PREPRINT[1]

Team Performance in Software Development: Research versus Current Advice

Torgeir Dingsøyr, Tor Erlend Fægri, Tore Dybå, Børge Haugset, Yngve Lindsjørn

In 1993, Walz, Elam and Curtis [1] stated that research on "how teams actually go about" making requirement determinations and design decisions can provide valuable insights for improving the quality and productivity. Since then, more and more tasks are performed by software, and software development is increasingly undertaken by teams. This article reviews studies on factors that influence team performance for co-located teams, and make propositions on five factors based on solid scientific studies. These propositions are relevant for practitioners in software teams, project managers, managers and researchers interested in software development. We ask how these propositions compare with current advice on software development as condensed in the *Agile manifesto.*

Team performance has been studied in many disciplines, from management science [2] and organizational psychology [3] to information systems [S12]. From these disciplines we find thorough review articles, providing insight on key findings such as the importance of establishing a common "mental model" within a team. Many of the studies conducted in other disciplines have been performed on software development teams [4], because such teams are examples of knowledge work in an innovative setting. What can the software engineering discipline learn from these studies? We investigate *what main factors influence the performance of software development teams.*

Reference: Dingsøyr, T., Fægri, T. E., Dybå, T., Haugset, B., and Lindsjørn, Y., "Team Performance in Software Development: Research Results versus Agile Principles," *IEEE Software*, vol. 33, pp. 106-110, 2016. DOI: 10.1109/MS.2016.100

Teamwork and team performance

A team is a small number of people with "complementary skills who are committed to a common purpose, set of performance goals, and approach for which they hold themselves mutually accountable" [2]. A team further has common tasks, interact socially, and experience the same organizational context [5].

Based on a systematic review of empirical studies of factors that influence performance of software development teams (studies in Table 1, see online appendix), we have constructed the software team performance model as in Figure 1. Software team performance is particularly influenced by: team coordination, goal orientation, team cohesion, shared mental models and team learning. We explain the importance and underlying findings related to each of these factors:

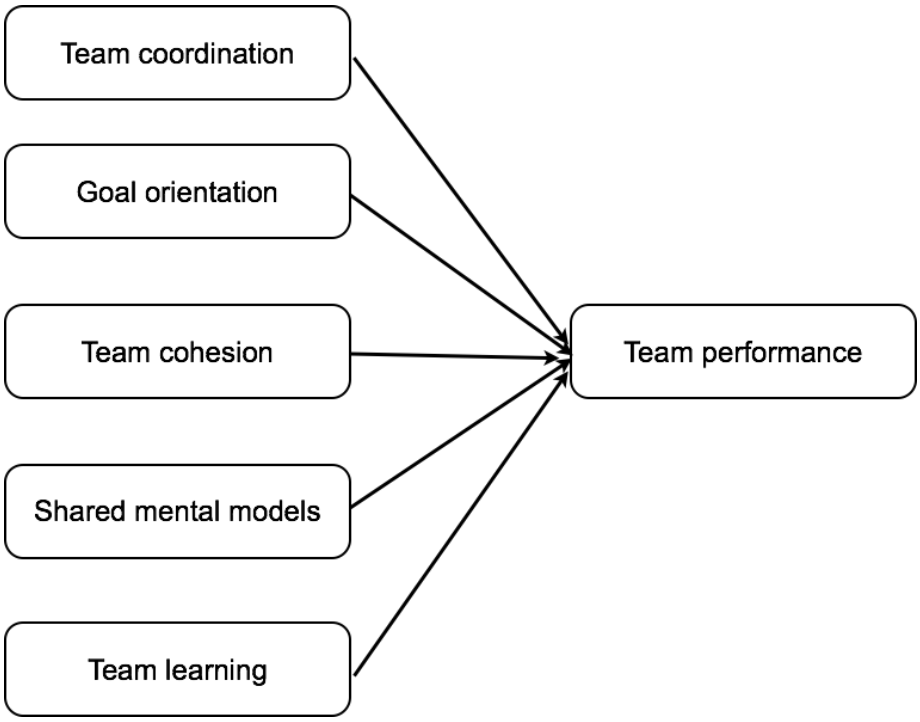

Figure 1: Software team performance model.

Team coordination

Software development involves ill-defined, ambiguous and non-routine work, which is incompatible with detailed, up-front planning [S13]. Coordinating team members is important for project success [S19], and that the team is able to efficiently adapt to changes in technology and business needs is important to achieve high software quality [S15]. Coordination is "managing dependencies between activities" [6]. Such dependencies include

shared resources, task assignments and task/subtask relationships. Team coordination involves creating a common understanding within the team with respect to these dependencies. Synchronizing and harmonizing individual contributions involves establishing mechanisms for coordination, like common work breakdown structures, schedules, budgets and deliverables [S19]. This can involve coordinating work processes, establishing internal procedures and mechanisms for feedback, and coordinating team member contributions.

