# Pair Programming Teams and High-Quality Knowledge Sharing: A Comparative Study of Coopetitive Reward Structures

**Shahla Ghobadi (corresponding author)**
Shahla.ghobadi@manchester.ac.uk
University of Manchester, UK

**John Campbell**
John.campbell@canberra.edu.au
University of Canberra, Australia

**Stewart Clegg**
Stewart.Clegg@uts.edu.au
University of Technology Sydney, Australia

## Abstract

There has been a growing research interest in understanding knowledge sharing in agile development. Yet, empirical research that sheds light on its underlying practices, such as pair programming, is evolving. This study uses insights from coopetition and software literature to focus inquiry on the relation between coopetitive rewards and high-quality knowledge sharing in pair programming teams. Theoretical hypotheses are developed and validated, suggesting that 'coopetitive rewards influence high-quality knowledge sharing both directly and over time through their impact on the level of knowledge sharing satisfaction', and, 'the impact of coopetitive rewards on high-quality knowledge sharing is dependent upon task complexity and the history of working under similar reward structure'. This study generates new understanding related to the use of rewards in pair programming teams, and offers a rigorous and replicable seven-step experimental process for simulating coopetitive structures and investigating their role in pair programming and in similar collaborative contexts.

**Keywords**. Pair programming; agile development; software team; knowledge sharing; information sharing; rewards; coopetition; competitive reward; cooperation; competition; experiment; laboratory experiment

## 1. Introduction

Popularization of agile development methodologies has attracted interest in core practices such as pair programming (Bellini et al., 2005, Canfora et al., 2007, Dingsøyr et al., 2012), where two developers share the same computer to collaborate in all aspects of software development (Canfora et al., 2004, Müller, 2005, Williams et al., 2000, Sharp and Robinson, 2008). Both collocated and distributed pair programming teams provide beneficial teamwork experiences (Hanks, 2008, Zacharis, 2011, Müller, 2007). For example, ongoing collaboration and informal conversation among peers may foster sharing of embedded knowledge such as 'smart tool usage techniques' and 'knowing from war stories when to

or not to use a specific technology' (Bellini et al., 2005, Chau and Maurer, 2004). The shared knowledge between peers can potentially inform better design and improve software quality (Hulkko and Abrahamsson, 2005, Williams et al., 2000), or enhance team confidence and satisfaction experiences of peers (Balijepally et al., 2009).

Nonetheless, knowledge sharing in agile and pair programming teams may face several challenges (Ghobadi, 2014). For example, developers may become overburdened with coding tasks and overlook the importance of sharing and documenting embedded knowledge for future use (Ryan and O'Connor, 2009), or knowledge sharing may appear threatening since frequent communication can expose weakness and incompetency of peers regarding different aspects of software development (Ally et al., 2005). The contradictory nature of system development methodologies may also cause further difficulties (Beath and Orlikowski, 1994). For example, pair programming emphasizes collaborative interaction between developers. Yet managers may use competitive strategies to improve developers' motivation. As competitive strategies signal the possibility of obtaining individual acknowledgement, they may inhibit knowledge sharing between peers (Ghobadi and D'Ambra, 2013).

Despite growing research interest in understanding knowledge sharing in agile development, empirical research that sheds light on its underlying practices such as pair programming is still evolving (Bellini et al., 2005, Balijepally et al., 2009). A better grasp of knowledge sharing practices in pair programming contexts is required to address its increasing popularity and to develop productivity implications for software community. Furthermore, research highlights the importance of studying 'high-quality knowledge sharing' in software development to emphasize not only the mere sharing of knowledge, but the sharing of useful knowledge that helps better accomplish development objectives (Ghobadi and D'Ambra, 2012, Shih and Huang, 2014). Yet, the concept of high-quality knowledge sharing has not yet received sufficient attention in the pair programming literature.

This study takes a step toward addressing the discussed voids by integrating social interdependence theory (Deutsch, 1949) and insights from software and coopetition literature (Ghobadi and D'Ambra, 2013, Loebbecke et al., 1999). Social interdependence theory provides a theoretical foundation for understanding knowledge sharing in software teams, and it is congruent with the collaborative nature of development processes (Pee et al., 2010). The theory has three key foci. First, it recognizes the mediating role of knowledge sharing in the relation between team interdependencies and group-related outcomes, thus highlighting the importance of studying knowledge sharing in collaborative teams (Deutsch, 1949, Johnson et al., 2006). Second, the theory identifies the essential role of reward structures in shaping knowledge sharing behaviors (Ferrin and Dirks, 2003). It suggests that although changing communication patterns can be challenging and may require considerable time, flexible rewards may be easily used to influence knowledge sharing behaviors. Third, the theory draws attention to two typical reward structures that may have an impact on knowledge sharing, including:

(i) *cooperative rewards* (are based on joint performance and divided equally between group members), and, (ii) *competitive rewards* (are based on individual performance and determined competitively; e.g., individuals are rewarded for outperforming team members) (Johnson and Johnson 1989). The relation between cooperative and/or competitive rewards, team outcome variables (e.g., task speed and accuracy, social connectedness, trust), and contingent factors (e.g., task complexity) is studied frequently (Beersma et al., 2009, Beersma et al., 2003, Serrano and Pons, 2007, Slavin, 1977). For example, research has shown that (i) cooperative rewards may lead to superior team performance in extremely high or low task-interdependence environments (Wageman and Baker, 1997), while in moderately-interdependent environments competitive reward systems may be preferred (DeRue and Hollenbeck, 2007), and, (ii) competitive atmospheres help individuals develop better analytical skills, whereas collaborative atmospheres prompt higher synthetic skills (Fu et al., 2009). Notwithstanding our existing understanding, real situations are a mixture of simultaneous cooperative and competitive reward structures (referred to as *'coopetitive reward structures'*) (Gordon et al., 2000). As such, investigating the relation between reward structures and knowledge sharing necessitates paying attention to the *coopetitive nature of rewards*. Coopetitive structures *(either 'dominant cooperative'* or *'dominant competitive')* are, however, a relatively unexplored area of research. This might be due to Deutsch's argument that hybrid structures, as weaker and less stable versions of strong cooperative or competitive structures, do not need independent research (Gordon et al., 2000). Hence, further research is required to explore coopetitive reward structures in less-explored areas such as pair programming teams, identify possible differences between different types of these structures, and investigate their overall impact on knowledge sharing practices.

With this background, the present study compares the impact of two types of coopetitive reward structures on high-quality knowledge sharing in pair programming teams. The research question asks: *how do coopetitive reward structures differ in shaping high-quality knowledge sharing in pair programming teams?* The concept of *'high-quality knowledge sharing'* refers to situations in which peers are satisfied with the quality and usefulness of the shared knowledge for accomplishing development activities (Li and Hsieh, 2009, Ghobadi and D'Ambra, 2012, Ghobadi and D'Ambra, 2013). In addressing the research question, this study makes two principal contributions. First, it develops hypotheses that explain how coopetitive rewards are different in shaping high-quality knowledge sharing practices, thus generating new understanding of the use of rewards for influencing knowledge sharing behaviors in pair programming teams. Second, an experimental process is designed and implemented to validate the hypotheses in a controlled laboratory environment, thus offering a rigorous and replicable process for simulating coopetitive structures and investigating their role in programming contexts.

