SCIENTIFIC PAPERS, UNIVERSITY OF LATVIA, 2011. Vol. 770
COMPUTER SCIENCE AND INFORMATION TECHNOLOGIES 15–26 P.

Towards a Contemporary Understanding of Motivation in Distributed Software Projects: Solution Proposal

Līva Šteinberga[1], Darja Šmite[1,2]
[1]University of Latvia, [2]Blekinge Institute of Technology
liva.steinberga@lu.lv, darja.smite@{lu.lv | bth.se}

Team motivation in software engineering is reported to have the largest impact on productivity, software quality and project overall success. Yet, as it is a soft factor and thus difficult to quantify, it usually takes a backseat. In fact, research on motivation and its interplay with different environmental factors in software engineering projects is scarce. Built on the basis of a recent systematic review on software engineers' motivation, in this paper, we emphasize the lessons learned from related studies regarding the factors determining team motivation. Further implications of distributed environment on motivation and the positive impact of agile practices are discussed and illustrated by examples from related studies. The contrasted project types (distributed and agile) suggest that blending agility into distribution might solve problems inherent in this type of environment. Finally, future research directions are suggested for project improvement through motivation.

Keywords: Motivation, Distributed Teams, Agile Teams, Distributed Agile Teams.

1 Introduction

Nowadays, in a quest for cheaper and faster high-quality software development many organizations have turned towards globally distributed software development [1]. Global software development (GSD) claims to enable the benefits of accessing larger resource pool, and reducing development costs in organizations that overcome the challenges of geographical distance [2]. As a result, nowadays, software projects more often involve geographically and temporally distributed and culturally diverse team members. These unique characteristics create significant challenges for communication, coordination and social networks [3] and being unable to overcome the new challenges a high number of global projects fails [4]. The emphasis in global projects is thus believed to be required for human aspects [5], which among other potential reasons of failure are often neglected since they are not easy definable and quantifiable.

Team motivation in software engineering is reported to have the largest impact on productivity, software quality and project overall success [6]. Nonetheless, motivation in global software teams has been relatively unexplored. To fill this gap we investigate what

motivates and what de-motivates software engineers, and how these influencing factors are manifested in global software projects. Furthermore, we consult related studies of agile teams from the past few decades, in which considerable effort has been devoted to exploring the social characteristics of software projects, as compared to research on traditional projects. This approach is built on team-centred philosophy and has inherent factors influencing motivation of software engineers. Inspired by related studies, we draw attention to potential areas of improvement with implications for practice and future research.

2 Background — Motivation in Software Teams

Motivation has been described as a source of performance improvement, which leads to productivity gains through exploitation of effective teamwork, in which team members act selflessly and contribute to the greater good [7]. In such teams, the team is more than the sum of its parts [7].

First experiments that demonstrated the influence of the team motivation on productivity were conducted by Elton Mayo in 1924, but later studies have confirmed the same results in several industries [8]. Mayo's experiments for the first time evidenced that workplaces are social environments, where people are motivated by many factors other than economic interest. He concluded that recognition, security, and sense of belonging were more important to productivity and morale or motivation, and a friendly relationship with the supervisor was very important in securing the loyalty and cooperation of the team [8].

In 1981, Barry Boehm reported that motivation has the single largest influence on quality and productivity than any other factor in software development, while in 1999, DeMarco and Lister's survey showed that the lack of motivation is one of the most frequently cited causes of software development project failure [6]. So far there are no other aspects claimed to have bigger influence on project effectiveness than developers' motivation, therefore it is important to understand what motivates software engineers to perform well and also what de-motivates them.

In 1978, Couger and Zawacki began a discussion whether software engineers form a distinct occupational group with similar needs and motives, and the discussion is still on-going. They surveyed the job perceptions of more than 6000 people from different professional areas and concluded that software engineers found their work less meaningful and rated their jobs less favourably than other professionals, their need to interact with others was negligible; they had very high growth needs and were concerned about learning new technology [9]. Beecham et al. in their systematic review on motivation confirmed that a little more than a half of empirical studies (54%) conducted in this field until year 2006 regarded software engineers as a distinct occupational group, while 24% of the studies denied this categorization [6].