Team coordination requires interaction amongst team members. Higher levels of quality and quantity of team interaction is positively related to project success [S18]. Coordination through plans is used to assign tasks, allocate physical and economic resources, manage resource dependencies, and integrate outputs. Tools used for administrative coordination include budgets, staffing tables, critical path analysis, milestones, inspections, and review meetings. This type of coordination has a positive impact on team performance [S4]. For software teams, development methods prescribe many mechanisms for coordination, like how planning is to be conducted, resources allocated and tasks distributed. The extent of which development methods are used is positively related to team performance [S13, S4]. Another practice for coordinating work in software teams, the use of coding standards, is also positively related to team performance [S9]. Managing dependencies in software development will typically involve providing feedback, as dependencies cannot always be identified prior to engaging in work tasks. Frequent feedback on work products will impact the performance of development teams [S12].

There are a number of coordination mechanisms that have been found to influence team performance, from administrative coordination, use of systems development methods, coding standards to feedback on work tasks:

Proposition 1: Team coordination practices have a strong positive impact on team performance.

Goal orientation

A team has a common purpose and set of performance goals. Goal or achievement orientation as well as the ability of the team to define clear goals are factors that influence team performance.

Goal orientation is important for team performance, both directly as in having clear goals, and indirectly as in having a goal-oriented team leader. Clear goals and milestones should be

established as a part of making effective plans and procedures. “When performing a task as complex as software development, team members must stay on track and achieve specific intermediate goals in order to increase their team’s performance” [S1]. Clearly articulated goals adopted by the team members have a positive influence on team performance [S10]. For a team leader, goal orientation is an action style – a propensity to act. Goal orientation can be regarded as a personality trait, denoting a person’s inclination to set goals, pursue and achieve them [S13]. Highly goal-oriented persons develop long-range and clear goals. They are persistent in pursuing them, especially when difficulties occur. The goal orientation of a team leader contributes substantially not only to individual performance but also to team performance, such as in the likelihood to keep schedule and budget [S13].

The team leadership is focused on influencing team members to work according to project goals. Software team leaders should understand the dynamics of the software development process, and concentrate on leading through monitoring and evaluating the behaviour of team members. An alternative approach is to lead through evaluating the outcome that is produced by team members. Team performance is better if the outcome is evaluated by the whole team than if a team leader is evaluating [S12]. In addition, having a team champion interpreting and influencing the team’s environment is positively related to team performance as seen by external stakeholders: “…reinforcing the importance of maintaining good relations upward in the organization and managing team progress to higher organizational levels” [S1].

Some development methods such as agile methods argue for teams to self-manage. However, that self-management increases team performance received only limited support [S12]. Many development projects have a project manager who is responsible for overall plans and external communication. So, whether there is a defined team leader or the leadership is distributed in a self-managing team, the goal-orientation of the leadership impacts team performance:

Proposition 2: Goal-oriented team leadership has a strong positive impact on team performance.

Team cohesion

A topic that has been widely studied in the teamwork literature is team cohesion. Team cohesion is "the tendency for a group to stick together and remain united in the pursuit of its

goals and objectives" [7]. Cohesion primarily involves commitment to team tasks, but also interpersonal attraction of team members and group pride [8].

How important is cohesiveness for software teams? A study of the influence of team cohesiveness, team experience and team capability on performance, found that cohesiveness was the dominating factor [S8]. Likewise, team cohesion was the most important team quality factor in a study linking teamwork quality to team performance [S19]. Some agile development methods like extreme programming emphasize collective code ownership. If developers are able to edit code developed by others, this is a form of intensive collaboration. Such intensive collaboration is unlikely unless the team has a high cohesion. More collective code ownership leads to fewer program bugs, which is a measure of software product quality [S9]. That collective code ownership leads to product quality then suggests that team cohesion has an indirect positive impact on product quality.

The opposite of cohesive teams are teams with conflicts. Conflicts over priorities of tasks or how to organize work can negatively influence the performance of a team. Conflicts can have strong negative impact on both software product success and customer satisfaction [S2]. However, there are different types of conflicts. Relationship conflicts have a negative influence on performance, while task conflicts have a positive influence on performance [S7]. Why do conflicts have a positive impact on performance? It could be because task conflicts make a team see new possibilities, avoiding "groupthink". Conflicts are probably inevitable in teamwork. The question is then how to manage conflicts when they appear. Teams that focus on conflict management are shown to perform better, according to a study measuring performance both as team and product performance [S6]. Conflicts can be managed in a number of ways. Imposing a solution on a team has a negative impact on performance, while recognizing disagreements and either engaging in a collaborative problem solving or seeking a compromise solution has a positive impact on performance [S2].

To summarize, there are several studies connecting cohesiveness and performance, with different operations of performance: team efficiency, team effectiveness and software product quality. Relationship conflicts will negatively influence performance, while task conflicts has a positive influence on performance. Conflict management is important in development teams, and teams should primarily engage in collaborative problem solving to reach an agreement to which team members can commit. We make the following proposition:

Proposition 3: Team cohesion has a strong positive impact on team performance.

Shared Mental Models

Software development is non-routine work that depends on the ability to acquire, communicate and make use of relevant knowledge. Shared mental models is knowledge held by team members that enable them to understand the tasks, the relationships among tasks, and to coordinate their actions and interactions [3, 9]. A development process such as Scrum can then be a shared mental model if a team has the same understanding of main activities and how they are related. If a team has established a shared mental model, this is believed to lead to team members anticipating each other's needs, and adjusting work strategies in accordance to changes in the team or in tasks. Shared mental models can thus be useful in understanding and explaining a wide range of collaboration patterns among team members.