The remainder of the paper proceeds as follows. First, theoretical hypotheses are postulated, followed by a discussion on research methodology and data collection procedures. Next, the empirical results

are provided in detail. Theoretical implications are discussed, and strategies for managing knowledge sharing in similar contexts are outlined. The paper concludes by recommending future directions that help extend this research.

## 2. Theoretical background

### *2.1. Hypotheses Development*

On the one hand, cooperative rewards motivate individuals to engage in collaborative behaviors, such as high-quality knowledge sharing, oriented to producing better team outcomes and acquiring joint rewards (Brandenburger and Nalebuff, 1996, Beersma et al., 2003, Tan et al., 2015). On the other hand, competitive rewards may motivate individuals to engage in the opposite set of behaviors, such as less high-quality knowledge sharing simply, to maximize individual performance and subsequent rewards (Brandenburger and Nalebuff, 1996, Lucker et al., 1976, Crozier and Friedberg, 2009, Lin and Huang, 2010, Johnson and Johnson, 1998). We can therefore expect that the quality of knowledge sharing in pair programming teams is higher under cooperative reward structures than competitive reward structures.

***Hypothesis 1:*** *The quality of knowledge sharing is higher under dominant cooperative reward structures than dominant competitive structures.*

Task complexity has been recognized as a multi-dimensional concept (Campbell, 1988, Wood, 1986) that can influence knowledge sharing practices (Byström and Järvelin, 1995). According to Wood (1986), all tasks contain three sub-components that affect their levels of complexity, including: (i) their product, (ii) the acts required to enact them, and, (iii) the information cues that need to be processed. The major difference between simple and complex tasks lies in 'the number of cues that must be processed' as well as in 'the number and complexity of the required processes' (Speier et al., 2003, Wood, 1986). Simple tasks are routine information processing tasks where inputs, process, and outcomes can be determined a priori, whereas difficult or complex tasks are new and genuine decision making tasks that require innovation, thereby they cannot be determined in advance (Byström and Järvelin, 1995). Since complex tasks involve acquisition, processing and creation of complex knowledge, they require deep understanding and problem formulation (Esterhuizen et al., 2012). Therefore, complex tasks are largely dependent upon high-quality knowledge sharing compared to simple tasks. We can therefore argue that complex tasks may benefit more from cooperative rewards than simple tasks:

***Hypothesis 2:*** *There is a greater difference in the quality of knowledge sharing under dominant cooperative reward structures and dominant competitive reward structures for complex tasks compared to simple tasks.*

Teams are evolving, adaptive, and dynamic organizational systems (Johnson et al., 2006, McGrath et al., 2000, Gersick, 1988). Research suggests the importance of studying the impact of past history on group-related processes (Harrison et al., 2002, Johnson et al., 2006, Roe, 2008). Structural adaptation theory, for example, introduces two concepts of 'friendly competition' and 'cutthroat cooperation' to explain how teams may react to changes in reward structures (e.g., from cooperative to competitive structures, and vice versa) (Johnson et al., 2006). Friendly competition refers to a situation where team members are assigned to work under competitive reward structures, after they have previously worked under cooperative rewards. Cutthroat cooperation refers to a situation where teams switch from working under competitive reward structures to cooperative structures. Structural adaptation theory suggests that (i) the benefits associated with cooperative reward structures will be less in groups with a past history of competition (cutthroat cooperation), and, (ii) the negative impacts of competition will be less in groups with a past history of cooperation (friendly competition). We can, therefore, expect that the history of working under similar reward structure can influence team communication as well as the overall team performance. In the context of pair programming, prior cooperative experience encourages peers to engage in high-quality knowledge sharing. By contrast, prior competition in projects or other organizational settings may decrease the quality of knowledge sharing practices. We therefore state:

***Hypothesis 3:*** *The difference in the quality of knowledge sharing under dominant cooperative reward structures and dominant competitive reward structures is greater when the pair has a history of working under a similar reward structure.*

Coopetition literature views knowledge sharing using insights from game theory (Loebbecke et al., 1999, Shih et al., 2006). According to this literature, knowledge may have two value components, including: (i) *the basic value*, and, (ii) *the value-added* (Ghobadi and D'Ambra, 2011). The component of 'basic value' refers to the recognized value of knowledge. The 'value-added' component reflects the competitive advantage of receiving knowledge, which is not fully known for the person who shares it. These two components can turn individuals into sharing or withholding knowledge to maximize their advantage. The value-added, for example, explains why sometimes people may share knowledge easily based on their estimation of the basic value of knowledge, without being aware of the competitive advantage of knowledge for receivers. Similarly, people's perception of the value-added, which may not necessarily be true, may lead to them hoard knowledge even though sharing that knowledge may produce synergic results. In this complex context, peers may largely rely on their prior knowledge sharing experiences whether to share or not to share high-quality knowledge. For example, if they were satisfied with the outcome and consequences of engaging in knowledge sharing (e.g., they were rewarded, they learnt new skills, they felt the experience was fair), they tend to continue this practice in future as well (Willem and Buelens, 2009). We therefore argue that:

***Hypothesis 4:*** *The quality of knowledge sharing among peers is affected by their level of satisfaction from engaging in prior knowledge sharing activities.*

## 3. Research Methodology

### *3.1. Method*

Research on software engineering (Sjøberg et al., 2005), pair programming teams (Balijepally et al., 2009), and reward systems (Ferrin and Dirks, 2003, Beersma et al., 2009) suggests the relevance and strength of laboratory experiments for observing and examining causal relationships in controlled environments. Following this tradition, the present study designs and implements a rigorous procedure for conducting experiments and testing the postulated hypotheses. Specifically, a 2 X 2 factorial experimental design is utilized to examine the four hypotheses. The factors include: (i) *tasks (T2: simple/ complex)*, and, (ii) *rewards (R2: dominant cooperative/ dominant competitive)*, as discussed in below.

### *3.2. Factors*

**Tasks:** The main lecturer in charge of the Java programming course, at the university where the experiments were conducted, was consulted for designing the experimental tasks. As a result of discussions and investigating several options, a number of *pair testing tasks* were designed. Pair testing involves discovering and resolving errors in a code (Williams et al., 2000, Sjøberg et al., 2005). Pair testing has a hybrid nature since pairs need to split up to run test cases, but they also need to collaborate and find the best resolutions. Pair testing also allows for quantifying team and individuals' performance through identifying individuals' contributions. Pair testing is suitable for this study because it aligns with the mixed nature of coopetition and with our interest in studying coopetitive rewards.

Two information-related criteria guided the design of simple and complex tasks, including (i) routine (simple) *versus* genuine information processing (complex), and, (ii) no need to seek information through different channels (simple) *versus* a need to seek information through different channels (complex) (Speier et al., 2003, Byström and Järvelin, 1995). Simple tasks involved identifying and resolving five manipulated errors in a Java code related to *'add or subtract days to current date'.* Complex tasks involved identifying ten manipulated errors in a Java code related to *'calculate a rectangle area'*. This study did not assign the conventional roles of driver and observer to peers, rather we followed the concept of 'self-organizing agile teams' to allow peers to organize themselves (Cockburn and Highsmith, 2001, Highsmith and Cockburn, 2001).