There are several classical motivation theories, which can be applied in order to explain what motivates software engineers (such as Abraham Maslow's Hierarchy of needs, Clayton Alderfer's ERG theory, Frederick Herzberg's Motivation-hygiene theory, David McClelland's achievement motivation theory and others). One of the contemporary aspects of understanding motivation prescribes a division of the influencing factors into intrinsic and extrinsic [8, 9]. Intrinsic factors address the work

itself and the goals and aspirations of the individual, such as achievement, possibility for growth, social relationships, security, etc. Extrinsic factors are concerned with the surrounding environment brought by the organization to the individual, such as praise, communication, office space, responsibility, money etc.

Beecham et al. in their report on the motivation of software engineers have gathered motivators and de-motivators for software engineers from 92 empirical studies [6]. They have identified some of those factors as inherent for software engineering. In total, 29 motivators and 15 de-motivators have been collected; those manifested in distributed and agile projects are listed in tables 1 and 2.

It is worth noting that authors of the systematic review have also captured the external signs associated with motivated and de-motivated software engineers, such as retention, productivity, project delivery time, budgets, absenteeism and project success [6]. While global projects are often suffering from similar negative impacts [4], our investigation in this paper is driven by the question whether a lack of motivation and presence of de-motivators can be the cause of failure.

3 Research Overview

The aim of our investigation is to understand the implications of motivational research regarding distributed software teams. More specifically, we consult research on agile software development projects for learning from team-centred approaches to increase the productivity and success ratio of distributed software projects. The research is thus driven by the following research questions:

RQ1: How the known motivators and de-motivators of software engineers are manifested in distributed software development projects?

RQ2: What can we learn from agile projects to eliminate de-motivators and enable motivators in distributed software development projects?

To address these research questions, we base our investigation on the results of the systematic review performed by Beecham et al. [6]. Manifestation of the identified motivators and de-motivators is explored through relating research findings from the field of global software engineering. It is worth noting that no extensive literature review is performed. Our goal in this paper is to exemplify manifestation of motivating and de-motivating factors. Further, we highlight the main areas of concern and conjure these as problems inherent in the nature of the distributed environment. We explore the factors related to motivation enabled by inherent characteristics of agile methods, and propose the potential areas of improvement for distributed projects inspired by agile approaches.

4 Findings

4.1 Motivation in Distributed Projects

To the best of our knowledge, no research dedicated to motivation can be found in the area of commercial distributed or global software development (with an exception of Open Source development projects, which are also distributed and global, but not

commercial). By distributed projects, we mean such software development projects, in which tasks (be it a phase or a development task) are split among several geographically distant locations. These locations can be represented by sites of the same company (offshore insourcing) or different companies (offshore outsourcing). The findings from related research in the area of global software engineering point out evidences of various de-motivating factors caused by distribution. Our observations suggest that some of these factors are inherent in the very nature of distributed projects, and thus de-motivation among software engineers in such projects may be manifested more often than in similar projects with co-located members. Hereby, we discuss motivators and de-motivators that have a special meaning in distributed projects in a concise way with examples of specific manifestation in distributed projects supported by the references to related research.

4.1.1 Manifestation of motivators in distributed projects

We have found that motivators such as Change, Benefit and Problem solving, Science, Experiment, Identification with the task, Career path, Variety of work, Recognition for work done, Development needs addressed, Making contribution or task significance, Rewards and incentives, Feedback, Job security, Good work life balance, Appropriate working conditions, Working in a successful company, and Sufficient resources are more generic and are thus not affected by distribution.

Challenge / Intrinsic motivator: Enabled. Software engineering is regarded as a challenging profession [6]. In its turn, globally distributed development is recognized to be considerably more challenging than even the most complex project managed entirely in-house [4, 10]. Thus, we can claim that challenges are triggered by distribution.