Shared mental models result from knowledge sharing and subsequent discussions within the team. As has been found in other disciplines, the shared mental models positively affects team performance [S10]. The models may include a shared understanding of the team's goals, which also is shown to contribute to software team performance [S10]. A facet of mental models is the degree of knowledge and expertise of the project team, which is shown to lead to lower development costs. With less shared mental models there will be a reduction in team performance measured as the speed to market [S16].

The importance of shared mental models is shown by a study comparing the relative effect of mental models and demographic similarities such as age, tenure and gender, which finds that shared mental models has a larger effect on team performance [10].

Proposition 4: Shared mental models have a strong positive impact on team performance.

Team learning

While shared mental models reflect the state of the team, team learning blends process and state [5]. Team learning is an "ongoing process of reflection and action, characterized by asking questions, seeking feedback, experimenting, reflecting on results, and discussing errors or unexpected outcomes of actions" [11]. By monitoring and reflecting upon past events the team becomes reflexive and thereby able to adjust and adapt the team's objectives, strategies and processes to current or anticipated circumstances [12, 13]. Team learning results in "a

relatively permanent change in the team's collective level of knowledge and skill produced by the shared experience of team members" [14]. Through the team learning process the team makes changes to adapt or improve [15].

For software teams, there is a positive relationship between skills and expertise and team performance [S8]. A variety of skills, like task skills, development method skills and application domain skills have a positive influence on performance [S5]. The presence of expertise in a development team leads to increased performance, and further, expertise coordination has been found to be more important than years of experience and project plans for team performance [S4].

Skills and expertise can be increased when teams learn. Team learning also has a direct positive effect on software team performance [S11], and in particular software team effectiveness [S17]. Work satisfaction among developers can also be improved from team learning [S11].

Proposition 5: Learning has a strong positive impact on team performance.

Team performance and current advice on development

The five factors above are also found in general team performance models [3, 5], and we propose that these five factors have strong impact on the performance of software development teams. How do the propositions compare with what is communicated in current advice on development? We choose to discuss our propositions in light of the twelve principles behind the Agile manifesto, as these principles have had a large influence on transformation of software development practices in recent years, and the agile methods also have a strong focus on teamwork:

Team coordination: One principle states that software should be delivered frequently. Short development iterations will put emphasis on coordination in a team. The principles do not state how coordination should be conducted, but in Scrum this is done through frequent short meetings (daily stand-ups), and through common planning, review and retrospective meetings for each iteration. Thus, current advice is somewhat implicit on the importance of coordination.

Goal-oriented leadership: An agile principle states that the "best architectures, requirements, and designs emerge from self-organizing teams". Further a principle places

emphasis on customer satisfaction through focus on delivering "valuable software", and that "working software is the primary measure of progress". We find principles giving advice on types of goals that teams should have, and what constitutes project success. However, there is little guidance in agile methods as to how self-management can be successfully utilized to reach project goals. It is an open question whether self-management leads to goal-orientation, and indeed there are mixed findings regarding the connection between self-management and team performance [16]. Also, through empirical studies of agile development teams, we have seen few examples of teams that can be said to be self-managing [17].

Team cohesion: A principle states that "business people and developers must work together daily throughout the project". In Scrum, there are practices for making the team stick together and remain united like the daily stand-up meetings as well as joint planning and review meetings for each iteration. Extreme programming puts more emphasis on development practices, of which pair programming and shared code ownership are practices that could foster and show team cohesion. Thus, we find little advice on principle level, but concrete practices that support cohesion in various agile development methods. Conflicts is a theme which is not discussed in agile methods apart from providing arenas for making decisions and facilitating a process of negotiating task conflicts such as the practice of planning poker for effort estimation.

Shared mental models: There are no explicit mentioning of shared mental models in principles or practices of agile development, but one principle implicitly describes the importance of establishing shared mental models in stating that face-to-face conversation is "the most efficient and effective method of conveying information to and within a development team". Agile methods themselves can be seen as powerful shared mental models. Scrum for example has very few roles, practices and artifacts. This simplicity could facilitate a clear common understanding of how the development process is to be conducted. In Scrum, the focus on detailed short-term planning and frequent information exchange within the team are factors that could enable shared mental models.

Team learning: One principle focuses on the importance of team learning: "at regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly". This form of learning focuses on process improvement in retrospectives, but there are other practices in agile methods to foster learning about domain and technologies, like demonstrations, planning poker and coding dojos for whole teams and pair programming between two individuals in a team.

We have made five propositions on factors that have a strong positive impact on team performance for co-located software development teams. We see that only team learning is directly addressed in a principle of agile software development, the other four factors are indirectly addressed in agile practices found for example in Scrum and Extreme programming. Our findings are important for practitioners, because they highlight what effect practices should have, which could lead to increased understanding of why practices should be followed, leading to changes in how they are performed. An example is that a lack of precision in discussions on work tasks in daily stand-ups will limit the coordination effects of such meetings. Further, our five factors highlight areas that could be the focus of new practices, which could further increase team productivity.