**Rewards:** Coopetition in collaborative contexts is both a reality and necessity (Lin et al., 2010). In pair programming, developers are encouraged to cooperate and engage in high-quality knowledge sharing (Balijepally et al., 2009). Yet, peers' perceptions of how management recognizes expertise and efforts, assigns credit for success, and judges who contributed most to the overall performance, may create competitive sensations, which inhibit high-quality knowledge sharing (Wray, 2010). Therefore, the research design created controlled coopetitive structures to examine carefully their role in shaping knowledge sharing behaviors. Specifically, we follow a 70% and 30% reward mix to create dominant cooperative and dominant competitive reward structures (Serrano and Pons, 2007). The dominant cooperative structure (type 1) includes a 70% cooperative and 30% competitive reward mix, whereas the dominant competitive structure (type 2) includes a 70% competitive and 30% cooperative reward mix. Table 1 shows the reward system for each type of task. The pair received *cooperative points* based on the overall number of the pair's resolved errors; each peer was awarded the same number of points. The peer who resolved the most errors received competitive points; however, if peers resolved the same numbers of errors the competitive points were equally divided between them. For example, (i) a peer assigned to a simple task and a dominant cooperative reward system (*simple/ cooperative)* could receive up to 70 points for cooperative rewards (70/5=14 points for resolving each error) and 30 points for competitive rewards (as a winner), (ii) a peer assigned to a simple task and a dominant competitive reward system (*simple/ competitive)* received up to 30 points for cooperative rewards (30/5=6 points for resolving each error) and 70 points for competitive rewards (as a winner), and (iii) a peer assigned to a complex task and a dominant cooperative reward system (*complex/ cooperative)* received up to 70 points for cooperative rewards (70/10=7 points for resolving each error) and 30 points for competitive rewards (as a winner).

**Table 1. Reward Systems**

| **Experiment Condition (Task/ Reward)** | **Cooperative Points** *(for each peer)* *Based on the number of resolved errors by pair* | | | | | | | | | | **Competitive Points** | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *10* | *Winner* | *Equal* |
| Simple/ Cooperative | 14 | 28 | 42 | 56 | 70 | | | | | | 30 | 15 |
| Simple/ Competitive | 6 | 12 | 18 | 24 | 30 | | | | | | 70 | 35 |
| Complex/ Cooperative | 7 | 14 | 21 | 28 | 35 | 42 | 49 | 56 | 63 | 70 | 30 | 15 |
| Complex/ Competitive | 3 | 6 | 9 | 12 | 15 | 18 | 21 | 24 | 27 | 30 | 70 | 35 |

### *3.3. Variables*

The variables for examining the hypotheses (H1- H4) include: *'high-quality knowledge sharing'* and *'knowledge sharing satisfaction'*. *High-quality knowledge sharing* refers to situations in which peers are satisfied with the quality and usefulness of the shared knowledge for accomplishing development activities, and it was measured using three items adopted from the previous literature (Li and Hsieh, 2009, Ghobadi and D'Ambra, 2013, Ghobadi and D'Ambra, 2012). Knowledge sharing satisfaction was defined as the overall level of satisfaction gained from prior knowledge sharing experience. For

example, the peers may have shared highly useful knowledge (high-quality knowledge sharing), but they may have found the overall atmosphere competitive or unfair (e.g., only one person shard useful knowledge; low level of knowledge satisfaction). Knowledge sharing satisfaction was measured using four items adopted from the previous literature (Willem and Buelens, 2009).

To create a '*history of working under a similar reward structure'* (H3) as well as to examine *'knowledge sharing satisfaction'* (H4), two rounds of experiments were designed in which pairs were working on a different task but with similar level of complexity and under a similar reward system. Suggested by existing literature, co-variant variables were also included (*age* and *personality traits (extroversion; agreeableness))* as they may influence knowledge sharing behaviors (Beersma et al., 2003). The majority of participants were in their second or third year of undergraduate studies (with some being postgraduates). Considering age (20-26 years old) and education levels, such individuals are perceived to serve as an important source of part-time labor (Biron and Bamberger, 2010). Measures are included in section 3.5.

### *3.4. Recruiting Participants*

The research methodology involved 64 individuals recruited from the database of computer science and information systems students in an Australian university. The students formed 32 pair programming teams, and participated in 32 two-round experiments. The lab administrator sent a letter of invitation to the students registered in the database. The letter stated that (i) a background of programming in Java is required, (ii) students can practice their Java skills during tasks, and, (iii) students would be paid *up to $50* based on their performance. Students were asked to send an expression of interest stating their prior experience with Java programming. 65 students contacted the researcher, and based on their relevance to the study (experience with Java), 64 students were selected.

### *3.5. Experimental Procedure*

A seven-step design process was designed and implemented, including: (i) *planning before experiments*, (ii) *entering the lab*, (iii) *experiment (round 1)*, (iv) *after experiment and point estimations*, (v) *break between experiments*, (vi) *experiment (round 2)*, (vii) *final point estimations and payments*.

**Planning before experiments:** The planning aimed to make sure that each pair includes homogenous peers and the peers are anonymous to each other. First, students were asked to report their Java programming experience in their expression of interest. For example, they were asked to state: (i) *when did they start programming, (ii) when did they start programming in Java,* and, *(iii) whether they have attended a Java programming course, and if yes what was their final score.* The researcher reviewed responses and pre-assigned each student to work on either a 'simple' or 'complex' task. For

example, if student 'A' did not have considerable experience in Java programming, the researcher pre-assigned her/him to work on 'simple' tasks during experiments. In this way, we controlled the coding experience of individuals and made sure that the pairs included homogenous peers with similar levels of Java programming competency (Sjøberg et al., 2005). Second, to eliminate possible confounding factors such as similarity, liking, attractiveness, and nonverbal signals of trustworthiness, experiments were designed in a way that peers did not know the identity of the other (Ferrin and Dirks, 2003). Prior to experiments, two lists of pair/peer ID numbers and their assigned seats were created. The first list included IDs and seat numbers for 16 pairs (and their peers) who worked on simple tasks, out of which 8 pairs worked under cooperative rewards (SCP1 to SCP8) and 8 pairs worked under competitive rewards (SCM1 to SCM8). The second list included IDs and seat numbers for 16 pairs (and their peers) who worked on complex tasks, out of which 8 pairs worked under cooperative rewards (CCP1 to CCP8) and 8 pairs worked under competitive rewards (CCM1 to CCM8). For example, SCP11 and SCP12 (peers in pair SCP1) were pre-assigned to sit at the lab stations of 1 and 6 (according to the first list). The lists were used as a guide at the time of the students' arrival in the lab, as discussed in the following section. In addition, pairs only communicated using anonymous ID codes and via an instant messaging application that was available in the lab (http://spark.en.softonic.com). Finally, the procedure, sample and measures, as well as the order in which the questions were asked, were carefully discussed with experts in experimental design.

**Entering the lab:** Upon arrival, the researcher asked the name of each individual to check the type of tasks that s/he was pre-assigned to (based on her/his coding competency). Looking at the ID lists (either for simple or complex tasks), each individual was assigned randomly to a pair ID and was given a briefing form. The form stated her/his individual ID, her/his seat number, and her/his peer's ID. The form explained that the experiment includes two rounds of 20-minute Java coding activity and a 10-minute break in between. After participants were located, the researcher gave them a form to note their demographic details (age, personality traits). Regarding extroversion, participants were asked to indicate their level of extroversion from '1 = I prefer to work independently from others' to '5 = I enjoy working with others in a group'. Regarding agreeableness, respondents were asked to indicate their level of agreeableness from '1 = I am competitive rather than cooperative' to '5 = I am eager to help and be helped in return.

