THE IMPACT OF PEER-BASED SOFTWARE REVIEWS ON TEAM PERFORMANCE: THE ROLE OF FEEDBACK AND TRANSACTIVE MEMORY SYSTEMS

Research-in-Progress

Christoph T. Schmidt
SAP Research &
Institute for Enterprise Systems,
University of Mannheim
69190 Walldorf, Germany
christoph.schmidt01@sap.com

Kai Spohrer
University of Mannheim
68161 Mannheim, Germany
spohrer@uni-mannheim.de

Thomas Kude
University of Mannheim
68161 Mannheim, Germany
kude@uni-mannheim.de

Armin Heinzl
University of Mannheim
68161 Mannheim, Germany
heinzl@uni-mannheim.de

Abstract

Peer-based software review techniques such as pair programming and peer code review are increasingly used within information systems development teams. How do these techniques influence team performance? While previous research has treated the decision to apply peer-based review techniques as a trade-off between increased effort and improved code quality, this study provides a deeper theoretical understanding of how these techniques affect team performance. Based on extant literature on team feedback and transactive memory systems, a theoretical model is developed which links pair programming and peer code review to different forms of feedback. Depending on task complexity, different feedback mechanisms are suggested to influence team performance directly through improved code quality, but also indirectly by strengthening teams’ transactive memory systems. Findings from ongoing extensive empirical research are expected to contribute to existing literature on peer-based software review techniques, which mostly examined effects on the performance of individuals in experimental settings.

Keywords: Information Systems Development, Team Performance, Peer Code Review, Pair Programming, Transactive Memory System, Outcome Feedback, Process Feedback
Introduction

Teamwork and continuous quality assurance are at the core of agile information systems development (ISD) (Beck 1999; Schwaber and Beedle 2001). In particular, instead of testing code at the end of a predefined, sequential process, agile ISD relies on self-organized teams which continuously use iterative quality assurance techniques, such as pair programming and peer code reviews (Balijepally et al. 2009).

Pair programming (PP) means that two developers share a single workstation and collaboratively develop software. One of them is actively writing code while the other observes, supports, and challenges his or her approach in order to develop an even better solution (Williams and Kessler 2000). When applying peer code review (PCR), code is developed by one single software developer and then submitted to a review system, where it is commented and evaluated by colleagues from within or outside the team. The original author then revises the code according to the feedback and resubmits a further version. This iterative cycle of review and revision ends when both author and reviewer are satisfied with the solution at hand (Rigby and Storey 2011). For years, PCR has been applied successfully as an important quality assurance technique in open source communities (Liang and Mizuno 2011; Rigby and Storey 2011) and is currently diffusing to ISD enterprises. PCR has been proposed as a less costly alternative to pair programming in practice (Paulk 2001), and initial research has been conducted to examine a potential substitution of PP by PCR (Müller 2004, 2005; Winkler and Biffl 2006). A closer examination of these techniques, consequently, appears worthwhile from the perspectives of both research and practice.

The decision to apply these peer-based review techniques has often been discussed as a trade-off between code quality and output quantity (e.g. Cockburn and Williams 2000). So far, positive effects of continuous reviews on code quality have been acknowledged (Salleh et al. 2011). Yet, the allocation of programming time was perceived as a short-term cost-benefit decision, directly relating to the quality of the code that is currently being developed. More specifically, the question was raised if the additional effort associated with review techniques is outweighed by the immediate benefits resulting from improved code quality. This study aims at moving beyond a narrow focus on code quality, and instead examining the impact of peer-based review techniques on ISD team performance in a more comprehensive way.

In addition to the immediate effects on code quality, peer-based review techniques may positively affect team performance in two major ways. First, by interacting with peers from within or outside the team, developers may create new knowledge by combining existing explicit or tacit knowledge (Nonaka et al. 2006). Thus, the interaction and socialization involved in peer-based review techniques may positively affect team performance by enabling team learning (Edmondson et al. 2007; Goodman and Dabbish 2011; Mathieu et al. 2008).

Second, peer-based review techniques essentially provide team members with feedback regarding the developed software and the way it is developed. When software developers receive feedback from individuals within or outside the team, team members’ awareness of their colleagues’ skills and specialized knowledge may increase. Coordinating specialized knowledge is paramount for ISD teams, given that software development is non-repetitive knowledge work. For agile development teams, knowledge coordination is particularly important as it allows for sustaining the capability to respond to and incorporate changing user requirements (Lee and Xia 2010).

