Shared knowledge in virtual software teams

A preliminary framework

Tor Erlend Fægri
Dept. of Software Engineering, Safety and Security
SINTEF ICT
Trondheim, Norway
Tor.E.Fegri@sintef.no

Viktoria Stray
Dept. of Informatics, Software Engineering
University of Oslo
Oslo, Norway
stray@ifi.uio.no

Nils Brede Moe
Dept. of Software Engineering, Safety and Security
SINTEF ICT
Trondheim, Norway
Nils.B.Moe@sintef.no

***Abstract*—Shared knowledge allows virtual teams to collaborate more effectively. Shared knowledge in teams, hereafter called team knowledge, must be established and maintained. This is a key enabler for agile development in a distributed context. Hence, organizations may benefit from efforts to ensure sufficient levels of team knowledge. Such efforts may include different measures, such as project kick-offs, frequent visits across locations, knowledge sharing tools and practices. However, team knowledge includes many types of knowledge, with different impacts on the team's work. This paper outlines a framework for conceptualizing the breadth of team knowledge relevant for virtual software teams. With the help of this framework, organizations can think more strategically about how to improve team knowledge – for example the planning of kick-offs, what to focus on in face-to-face meetings and how the team members work together on a day-to-day basis. The framework may also be used to assist in planning team composition, for example based on individuals' knowledge and the overlap with other team members' knowledge. The framework uses four broad categories of team knowledge: task-related, team-related, process-related and goal-related. Beneath these four categories the framework details and describes more concrete knowledge types. We also provide examples from software practice for each knowledge type.**

Keywords–virtual teams, globally distributed teams, shared team knowledge, agile, distributed agile, shared mental models

I. INTRODUCTION

Virtual teams are teams that collaborate on a task while being dispersed geographically or organizationally. Both dispersion factors make communication more difficult. In such contexts effective work coordination becomes more dependent upon the shared knowledge already held by the individuals. We use the term team knowledge to describe knowledge that is shared among the team members. With team knowledge, team members can make valid assumptions about the activities done by other team members and thereby reduce the negative impact of distribution. Team knowledge is especially important in virtual teams because members in virtual teams often have little knowledge of what people at other sites are doing day to day which may lead to misunderstandings [13].

Many companies face challenging decisions regarding the application of agile methods in virtual teams [24]. One topic of concern is to enable effective collaboration and self-organization. Self-organization is a key agile principle [4, 12, 22]. Shared knowledge within the team is particularly important for collaboration and self-organization. Only by a sufficient level of shared knowledge can teams reap the benefits of agile methods. Shared knowledge allows the team to utilize holographic organization principles – creating teams that can take on difficult tasks, and adapt to emerging situations. Hence, establishing and maintaining team knowledge is an effective vehicle for achieving high-performance virtual software teams.

Software teams share knowledge about many issues to get their work done. The main contribution of this paper is to outline a framework of team knowledge that classify and describe classes of knowledge with particular importance for virtual agile teams. The main classifications of the framework is adopted from Wildman et al. [29] and amended with more detailed attributes from relevant research streams. We also include examples from the software development context.

The main value of the framework is that it constitutes a tool for structured thinking in software process improvement work in virtual teams context. For example, having some idea about the classes of knowledge to share within a team can be useful because establishing and maintaining team knowledge comes with a cost, particularly in situations where existing knowledge must be challenged and changed. Being able to prioritize different knowledge types of importance to virtual teams can be a benefit here. It could thus serve as a foundation for knowledge management initiatives as well. Furthermore, the framework may also be used as foundation for quantitative investigations of team knowledge in virtual teams.

II. APPROACH

Numerous strands of research have investigated team knowledge. Team mental models [12, 20], transactive memory systems [28], strategic consensus, collective mindfulness and situation awareness [6], group learning, group sociology of knowledge [2] are examples of research that have contributed to our understanding of shared knowledge in teams.

