Remote working in an Agile team

Date September 23, 2016

Author(s) TheAgileResearchNetwork
Helen Sharp, Leonor Barroca and Advait Deshpande (The Open University, UK)
Peggy Gregory and Katie Taylor (University of Central Lancashire, UK)

Agile Research Network
Bridging the gap between research and industry

DSDM®
CONSORTiUM

1 Summary

Co-location is the 'gold standard' of agile development, but distribution in one form or another is a necessary reality of today's development activity for many organisations. Distribution comes in three main flavours: distributed teams, dispersed individuals and hybrid teams. A *distributed team* consists of sub-teams in different locations, i.e. everyone in the sub-team is co-located with others in the sub-team, but the overall team is split between locations; this is commonly used in offshoring and global software development contexts. A *dispersed team* describes the situation when each individual in the team is located in a different place so each individual is on their own. A *hybrid team* is a combination of co-location and dispersed working, where there is a co-located team and one or two individuals who are located elsewhere and are on their own; these workers are referred to as remote workers. This white paper focuses on the experience of one organisation that utilises hybrid teams for its software development. It explores the challenges faced by this organisation in using remote working, investigates one team in detail and summarises relevant findings from existing literature that help to address these challenges.

The organisation in this white paper is Workplace Systems Ltd – a Milton Keynes-based company that specialises in workforce management software and employs primarily scrum-based agile practices. They are keen to support remote working but need to ensure that:

1. remote working is compatible with their agile working; and
2. remote workers' participation is effective throughout the sprints

Agile Research Network (ARN) explored the remote working situation with members of the Workplace Systems (WPS) software development team to answer these questions. Overall, the remote working arrangements work well and experience and literature show that remote working is compatible with agile software development, but there are also challenges concerned with making remote workers' participation effective. At WPS, these challenges fall into six overlapping themes: tooling and infrastructure, knowledge sharing and working together, (remote) pair programming, large group meetings, awareness (presence, activity, important issues), and social interaction and familiarity.

Findings indicate that making remote workers' participation effective requires effort by all team members to overcome challenges and to develop and maintain a shared understanding of the project between all members of the team. The following suggestions summarise those detailed in the white paper:

- balance the needs of the co-located team and remote worker to avoid detrimental effects on either, and involve the remote worker effectively in discussions
- have a robust communication and tooling infrastructure for online interactions
- use multiple communication modes to allow for individual differences across staff members
- consider pairing to facilitate knowledge sharing more effectively, build more trust, and integrate remote workers better into the teams
- use suitable tools to support pairing between co-located and remote team members
- use published pairing guidelines to support novice pairs
- use social discipline and high quality communication technology to engage remote workers in large group meetings, and set up a dedicated meeting room so that appropriate equipment may be consistently available

- provide collaborative platforms that support meaningful engagement of remote workers and co-located workers in meaningful tasks
- have both formal and informal channels of communication and distinguish important information from social chatter for all team members
- be aware of possible breakdowns in communication and information exchange, particularly with the remote worker

2 Introduction

Agile software development advocates the use of small co-located teams to help reduce misunderstanding, aid face-to-face communication and improve decision-making. The benefits of co-location have been known for many years, and in the early days of agile working, this co-location principle was stubbornly adhered to. Yet, the business reality for many organisations means the use of outsourced developers, third party external specialists or geographically scattered team members. Co-location cannot be supported in many cases and for various reasons, including cost, and this has led to remote working in one form or another.

One form of remote working is often referred to as distributed teams where sub-teams are situated in different locations, even different countries. Another is the dispersed team where most or all team members work alone in different locations. A third type of distribution is the hybrid team, with a combination of some co-located members and some working remotely. Organisations typically use remote working to retain talented staff or to gain access to specialist skills. In this case, integrating workers' capability and retaining expertise and experience are more important than simply cutting costs.