Team work / Intrinsic motivator: Challenged. Developers are recognized to be motivated by working in a team of other professionals rather than alone [6]. Distributed teams on the contrary tend to be divided into sub-teams by location and often experience cross-site competition instead of collaboration [10], which is regarded as "us and them" attitude [11]. Also, distribution makes it difficult to apply mutual adjustments and supportive behaviour, which are commonly used in effective teamwork and especially important for dealing with complex tasks [12]. Thus, we can conclude that although teamwork can be established inside the co-located sub-teams, motivation from the cross-site teamwork perspective is likely to be challenged.

Development practices / Intrinsic motivator: Challenged. While developers can feel united around the use of a certain methodology or development practice, for example, object-oriented, agile, or prototyping practices [6], cultural and also organizational differences associated with distributed projects often lead to discrepancies in the work habits [13] and sometimes to enforced use of the other site's methodology, which might de-motivate the developers.

Technically challenging work / Intrinsic, general motivator: Challenged. Work is found to be motivating if it is not mundane and is technically challenging [6]. Unfortunately, companies often practice outsourcing of routine tasks and keeping the more interesting work for themselves. This could be one of the reasons why offshoring, in particular to India, is often associated with rapid turnover of employees.

Autonomy / Intrinsic, general motivator: Challenged. Freedom to carry out tasks letting roles evolve is regarded as a motivator for software engineers [6]. In distributed

projects, remote sites are often closely supervised by their headquarters [14] and even regarding technical decisions autonomy may be limited [15]. This is one of important factors to be considered in distributed projects, in which site inequity (discussed later) can further complicate the morale of the remote sites.

Empowerment or responsibility / Intrinsic, general motivator: Challenged. Responsibility for the task is recognized as a motivator for software engineers to perform better [6]. However, remote sites, e.g., in offshore development relationships, are more often involved only in partial activities (only coding, only testing or only maintenance activities [16]) and do not receive the responsibility for the whole development project. While the onshore part of the team is often working on the functionality holding the ownership and thus, in addition, creating an iniquity between the parts of the team.

Trust, respect or equity / Intrinsic, general motivator: Challenged. Trusting and respecting other people, treating and managing them fairly has been identified as one of the key motivators for software developers [6]. Meanwhile, there is a number of studies reporting lack of trust as one of the major problems in globally distributed software development teams (such as [14], [17], [18]). Key factors that cause the lack of trust are poor socialization and socio-cultural fit, increased monitoring, inconsistency and disparities in work practices, reduction of communication, lack of face-to-face meetings, poor language skills, lack of conflict handling, lack of cognitive-based trust [14], and others. Those factors also relate to other motivators identified by Beecham et al. [6]. For instance, autonomy is another key motivator, hindered by increased monitoring.

Employee participation / Intrinsic, general motivator: Challenged. This motivator can be described by involvement in the company and working with others [6]. Herbsleb and Mockus have found that distributed work items appear to take about two and a half times as long to complete as similar items where all the work is co-located [3]. Aiming to find the reasons behind such a gap in productivity, they had observed that communication, coordination and social networks in distributed teams may differ from single-site counterparts in a way that it requires more people to participate thus introducing a delay [3].

Sense of belonging / Extrinsic motivator: Challenged. Sense of belonging to the team and supportive relationships between team members is regarded as an extrinsic motivator. In distributed teams, however, this remains one of the greatest challenges. According to Herbsleb and Mockus, people at different sites are less likely to perceive themselves as a part of the team [3]. This is illustrated by conflicting work styles adopted in the remote locations and by the perception that distant colleagues are less likely to help out when workloads are especially heavy. They thus conclude that cross-site relationships compared to same-site relationships are less oriented towards mutual benefit [3].

4.1.2 Manifestation of de-motivators in distributed projects

We have found that de-motivators such as Risk, Stress, Unfair reward system, Uncompetitive or poor pay and unpaid overtime, Unrealistic goals and phoney deadlines, and Poor management are more generic and thus are not directly attributed to distribution.

Inequity: Triggered. Recognition based on management intuition or personal preference is regarded as de-motivating in software engineering [6]. Inequity in distributed

projects is expressed in various ways. It can be related to the already mentioned "us and them" attitude [11]. It is also manifested in various ways through unequal rules and requirements on remote sites. For example, members of highly distributed teams that work across multiple time-zones are often required to shift their working hours [19] creating dissatisfaction in long term. In addition, the time shifting is often tolerated by the offshore sites (late shifts [19]), while the work hours for the headquarter site or onsite teams often remain unchanged.