For researchers, these findings are important because they connect practices in software development to a wider body of knowledge. This is important, for example in evaluating the outcome of practices, where future work could make use of established concepts in team research to increase internal validity of studies.

Comparing the results of our systematic review to overview articles on teamwork reveals that there are a number of factors identified in other domains that have not yet been sufficiently studied on software teams. In particular, central claims in agile software development, like that self-management leads to higher team performance needs further investigation. Further, team performance is challenged by more and more complex development projects, work distribution, time zones and sociocultural differences in global software development. Software development remains an archetype of knowledge work, which should grant attention from a number of research disciplines. The team remains the core organizational form in software development.

References

[1] Walz, D. B., Elam, J. J., and Curtis, B., "Inside a software design team: Knowledge aqcuisition, sharing and integration," *Communications of the ACM,* vol. 36, pp. 63-77, 1993.

[2] Katzenbach, J. R. and Smith, D. K., "The Discipline of Teams," *Harvard Business Review,* vol. 71, pp. 111-120, Mar-Apr 1993.

[3] Salas, E., Sims, D. E., and Burke, S. C., "Is there a "Big five" in teamwork?," *Small Group Research,* vol. 36, pp. 555-599, 2005.

[4] Dingsøyr, T. and Dybå, T., "Team Effectiveness in Software Development: Human and Cooperative Aspects in Team Effectiveness Models and Priorities for Future Studies,"

in *Workshop on Co-operative and Human Aspects of Software Engineering, International Conference on Software Engineering (ICSE)*, Zürich, Switzerland, 2012, pp. 27-29.
[5] Mathieu, J., Maynard, M. T., Rapp, T., and Gilson, L., "Team effectiveness 1997-2007: A review of recent advancements and a glimpse into the future," *Journal of Management,* vol. 34, pp. 410-476, Jun 2008.
[6] Malone, T. W. and Crowston, K., "The interdisciplinary study of coordination," *ACM Computing Surveys,* vol. 26, pp. 87-119, Mar 1994.
[7] Mudrack, P. E., "Defining group cohesiveness - a legacy of confusion," *Small Group Behavior,* vol. 20, pp. 37-49, Feb 1989.
[8] Mullen, B. and Copper, C., "The relation between group cohesiveness and performance - an integration," *Psychological Bulletin,* vol. 115, pp. 210-227, Mar 1994.
[9] Cannon-Bowers, J. A., Salas, E., and Converse, S., "Shared mental models in expert team decision making," in *Individual and group decision making: Current issues*, N. J. Castellan, Ed., ed Hillsdale, NJ: Lawrence Erlbaum, 1993, pp. 221-245.
[10] Kang, H. R., Yang, H. D., and Rowley, C., "Factors in team effectiveness: Cognitive and demographic similarities of software development team members," *Human Relations,* vol. 59, pp. 1681-1710, 2006.
[11] Edmondson, A. C., "Psycological safety and learning behavior in work teams," *Administrative Science Quarterly,* vol. 44, pp. 350-383, 1999.
[12] Müller, A., Herbig, B., and Petrovic, K., "The Explication of Implicit Team Knowledge and Its Supporting Effect on Team Processes and Technical Innovations," *Small Group Research,* vol. 40, pp. 28-51, February 1, 2009 2009.
[13] Gebert, D., Boerner, S., and Kearney, E., "Cross-functionality and innovation in new product development teams: A dilemmatic structure and its consequences for the management of diversity," *European Journal of Work and Organizational Psychology,* vol. 15, pp. 431 - 458, 2006.
[14] Ellis, A. P. J., Hollenbeck, J. R., Ilgen, D. R., Porter, C. O. L. H., West, B. J., and Moon, H., "Team learning: Collectively connecting the dots," *Journal of Applied Psychology,* vol. 88, pp. 821-835, 2003.
[15] Edmondson, A. C., "The local and variegated nature of learning in organizations: A group-level perspective," *Organization Science,* vol. 13, pp. 128-146, Mar-Apr 2002.
[16] Cohen, S. G. and Bailey, D. E., "What makes teams work: Group effectiveness research from the shop floor to the executive suite," *Journal of Management,* vol. 23, pp. 239-290, 1997.
[17] Moe, N. B., Dingsøyr, T., and Dybå, T., "Overcoming Barriers to Self-Management in Software Teams," *IEEE Software,* vol. 26, pp. 20-26, 2009.
[18] Kitchenham, B. A., "Guidelines for performing Systematic Literature Reviews in Software Engineering Version 2.3," Keele University and University of Durham, EBSE Technical Report2007.
[19] Dybå, T. and Dingsøyr, T., "Empirical Studies of Agile Software Development: A Systematic Review," *Information and Software Technology,* vol. 50, pp. 833-859, 2008.

Author biographies

Torgeir Dingsøyr (torgeir.dingsoyr@sintef.no) is a chief scientist at the SINTEF research foundation, Trondheim, Norway. He is also an adjunct professor at the Department of Computer and Information Science, Norwegian University of Science and Technology.