To improve our capacity to understand and respond to the needs of virtual software teams, it makes sense to seek support in the range of research strands addressing team knowledge. Therefore, we decided to build on an organizing framework that summarizes the current team knowledge research [29]. This framework is both holistic and inclusive – hence it befitted our purpose well.

This is the author accepted version of an article published in
2016 IEEE 11th International Conference on Global Software Engineering (ICGSE)
https://doi.org/10.1109/ICGSE.2016.22

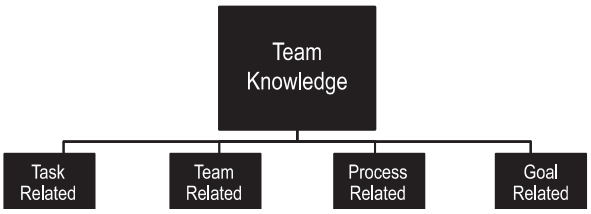

Figure 1. Team knowledge framework (from Wildman et. al. 2012)

Wildman et al. [29] divided team knowledge into four categories: *task-related, team-related, process-related and goal-related* (see Fig. 1). By means of these categories, the framework includes and systematizes key streams of shared team knowledge conceptualizations – an important benefit for software research which is, by being research of a highly applied nature, encouraged to relate to current streams in a number of adjacent research topics. For example, team mental model research has predominantly assumed a strictly causal I-P-O (Input-Process-Output) model of understanding team knowledge that seems inadequate to describe and explain the complex reality of software teams. Rather, learning and the emergent nature of behavior, motivation, and cognitive structures seems a more appropriate point of origin – as is now also recommended by contemporary team research [6, 17]. Agile software development, both in non-virtual and virtual teams, is motivated by the capacity to embrace uncertainty and learning opportunities. Therefore it seems reasonable to base our understanding of team knowledge on a framework that accepts these assumptions.

A second benefit of the classifications by Wildman et al. is that the authors made no assumptions about causal relationships among the shared team knowledge types. Practitioners may therefore use this framework irrespective of cause-effect hypothesis. E.g. it could be used to determine weak areas in team knowledge. Being very broad, this framework may be too extensive for some virtual software team improvement efforts, but adopters of the framework are free to select those categories of knowledge they want to work with – e.g. based on a consideration of which categories are most important in the actual context. The absence of assumed causal relationships among the categories makes this straightforward. This does not mean that investigations of causal relationships are incompatible with the framework. Such relationships can be introduced in specific applications of the framework.

A third benefit is that the framework may open the field for new research opportunities within virtual software teams. One example is the behavioral research approach that has predominantly been addressing the team-related knowledge category – using the transactive memory systems ideas – but could in fact be just as interesting from a holistic, broad perspective addressing both tasks and processes. Team behavior, i.e. the application of team knowledge in practice, may give new insights into those other knowledge categories. Software process research should be equally interested in actual behavior as in the formal doctrine.

Motivated by our understanding of software development, we have decided to deviate from Wildman et al. in an important consideration. Wildman et al. noted that it has been tradition to distinguish between static and dynamic team knowledge. They argued that it might be useful to treat team knowledge differently with respect to its expected longevity. Hence, each of the four categories – task-related, team-related, process-related and goal-related – have a static and a dynamic sub-category in the framework of Wildman et al. We have chosen not to carry this distinction forward in our proposal. Although we think the distinction may have merit in certain practical cases, it is difficult to make this distinction on a generic level for all virtual software teams. The main reason for this is that virtual agile software development teams work in a highly dynamic environment where it is difficult, up-front, to identify any types of knowledge that should be considered static. For example, the decision of Wildman et al. to classify task mental models as static knowledge might be appropriate for a manufacturing team, but would not befit a virtual software development team. In software practice, task knowledge is an emergent property – widely recognized by the agile movement in the appreciation of customer collaboration and working software before contracts and specifications. Furthermore, once a software team has completed a task, they move on to a new task with a new task mental model. Other knowledge types may lend themselves better to such classification though – for example those pertaining to overarching goals and visions, in addition to team norms and team membership. However, even here, we see good reasons to treat such knowledge as potentially emergent. Challenging team values and norms are key activities in team coaching practice thus suggesting that even values and norms can be altered [9].