**Experiment (round 1):** Each participant was given the relevant coding task and the reward system (one of the rows of Table 1). The reward system explained how they can earn points and subsequent cash based on their performance. The researcher asked participants to log into the instant messaging system, add their peer ID, and start working in the first round of the experiment. All the discussions online were monitored to make sure that students do not reveal identity. After 20 minutes of activity, students reported their resolutions and noted who has resolved each of the errors; they sent this

information to the researcher ID that was visible to them. In the case of conflict between peers, a third-party (a PhD candidate) checked the archive of communications to provide resolution.

**After experiment and point estimations:** The students were given a questionnaire asking them questions that measure high-quality knowledge sharing in the first round (Li and Hsieh, 2009, Ghobadi and D'Ambra, 2013). They were asked to indicate their level of agreement to the following statements: (i) *I am satisfied with the quality of the knowledge that my peer shared (1=Strongly Disagree to 5=Strongly Agree),* (ii) *I found the shared knowledge by my group mate useful in accomplishing the task* (1=Strongly Disagree to 5=Strongly Agree), and (iii) *Overall, the quality of the shared knowledge by my peer was* (1=Very Low to 5=Very High). After receiving their answers, the researcher calculated the acquired points for the components of each pair, and then announced the results via the messaging system. For example, the researcher informed SCP1 that SCP11 and SCP12 are awarded 56 and 86 points, respectively (the team resolved four errors all together). The purpose of informing peers about each other's points was to influence their satisfaction of knowledge sharing practices, and in turn influence their knowledge sharing behaviors in the second round of the experiment (e.g., the person who received 56 (30 points less) may become more reserved in sharing knowledge in the second round to be able to receive the competitive point).

**Break between experiments:** The participants were asked to stay within the lab and not to talk to others during the 10-minute break. Toward the end of the break, the researcher administrated a questionnaire that included 6 questions for checking 'how well rewards and tasks were manipulated in the previous round' and for measuring 'knowledge sharing satisfaction from the previous round'. Regarding task complexity, respondents rated complexity of the task they had completed on a scale ranging from '1 = not at all complex' to '5 = complex'. Regarding reward manipulation, participants indicated their degree of agreement with a statement using a scale ranging from 1 to 5; one end was '*To be rewarded, the assigned task required more cooperative work than competing with the group mate'* (referring to dominant cooperative rewards), and the other end was *'To be rewarded, the assigned task required more outperforming the peer than cooperative work'* (referring to dominant competitive rewards). Regarding knowledge sharing satisfaction, respondents were asked to indicate their opinion regarding the following statements (Willem and Buelens, 2009): (i) *How satisfied are you with the exchange of knowledge and experiences during the task?* (1=Very Low to 5=Very High), (ii) *There was some knowledge that was not shared, and caused delay or lower performance* (1=Strongly Disagree to 5=Strongly Agree), (iii) *There was sufficient sharing of experiences and ideas during the cooperation* (1=Strongly Disagree to 5=Strongly Agree), (iv) *Lack of sufficient knowledge sharing disturbed cooperation and task accomplishment* (1=Strongly Disagree to 5=Strongly Agree); the second and fourth items were reverse coded during analysis.

**Experiment (round 2):** After the break, each pair worked on a new coding task with similar level of complexity and under the same reward system. After 20 minutes of activity, students reported their resolutions and noted who resolved each of the errors. A questionnaire asking about manipulation checks and high-quality knowledge sharing in the second round was administrated.

**Final point estimations and payments:** The researcher calculated the final points for each individual in both rounds, and each person was paid accordingly.

## 4. Results

*Manipulation Checks.* As described in the experimental procedure, manipulation questions were asked to check how well tasks (simple/complex) as well as rewards ('dominant cooperative/dominant competitive) were simulated. To avoid non-independence of observations, only data provided by one random member of each pair was used (Ferrin and Dirks, 2003). The results of t-tests revealed significant differences across the manipulated conditions. Specifically, (i) the respondents assigned to the dominant cooperative reward structure viewed the reward system as more cooperative (M=1.75, SD=0.577), whereas the respondents assigned to the dominant competitive reward structure viewed the reward system as more competitive (M=3.63, SD=0.5) (t (16)=9.82, $p<0.05$), and, (ii) the respondents assigned to work under simple tasks viewed the task more simple (M=2.31, SD=0.793) and the respondents assigned to work under complex tasks viewed the task more complex (M=3.56, SD=0.934) (t (16)=4.005, $p<0.005$). Although 2 X 2 factorial ANOVA conducted on both reward and task checks yielded the main effect for reward and task conditions, neither task or reward conditions did have significant unintended interaction effect on the task and reward manipulation checks, respectively (F(3, 28)=0.832, p=0.488), (F(3, 28) =0.048, p=0.827). The descriptive statistics and the intercorrelations are displayed in Table 2. To avoid non-independence of observations, we only use data provided by one random member of each pair (Ferrin and Dirks, 2003).

**Table 2. Means, Standard Deviations, and Intercorrelations**

| | M | SD | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|---|
| (1) Task Complexity | 2.69 | 1.09 | 1 | | | | | | | |
| (2) Reward System | 2.94 | 1.08 | -0.21 | 1 | | | | | | |
| (3) Average Age | 21.91 | 2.10 | .621** | -0.303 | 1 | | | | | |
| (4) Average Extroversion | 3.38 | 0.87 | .433* | -0.146 | 0.126 | 1 | | | | |
| (5) Average Agreeableness | 2.94 | 0.67 | 0.105 | -0.006 | .386* | -0.235 | 1 | | | |
| (6) High-Quality Knowledge Sharing(round1) | 2.96 | 0.64 | -0.111 | -0.33 | 0.108 | -0.029 | 0.044 | 1 | | |
| (7) Knowledge Sharing Satisfaction | 3.17 | 0.59 | -0.114 | -.503** | 0.02 | -0.051 | -0.217 | 0.331 | 1 | |
| (8) High-Quality Knowledge Sharing(round2) | 3.08 | 1.23 | 0.044 | -.589** | 0.178 | 0.09 | -0.098 | .729** | .424* | 1 |

*N=32; ** Correlation is significant at the 0.01 level (2-tailed). * Correlation is significant at the 0.05 level (2-tailed).*

*Validities.* A confirmatory factor analysis of the high-quality knowledge sharing in both rounds and

knowledge sharing satisfaction indicated a good fit and supported convergent validity of the constructs (Chi-Square=261.494 df=45, p<0.001; most of the factor loadings> 0.7 (two were close at 0.68). Table 3 demonstrates the results of factor analysis for variable measures (six items measuring high-quality knowledge sharing in the first round (HQ Knowledge Sharing (round1)-1 to HQ Knowledge Sharing (round1)-3) and the second round (HQ Knowledge Sharing (round2)-1 to HQ Knowledge Sharing (round2)-3), and four items measuring knowledge sharing satisfaction (Knowledge Sharing Satisfaction-1 to Knowledge Sharing Satisfaction-4). Table 3 also shows correlations between the three identified factors. Extracted variances between high-quality knowledge sharing in both rounds and knowledge sharing satisfaction were considerably greater than the correlation square between them, establishing discriminant validity of the constructs (high-quality knowledge sharing (round1) and knowledge sharing satisfaction: 0.6 > 0.11; high-quality knowledge sharing (round1) and high-quality knowledge sharing (round2): 0.74 > 0.47; high-quality knowledge sharing (round2) and knowledge sharing satisfaction: 0.75 > 0/17).