To produce this case study the Agile Research Network (ARN)[1] worked with a small software house, Workplace Systems Ltd (WPS). They specifically wanted to know whether remote working is compatible with agile working, and how best to make remote workers' participation as effective as possible throughout the sprints. The aim of this case study is to:

1. Seek others' published experiences of using remote working with agile software development.
2. Investigate how hybrid teams collaborate: the differences that exist between the co-located team members and the remote worker; and the challenges faced in making collaboration effective.
3. Identify potential mitigation strategies from research and practitioner literature to overcome the challenges identified.

1 The Agile Research Network (agileresearchnetwork.org) is funded by the DSDM Consortium Board, The Open University and University of Central Lancashire. The model operated by the network is that DSDM members propose the challenge they'd like to investigate, and then work closely with the research team to understand the causes and consequences of the challenge and to identify alternative ways of working gleaned from published literature.

3 The company and the context

WPS is based in Milton Keynes and specialises in workforce management software. A change in ownership in 2011 resulted in a transition from a waterfall based software development approach to an agile one. The company primarily uses Scrum, and full-time or part-time remote working has become part of their culture. For example, one developer works remotely for 2-3 days of the week and joins the co-located office team for the rest of the week. Of particular interest to this investigation were two teams that each had one full-time remote worker. Both full-time remote workers live over 100 miles away and come into the office at most once per release (eight weeks). The full-time remote workers are software developers, have been employed by the company for a long time and bring essential experience and subject matter expertise to the teams.

At the time of the case study (third quarter of 2015), WPS had 4 scrum teams, 2 of which each included one full-time remote worker, and one team also had a part-time remote worker. In addition, several development partners and clients/product owners work at a distance because they are based in Australia or North America. This means that the whole organisation has direct experience of remote working, although this case study focuses specifically on hybrid development teams.

WPS have an eight-week release cycle consisting of three 2-week sprints followed by a final sprint devoted to fixes and regression testing. Their key agile practices are backlog grooming and planning poker, daily stand-ups, sprint planning, show & tells, and retrospectives. Each of the main agile practices is conducted with full team participation including the remote worker, and there is regular interaction between individual team members. Data sharing and collaboration is enabled using a range of software tools (see Table 1) that were adopted prior to ARN's engagement. Regular face-to-face interactions with remote workers (referred to as "touchpoints" at WPS) take place every 8 weeks.

At the start of the sprint, the product backlog is agreed by the team. Planning poker is conducted separately using Hatjitsu. Scrum stand-ups take place daily for each team at 9:15 or 9:30. Workflow and workload tracking data is maintained using Jira, which is the main source of up-to-date information. During stand-ups the team also write items of work on tickets (post-it notes) and move them around on a physical board. The remote worker conveys information by voice call and one of the team members in the office moves their post-its around for him. Jira is updated after the stand-up. During the day, Jira is the main source of information and is updated continually. The physical board is then modified to mirror the data on Jira before the next stand-up. The physical scrum boards are used differently by different teams, and the use of different tools is embraced, including trialling new software as it comes along.

Retrospectives follow a similar combination of online and offline practices. Using a standard retrospective format designed to elicit 'what should we stop doing?', 'what should we start doing?' and 'what should we continue to do?' the office team uses post-it notes and the remote worker participates via video or voice call, or through an online support tool such as Appear.in. Information from the remote worker is added to the post-it notes by one of the co-located team members. Show & tells are not confined to one project team; all the project teams have a combined show & tell and discuss the features delivered in a release for the benefit of other teams. Remote members dial in to these meetings using WebEx.

Pair programming was not used at WPS at the time of the investigation, but teams were starting to explore this practice. Initial attempts at pairing with remote workers were cumbersome, but no specific tool support was used, and in particular although screen sharing was supported, joint editing was not. This last point is reflected in the group retrospective summarised in Table 2.