Tor Erlend Fægri (tor.e.fagri@sintef.no) is a research scientist at the SINTEF research foundation.

Tore Dybå (tore.dyba@sintef.no) is a chief scientist at the SINTEF research foundation.

Børge Haugset (borge.haugset@sintef.no) is a research scientist at the SINTEF research foundation.

Yngve Lindsjørn (ynglin@ifi.uio.no) is a lecturer at the Department of Informatics, University of Oslo.

Table 1: Primary studies on team performance.

Study	Study aim	Reference
S1	Behavioural and technical factors in team performance	Guinan, P. J., Cooprider, J. G., and Faraj, S., "Enabling software development team performance during requirements definition: A behavioral versus technical approach," Information Systems Research, vol. 9, pp. 101-125, Jun 1998.
S2	Conflicts in software development teams	Gobeli, D. H., Koenig, H. F., and Bechinger, I., "Managing conflict in software development teams: A multilevel analysis," Journal of Product Innovation Management, vol. 15, pp. 423-435, Sep 1998.
S3	Cooperation skills and personality for shared mental models	Yang, H. D., Kang, H. R., and Mason, R. M., "An exploratory study on meta skills in software development teams: antecedent cooperation skills and personality for shared mental models," European Journal of Information Systems, vol. 17, pp. 47-61, Feb 2008.
S4	Coordinating expertise in software development teams	Faraj, S. and Sproull, L., "Coordinating expertise in software development teams," Management Science, vol. 46, pp. 1554-1568, Dec 2000.
S5	Develop a model that considers team task skills as a moderator to the more specific application domain and development methods skills.	Chan, C. L., Jiang, J. J., and Klein, G., "Team task skills as a facilitator for application and development skills," IEEE Transactions on Engineering Management, vol. 55, pp. 434-441, 2008.
S6	Effect of intra-group conflict on team performance	Sawyer, S., "Effects of intra-group conflict on packaged software development team performance," Information Systems Journal, vol. 11, pp. 155-178, Apr 2001.
S7	Effect of team diversity on performance	Liang, T. P., Liu, C. C., Lin, T. M., and Lin, B., "Effect of team diversity on software project performance," Industrial Management & Data Systems, vol. 107, pp. 636-653, 2007.
S8	Examine the influence of team cohesiveness, experience and capability on team performance	Lakhanpal, B., "Understanding the factors influencing the performance of software-development groups - an exploratory group-level analysis," Information and Software Technology, vol. 35, pp. 468-473, Aug 1993.
S9	Examine the role of collective ownership and coding standards on team performance	Maruping, L. M., Zhang, X. J., and Venkatesh, V., "Role of collective ownership and coding standards in coordinating expertise in software project teams," European Journal of Information Systems, vol. 18, pp. 355-371, Aug 2009.
S10	Influential characteristics of information systems development team performance	Lu, Y. B., Xiang, C. J., Wang, B., and Wang, X. P., "What affects information systems development team performance? An exploratory study from the perspective of combined socio-technical theory and coordination theory," Computers in Human Behavior, vol. 27, pp. 811-822, Mar 2011.
S11	Knowledge integration in information systems	Janz, B. D. and Prasarnphanich, P., "Freedom to cooperate:

	development teams	Gaining clarity into knowledge integration in information systems development teams," IEEE Transactions on Engineering Management, vol. 56, pp. 621-635, 2009.
S12	Managerial control and team level control in design teams	Henderson, J. C. and Lee, S., "Managing i/s design teams - a control theories perspective," Management Science, vol. 38, pp. 757-777, Jun 1992.
S13	Relationship between design method, goal orientation and team effectiveness	Sonnentag, S., Frese, M., Brodbeck, F. C., and Heinbokel, T., "Use of design methods, team leaders' goal orientation, and team effectiveness: A follow-up study in software development projects," International Journal of Human-Computer Interaction, vol. 9, pp. 443-454, 1997.
S14	Shared mental models and team effectiveness	Kang, H. R., Yang, H. D., and Rowley, C., "Factors in team effectiveness: Cognitive and demographic similarities of software development team members," Human Relations, vol. 59, pp. 1681-1710, 2006.
S15	Software development team flexibility antecedents	Li, Y. Z., Chang, K. C., Chen, H. G., and Jiang, J. J., "Software development team flexibility antecedents," Journal of Systems and Software, vol. 83, pp. 1726-1734, Oct 2010.
S16	Team memory in software development projects	Keskin, H., "Antecedents and consequences of team memory in software development projects," Information and Management, vol. 46, pp. 388-396, 2009.
S17	Team reflexivity in innovative teams	Hoegl, M. and Parboteeah, K. P., "Team reflexivity in innovative projects," R&D Management, vol. 36, pp. 113-125, 2006.
S18	The effect of team dynamics and organizational support on ICT project success	Gelbard, R. and Carmeli, A., "The interactive effect of team dynamics and organizational support on ICT project success," International Journal of Project Management, vol. 27, pp. 464-470, Jul 2009.
S19	Understand the impact of teamwork quality on team performance	Hoegl, M. and Gemuenden, H. G., "Teamwork quality and the success of innovative projects: A theoretical concept and empirical evidence," Organization Science, vol. 12, pp. 435-449, Jul-Aug 2001.