Within the four categories task-related, team-related, process-related and goal-related team knowledge, we have populated the framework with more specific team knowledge elements relevant and representative for the corresponding category (called “relevant knowledge types” in the tables). In this way, we extend the framework of Wildman et al. We have also, due to our ambition to support the agile, virtual software teams, paid special attention to self-organization. Self-organization is key to distributed agile development. Because self-organization puts particular demands on shared knowledge, we have identified particular knowledge elements from research on self-organizing teams. We have also included a brief description for each.

Subsequently, we have added examples from the software development context. These examples aim to assist in the application of the framework in a software development situation by making it easier to translate. We have made comments where we were unable to find examples from software development research. This can indicate a gap in the knowledge front.

TABLE I. TASK RELATED KNOWLEDGE

Relevant knowledge types	*Description*	*Software examples*
Task strategies [11] task performance strategies [18], task plans [25].	Shared understanding about how a task is supposed to be accomplished by the team so that a sufficient level of performance can be achieved. How task work is allocated to members. Use of team subgroups working on parts of a task.	Collaborative programming or independent programming [23].
Task knowledge [17, 18, 28]. Likely contingencies, scenarios, environmental constraints [18]. Task component relationships [18].	Shared understanding about the content of the task. Shared understanding about how the parts of the tasks interact. Shared understanding about how a task is connected to its environment.	Shared knowledge about that state of the code for a particular feature [7].

It is important to note that the tables are neither exhaustive nor definitive. Our main ambition has been to create a preliminary framework that is analytically sound – i.e. provides a categorization that is useful for deeper analysis. Furthermore, our process for identifying examples in software literature is best described as ad-hoc and exploratory. Nevertheless, some of the example sections are thin and reflects the low availability of software research on the particular types. We found fewer still in the virtual software team domain.

III. A FRAMEWORK OF TEAM KNOWLEDGE FOR GLOBAL SOFTWARE DEVELOPMENT

Knowledge from all the categories of the framework allow team members to make assumptions about the work of other team members, and thereby contribute to make the team more effective in a distributed setting. What separates the knowledge categories is what type the knowledge belongs to, and how it may help team members make assumptions about the others.

We will now give examples from the framework from each of the knowledge categories.

A. Task-related knowledge

This category of knowledge concerns team members' shared understanding about the tasks, e.g. items in a product backlog or other divisions of the whole work to be done. Table I introduces key knowledge types for this category.

In this category we find two primary types of knowledge. 1) Knowledge regarding how a task should be accomplished, e.g., the problem-solving strategy. 2) Knowledge about what the task entails, e.g. a profound understanding of the criteria used to determine if the task was successfully completed or not.

Note that task-related knowledge is distinct from process-related knowledge in that the former will be particular for each task, while the latter will be generic for a range of tasks (e.g. how the team resolves coordination or leadership issues for work in general).

B. Team-related knowledge

Team-related knowledge (see Table II) refers to the characteristics and qualities of each of the team members or of the team as a whole [29]. These include characteristics such as team members´ knowledge, skills, attitudes, strengths, weaknesses, preferences and tendencies [18]. This category also concerns knowledge about who is on the team and share responsibility for team outcomes [11] who of the team members need assistance, and who is losing focus [29]. Expertise location, the extent to which team members know who knows what on the team, has an important role in virtual teams [14]. Expertise location is one dimension of a larger construct known as team cognition [12].