Table 1: A range of software tools had been adopted to support co-located and remote working

Tool	Purpose	What the team used it for
Jira	Project planning / management/ticket tracking	Used as a dashboard for work coordination activities – In the sprint view mode it provided a complete dashboard of the tickets to be completed, any blockers, and provided visibility of what the team members are working on. Also used as a repository for user stories and bug tracking
Hatjitsu	Online disposable poker rooms	Used for estimating and sizing user stories as part of the backlog grooming meetings. Fibonacci numbers are used for the planning poker sessions.
Bitbucket	Online Git repository	Used for code collaboration and source control - pull requests, code review, and code commits. The code review comments appear inline against the code and allow interactive conversations about the code to take place online.
WebEx	On-demand collaboration, online meeting, web conferencing and video conferencing	Used for show & tell sessions per release that included all the project teams in the company and also sales teams.
HipChat	Chat and Instant Messaging (IM) application.	Used for IM and group chats within the project team. Also used for voice calling mainly by the office team.
SourceTree	Desktop client for the Git code repository	Used for code commits and maintenance along with comments on the nature of change – user stories, tickets, and bug fixes
MySQL	Database	Used as a backend for the PHP-based online system. Test environment of the database used extensively for debugging code with dummy data
Outlook	Email / Calendar	Used primarily for setting up tasks, meetings in the Outlook calendar - the meeting reminders were synchronised and thus all team members were aware of the various scheduled project-related activities.

Skype for Business	Voice calls / Group voice calls / Video calls / Group video calls	Used in integration with Outlook, mainly for video calls during meetings – mainly daily standups and retrospectives.
Appear.in	Video calls / Group video calls	Used for videoconferencing. Could be integrated with an HipChat IM chats and often used to declare impromptu requests for group video calls on the project IM channels
PHPStorm	Integrated Development Environment (IDE)	Used for writing, debugging, and testing PHP code. Often used in conjunction with SourceTree and Bitbucket.

The investigation took several forms: spending time with remote and co-located members of the development teams to learn about their environment and ways of working; running an open focus group retrospective to which all co-located and remote developers and testers were invited; interviewing a range of developers and testers; and studying one hybrid team in detail.

The hybrid team studied in detail consisted of one scrum master, two developers (one of whom was a remote worker), two part-time testers, and a product owner shared with other teams. The product owner was not available as a participant in the study.

4 Successes and challenges

Since their change of ownership in 2011, WPS have been evolving their agile practices and implementing continuous improvement. During the focus group retrospective, there was a lot of support for remote working and several areas were identified as "doing well". None of the teams felt that remote working was getting in the way of productivity, and everyone was committed to supporting team members, whether co-located or remote. Throughout the interviews, members of staff agreed that there are benefits to remote working and there was a strong sense of willingness to continue improving the practice.

What are we doing well?

- Small group meetings and regular one-to-one conversations
- Using a good selection of communication tools
- Stand-ups, retrospectives, bug-no bug meetings
- Visibility of presence, i.e. knowing who is online and whether they are available
- Online tool use: Jira and HipChat and Bitbucket
- Keeping everyone up-to-date, e.g. using Jira
- Regular visits to co-located team by remote workers
- Screen sharing
- Developer / Tester interactions

Through this continuous improvement, and before ARN became involved, a number

of concerns around remote workers and their integration into the co-located teams had been identified, specifically:

- expert knowledge was not shared between the remote worker and co-located team mates
- participation of the remote worker in meetings was limited, and
- team building remained a "work in progress"

These were reflected in the investigation for this case study. The rest of this white paper focuses on the challenges and how they may be overcome.