WEB EXTRA: Team Performance in Software Development: Research versus Current Advice

Torgeir Dingsøyr, Tor Erlend Fægri, Tore Dybå, Børge Haugset, Yngve Lindsjørn

This material complements the Voice of Evidence column, "Team Performance in Software Development: Research versus Current Advice" (*IEEE Software,* XXX). That column summarized results from a systematic review on team performance for software development teams. This extra material provides details on research method and shows connections between findings and primary studies.

A central concept in studies of teams is team performance.[2] We use this term to refer to evaluations of the results of the teamwork. There are a number of results of the work of software development teams, from the quality of the software to the ability of the team to meet project goals and budgets. But also the motivation of team members to work together in the future, often measured by job satisfaction, is included in this broad definition of performance.

How this review was conducted

A systematic review [18] is characterized by a defined research question, identification of inclusion and exclusion criteria, search for relevant studies, critical appraisal, data extraction and synthesis. We conducted this review as follows:

Research question: What main factors influence the performance of software development teams?

Inclusion and exclusion criteria: We included empirical survey studies of team performance conducted on co-located professional software development projects, published in scholarly

[2] Some studies refer to team performance as the process of conducting teamwork, while the evaluation of the outcome of the teamwork is referred to as team effectiveness [8]. We consequently use team performance to refer to evaluation of all team outcomes, like ability to meet project goals, budget and schedule, the quality of software developed, development effectiveness and efficiency, and also include team member's job satisfaction. Some studies also include learning as one indicator of team performance.

journals. We excluded studies on students, on particular development practices, and on distributed and global development teams. The reason for focusing on surveys was that these studies are conducted on industry participants; show causal relationships and it is also a way of limiting the number of studies. We focus on co-located teams to address team factors only, and exclude factors related to temporal, geographical or sociocultural distance.

Data sources and search strategy: We conducted searches in the *ISI Web of Knowledge* and *Scopus* in October 2011 with the following search string:

Title=(Team OR group OR teamwork) AND Topic=Software AND Document Type=(Article OR review).

Figure 1 shows the study selection process.

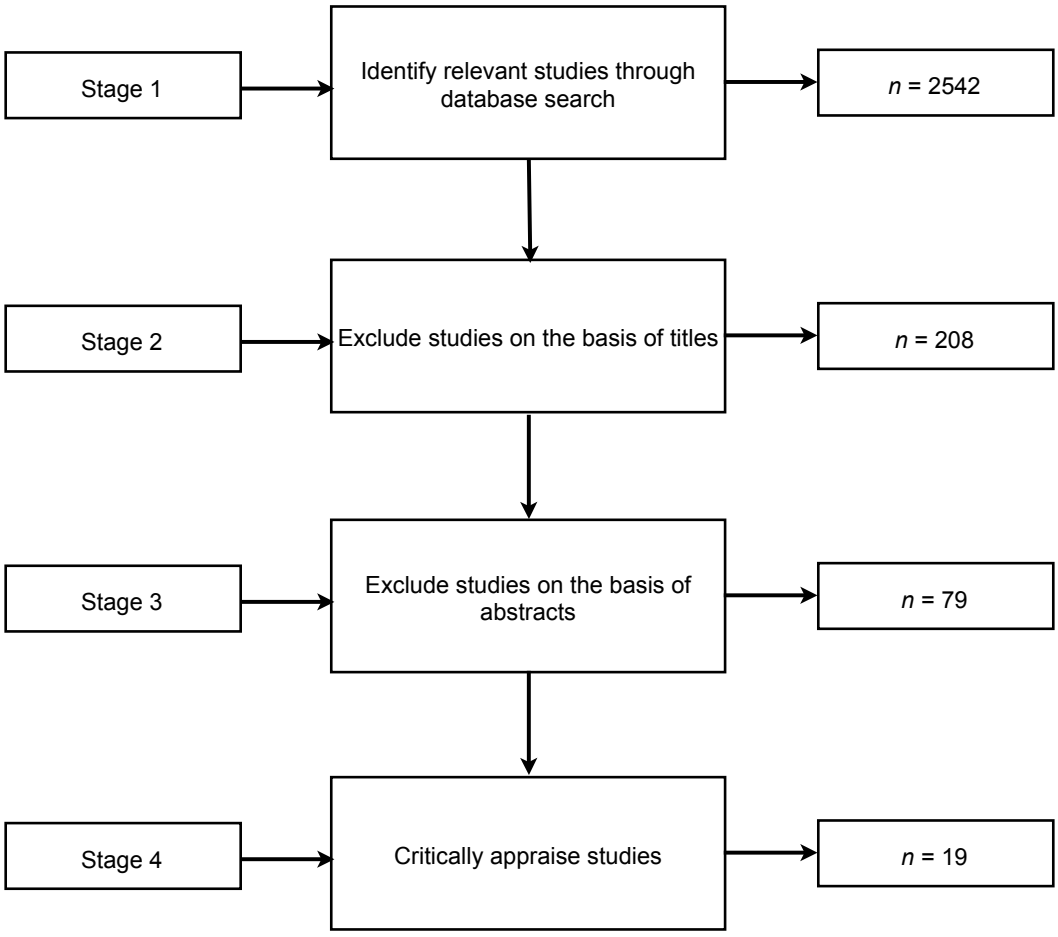

Figure 1: Stages of study selection process.