Joint learning through continuous interaction as well as improved knowledge coordination may both be induced by peer-based review techniques. Thus, partially mediate the link between the extent to which these techniques are used and the performance of an ISD team. While knowledge creation may as well result from other types of interaction and socialization, giving feedback to fellow programmers regarding their work is salient to peer-based review techniques. Therefore, this study focuses on the role of feedback inherent in PP and PCR and its effect on team performance. Specifically, we aim at answering the following research questions:

1) What are the underlying feedback mechanisms behind pair programming and peer code review?

2) How do these feedback mechanisms directly and indirectly affect ISD team performance when using pair programming and peer code review?
In answering these research questions, we expect our study to contribute to previous literature in several important ways. First, research on peer-based software review techniques has primarily addressed single techniques and examined their effects in experimental settings (Balijepally et al. 2009; Dingsøyr et al. 2012; Müller 2004, 2005). While these studies provide important insights, the question of how peer-based software reviews affect team performance has so far been mostly neglected. Our study aims at filling this gap by providing a better understanding of direct and indirect effects of peer-based review techniques on team performance. In order to do so, we draw on existing work on team feedback and transactive memory systems (TMS) in the field of socio-psychology.

Second, the level of analysis of prior studies on PP or PCR was mostly individual developers or pairs (Balijepally et al. 2009; Dingsøyr et al. 2012; Müller 2004, 2005; Salleh et al. 2011). However, self-organizing teams increasingly become the crucial entity of software development organizations. This is mainly due to the widespread application of Scrum, a popular agile project management method with an explicit focus on teams (Schwaber and Beedle 2001). Consequently, the performance of development teams as a whole, rather than of individual developers, becomes the center of interest in software development organizations. This study is among the first to elevate the level of analysis from single developers or pairs to ISD teams.

Managers and ISD teams in software development organizations may learn from our study. We expect to foster a better understanding of how to allocate scarce resources on peer-based software review techniques. Specifically, our study is expected to show that for certain development settings, PP and PCR as well as their associated feedback mechanisms may act as complements instead of substitutes, as previously suggested (Müller 2004; Paulk 2001). By introducing task complexity as a contingency factor, our study aims to provide recommendations on when to use PP and PCR.

The next section briefly introduces the theoretical foundations of our study. Then, based on existing literature, we deduce a preliminary theoretical model that explains direct and indirect effects of the feedback mechanisms associated with peer-based software review techniques on ISD team performance. The paper closes with outlining the current state as well as the next steps of our research project and discussing in more detail the expected contributions to theory and practice.

**Theoretical foundations**

**Transactive Memory Systems**

In order to examine team level effects of individual level techniques, we draw from the theory of transactive memory systems (TMS). It posits that teams of interacting individuals develop a form of social cognition, the so-called TMS, through which team members distribute specialized knowledge across individuals, retrieve it from each other, and apply it in appropriate situations (Choi et al. 2010), allowing them to exceed the combined performance of individual developers (Wegner 1987). By its very nature, this theory details the relationship between individual memory systems and a collective one, based on communication (Hollingshead 1998). In accordance with TMS theory, teams with well-developed TMS have been found to show better recall of information and hence work performance (Hollingshead 1998; Kanawattanachai and Yoo 2007; Liang et al. 1995; Moreland and Myaskovsky 2000; Wasko and Faraj 2000). TMS are recognized as a powerful theoretical lens and have frequently been applied in IS, management and organizational research (Lewis and Herndon 2011; Ren and Argote 2011; Spohrer et al. 2012). Different studies show that “TMS should benefit teams through enhanced communication and coordination” (Mathieu et al., 2008) and report a positive relationship between the presence of TMS and the performance of teams (Lewis 2004). Consequently, it appears promising to study indirect performance effects of peer-based review techniques from a TMS perspective.

In more detail, a TMS consists of two components: an organized store of knowledge, the so-called TMS structure, and transactive processes that allow encoding, storage and retrieval of knowledge (Lewis and Herndon 2011). Teams with a well-developed TMS rely on the coexistence of three criteria: (1) different team members are specialized in different areas of expertise while they know about each other's
specialization, (2) team members have high credibility and trust in each other’s expertise to be correct and valuable, and (3) they are able to efficiently match team task requirements with specialized expertise (Lewis 2003).