Practices that work, practices that don't work, and areas for improvement

Practices that work

- Collaborative platforms such as Bitbucket and Jira
- Communication tools such as HipChat and appear.in
- Self-organisation and lack of hierarchy

Practices that don't work or struggle to be effective

- Remote worker's participation in meetings
- Lack of video capability – missing visual, verbal cues
- Ownership and integration of remote workers' work
- Making the regular face-to-face visits of the remote worker to the main office relevant

Areas for consideration/improvement

- Making ambient knowledge available to all workers
- Informal communication and its importance
- Differentiating useful and not-so-useful information in communication

Challenges identified through the investigation divide into six overlapping areas: tooling and infrastructure, knowledge sharing and working together, remote pair programming, large group meetings, awareness, and social interaction and familiarity. Pair programming is one way to address challenges around knowledge sharing and working together, although pairing was not mentioned at WPS in that context. Instead, pair programming was attempted because it's an agile practice. A summary of these challenge areas as expressed through the group retrospective is in Table 2.

Table 2. Themed entries from the group retrospective on remote working

Theme	Q2. What do we need to revise/rethink?	Q3. What should we be doing that we are not doing?	Q4. What are the blockers?
Tooling and infrastructure	• Conference facilities • Many communication tools can cause confusion • Connections /infrastructure	• Choose single tool communication • Improve electronic communications at MK • Better audio devices	• Technical issues • Poor infrastructure • Noisy office environment
Knowledge sharing and working together	• Planning work together • Way of sharing ideas on the whiteboard		• Fast response times for getting answers
Remote pair programming	• How/whether to attempt paired programming with remote workers • Pair programming is rather cumbersome with remote worker	• Pair programming	
Large group discussions	• Very large group meetings • Poor communications for remote workers with large group sessions		• Communication in big groups
Awareness	• Multiple interruptions (others not aware you are busy or on a call) • Getting out of step with what is expected		• Presence awareness; • Being aware of the background "vibe"

Social interaction and familiarity	• Social interactions • Team building	• "Office life" webcam • Invite remote workers to virtual coffee breaks • One communication tool but without losing out on office banter • Remote team building	• Interpersonal familiarity

From the team's perspective, the biggest challenge is that remote workers miss out on ambient knowledge and building technical know-how through peer engagement. This can lead to breakdowns in shared understanding and hence "getting out of step with what is expected" (Table 1).

One of the scrum masters raised the point that whatever steps are taken to integrate remote worker capability into the team, the impact on co-located workers should also be considered.

5 How to Mitigate the challenges

This section summarises published literature that is relevant to WPS's situation and extracts suggestions for how the challenges may be overcome in their circumstances. Four of the six challenge areas are discussed explicitly (knowledge sharing and working together, remote pair programming, large group meetings, and awareness), while challenges in the other two areas (tooling and infrastructure and social interaction and familiarity) are addressed coincidentally.

5.1 Knowledge sharing and working together

Physical separation of project members causes problems from strategic, individual, and cultural perspectives [Holmstrom et al, 2006]. These problems fall into three broad areas – *Communication*, *Coordination*, *and Control*. A shared understanding of the project and it goals are crucial to the success of an agile project, but without clear integration of the remote worker in the team's discussions this shared understanding is at risk [Abdullah et al, 2010].

There is a strong positive relationship between trust and knowledge sharing [Staples and Webster, 2008]. In addition, this relationship is stronger when task interdependence is low so trust is more critical in weak structural situations such as self-organising agile teams. Knowledge sharing is positively associated with team effectiveness. Hence, team 'imbalance' and 'hybrid structures' lead to weaker relationships between sharing and effectiveness.

Communication tools underpin any attempts to improve knowledge sharing in any form of remote working. Without robust infrastructure and appropriate support

tools, communication and hence knowledge sharing and working together will suffer. Hummel et al [2012] echo the findings from [Paasivaara et al. 2008, 2009] that multiple communication modes, including a variety of differing media, should be provided in order to bridge breakdowns of communication tools and to comply with differing individual preferences.