**Table 3. Factor Analysis Results & Correlations**

| | Factor 1 | Factor 2 | Factor 3 |
|---|---|---|---|
| HQ Knowledge Sharing (round2)-2 | 0.977 | | |
| HQ Knowledge Sharing (round2)-3 | 0.917 | | |
| HQ Knowledge Sharing (round2)-1 | 0.786 | | |
| Knowledge Sharing Satisfaction-2 | | 0.974 | |
| Knowledge Sharing Satisfaction-3 | | 0.846 | |
| Knowledge Sharing Satisfaction-1 | | 0.708 | |
| Knowledge Sharing Satisfaction-4 | | 0.812 | |
| HQ Knowledge Sharing (round1)-1 | | | 0.927 |
| HQ Knowledge Sharing (round1)-2 | | | 0.689 |
| HQ Knowledge Sharing (round1)-3 | | | 0.686 |

| Factor/Correlation | 1 | 2 | 3 |
|---|---|---|---|
| HQ Knowledge Sharing (round 2) | 1 | 0.411 | 0.682 |
| Knowledge Sharing Satisfaction | 0.411 | 1 | 0.334 |
| HQ Knowledge Sharing (round 1) | 0.682 | 0.334 | 1 |

*Hypothesis Testing.* The Mann-Whitney U test is a nonparametric test of the null hypothesis that compares two samples that come from the same population against an alternative hypothesis. As such, this test was used to examine the difference between the two types of reward structures (two samples) in determining high-quality knowledge sharing (H1). The comparison of two independent samples showed significant difference in high-quality knowledge sharing of participants (round 1) under dominant cooperative (Mean: 22.63, n=16) and dominant competitive (Mean: 10.38, n=16) reward structures (confirming hypothesis 1; U=30.00, z= -3.774, p<0.001). Similarly, in round 2, the Mann-Whitney U test revealed significant difference in high-quality knowledge sharing of subjects under dominant cooperative (Mean: 24.5, n=16) and dominant competitive reward structures (Mean: 8.5, n=16) (U=. 000, z= -4.903, p<0.001). These results, as shown in Table 4, confirmed Hypothesis 1 (*the quality of knowledge sharing is higher under dominant cooperative reward structures than dominant competitive structures*). Please note that the ranks results in the table show the largest rank as best.

**Table 4. Mann-Whitney U test**

| Variable | Reward Type | N | Rank Mean |
|---|---|---|---|
| HQ Knowledge Sharing (round1) | Dominant Cooperative | 16 | 22.63 |
| | Dominant Competitive | 16 | 10.38 |
| HQ Knowledge Sharing (round2) | Dominant Cooperative | 16 | 24.5 |
| | Dominant Competitive | 16 | 8.5 |

| | HQ Knowledge Sharing (round1) | HQ Knowledge Sharing (round2) |
|---|---|---|
| Mann-Whitney U | 30 | 0 |
| Z | -3.774 | -4.903 |
| Asymp. Sig. (2-tailed) | <0.001 | <0.001 |

Univariate analysis is a statistical test for examining the association of one specific variable with the phenomenon of interest. As such, this test was used to examine whether the difference in the quality of the knowledge being shared under dominant cooperative reward structures and dominant competitive reward structures (H2) is greater for complex tasks compared to simple tasks (variable). The interaction effect analysis confirmed the hypothesis ($F=16.04$, $df=1$, $p<0.001$; $F=5.04$, $df=1$, $p=0.03<0.05$). The visual comparison of the difference between high-quality knowledge sharing in dominant cooperative and dominant competitive rewards for simple and complex tasks (the end points of lines), shown in Figure 1, was consistent with Hypothesis 2 (*There is a greater difference in the quality of knowledge sharing under dominant cooperative reward structures and dominant competitive reward structures for complex tasks compared to simple tasks.* For example, there is greater difference between the end points of the bold lines (associated with complex tasks) than the end points of the normal lines (associated with simple tasks).

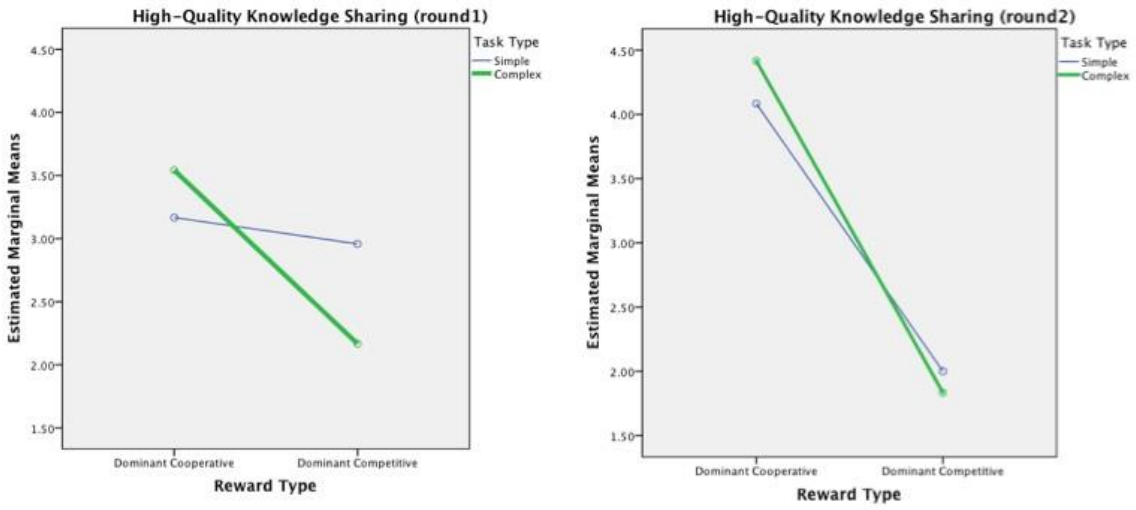


**Figure 1. Hypothesis 2 Plot**

Similarly, Univariate analysis was carried out to assess the differences in the quality of knowledge sharing under dominant cooperative reward structures and dominant competitive reward structures

(H3) in both experimental rounds (specific variable of 'history'). To do this analysis, (i) a new variable column (labeled 'history') was added in SPSS, (ii) the high-quality knowledge sharing values for both rounds were merged (64 rows), and, (iii) for the values of high-quality knowledge sharing in round 1 and round 2, values of 0 and 1 were assigned to the 'history' variable, respectively. The results of univariate analysis confirmed that the difference in the quality of the knowledge sharing under dominant cooperative reward structures and dominant competitive reward structures was greater when pairs had the experience of working under similar reward structure (history value=1) ($F=51.146$, $df=1$, $p<0.0.01$).

Spearman Correlation analysis helps assess how well the relationship between two variables can be described using a monotonic function. To examine Hypothesis 4, this analysis was employed to check the impact of knowledge sharing satisfaction on high-quality knowledge sharing. The analysis showed support for Hypothesis 4 (correlation coefficient = 0.423 $p=0.016<0.05$), suggesting that the quality of knowledge sharing among peers is affected by their level of satisfaction from engaging in prior knowledge sharing activities. Separate analysis for the pairs working under dominant cooperative and dominant competitive reward structures also supported this hypothesis (correlation coefficient = 0.46 $p<0.05$).