TMS theory relies on the underlying assumption that specialization of knowledge within a team yields positive team outcomes and that a well-developed TMS is, by definition, associated with a higher knowledge specialization of its members (Ren and Argote 2011). This differentiates TMS from other types of social cognition, like shared mental models (Cannon-Bowers et al. 1993). While shared mental models shrink with increasing specialization of team members (Levesque et al. 2001), TMS even grow with increasing specialization as long as the team can coordinate its tasks in a way that allows to use individuals’ specialized expertise (Lewis and Herndon 2011). Nonetheless, there are also dynamic effects that accompany the development of a team’s TMS over time (Lewis and Herndon 2011; Ren and Argote 2011). In particular, intensive and task-related communication plays a central role in stimulating the development of a TMS (Hollingshead 1998; Kanawattanachai and Yoo 2007). Thus, TMS is an appropriate lens to examine the effects of review techniques, which essentially incorporate the communication of aspects regarding a team’s tasks and their potential solutions.

ISD teams may generally have well-developed TMS, given that they typically consist of relatively “close personal relationships in which people share responsibilities, engage in conversations about many different topics, and make joint decisions” (Hollingshead, 1998, p. 425). In line with this thought, several scholars have researched antecedents, constituents and effects of TMS in software development. ISD team performance has been found to improve with better developed TMS regarding their effectiveness, efficiency, knowledge integration and the ability to transfer knowledge to others (Faraj and Sproull 2000; Kotlarsky and Oshri 2005; Maruping et al. 2009; Oshri et al. 2008). Related disciplines have shown that TMS quality increases teams’ creativity as well as the success of new products they develop (Akgün et al. 2005; Gino et al. 2010).

Feedback

Feedback is argued to be a powerful mechanism to shape achievement and performance of a team (Hattie and Timperley 2007; Kozlowski and Ilgen 2006; London and Sessa 2006). By transferring information to individual team members reflecting their performance, processes, or behaviors, feedback can help teams develop a joint view of expectations and awareness about behaviors, abilities, and skill levels (London and Sessa 2006).

Feedback can serve several purposes, such as providing information when people move away from preset goals, guiding activities, and promoting critical reflection on a certain task and ways of working in order to obtain new insights and come up with novel approaches. London and Sessa (2006) argue that feedback helps teams monitor and regulate themselves in order to efficiently complete their tasks. Feedback is widely acknowledged as a valuable mechanism to stimulate individual learning and motivation (Gabelica et al. 2011; Nadler 1979). However, more clarity is needed regarding the question of how teams may benefit from feedback in improving their performance (Barr and Conlon 1994; Hinsz et al. 1997), or which form of feedback is actually efficient for a given setting.

In accordance with prior literature, we distinguish outcome feedback and process feedback (Earley et al. 1990; Gabelica et al. 2011; London and Sessa 2006). Outcome feedback refers to information about performance in a certain task, provided in order to reinforce good performance or to correct poor performance. By contrast, process feedback refers to information provided about the way one performs a task and comes to a solution (Gabelica et al. 2011).

Process feedback includes information about displayed behaviors and strategies, cognitive interactions with others and social conditions that are created. Process feedback, thereby, exceeds the evaluative role of outcome feedback by providing more precise directional advice in a social context, which is particularly helpful for individuals who adapt their actions in order to complete very complex tasks (Earley et al. 1990). Reviews of prior research indicate that there is a major influence of outcome feedback on team performance. By contrast, process feedback is considered more effective in improving the functioning of the team than in fostering its performance (Gabelica et al. 2011).
Feedback is a central mechanism to both review techniques, PP and PCR, as they provide developers with peers’ opinions on their work and ways of working. Through feedback, developers are able to obtain information about quality and quantity of their output as well as the effectiveness of the applied methods (Nadler 1979). Given their different nature, the studied techniques may build on different forms of feedback which may make their application more or less effective according to a given context. In the following, we discuss in more detail how applying PP and PCR leads to a different intensity of process and outcome feedback within the team. Moreover, we introduce team awareness of feedback and inclusion of expertise from outside the team as relevant feedback characteristics for direct and indirect effects of PP and PCR on team performance.