Interesting work going to other parties: Triggered. Beecham et al. [6] also indicate that software engineers may be de-motivated by being separated from the other team members and by outsourcing the most interesting work to other sites [6]. This is the case in companies that are downsizing their development budgets and sending the work to outsourcing suppliers. In such projects (we call them onsite), personnel will be de-motivated. In other projects, where boring work is sent offshore to free up onshore employees for new projects [20], the offshore site will become de-motivated.

Lack of promotion opportunities/ stagnation/ career plateau/ boring work/ poor job fit: Triggered. In line with the above, it appears that some companies practice offshoring through a relocation of more stable work, such as software maintenance and bug fixing, in order to free up onsite resources for the new development and thus more interesting work [20]. This is another example of inequity, boring work and stagnation, and also a manifestation of poor motivation in offshore sites, which could be an explanation for the sequential rapid turnover of employees in the sites that receive boring work.

Poor communication: Triggered. Feedback deficiency from colleagues and software users is de-motivating for software engineers [6], but one of the most demoralizing bad practices is the loss of direct contact to all levels of management for each staff member [21], which is very likely to occur in distributed setting due to geographic and often temporal distance.

Bad relationship with users and colleagues: Triggered. Software engineering is a complex field and the most problems can nowadays be effectively solved only by effective teamwork performed by engineers with a help from software users. Therefore, it is very important to form a cohesive team and maintain friendly and supporting relationships with users. Thus, bad relationships with users and colleagues can hinder one from motivation to work effectively [6]. In a distributed setting, it can be complicated to establish good and close relationships with remote colleagues and users because of the mentioned geographic and temporal distances, because it is more and more common that developers from different sites never meet [14].

Poor working environment: Triggered. Wrong staffing levels and unstable, insecure working environment lacking investment and resources is likely to de-motivate employees [6]. Being physically separated from the team can negatively affect software engineer's performance [6]. This factor is inherent for staff from remote sites in distributed software development.

Poor cultural fit/ stereotyping/ role ambiguity: Triggered. Software engineers share national, occupational and organizational cultures. If one does not feel like belonging to the culture, it could be disturbing and serve as a de-motivating factor. There is evidence about reciprocal stereotyping of software engineers by managers as a de-motivator [22] and role ambiguity, which involves uncertainty about role responsibilities,

expectations, or tasks [23]. These factors are likely to occur in a distributed setting due to socio-cultural distances that are associated with these projects [19].

Producing poor quality software: Triggered. It is observed that poor outcome of the work de-motivates software engineers [6]. Meanwhile, a transfer of software work from one site to another is recognized to have a negative impact on both productivity of software engineers and resulting quality of the outcome. It happens due to significant challenges of learning to handle the already developed software being unknown [20].

Lack of influence or no involvement in decision-making: Triggered. Distribution among the offshore sites and the headquarters, and especially temporal dislocation, leads to the feeling of lagging behind [19]. This is likely to have an impact on teamwork and all associated activities, such is decision-making. Thus, offshore sites are naturally left with a limited influence on the project.

4.2 Motivation in Agile Projects

Agile movement has emphasized the importance of human aspects in software development by manifesting one of its values "Individuals and interactions over processes and tools" and its principle "Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done". Melnik and Maurer have described agile methods as human-centric bodies of practices and guidelines for building software in unpredictable and highly-volatile environments [24].

So far, the most popular agile practices are Extreme Programming (XP) practices, which among other benefits originally hold values of daily face-to-face communication between all project team members and have a regular feedback on the work done. There are several evidences that XP environment gives a grater job satisfaction and increases code quality [7]. Since those two signs are closely connected with external signs of the motivation of software engineers mentioned beforehand in section 2, it can be derived that XP environment is highly motivational. Besides, there are studies, which confirm that the more practices are incorporated in the development process, the higher is the positive emotions experienced by the developers [25]. Further, we discuss how different practices address motivation of software engineers.