Analyzing communication between peers via the instant messaging system supported these insights. For example, in dominant cooperative structures, satisfying and friendly messages before and after the first break were observed: *"no worries mate, it's time to show your java skills" or "good game. Hopefully we can get through this"*. However, peers did communicate and chat less in competitive structures. Finally, the possible impact of demographic variables (age and personality traits) on high-quality knowledge sharing and knowledge sharing satisfaction were examined. The regression analysis and coefficients did not show a significant relation between the demographics and the study's variables ($p>0.05$). The results are discussed in the following section.

## 5. Discussion

### *5.1. Theoretical Implications*

Although research on knowledge sharing in agile development is gaining theoretical rigor (Ramesh et al., 2012, Ghobadi and Mathiassen, 2014), focused research on its underlying knowledge-intensive practices (e.g., pair programming) is scarce (Balijepally et al., 2009). This study is undertaken to shed more light on high-quality knowledge sharing practices in pair programming teams, where knowledge sharing may face difficulties with regard to the nature of pair programming tasks, daily social relationships, and the competitive nature of knowledge (Cockburn, 2006, Conboy et al., 2010, Nerur et al., 2005, Chan and Thong, 2009).

To investigate knowledge sharing in this context, the present study borrows the perspective of the 'coopetitive reward structures' from social interdependence theory (Deutsch, 1949) and insights regarding *'coopetitive knowledge sharing'* from the software development and coopetition literature (Ghobadi and D'Ambra, 2013, Loebbecke et al., 1999, Ghobadi and D'Ambra, 2012). More specifically, the impact of two types of coopetitive reward structures on high-quality knowledge sharing is investigated and compared, asking '*how do coopetitive reward structures differ in shaping high-quality knowledge sharing in pair programming teams?*' In addressing this research question, two principal theoretical contributions extend existing research.

First, this study contributes to the extant literature by (i) simulating two types of coopetitive reward structures (dominant cooperative and dominant competitive) in the less-explored area of pair programming teams, and (ii) identifying differences between them in terms of their overall impact on high-quality knowledge sharing behaviors. More specifically, four hypotheses are developed and validated, stating that: (i) the quality of knowledge sharing is higher under dominant cooperative structures than dominant competitive structures, (ii) there is a greater difference in the quality of knowledge sharing under dominant cooperative structures and dominant competitive structures for *complex tasks compared to simple tasks* as well as for *when the pair has a history of working under a similar reward structure*, and, (iii) the quality of knowledge sharing in both reward structures is higher when peers are overall satisfied from their engagement in prior knowledge sharing activities (e.g., fairness). The implications of these results include: (i) coopetitive rewards influence high-quality knowledge sharing in pair programming teams both directly and over time through their impact on the level of knowledge sharing satisfaction, and, (ii) the impact of coopetitive rewards on high-quality knowledge sharing in pair programming teams is dependent upon task complexity and the history of working under similar reward structure. By showing the importance of knowledge sharing satisfaction, the findings affirm prior research suggesting that pre-existing conditions determine how rewards influence group outcomes (Ferrin and Dirks, 2003). By demonstrating the superiority of dominant cooperative reward structures, the results indicate that dominant cooperative and competitive reward structures still correspond to their extreme version of cooperative and competitive reward structures, thereby adhering to the coopetition literature in the context of pair programming teams (Deutsch, 1949, Johnson and Johnson, 2006). Existing research reports that flaws and inconsistencies in measuring 'coopetition' often lead to contrary results about the role of cooperation and competition in collaborative contexts (Ghobadi and D'Ambra, 2011). Through designing and following a rigorous and controlled procedure for simulating and examining 'coopetitive structures', this study establishes theoretical results supporting prior research rather than contributing to divergent research outcomes.

Second, this study offers a rigorous and replicable experimental process for simulating coopetitive structures and exploring their role in pair programming teams and in similar collaborative contexts. The process covers *creating simple and complex pair programming tasks* (here, pair testing), *designing coopetitive rewards systems*, and *developing a comprehensive seven-step experimental procedure for examining the role of coopetitive rewards in pair programming contexts*. The procedure takes into account *careful team planning* (team composition, anonymity of peers) and *the order* in which the experimental steps should be executed to create carefully a sense of cooperation and competition among peers. In addition to being useful for replication in pair programming research, the experimental process contributes to advancing the recent emphasis on understanding inevitable coopetitive structures at intra-organizational levels (Ghobadi and D'Ambra, 2012). Experimental setting has been useful in examining the relation between rewards and knowledge sharing for at least three reasons. First, it has allowed us to control for confounding effects (e.g., age, personality traits) which may influence this relationship (evidence: no significant impact of cofounding effects on knowledge sharing). Second, it has enabled us to simulate coopetitive rewards in a robust manner (evidence: significant supporting results in the manipulation checks). Third, it allowed comparing the knowledge sharing consequences of the contrasting styles of rewards in the controlled context *(*evidence: checking hypotheses*).*

### *5.2. Practical Implications*

Team-based incentives are commonly used to promote better performance and improve coordination of efforts (Garvey, 2002). In response, research has studied how rewards may best support team-based structures (Lin et al., 2008, Taylor, 2006). Yet, coopetitive rewards in the highly collaborative context of software development have not received adequate theoretical attention. Furthermore, in practice, software development companies largely acknowledge and broadcast cooperative atmospheres and structures. There is less acknowledgment of coopetitive rewards inherent in software development works.

This study improve our understanding of how cooperative and competitive reward structures may coexist in pair programming teams and how these structures may influence high-quality knowledge sharing among peers. We discuss the influence that reward structures, task complexity, knowledge sharing satisfaction, and the history of group-based rewards may have on collaborative behaviors in pair programming teams. Based on the empirical results, the following considerations (Ensure Minimize Leverage Develop) can be used to support high-quality knowledge sharing in pair programming teams: (i) Ensure that reward structures are clear, communicated and understood well at different organizational levels, (ii) overall, Minimize hidden competitive rewards (e.g., signals from management in acknowledging individual contribution) and promote dominant cooperative rewards structures, *especially if the pair programming task is complex and the team members are more experienced*, (iii)

If the peers have previously cooperated, Leverage prior experiences through engaging in cooperative structures, and, (iv) Develop collaborative strategies that increase knowledge sharing satisfaction of peers *(e.g., making sure that team members are homogeneous in their work and social skills so that they can contribute equally and thus they feel appreciated)*.

### *5.2. Limitation and Concluding Points*

This study contributes to the literature by (i) generating new understanding of the use of rewards for promoting knowledge sharing in pair programming teams, and (ii) offering a rigorous and replicable experimental process for simulating coopetitive structures and investigating their role in pair programming and in similar collaborative contexts. Experimental tasks were designed with the input from an expert who had taught the course for more than fifteen years and has remained active in corporate software development endeavors. The median duration of experiments in software engineering is reported to be one hour (Sjøberg et al., 2005), which is consistent with the duration of experiments in this study. We acknowledge a number of limitations that provide the opportunity for future research.

First, the sample of 64 students and 32 experiments may be adequate for the statistical analysis (2 X 2 design), yet the sample may be extended. Future research may check the results in larger samples and with different characteristics (e.g., different tasks). Second, the timing of each round as well as the break between the two rounds were relatively short to allow possible learning effects to unfold and allow more high-quality knowledge sharing to emerge (Wang et al., 2015). It is recommended to develop controlled experiments with extended periods. Third, this study used quantitative measures that are proposed and validated in software, coopetition, and knowledge management literature. Both qualitative and quantitative analysis using objective data such as observation of task processes and conversations may extend the findings and lead to development of new hypotheses and propositions.