**Theoretical model**

In order to improve our understanding of how peer-based review techniques influence team performance, we propose a theoretical model that links these techniques to the feedback that is provided within the team. On the one hand, feedback is suggested to directly increase team performance. On the other hand, certain feedback characteristics are theorized to strengthen the team’s TMS, which in turn is suggested to entail positive effects on team performance. Both theoretical links are proposed to be moderated by the complexity of the team’s task. The theoretical model is depicted in Figure 1 and developed subsequently.

![Figure 1: Research model](image)

**Peer-based review techniques and feedback**

Feedback is salient to both PCR and PP as ISD team members receive information on how other developers perceive their work and way of working. However, the particular form of feedback is arguably not identical for both techniques.

When conducting PCR, code is developed by one software engineer (the author), and submitted to a review system. One or more colleagues from within or outside the team then examine the code regarding its quality and correctness in covering functional task requirements in order to propose changes, highlight defects, point out inaccuracies, or suggest potential improvements. In reaction, the author revises the code and resubmits it to the system. Several iterations of such review and revision cycles may be required before the reviewers are satisfied with the result. Throughout this process, the original author is provided with outcome feedback that is given on the developed code in order to stimulate improvement. However, the actions of the author that resulted in the developed code (i.e., the process of development) remain undisclosed to reviewers who only evaluate the submitted result.

When using PP, two developers use a single computer while collaboratively developing software (Williams...
and Kessler 2000). One of them is in the role of a driver who actively develops the solution, while the other one, the so-called observer, is continuously “observing the work of the driver and identifying tactical and strategic deficiencies in their work” (Williams 2000, p. 3). Paired developers frequently switch roles between driver and observer. Hence, one developer of a PP dyad is constantly giving feedback on the code that is being developed, thus also providing outcome feedback. In contrast to PCR, however, the observer additionally identifies and communicates issues with the driver’s approach to complete the task. This provides the coding developer with process feedback on the strategy and behaviors how to come to the final work result. This process feedback includes not only task-related, but also cognitive aspects such as differences in the perceptions of what has to be done as well as interpersonal aspects that address how the two individuals collaborate (Akkerman et al. 2007; Gabelica et al. 2011). Consequently, we propose:

P1a, b: The more an ISD team engages in PP, the higher the intensity of (a) process feedback and (b) outcome feedback it receives.

P2a: The more an ISD team engages in PCR, the higher the intensity of outcome feedback it receives.

Besides differences in the nature of feedback, PP and PCR differ with regard to a further characteristic of provided feedback: the range of potential feedback sources and recipients. During PP, feedback is exclusively given by one team member to the single partner. The PP technique does not incorporate any mechanism for sharing feedback with other team members apart from the two involved (Williams 2000). Feedback in PCR, in contrast, can come from all developers who use the same review system, typically all developers active in a certain project (Liang and Mizuno 2011). Moreover, all review requests as well as feedback information received through PCR are not only provided to the original author of the code. All information is publicly disclosed to every PCR participant, including the author’s entire team (Rigby and Storey 2011). Consequently, team members become aware of the fact that the author receives feedback including feedback details. Therefore, we propose:

P2b: The more an ISD team engages in PCR, the higher the team members’ awareness of the feedback that one of them receives.

Outcome feedback and ISD team performance.

Prior research has emphasized that outcome feedback on the individual level directly influences team performance in the way that individuals are motivated to improve exactly those dimensions of performance they receive feedback on (Gabelica et al. 2011). If ISD team members are provided with outcome feedback on the code they develop, they gain information about the quality of their code. This allows them to directly compare others’ quality expectations with their own code by being informed about concrete errors and potential improvements. In addition to providing the opportunity to remove identified errors immediately, outcome feedback should thus motivate team members to improve the code quality. Consequently, outcome feedback should increase team performance. The following proposition is put forth:

P3: The higher the intensity of outcome feedback which individual ISD team members receive, the better the performance of the team regarding quality of the code.

Feedback mechanisms and transactive memory systems

In stark contrast to outcome feedback that mainly impacts individual team members’ performance rather than interactions with others (Gabelica et al. 2011), process feedback primarily yields effects on team functioning and learning (Gabelica et al. 2011; Nadler 1979). We posit that process feedback provided to individual ISD team members strengthens team TMS as a primary effect.