Iterations and small releases. Iterations and small releases enable software engineers to receive early and frequent feedback and recognition for well-done job from all the stakeholders. It increases the self-esteem of the software engineers and the level of trust between them and the customer [8].

Enables: Feedback, Recognition, Trust/ respect.

Helps to avoid: Unrealistic goals/ phoney deadlines.

Simple design, continuous testing and continuous integration. Continuous software development process provides early feedback which prevents from delays and integration problems later thus avoiding employee dissatisfaction.

Enables: Feedback [25].

Helps to avoid: Unrealistic goals/ phoney deadlines.

Regular face-to-face meetings. Daily stand-up or weekly face-to-face meetings address the need for developers to feel they are making progress, which is common in groups of professionals with a high need for personal growth and development [26]. Besides, regular meetings ease and speed communication and collaboration [27]. This

practise and small chunks of action assigned to individuals support sense of belonging to the team and maintain team awareness of member activity [26].

Enables: Sense of belonging, Good management, Identify with the task, Empowerment / responsibility, Employee participation, Team work.

Helps to avoid: Poor communication, Poor management, Producing poor quality software [27], Unrealistic goals/ phoney deadlines [7].

Pairing. Pairing motivates software engineers because it addresses the need for learning, autonomy and social activity [7], and it also provides feedback [25]. However, pairing can be de-motivating if pairs have personality conflicts [7] or continuous pairing is too intense [26]. Individuals can become bored with regular rhythm of development and become dependent on working in pairs thus losing their confidence to work alone [26].

Enables: Autonomy, Feedback, Development needs addressed, Bad relationships with colleagues.

Helps to avoid: Poor communication, Producing poor quality code.

Self-organizing team. Software engineers in XP environment are working in self-organising teams which means that individuals are given an autonomy or freedom to carry out tasks themselves, allowing roles to evolve. As reported by Beecham et al., this trait of XP teams is one of the main motivators for software engineers in general [6]. It also discourages from working in an insular myopic fashion that de-motivates other members of the team [26]. In self-organizing teams, a principled leadership is applied which allows software engineers to make the decisions that affect them and participate in personal goal setting which is found to increase the job satisfaction [6]. Agile developers have been found to be motivated by working with people who possess very good communication skills [26] and who are highly competent [8]. On the other hand, the lack of these traits has been found de-motivating. De-motivating factor in XP environment is that individual inputs into the project might be subsumed by the whole team, thus impacting promotion opportunities [26].

Enables: Autonomy, Empowerment/ responsibility, Career path, Lack of promotion opportunities.

Helps to avoid: Lack of influence/ not involved in decision making/ no voice.

5 Discussion — Blending Agility with Distribution

Evidence from empirical studies in the area of global software engineering suggests that this type of environment is full of challenges that trigger de-motivators and hinder motivators for software engineers. In the following tables, we present a summary of the findings from exploring distributed projects and agile projects.

Table 1

Motivators (based on [6]) Manifested in Distributed and Agile Projects

Motivators	**Challenged by distribution**	**Triggered by distribution**	**Solved in Agile**
INTRINSIC MOTIVATIORS			
Challenge		✓	
Team work	✓		✓
Development practices	✓		✓
General motivators			
Identify with the task	✓		✓
Variety of work	✓		✓
Recognition for work done	✓		✓
Technically challenging work	✓		
Autonomy	✓		✓
Empowerment / responsibility	✓		✓
Trust / respect / equity	✓		✓
Employee participation	✓		✓
EXTRINSIC MOTIVATORS			
Sense of belonging	✓		✓
Feedback	✓		✓

Table 2

De-motivators (based on [6]) Manifested in Distributed and Agile Projects

De-motivators	**Triggered by distribution**	**Solved in Agile**
Inequity	✓	
Interesting work going to other parties	✓	
Lack of promotion opportunities/ stagnation/ career plateau/ boring work/ poor job fit	✓	
Poor communication	✓	✓
Bad relationship with users and colleagues	✓	
Poor working environment	✓	
Poor cultural fit/ stereotyping/ role ambiguity	✓	
Lack of influence/ not involved in decision making/ no voice	✓	✓

Although our study does not differentiate and systematically evaluate diverse implementations of global projects, it raises an important question about the motivators hindered and de-motivators triggered by geographic and temporal distribution. We therefore conjure that a contemporary understanding of motivation in software projects is required as more and more companies become distributed and utilize resources from all around the world. Another contemporary aspect that is not well understood to date is related to cultural diversity. In particular, one may wonder whether motivators and de-motivators are equally strong across different countries. Thus, further empirical studies into the phenomenon of motivation in distributed projects are important.