Team related knowledge bears strong resemblance to the transactive memory concept that seeks to describe and explain how a group functions as a knowledge creating entity (or collective) larger than the individual group member. Research on transactive memory systems has shown how team-related knowledge allows the team to function effectively with three dimensions of behavioral

TABLE II. TEAM-RELATED KNOWLEDGE

Relevant knowledge types	*Description*	*Software examples*
Team membership [11].	Defined boundaries of the team allow determination of who shares responsibility for the team's work.	Internalizing team membership is associated with higher levels of team performance in software teams [21].
Team member model [10, 18].	Team members´ knowledge, skills, attitudes, preferences and tendencies.	Shared mental models in software development [16, 30].
Expertise location [14, 28].	The common awareness of each team member’s specialized knowledge and unique expertise.	Software development teams must be able to know where expertise is located and where it is needed because this positively affects team performance [8, 12]. Expertise location can be improved by techniques such as Planning Poker [9].

TABLE III. PROCESS-RELATED KNOWLEDGE

Relevant knowledge types	*Description*	*Software examples*
Team interaction mental models [3, 29].	Team processes such as communication, leadership and coordination.	Effective global virtual teams need a rhythmic temporal pattern of interaction [1]. Use of video in distributed agile team meetings positively affect team communication [26].
Team norms [11].	Shared expectations of how to behave. Norms are formed and adopted as patterns of action are found useful or effective.	Manipulation of norms can help teams perform software engineering tasks better [27].

abilities – recognizing, trusting, and coordinating specialized knowledge among team members – and such dimensions have a positive impact on team performance [14]. It is also suggested that transactive memory systems can be formed in virtual teams, although with more effort [14].

C. Process-related knowledge

Process-related team knowledge refers to team member interaction and interpersonal processes in a team such as communication, coordination and leadership [29]. Knowledge about team interactions creates expectations and drives team member behavior [18]. Having this type of knowledge enables virtual teams to be adaptable. Norms are codes of conduct that are accepted by the team members and these may enhance or reduce team effectiveness [27].

Table III introduces key knowledge types from the process-related knowledge category.

D. Goal-related knowledge

Goal-related knowledge concerns the goals, visions and overall agreements pertaining to the team's work. Table IV introduces key knowledge types in this category. Goal-related knowledge is distinct from task-related knowledge in the sense that goals pertain to higher-level ambitions for the team – both superordinate or subordinate [29]. Furthermore, goal-related knowledge is distinct from team-related knowledge in that goals pertain to external objectives rather than the socio-collective properties of the team.

Goal-related knowledge is very important for virtual agile teams. Goals and visions are typically established early on in the team's lifespan. For self-organization to work, team members must have a profound interest and commitment to the overall objectives of the team [22]. In virtual teams, the social bonds that enact social contracts amongst the members are more fragile. In consequence, establishing and maintaining shared goals becomes more challenging – but no less important.

In our search, we found no agile software studies that have looked into more strategic-level knowledge. We believe that the ability to involve software teams in planning at different levels will be increasingly important in the time to come as turbulence seems to be on the rise.

IV. DISCUSSION AND CONCLUSION

We have highlighted team knowledge as an important topic in virtual teams, and described a framework for understanding shared knowledge in such teams. We believe that the concepts presented in this overview of team knowledge need to be further operationalized and studied. We briefly present some of the highlights of a research agenda that we envisage. Research activities focus on the three types of stakeholders that are important for success of virtual software teams: executive management, team leaders and team members.

Only a few of the studies of team knowledge in software development has addressed virtual teams. We know that achieving high performance in virtual teams is more demanding than for collocated teams. These gaps constitute potentially interesting research topics for further study. One example is how can management support the development for team knowledge in virtual teams?

Team situation awareness is studied almost entirely in laboratory settings [29]. There is a clear lack of studies in real-life settings. We have limited understanding about how team situation awareness develops and functions over time.