Process feedback provides developers not only with information on how others judge the pursued development approach and how they perceive the challenges associated with it. In addition, it constitutes an intensive act of social interaction in which interpersonal information is exchanged as one tries to understand the other’s solution strategies (Gabelica et al. 2011). Thereby, interacting developers are enabled to monitor each other, build credibility by showing competence to their colleagues, gain insights about the specialized knowledge of others, and understand how knowledge fits with the addressed tasks.
Thirty Third International Conference on Information Systems, Orlando 2012

(Hollingshead 1998; Marks and Panzer 2004). This suggests process feedback to strengthen the TMS of a team (Decuyper et al. 2010), and we posit:

**P4:** The higher the intensity of process feedback which individual ISD team members receive, the stronger is a team’s transactive memory system.

Teams with well-developed TMS can rely on specialized knowledge of their single members. The mere specialization of a member, however, does not necessarily influence a team’s TMS as long as team members cannot locate specific knowledge domains within the team (Lewis and Herndon 2011). Making individual feedback available to the entire team and making sure that the other team members are aware of it may change this situation: the team’s understanding of each other’s roles and capabilities increases (London and Sessa 2006), they extend their knowledge about their specializations, and can more easily decide whose knowledge fits given task requirements in order to efficiently coordinate their team work.

The positive effect of the social interaction inherent in process feedback on the team’s TMS may be strengthened if the team is able to distribute the gained insights among a larger number of team members. When developers become aware of process feedback that another team member gives or receives, they learn about his/her expertise and competence, as well as tasks that are addressed in the scope of the feedback. Also, the positive effect of process feedback on the credibility of a team member may more quickly disseminate across the team. Hence, the following moderator effect is proposed.

**P5:** The higher team members’ awareness of feedback that one of them receives, the stronger the influence of the intensity of process feedback that individual team members receive on the team’s transactive memory system.

**Task complexity**

Scholars have argued that investigations of a team’s TMS should also account for differences in team tasks (Lewis and Herndon 2011). More specifically, task characteristics appear to influence how TMS can develop and operate (Jarvenpaa and Majchrzak 2008; Lewis and Herndon 2011). We proposed that process feedback stimulates the development of a team’s TMS through social interactions. These interactions help exchange opinions about approaches towards task completion based on specialized knowledge and enable team members to demonstrate credible expertise and to learn about each other’s roles in completing tasks. This appears particularly prevalent when teams tackle complex tasks that require task-specific actions and novel solutions rather than established knowledge (Akgün et al. 2005), because then developers have the chance to demonstrate and combine deep specialist expertise and to learn how to coordinate it.

Outcome feedback, in contrast, does not include the exchange of information about solution strategies, but rather about the appropriateness of an existing result (Gabelica et al. 2011). Thereby, it lacks the directional, strategy-shaping facet of process feedback. Thus, higher task complexities lead to a growing risk of feedback misinterpretation and inappropriate behavioral adjustment based on outcome feedback (Earley et al. 1990). Consequently, we propose:

**P6:** Increasing task complexity (a) amplifies the positive effect of process feedback on team TMS and (b) reduces the positive effect of outcome feedback on ISD team performance.

**Transactive memory systems and team performance**

Teams with well-developed transactive memory systems consist of specialized experts that hold the knowledge to complete the given tasks and are able to coordinate effectively and efficiently (Lewis and Herndon 2011). Through team members’ specialization and coordination, teams can combine different perspectives and expertise in order to come to more creative results and can accomplish their work more effectively and efficiently. In accordance with much prior research (c.f., Lewis and Herndon 2011; Ren and Argote 2011; Spohrer et al. 2012), we propose:

**P7:** The better developed an ISD team’s transactive memory system, the better its performance with regard to innovativeness of developed solutions, quality of the code and efficiency.
Research design and expected contributions

Research design

Research on peer-based software reviews has been described to be at the intermediate state (Dybå and Dingsøyr 2008) for which Edmondson and McManus (2007) propose qualitative studies in combination with quantitative research methods. We share this view and pursue a two-stage study approach. In the first phase of our research project, we plan to refine the proposed research model with insights from in-depth qualitative case study research at our industry partner (Yin 2009). This research setting provides us with unique access to one of the largest software development companies worldwide with several hundreds of professional development teams distributed on a global scale. We have started a series of case studies at the company’s headquarters in Europe, gathering data from a selection of teams with varying adoption intensities of PP and PCR.