Fourth, the design process allocated the less experienced students to the simple tasks. In this study, postgraduate students, generally older, were information systems students, taking this course to be familiar with programming principles. Thus, they were not necessarily more experienced in Java than the first or second year undergraduate software engineering students. Yet, the results showed a correlation between age and task complexity (Table 2), suggesting that there is a greater difference in the quality of knowledge sharing under dominant cooperative reward structures and dominant competitive reward structures for more experienced programmers compared to less experienced programmers. Future research should pay attention to the role of ‘programming experience’ while investigating the difference in the quality of knowledge sharing under dominant cooperative reward structures and dominant competitive reward structure (Hypothesis 2).

Fifth, this study compared the impact of two types of reward structure on high-quality knowledge sharing in pair programming teams. Knowledge sharing is noted to mediate the impact of rewards on

group outcomes (Deutsch, 1949). Extant literature also acknowledges high-quality knowledge sharing as an essential aspect of agile development teams (Gupta and Bajwa, 2012, Ghobadi and Mathiassen, 2014). Yet, it would be of absolute value to develop new qualitative and quantitative studies that include other outcome variables such as software quality or project success (Hulkko and Abrahamsson, 2005) and investigate the role of coopetitive rewards on them. Such studies, as also advocated by existing research reviews (Pais and dos Santos, 2014), will advance our understanding of how the three layers of 'coopetitive structures', 'high-quality knowledge sharing' and 'performance outcomes' interact and influence each other.

## Acknowledgment

This research received the experimental small project grant from the ASB experimental research laboratory in 2011. Thanks to Ben Greiner for his feedback in designing the experiments, Claude Sammut for his input in designing the tasks, and Matthew Tolhurst for his support in running the experiments. Many thanks also to the review team for their useful and constructive comments.

## References

Ally, M., Darroch, F. & Toleman, M. (2005) A framework for understanding the factors influencing pair programming success, eXtreme Programming and Agile Processes in Software Engineering, Sheffield, UK, 1-10.

Balijepally, V., Mahapatra, R., Nerur, S. & Price, K. H. (2009) Are two heads better than one for software development? The productivity paradox of pair programming. MIS Quarterly, 33 (1), 91-118.

Beath, C. M. & Orlikowski, W. J. (1994) The contradictory structure of systems development methodologies: deconstructing the IS-user relationship in information engineering. Information Systems Research, 5 (4), 350-377.

Beersma, B., Hollenback, J. R., Conlon, D. E., Humphrey, S. E., Moon, H. & Ilgen, D. R. (2009) Cutthroat cooperation: The effects of team role decisions on adaptation to alternative reward structures. Organizational Behavior and Human Decision Processes, 108 (1), 131-142.

Beersma, B., Hollenbeck, J. R., Humphery, S. E., Moon, H., Conlon, D. E. & Ilgen, D. R. (2003) Cooperation, competition, and team Performance: toward a contingency approach. Academy of Management Journal, 46 (5), 572-590.

Bellini, E., Canfora, G., García, F., Piattini, M. & Visaggio, C. A. (2005) Pair designing as practice for enforcing and diffusing design knowledge. Journal of Software Maintenance and Evolution: Research and Practice, 17 (6), 401-423.

Biron, M. & Bamberger, P. (2010) The impact of structural empowerment on individual well-being and performance: Taking agent preferences, self-efficacy and operational constraints into account. Human Relations, 63 (2), 163-191.

Brandenburger, A. M. & Nalebuff, B. J. (1996) *Co-Opetition: a revolutionary mindset that combines competition and co-operation*, Bantam Dell Pub Group, New York.

Byström, K. & Järvelin, K. (1995) Task complexity affects information seeking and use. Information processing & management, 31 (2), 191-213.

Campbell, D. J. (1988) Task complexity: A review and analysis. Academy of Management Review, 40-52.

Canfora, G., Cimitile, A., Garcia, F., Piattini, M. & Visaggio, C. A. (2007) Evaluating performances of pair designing in industry. Journal of Systems and Software, 80 (8), 1317-1327.

Canfora, G., Cimitile, A. & Visaggio, C. A. (2004) Working in pairs as a means for design knowledge building: an empirical study, Program Comprehension, 2004. Proceedings. 12th IEEE International Workshop on: IEEE, 62-68.

Chan, F. K. Y. & Thong, J. Y. L. (2009) Acceptance of agile methodologies: A critical review and conceptual framework. Decision Support Systems, 46 (4), 803-814.

Chau, T. & Maurer, F. 2004. Knowledge sharing in agile software teams. In*: Logic versus approximation*, Wolfgang, L. (ed.) pp. 173-183. United States, Springer.

Cockburn, A. (2006) *Agile software development: the cooperative game (agile software development series)*, Addison-Wesley Professional, Boston, United States.

Cockburn, A. & Highsmith, J. (2001) Agile Software Development:The People Factor. Computer, 34 (11), 131-133.

Conboy, K., Coyle, S., Wang, X. & Pikkarainen, M. (2010) People over process: key people challenges in agile development. IEEE Software, 99 (1), 47-57.

Crozier, M. & Friedberg, E. (2009) *The bureaucratic phenomenon*, Transaction Pub, London.

Derue, D. S. & Hollenbeck, J. R. (2007) The search for internal and external fit in teams. Perspectives on organizational fit, 259-285.

Deutsch, M. (1949) A theory of cooperation and competition. Human Relations, 2 (2), 129-152.

Dingsøyr, T., Nerur, S., Balijepally, V. G. & Moe, N. B. (2012) A decade of agile methodologies: Towards explaining agile software development. Journal of Systems and Software, 85 (6), 1213-1221.

Esterhuizen, D., Schutte, C. S. L. & Du Toit, A. S. A. (2012) Knowledge creation processes as critical enablers for innovation. International Journal of Information Management, 32 (4), 354-364.

Ferrin, D. L. & Dirks, K. T. (2003) The use of rewards to increase and decrease trust: mediating processes and differential effects. Organization Science, 14 (1), 18-31.

Fu, F.-L., Wu, Y.-L. & Ho, H.-C. (2009) An investigation of coopetitive pedagogic design for knowledge creation in Web-based learning. Computers & Education, 53 (3), 550-562.

Garvey, C. (2002) Steer teams with the right pay. HR Magazine, 47 (1), 71-78.

Gersick, C. J. G. (1988) Time and Transition in Work Teams: Toward a New Model of Group Development Academy of Management Journal, 31 (1), 9-41.

Ghobadi, S. (2014) What drives knowledge sharing in software team: a review and classification framework. Information & Management 52 (1), 82-97.

Ghobadi, S. & D'ambra, J. (2011) Coopetitive Relationships in Cross-Functional Software Development Teams: How to Model and Measure? Journal of Systems and Software, 85 (5), 1096-1104.

Ghobadi, S. & D'ambra, J. (2012) Knowledge sharing in cross-functional teams: a coopetitive model. Journal of Knowledge Management, 16 (2), 285-301.

Ghobadi, S. & D'ambra, J. (2013) Modeling High-Quality Knowledge Sharing in Cross-Functional Software Development Teams Information Processing & Management, 49 (1), 138-157.

Ghobadi, S. & D'ambra, J. (2011) Coopetitive Knowledge Sharing: An Analytical Review of Literature. Electronic Journal of Knowledge Management, 9 (4), 307-317.

