important to the developers in order to stay engaged, focused and actively involved in a PP session (as discussed in the next two sections). The degree of contribution that is needed to stay focused might depend on the personality of the developers.

Opportunities to contribute
During the interviews, developers stated that there were situations during PP sessions in which they would have preferred to work on their own and split up the pair constellation. Developers described those situations as periods of time in which their contributions were not needed and they had to wait for their partner to finish a simple but time-consuming task such as setting up a test environment.

Risk of dropping out
Based on the interviews, we found that developers who do not contribute enough on a verbal or driving level seem to be more likely to drop out of the PP session. They reported that it is hard for them to follow their partners' actions and understand what their partner is currently working on. This in turn decreases the amount of their contribution. Hence, a very limited amount of contribution on both levels might be an indicator that one developer is dropping out.

6 Conclusion and Future Work

The quantitative concept of equity of participation (in terms of driving and verbal contributions) can't be used as a metric to evaluate whether PP is a waste of resource because there are many factors that need to be taken into account. Based on our results it seems to be important that developers have an opportunity to contribute during PP sessions. Contributions of both developers are a necessary precondition to achieve the benefits of PP, for example, better decision making by discussing the problem from two perspectives and knowledge transfer. However, the extent to which they have to contribute in order to stay actively involved and to make a beneficial contribution to the task at hand seems to depend on the developer. Therefore, a lack of equity of participation does not imply that the session is a waste of resources.

For our future work, we plan to investigate what kinds of interactions are important during PP sessions. Therefore, we will analyse PP sessions qualitatively.

References:

[Bachour 2010]
Bachour, K., Augmenting Face-to-Face Collaboration with Low-Resolution Semi-Ambient Feedback, École Polytechnique federale de Lausanne, 2010

[Begel et al. 2008]
Andrew Begel and Nachiappan Nagappan. Pair programming: what's in it for me? In ESEM '08: Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement, pages 120-128. ACM, 2008.

[DiMocco et al. 2004]
DiMicco, J.M., A. Pandolfo, W. Bender, Influencing group participation with a shared display. *Proc. of CSCw'04*. ACM 2004, 614-623

[Kent 99]
Kent Beck. Extreme Programming Explained: Embrace Change. Addison-Wesley Professional, 1999

[Layman et al. 2004]
L. Layman, L. Williams, and L. Cunningham. Exploring extreme programming in context: an industrial case study. In Agile Development Conference, 2004, pages 32-41, 2004.

[Marshall et al. 2008]
Marshall, P. and Hornecker, E. and Morris, R. and Dalton, S. and Rogers, Yvonne (2008). When the fingers do the talking: A study of group participation for different kinds of shareable surfaces. Proceedings of IEEE Tabletops and Interactive Surfaces

[Salomon et al. 89] G. Salomon & T. Globerson. When teams do not function the way they ought to, International Journal of Educational Research, vol. 13, pages 89–99+, 1989.

[Vanhanen et al. 2007] J. Vanhanen and C. Lassenius. Perceived effects of pair programming in an industrial context. In Software Engineering and Advanced Applications, 2007. 33rd EUROMICRO Conference on, pages 211-218, 2007.

[Vanhanen et al. 2007a] Jari Vanhanen and Harri Korpi. Experiences of using pair programming in an
agile project. In HICSS '07: Proceedings of the 40th Annual Hawaii International Conference on System Sciences, page 274b. IEEE Computer Society, 2007.

[Williams et al. 2002] Williams and R. Kessler. Pair programming illuminated. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 2002.

[Williams et al. 2000]
L. Williams, RR Kessler, W. Cunningham, and R. Jeffries. Strengthening the case for pair programming. IEEE software, 17(4):19-25, 2000.