Having said that distributed software projects trigger a lot of de-motivating factors, in this paper, we also aim at understanding how to solve these environmental flaws. One source of inspiration for us has been agile projects. Even though agility and distribution may seem as an incompatible mix [28], the team-centred approaches suggested by agile methods are already evidenced in distributed environment [28, 29]. While previous studies have solely focused on positive impacts of agile methods on team communication and coordination experiences, we emphasize the necessity to address motivation as the prime object of study.

In this study, we have explored the current understanding of motivation in traditional software teams based on related studies to date. However, one of our goals is to emphasize that it is fair to assume that environment including software projects changes over time and existing studies could provide a limited explanatory power to understand the motivation in, e.g., projects involving developers from all around the world. In order to be successful, organizations should strive to understand how to motivate their employees in actual project settings and keep an eye on the rapidly changing factors influencing the motivation.

Our future research includes empirical evidences of the motivation of software engineers in different types of distributed projects where software engineers with different cultural backgrounds are involved. We are also planning to investigate what we can learn from open-source projects in order to raise the team motivation in GSD.

6 Conclusions

Our observations suggest that many motivating factors are challenged and de-motivating factors are inherent in the very nature of distributed projects, and thus we argue that de-motivation among software engineers in such projects may be manifested more often than in similar projects with co-located members. We trace the negative effect on motivation to geographic and temporal distance and cultural diversity, and claim that without explicit concern of motivation, global projects will ultimately fall into the category of dissatisfactory for those involved.

Ramasubbu and Balan states that productivity and quality can be modelled as a function of personnel-related factors and software methodology-related factors [29]. In our investigation, we confirm that software methodology and also project environment matters. We study two distinct approaches – distributed projects and agile projects –, which are analysed in the aspect of motivational studies. We suggest that the equation

shall also include factors that influence motivators and de-motivators for software engineers.

Further, we discuss the potential of agile approaches to solve the problems of de-motivation in distributed projects and propose future research to focus on evaluating the impact of agility on motivation. Future work also involves empirical investigation of motivation in different types of culturally diverse distributed projects and investigation of what we can learn from open-source projects in order to raise team motivation in GSD.

Acknowledgements

This work has been supported by European Social Fund project
No. 2009/0216/1DP/1.1.1.2.0/09/APIA/VIAA/044.