Ghobadi, S. & Mathiassen, L. (2014) Perceived Barriers to Effective Knowledge Sharing in Agile Software Teams. Information Systems Journal (DOI: 10.1111/isj.12053).

Gordon, F. M., Welch, K. R., Offringa, G. & Katz, N. (2000) The complexity of social outcomes from cooperative, competitive, and individualistic reward systems. Social Justice Research, 13 (3), 237-269.

Gupta, N. & Bajwa, J. K. (2012) Analysis of Knowledge Sharing Practices in Distributed Agile Environment. International Journal of Computer & Communication Technology, 3 (6-7).

Hanks, B. (2008) Empirical evaluation of distributed pair programming. International Journal of Human-Computer Studies, 66 (7), 530-544.

Harrison, D. A., Price, K. H., Gavin, J. H. & Florey, A. T. (2002) Time, teams, and task performance: Changing effects of surface-and deep-level diversity on group functioning. Academy of Management Journal, 45 (5), 1029-1045.

Highsmith, J. & Cockburn, A. (2001) Agile software development: The business of innovation. Computer, 34 (9), 120-127.

Hulkko, H. & Abrahamsson, P. (2005) A multiple case study on the impact of pair programming on product quality, Proceedings of the 27th international conference on Software engineering: ACM, 495-504.

Johnson, D. W. & Johnson , R. T. (1989) *Cooperation and competition: theory and research*, Interaction Book Company, MN, US.

Johnson, D. W. & Johnson, R. T. (1998) Cooperative learning and social interdependence theory. Social psychological applications to social issues, 4, 9-36.

Johnson, D. W. & Johnson, R. T. (2006) New developments in social interdependence theory. Genetic, Social, and General Psychology Monographs, 131 (4), 285-358.

Johnson, M. D., Hollenbeck, J. R., Humphrey, S. E., Ilgen, D. R., Jundt, D. & Meyer, C. J. (2006) Cutthroat cooperation: asymmetrical adaptation to changes in team reward structures. Academy of Management Journal, 49 (1), 103-119.

Li, C. & Hsieh, C. (2009) The impact of knowledge stickiness on knowledge transfer implementation, internalization, and satisfaction for multinational corporations. International Journal of Information Management, 29 (6), 425-435.

Lin, C., Tan, B. & Chang, S. (2008) An exploratory model of knowledge flow barriers within healthcare organizations. Information and Management, 45 (5), 331-339.

Lin, S. P., Wang, Y. C., Y.H., T. & Hsu, Y. F. (2010) Perceived job effectiveness in coopetition: A survey of virtual teams within business organizations. Computers in Human Behavior, 26 (1), 1598-1606.

Lin, T. C. & Huang, C. C. (2010) Withholding effort in knowledge contribution: The role of social exchange and social cognitive on project teams. Information and Management, 47 (1), 188-196.

Loebbecke, C., Van Fenema, P. C. & Powell, P. (1999) Co-Opetition and knowledge transfer. The Database for Advances in Information Systems, 30 (2), 14-25.

Lucker, G. W., Rosenfield, D., Sikes, J. & Aronson, E. (1976) Performance in the interdependent classroom: A field study. American Educational Research Journal, 13 (2), 115-123.

Mcgrath, J. E., Arrow, H. & Berdahl, J. L. (2000) The study of groups: past, present, and future. Personality and Social Psychology Review, 4 (1), 95-105.

Müller, M. M. (2005) Two controlled experiments concerning the comparison of pair programming to peer review. Journal of Systems and Software, 78 (2), 166-179.

Müller, M. M. (2007) Do programmer pairs make different mistakes than solo programmers? Journal of Systems and Software, 80 (9), 1460-1471.

Nerur, S., Mahapatra, R. K. & Mangalaraj, G. (2005) Challenges of migrating to agile methodologies. Communications of the ACM, 48 (5), 72-78.

Pais, L. & Dos Santos, N. R. (2014) and Personal Development. The Wiley Blackwell Handbook of the Psychology of Training, Development, and Performance Improvement, 278.

Pee, L. G., Kankanhalli, A. & Kim, H. W. (2010) Knowledge sharing in information systems development: a social interdependence perspective. Journal of the Association for Information Systems, 11 (10), 550-575.

Ramesh, B., Mohan, K. & Cao, L. (2012) Ambidexterity in Agile Distributed Development: An Empirical Investigation. Information System Research, 23 (2), 323-339.

Roe, R. A. (2008) *Time in organizational research*, Routledge.

Ryan, S. & O'connor, R. V. (2009) Development of a team measure for tacit knowledge in software development teams. Journal of Systems and Software, 82 (2), 229-240.

Serrano, J. M. & Pons, R. M. (2007) Cooperative learning: we can also do it without task structure. Intercultural Education, 18 (3), 215-230.

Sharp, H. & Robinson, H. (2008) Collaboration and co-ordination in mature eXtreme programming teams. International Journal of Human-Computer Studies, 66 (7), 506-518.

Shih, H.-P. & Huang, E. (2014) Influences of Web interactivity and social identity and bonds on the quality of online discussion in a virtual community. Information Systems Frontiers, 16 (4), 627-641.

Shih, M.-H., Tsai, H.-T. & Wu, C.-C. (2006) A holistic knowledge sharing framework in high-tech firms: game and co-opetition perspectives International Journal of Technology Management, 36 (4), 354-366.

Sjøberg, D. I., Hannay, J. E., Hansen, O., Kampenes, V. B., Karahasanovic, A., Liborg, N.-K. & Rekdal, A. C. (2005) A survey of controlled experiments in software engineering. IEEE Transactions on Software Engineering, 31 (9), 733-753.

Slavin, R. E. (1977) Classroom reward structure: An analytical and practical review. Review of Educational Research, 47 (4), 633-650.

Speier, C., Vessey, I. & Valacich, J. S. (2003) The effects of interruptions, task complexity, and information presentation on computer-supported decision-making performance. Decision Sciences, 34 (4), 771-797.

Tan, K. H., Wong, W. & Chung, L. (2015) Information and Knowledge Leakage in Supply Chain. Information Systems Frontiers, 1-18.

Taylor, E. Z. (2006) The effect of incentives on knowledge sharing in computer-mediated communication: An experimental investigation. Journal of Information Systems, 20 (1), 103-116.

Wageman, R. & Baker, G. (1997) Incentives and cooperation: The joint effects of task and reward interdependence on group performance. Journal of Organizational Behavior, 18 (2), 139-158.

Wang, G. A., Liu, X., Wang, J., Zhang, M. & Fan, W. (2015) Examining micro-level knowledge sharing discussions in online communities. Information Systems Frontiers, 1-12.

Willem, A. & Buelens, M. (2009) Knowledge sharing in inter-unit cooperative episodes: the impact of organizational structure dimensions. International Journal of Information Management, 29 (2), 151-160.

Williams, L., Kessler, R. R., Cunningham, W. & Jeffries, R. (2000) Strengthening the case for pair programming. IEEE Software, 17 (4), 19-25.

Wood, R. E. (1986) Task complexity: definition of the construct. Organizational Behavior and Human Decision Processes, 37 (1), 60-82.

Wray, S. (2010) How pair programming really works. IEEE Software, 27 (1), 50-55.

Zacharis, N. Z. (2011) Measuring the Effects of Virtual Pair Programming in an Introductory Programming Java Course. IEEE Transactions on Education, 54 (1), 168-170.