Conclusion:

- Without careful planning of knowledge sharing and support to work together, the remote worker and co-located team mates will perceive an imbalance and this can lead to problems with trust and hence productivity.
- Multiple communication modes provide flexibility [Paasivaara et al. 2008, 2009] and allow the teams to:
    - choose appropriate tool for the circumstance, e.g. bandwidth, type of information,
    - bridge breakdowns of communication tools, and
    - comply with differing individual preferences

Suggestions:

- Balance the needs of the co-located team and remote worker to avoid detrimental effects on either, and involve the remote worker effectively in discussions
- Have a robust communication and tooling infrastructure for online interactions
- Use multiple communication modes to allow for individual differences, but introduce some social discipline to avoid potential confusion about their use

5.2 Remote pair programming

Pairing requires effort and experience to be effective [Dyba et al, 2006], and is still a controversial practice. In the early days of agile working, full-time pairing was encouraged, but it is now more common for pairing to take place for particular tasks rather than for it to be implemented full-time. Knowledge transfer is one of the widely acknowledged benefits of pair programming [Begel and Nagappan, 2008] [Schindler, 2008], and hence would address one of the key challenges faced at WPS. Pair programming compared to solo programming increases knowledge of the code [Luck, 2004] and the software system [Vanhanen and Korpi, 2007]. It also increased knowledge about development tools, work practices, refactoring old code, new technologies, and programming languages [Vanhanen et al 2007].

There are several examples where remote pairing is successful when appropriate support tools are used [Dou et al, 2010] [Schummer and Lukosch, 2008]. Successful pairing of any kind is hard and benefits from practice. Remote pairing is most beneficial if the two participants know each other and have paired before (preferably with each other). However appropriate support tools are key if remote pairing is to be successful. WPS's experiments with pairing had used screen sharing facilities but not shared editing facilities, and the participants were not experienced with pairing at all. Purpose-built remote pairing tools exist, such as Saros [Schenk et al, 2014] [Prechelt, 2013], and these provide several features including collaborative real-time file editing, various sharing options, awareness information and whiteboard (http://www.saros-project.org).

Guidelines have been developed for pairing [Plonka, 2013], particularly when knowledge transfer is the main goal of the session [Plonka et al, 2015], and although not specifically tailored to remote working they are applicable in co-located and distributed settings.

Conclusions:

- Remote pairing can be practised successfully
- Pairing can help with issues of knowledge sharing and trust

Suggestions:

- Consider pairing to facilitate knowledge sharing more effectively, build more trust, and integrate remote workers better into the teams
- Use a tool that provides shared editing capabilities and support for collaborative working, preferably one that has been developed with remote pairing in mind
- Practice pairing between all colleagues so that experience and skill can be grown and shared
- Use the pairing guidelines referred to above to support novice pairs

5.3 Large Group meetings

Involving remote participants in a large group meeting is widely-acknowledged as having difficulties, and a quick search of the web will show that many different video conferencing facilities and research systems have been developed to support this activity. However these systems are not specifically designed for software development teams, and can be prohibitively expensive. There are also many guidelines available to support the conduct of large group meetings.

Specific issues in large group meetings at WPS are:

- Remote workers are not able to actively engage or participate in group meetings where only audio communication is used.
    - Not being in the room, their virtual presence is easy to overlook
    - They miss verbal, visual cues that are key to an ongoing, participatory conversation
- In a group meeting, those in the office are unsure of how to engage remote workers without interrupting the flow of conversation or are wary of doing it to the detriment of the natural conversation.
    - The scrum masters are aware of this issue and although they attempt to engage remote workers at every possible opportunity, virtual presence has its limitations

Agile teams are no strangers to the need for 'social discipline', i.e. to modify their behaviour in order to accommodate team activities. For instance, this is a key element of self-organisation; self-organising teams adopt various practices that Hoda et al [2012] refer to as Balancing Acts, i.e. teams need to handle many variables at once and reach equilibrium between them all. The application of this kind of social discipline by all team members in large group meetings, coupled with improved communication technology will improve everyone's experience of large group meetings.