In the second research phase, we plan to test the model quantitatively using structural equation modeling and partial least square techniques (Chin 1998) via a field study. For that purpose, we are in the process of developing a questionnaire-based measurement instrument based on insights from our first round of interviews as well as our review of extant literature (Cummings 2004). We then plan to pilot the instrument for refinement with teams from our partner company, and present our final measurement instrument at ICIS 2012 in order to obtain feedback. Where suitable, we will use existing construct measurement instruments as, for instance, that of Lewis (2003) to measure TMS. In addition to our industry partner, we will be able to enrich our dataset with team data from other software companies to overcome a single-firm bias. Finally, we plan for a dataset comprising answers from more than 75 teams to test our theoretical model.

For those constructs that have not been properly operationalized in existing literature, scales will be developed as described in MacKenzie et al. (2011). The intensities of the studied peer-based software review techniques will be measured via two dimensions: first, the time developers spend applying a particular technique, and second, the percentage of new code that is developed using one of the techniques. In addition, we aim at triangulating the questionnaire responses with log data of the review systems, e.g. the submitted and reviewed code changes. While there are well-established and applicable operationalizations of TMS (Lewis 2003; Lewis and Herndon 2011), we will only partially be able to use existing scales for the feedback constructs (e.g. Gabelica et al. 2011; Geister et al. 2006; London and Sessa 2006). The performance of the teams will be measured via self-developed items for innovativeness and effectiveness in combination with efficiency items from Hoegl and Gemuenden (2001). In order to reduce common method bias, particularly common rater effects, we will not only collect ratings from team members, but also from team leaders and managers where possible (Podsakoff et al. 2003).

Contribution to theory

This study is expected to contribute to existing literature on performance implications of software development techniques (Dingsøyr et al. 2012; Dybå and Dingsøyr 2008). While previous research has studied the performance effects of peer-based review techniques such as PP and PCR (c.f. Balijepally et al. 2009), a deep understanding of how these techniques may improve ISD team performance is still missing. Our study focuses on feedback mechanisms, and explains their impact on the output quality, efficiency, and innovativeness of ISD teams. In doing so, this study replies to frequent calls for theory-based and empirical studies on software development techniques and their performance effects (e.g. Dingsøyr et al. 2012; Dybå and Dingsøyr 2008; Maruping et al. 2009). Previous studies on PP and PCR relied on insights from student experiments (Müller 2004, 2005). By contrast, we plan to test our model with empirical data from professional software development teams in different organizational contexts.

Previous studies discussed effects of PP or PCR for individual programmers or pairs of software developers (Balijepally et al. 2009). We consider the performance of development teams as the relevant dependent variable, because we are convinced that teams are becoming the decisive entity for software development companies due to the widespread use of Scrum and its strong focus on development teams (Schwaber and Beedle 2001). Thus, our study contributes to previous work by being among the first to analyze performance implications of peer-based review techniques on a team level.
Literature on software development education (Salleh et al. 2011) and practice (Balijepally et al. 2009) suggests that PP may be particularly valuable if task complexity is high. Yet, to the best of our knowledge, no conclusive argument was put forth to theoretically explain the role of task complexity as a contingency factor when explaining the value of PP. By introducing feedback mechanisms and TMS as mediators of the link between PP and team performance, our model provides a sound theoretical argument: PP is particularly valuable when task complexity is high because, in this case, the process feedback inherent in PP allows for the disclosure and combination of specialist knowledge, reinforcing the positive effect on the team's TMS.

The consideration of task complexity as a contingency factor of our model also shows that PP and PCR may indeed act as complements. This may particularly be the case if teams face varying task complexities. Such teams may benefit from a combination of both techniques in that for simple tasks, quality can be assured via PCR, while the team can strengthen its TMS through relying on PP if tasks are more complex.

**Practical implications**

Research on ISD team performance is highly relevant for software development organizations as it helps managers and teams better understand which and how different factors affect team performance. The results of our study are expected to guide development teams on how and when to adopt PP and PCR. They might use the techniques exclusively or complementary to benefit the most regarding their team performance. Among practitioners, PP and PCR are often one-dimensionally discussed to cause additional effort for developers while enabling better software quality. Our research is expected to clarify that a short-sighted perspective on PP's and PCR's effects on better software quality only, may be too limited. Additional effects of using review-based development techniques should be taken into account to appropriately assess overall benefits for software development teams.

**References**