Citation management, retrieval and inclusion decisions: The 2542 citations retrieved at stage 1 were imported to a reference management package, and then exported to a spreadsheet, where further decisions were recorded. At stage 2, two authors excluded studies that were clearly not related to teamwork in software development. At stage 3, reading full abstracts further excluded irrelevant studies. Many studies were excluded because they were conducted on student teams, or because the teams were not co-located. All text was read by two authors independently, and by a third author in case of disagreement. At stage 4, we excluded non-survey studies, and studies which did not have a research question or one or more hypotheses related to team performance. This left 19 studies for quality assessment (Table 1 in published article).

Quality assessment: The final selection of studies were assessed according to eight criteria, adapting Dybå and Dingsøyr´s [19] criteria for surveys, see Tables 1 and 2. For the 19 articles assessed for quality, each article was assessed by the three first authors, and final quality scores were calculated by taking the mode of all three scores. For four of the 152 scores, there was not agreement between two assessors, and there we used the mean value. The maximum possible score was 32. The average was 18.7, with 14 as the lowest score and 22 as the highest. The article with score 14 lacked a discussion of researcher bias (criteria 5) and study limitations (criteria 6). These were also the criteria with the lowest overall scores. The studies scored best on clear study aims (criteria 1) and description of questionnaire design and definitions of measures (criteria 3). We did not exclude any studies based on the quality assessment.

Data extraction: We extracted research questions, hypotheses, context description of surveys, and key information like number of teams studied, number of respondents, team size, way of measuring performance as well as test type and significance level. All information was recorded in a spreadsheet.

Synthesis of findings: We derived the factors in this article by grouping studies and identifying factors that had at least three studies showing an influence on team performance. We then assigned one researcher per group for thematic synthesis of identified findings, based on confirmed and rejected hypotheses related to team performance. The connection between factors and primary studies are shown in Tables 3-7.

Limitations: We only investigated empirical surveys, not other types of studies. However, the factors identified correspond to factors identified in general team performance models, based on broad literature reviews [3, 5].

Acknowledgement

The work was carried out in the projects TeamIT and Agile 2.0 supported by the Research council of Norway through grants 193236 and 236759. Agile 2.0 is also supported by the companies Kantega, Kongsberg Defence & Aerospace, Sopra Steria, and Sticos.

Table 1. Quality assessment of primary studies, see Table 2 for quality checklist. The reference refers to studies listed in Table 1 in main article.

Reference	1	2	3	4	5	6	7	8	Total score
S1	3	3	3	3	1	2	3	3	21
S2	2	3	2	3	0	2	2	2	16
S3	3	3	3	3	0	3	3	3	21
S4	3	3	2	2	1	3	3	3	20
S5	3	2	3	3	0	2	3	2	18
S6	3	3	3	3	1	3	3	3	22
S7	3	2	3	2	0	2	3	2	17
S8	3	2	2	2	0	1	2	2	14
S9	3	3	3	3	1	2	3	3	21
S10	3	2	3	2	1	2	3	2	18
S11	3	3	3	3	1	3	3	3	22
S12	3	3	3	3	1	1	2	2	18
S13	3	2	3	2	1	2	2	2	17
S14	3	3	3	2	0	3	3	3	20
S15	3	2	3	2	1	2	3	2	18
S16	2	2	3	2	0	2	2	2	15
S17	3	3	3	3	1	2	3	3	21
S18	3	2	2	2	0	2	3	2	16
S19	3	3	3	3	1	2	3	3	21

Table 2. Quality Checklist

#	Criteria	Things to consider
Category: Questions on Aims		
1.	Do the authors clearly state the aims of the research?	*Do the authors state research questions, e.g., related to time-to-market, cost, product quality, process quality, developer productivity, and developer skills?* *Do the authors state hypotheses and their underlying theories?*
Category: Questions on Design, Data Collection, and Data Analysis		
2.	Do the authors describe the sample and the target population?	*Do the authors explain how the sample and target population were defined and selected?* *Do the authors state to what degree the sample is representative of the target population?* *Do the authors explain why the sample they selected was the most appropriate for providing insight into the type of knowledge sought by the study?* *Do the authors report the sample size and response rate?*
3.	Do the authors describe the design of the questionnaire and define the measures?	*Do the authors explain how items and measurement scales were defined and selected (e.g., domain of concepts, multiple-item scales, units, counting rules)?* *Are quality control methods used to ensure consistency, completeness and accuracy of collected data?* *Are reliability and validity analyses performed (e.g. Cronbach's alpha, item-scale correlations, factor analysis)?* *Do the authors append the questionnaire?*
4.	Do the authors define the data analysis procedures?	*Do authors justify their choice / describe the procedures / provide references to descriptions of the procedures?* *Do the authors report significance levels and effect sizes?* *Do the authors perform analyses of possible nonresponse bias?* *Do the authors report or give references to raw data and/or descriptive statistics?*
5.	Do the authors discuss potential researcher bias?	*Were the authors the developers of some or all of the treatments? If yes, do the authors discuss the implications anywhere in the paper? (If the authors developed the treatments (or parts of them) without discussing the implications, the answer to question 5 is "not at all".)* *Do the authors critically examine their own role, potential bias and influence during the formulation of research questions, sample recruitment, data collection, and analysis and selection of data for presentation?*
6.	Do the authors discuss the limitations of their study?	*Do the authors discuss external validity with respect to subjects, materials, and tasks?* *If the study used novel measures, is the construct validity of the measures discussed?* *Do the authors discuss the credibility of their findings?*
Category: Questions on Study Outcome		
7.	Do the authors state the findings clearly?	*Do the authors present results clearly?* *Do the authors present conclusions clearly?* *Are the conclusions warranted by the results and are the connections between the results and conclusions presented clearly?* *Do the authors discuss their conclusions in relation to the original research questions?* *Are limitations of the study discussed explicitly?*
8.	Is there evidence that the survey can be used by other researchers / practitioners?	*Do the authors discuss whether or how the findings can be transferred to other populations, or consider other ways in which the research can be used?* *To what extent do authors interpret results in the context of other studies / the existing body of knowledge / theories?*