Conclusions:

- Large group meetings pose difficulties for integrating contributions from all members of the development team.
- Current attempts to facilitate engagement are only moderately successful
- The application of appropriate social discipline, e.g. structured behaviour, will improve the experience of large group meetings

Suggestions:

- In large group meetings involving remote workers, 'social discipline' can help ensure remote workers contribute effectively. This would involve:
  - Ensuring that the remote participant's opinion is sought at crucial junctures
  - Agreeing on a way to interject or intervene in a conversation where video communication is not possible (say, using an alarm sound)
- Use high quality communication technology in large group meetings that allow:
  - Using two-way video communication where possible to provide visual cues to all of the participants
  - Ensuring that all of the participants in the group meeting are visible to each other and to the remote participants (assuming a video call)
- Set up a dedicated meeting room so that appropriate equipment may be consistently available

5.4 Awareness

The need for awareness was referred to in WPS in terms of three issues: presence (who is available to be contacted), activity (who is working on what), and important issues (technical activities or meeting arrangements). To maintain awareness, information needs to be exchanged between people. Ko et al [2007] identified three types of information needed in a co-located software development team: about activity, about artefacts, and about coworkers. They found that the most frequently-sought information includes awareness about artefacts and co-workers, and that developers often had to defer tasks because the only source of knowledge was unavailable coworker(s). These findings are echoed in WPS.

As part of our investigations we studied the information flows around co-located and remote workers and were able to identify imbalances being experienced within the team. Specifically, key differences were observed in terms of the reliance on virtual artefacts and collaborative tools, the use of informal and formal channels of communication, and the accessibility of information that impacts on effectiveness. If all workers were remote or all were co-located then this imbalance would be less likely to occur. It becomes risky when trust or confidence start to suffer (see knowledge sharing mentioned above).

Examples where remote workers missed out on information that they needed are:

- Forgetting to invite or inform a remote worker about ad-hoc meetings
- Forgetting to inform a remote worker about cancellations of meetings
- Remote workers being overlooked in conversations in group meetings
- Remote workers not getting informed about database codebase issues

- Remote workers being overlooked in the decision-making of software engineering processes, key practices, and workplace issues

Conclusions:

- Virtual artefacts and tools are crucial for remote workers to work effectively – they are the 'office'
- Collaborative tools such as Bitbucket that create a level-playing field and provide the same information and opportunity to participate are important for teams with remote workers
- informal channels (and text-only-based interaction) may lead to miscommunication

Suggestions:

- provide collaborative platforms that support meaningful engagement of remote workers and co-located workers in meaningful tasks
- differentiate important communications clearly from 'chat', by using a different channel or some other form of distinction
- have both formal and informal channels of communication and distinguish important information from social chatter for all team members
- to avoid the possibility of miscommunication and breakdowns, the communication media and the task need to be aligned
- be aware of possible breakdowns in communication and information exchange, particularly with the remote worker

6 The bottom line

Is remote working compatible with agile development?

Overall, yes. There are challenges in knowledge sharing and team building, with the danger that trust may break down between colleagues. Regular face-to-face touchpoints are vital in this situation. Collaborative platforms are the key to effective working, provided they offer meaningful engagement with others collaborating on a meaningful task. 'Meaningful engagement' means collaboration, i.e. working together on the same piece of work at the same time, and 'meaningful task' means something that directly progresses the main goal of software development such as producing code. For example, Bitbucket provides support for all workers to focus on code and to directly interact with it, hence allowing meaningful interaction on a meaningful task.

How can remote workers participate more effectively throughout the sprints?

A hybrid team consists of a remote worker and his co-located teammates, and hence collaboration in this context has similarities with a dispersed team and with a co-located team. This is no surprise. However whereas in a dispersed context and in a co-located context, all team members have equal opportunity for collaborative activities, in a hybrid team, opportunities are imbalanced. This creates potential

disadvantages as well as advantages for both co-located and remote workers. For the remote worker, potential disadvantages include being isolated and hence excluded from the knowledge network that his co-located teammates are embedded within. For the co-located workers, disadvantages are that informal communication and additional information exchanges may distract them from their main purpose. Potential advantages are the reverse of these: that remote workers can focus on the task in hand, and that co-located workers have a rich set of information available to them.