References

1. D. Šmite, C. Wohlin, T. Gorschek, and F. Robert, "Empirical evidence in global software engineering: a systematic review," Empirical Software Engineering, vol. 15, no. 1, p. 91, 2010.
2. L. Layman, L. Williams, D. Damian, and H. Bures, "Essential communication practices for Extreme Programming in a global software development team," Information and Software Technology, vol. 48, no. 9, p. 781, 2006.
3. J. D. Herbsleb, "An empirical study of speed and communication in globally distributed software development," IEEE Transactions on Software Engineering, vol. 29, no. 6, p. 481, 2003.
4. C. Ebert, "Optimizing supplier management in global software engineering," in Proceedings of the International Conference on Global Software Engineering, 2007.
5. T. Hall, H. Sharp, S. Beecham, N. Baddoo, and H. Robinson, "What do we know about developer motivation?," Software, IEEE, vol. 25, no. 4, p. 92, 2008.
6. S. Beecham, N. Baddoo, T. Hall, H. Robinson, and H. Sharp, "Motivation in software engineering: a systematic literature review," Information and Software Technology, vol. 50, no. 9-10, p. 860, 2008.
7. A. Law and C. Raylene, "Effects of agile practices on social factors," in Proceedings of the 2005 workshop on Human and Social Factors of Software Engineering - HSSE 05, 2005, p. 1.
8. G. Asproni, "Motivation, teamwork, and agile development," Agile Times, IV (1), p. 8–15, 2004.
9. B. L. Mak and H. Sockel, "A confirmatory factor analysis of IS employee motivation and retention," Information & Management, vol. 38, no. 5, pp. 265-276, 2001.
10. A. Piri, T. Niinimäki, and C. Lassenius, "Descriptive analysis of fear and distrust in early phases of GSD projects," presented at IEEE International Conference on Global Software Engineering, 2009.
11. D. Šmite and C. Gencel, "Why a CMMI level 5 company fails to meet the deadlines?," Product-Focused Software Process Improvement, p. 87–95, 2009.
12. D. Šmite, N. Moe, and R. Torkar, "Pitfalls in remote team coordination: lessons learned from a case study," Product-Focused Software Process Improvement, p. 345–359, 2008.
13. J. S. Olson and G. M. Olson, "Culture surprises in remote software development teams," Queue, vol. 1, no. 9, p. 52–59, 2003.
14. N. B. Moe and D. Šmite, "Understanding a lack of trust in global software teams: a multiple-case study," Software Process Improvement and Practice, vol. 13, no. 3, p. 217, 2008.
15. R. Prikladnicki, J. L. N. Audy, and F. Shull, "Patterns in effective distributed software development," Software, IEEE, vol. 27, no. 2, p. 12–15, 2010.
16. S. Islam, M. M. A. Joarder, and S. H. Houmb, "Goal and risk factors in offshore outsourced software development from vendor's viewpoint," in Proceedings of IEEE International Conference on Global Software Engineering, 2009, p. 347–352.
17. A. Piri, T. Niinimäki, and C. Lassenius, "Fear and distrust in global software engineering projects," Journal of Software Maintenance and Evolution: Research and Practice, 2010.
18. S. Jalali, C. Gencel, and D. Šmite, "Trust dynamics in global software engineering," in Proceedings of

the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement - ESEM, Bolzano-Bozen, Italy, 2010.
19. H. Holmstrom, E. O. Conchuir, P. J. Ågerfalk, and B. Fitzgerald, "Global software development challenges: a case study on temporal, geographical and socio-cultural distance," IEEE International Conference on Global Software Engineering, 2006.
20. D. Šmite and C. Wohlin, "Software product transfers: lessons learned from a case study," IEEE International Conference on Global Software Engineering, 2010.
21. S. A. Frangos, "Motivated humans for reliable software products," Microprocessors and Microsystems, vol. 21, no. 10, pp. 605-610, Apr. 1998.
22. S. Ramachandran and S. V. Rao, "An effort towards identifying occupational culture among information systems professionals," in Proceedings of the 2006 ACM SIGMIS CPR, Claremont, California, USA, 2006, p. 198.
23. M. F. Reid, M. W. Allen, C. K. Riemenschneider, and D. J. Armstrong, "Affective commitment in the public sector," in Proceedings of the 2006 ACM SIGMIS CPR, Claremont, California, USA, 2006, p. 321.
24. G. Melnik and F. Maurer, "Comparative analysis of job satisfaction in agile and non-agile software development teams," Extreme Programming and Agile Processes in Software Engineering, p. 32–42, 2006.
25. S. L. Syed-Abdullah, J. Karn, M. Holcombe, T. Cowling, and G. Marian, "The positive affect of the XP methodology," in Extreme Programming and Agile Processes in Software Engineering, Sheffield, UK, 2005.
26. S. Beecham, H. Sharp, B. Nathan, T. Hall, and H. Robinson, "Does the XP environment meet the motivational needs of the software developer? An empirical study," in AGILE 2007, 2007, p. 37.
27. E. Whitworth and R. Biddle, "The social nature of agile teams," in AGILE 2007, 2007, p. 26–36.
28. D. Šmite, N. B. Moe, and P. J. Ågerfalk, "Agility across time and space: making agile distributed development a success," Springer, Heidelberg, Germany, 2010.
29. N. Ramasubbu and R. K. Balan, "Globally distributed software development project performance: an empirical analysis," in Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on the foundations of software engineering - ESEC-FSE 07, 2007, p. 125.