Each question is answered on a 4-point scale where:

- "3 = Fully" means all questions listed in the "consider" column can be answered with "yes"
- "2 = Mostly" means the majority of all (but not all) questions listed in the "consider" column can be answered with "yes"
- "1 = Somewhat" means some (but the minority) of the questions listed in the "consider" column can be answered with "yes"
- "0 = Not at all" means none of the questions listed in the "consider" column can be answered with "yes"

Table 3: Accepted hypotheses linking team coordination and team performance.

Hypotheses	Study
Adherence to coding standards is positively related to software project technical quality (i.e., it should negatively relate to the number of errors in the software code).	S9
Anticipation capability will have a positive relationship with the software development team's flexibility.	S15
Conventional team factors (presence of expertise, professional experience, administrative coordination, and software development methods) are positively related to team performance.	S4
In general, increases in both managerial control and team-member control hava a positive effect on I/S design team performance.	S12
In general, increases in team-member outcome control will have a significant positive effect on I/S design team performance.	S12
Organizational support will moderate the relationship between team dynamics and project success. The positive relationship between team dynamics and project success will be stronger among teams that report higher levels of organizational support.	S18
Team dynamics will be positively related to project success.	S18
The extent to which design methods are used in the software development process is positively related to team effectiveness.	S13

Table 4: Accepted hypotheses linking goal-oriented leadership and team performance.

Hypotheses	Study
Clarity of mission is positively related to ISD team performance.	S10
In general, increases in both managerial control and team-member control have a positive effect on I/S design team performance.	S12
Team leaders goal orientation is positively related to team effectiveness, in addition to the use of design methods.	S13
Visionary processes are positively related to team performance.	S1

•

Table 5: Accepted hypotheses and research questions linking team cohesion and team performance.

Hypotheses / Research questions	Study
Collective ownership is positively related to software project technical quality (i.e., it should negatively relate to the number of errors in the software code).	S9

Group cohesiveness would be positively related to the group's performance level	S8
How does the conflict management style relate to success?	S2
Teamwork quality (including cohesion) is positively related to the performance of teams with innovative projects.	S19
What effects do these factors have on packaged software development team performance?	S6
What factors most affect the level of intragroup conflict in packaged software development teams?	S6
What kinds of relationships exist between the composition of software teams and performance?	S7
What relation do conflict intensity and context have with project success?	S2

Table 6: Accepted hypotheses linking shared mental models and team performance.

Hypotheses	Study
As memory dispersion increases, the positive effect of procedural memory on speed- to-market will be increased.	S16
Clarity of mission is positively related to information systems development team performance.	S10
Declarative memory will be positively associated with less development cost.	S16
Team members' shared mental models have a more positive influence on software development team effectiveness than do demographic (age, tenure and gender) similarities.	S14
Team-members' shared mental models is positively associated with software development team effectiveness.	S3

Table 7: Accepted hypotheses linking team learning and team performance.

Hypotheses	Study
Conventional team factors (presence of expertise, professional experience, administrative coordination, and software development methods) are positively related to team performance.	S4
Cooperative learning exhibited in an ISD team is positively related to ISD work performance.	S11
Cooperative learning exhibited in an ISD team is positively related to individual work satisfaction.	S11

Expertise coordination processes (recognizing where expertise is needed, knowing where expertise is located, and bringing expertise to bear) are positively related to team performance.	S4
Expertise coordination processes are positively related to team performance above and beyond traditional factors.	S4
Team reflexivity is positively related to team effectiveness.	S17
The impact of application domain skills on software development project management success is increased for higher levels of the team task skills.	S5
The impact of application domain skills on software development process success is increased for higher levels of the team task skills.	S5
The impact of development methods skills on software development project management success is increased for higher levels of the team task skills.	S5
The impact of development methods skills on software development process success is increased for higher levels of the team task skills.	S5
The software development group's capability would be positively associated to the group's performance	S11
The software development group's experience would be positively associated to the group's performance	S11