Making the remote workers' participation effective requires effort by all team members to overcome the challenges being faced, and to develop and maintain a shared understanding of the project by everyone.

References

Begel, A. and Nagappan, N. 2008. Pair programming: what's in it for me?. In Proceedings of the Second international symposium on Empirical software engineering and measurement

Dou, W., Wang, Y., Luo, S. (2010) 'Analysis and Design of Distributed Pair Programming System' Intelligent Information Management, 2, 487-497

Dyba, T., Arisholm, E., Sjøberg, D. I. K., Hannay, J. E., and Shull, F. (2007). Are two heads better than one? on the effectiveness of pair programming. IEEE Software, 24(6):12–15, November 2007.

Hoda, R., Noble, J. and Marshall, S. (2012) 'Developing a grounded theory to explain the practices of self-organizing Agile teams', Empirical Software Engineering, 609–639, DOI 10.1007/s10664-011-9161-0

Holmstrom, H.; Fitzgerald, B.; Agerfalk, P.; Conchuir, E. (2006) Agile Practices Reduce Distance in Global Software Development

Hummel, M., Rosenkranz, C. & Holten, R., 2012. The Role of Communication in Agile Systems Development: An Analysis of the State of the Art. Business and Information Systems Engineering, 5(5), pp.343–355

Ko, A., DeLine, R. and Venolia, G. (2007) 'Information Needs in Collocated Software Development Teams', in Proceedings of ICSE '07 (29th international conference on Software Engineering), pp 344-353

Luck, G. 2004. Subclassing XP: breaking its rules the right way," Agile Development Conference.

Paasivaara M, Durasiewicz S, Lassenius C (2008) Distributed agile development: using scrum in a large project. In: International conference on global software engineering. IEEE Press, New York, pp 87–95

Paasivaara M, Durasiewicz S, Lassenius C (2009) Using scrum in distributed agile development: a multiple case study. In: International conference on global software engineering. IEEE Press, New York, pp 195– 204

Plonka, L. (2013) Unpacking Collaboration in Pair Programming in Industrial Settings, PhD thesis, The Open University

Plonka, L., Sharp, H., van der Linden, J. and Dittrich, Y. (2015) 'Knowledge transfer in pair programming: an in-depth analysis' International Journal of Human-Computer

Studies, **73**, doi: http://dx.doi.org/10.1016/j.ijhcs.2014.09.001.

Prechelt, L. (2013) Agile Offsharing: Using Pair Work to Overcome Nearshoring Difficulties, in Proceedings CHASE 2013, Workshop at ICSE 2013, San Francisco

Schummer, T. and Lukosch, S. (2008) Supporting the Social Practices of Distributed Pair Programming, in R.O. Briggs et al. (Eds.): CRIWG 2008, LNCS 5411, pp. 83–98

Schenk, J., Prechelt, L. and Salinger, S. (2014) 'Distributed-pair programming can work well and is not just distributed pair-programming', in ICSE Proceedings Companion pp74-83, doi: 10.1145/2591062.2591188

Schindler, C. 2008 "Agile Software Development Methods and Practices in Austrian IT-Industry: Results of an Empirical Study," In International Conference on Computational Intelligence for Modelling Control & Automation

Staples, D.S. & Webster, J. (2008) 'Exploring the effects of trust, task interdependence and virtualness on knowledge sharing in teams'. Information Systems Journal 18, pp.617-140

Vanhanen, J. And Korpi, H. 2007. Experiences of Using Pair Programming in an Agile Project. In *Proceedings of the 40th Annual Hawaii International Conference on System Sciences.*

Vanhanen, J., Lassenius, J. and Mantyla, M. 2007. Issues and Tactics when Adopting Pair Programming: A Longitudinal Case Study. In *Proceedings of the International Conference on Software Engineering Advances.*