TABLE IV. GOAL-RELATED KNOWLEDGE

Relevant knowledge types	*Description*	*Software examples*
Expectations, goal, mission, objectives [10, 29].	Team goals are mental representation of overall goal or mission for the team, team expectations or (challenging) performance objectives. Here is also included the mental representations concerning the achievement of these goal.	Software team outcomes [4, 5]. Overarching team goals and mission [30, 31].
Strategic consensus – strategic vision [14, 15, 19].	Agreement about strategic goals for the organization.	None found.

We are aware that for many virtual software teams, the interaction with people outside the team (external network) can bring tremendous and vital capacities to the teams' work. Closer investigation of this aspect is an interesting and valuable amendment to the framework.

ACKNOWLEDGMENT

This work was supported by the Smiglo project, partly funded by the Research Council of Norway under grant 235359/O30.

REFERENCES

[1] J. M. Bass, "How product owner teams scale agile methods to large distributed enterprises," *Empirical Software Engineering,* vol. 20, pp. 1525-1557, 2015.

[2] J. S. Brown and P. Duguid, "Organizational learning and communities-of-practice: Toward a unified view of working, learning, and innovation," *Organization Science,* vol. 2, pp. 40-57, 1991.

[3] J. A. Cannon-Bowers, E. Salas, and S. Converse, "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. 221-246.

[4] A. Cockburn and J. Highsmith, "Agile software development: The people factor," *IEEE Computer,* vol. 34, pp. 131-133, November 2001 2001.

[5] K. Crowston and E. E. Kammerer, "Coordination and collective mind in software requirements development," *IBM Systems Journal,* vol. 37, pp. 227-245, 1998.

[6] L. A. DeChurch and J. R. Mesmer-Magnus, "The Cognitive Underpinnings of Effective Teamwork: A Meta-Analysis," *Journal of Applied Psychology,* vol. 95, pp. 32-53, 2010.

[7] J. A. Espinosa, R. E. Kraut, S. A. Slaughter, J. F. Lerch, J. D. Herbsleb, and A. Mockus, "Shared mental models, familiarity, and coordination: A multi-method study of distributed software teams," in *Twenty-third International Conference on Information Systems*, Barcelona, 2002, pp. 425-433.

[8] S. Faraj and L. Sproull, "Coordinating Expertise in Software Development Teams," *Management Science,* vol. 46, pp. 1554-1568, 2000.

[9] T. E. Fægri, "Adoption of team estimation in a specialist organizational environment," in *Agile Processes in Software Engineering and Extreme Programming (11th International Conference, XP2010)*. vol. LNBIP 48, A. Sillitti, A. Martin, X. Wang, and E. Whitworth, Eds., ed Berlin Heidelberg: Springer-Verlag, 2010, pp. 28-42.

[10] J. R. Hackman, "The psychology of self-management in organizations," in *Psychology and work: Productivity, change, and employment*, M. S. Pallack and R. O. Perloff, Eds., ed Washington, DC: American Psycological Association, 1986.

[11] J. R. Hackman, "The design of work teams," in *Handbook of organizational behavior*, J. Lorch, Ed., ed Englewood Cliffs, NJ: Prentice Hall, 1987, pp. 315-342.

[12] J. He, B. S. Butler, and W. R. King, "Team cognition: Development and evolution in software project teams," *Journal of Management Information Systems,* vol. 24, pp. 261-292, Fal 2007.

[13] J. D. Herbsleb, "Global Software Engineering: The Future of Socio-technical Coordination," presented at the 2007 Future of Software Engineering, 2007.

[14] P. Kanawattanachai and Y. Yoo, "The impact of knowledge coordination on virtual team performance over time," *MIS Quarterly,* vol. 31, pp. 783-898, 2007.

[15] F. W. Kellermanns, J. Walter, C. Lechner, and S. W. Floyd, "The lack of consensus about strategic consensus: Advancing theory and research," *Journal of Management,* vol. 31, pp. 719-737, 2005.

[16] L. L. Levesque, J. M. Wilson, and D. R. Wholey, "Cognitive divergence and shared mental models in software development project teams," *Journal of Organizational Behavior,* vol. 22, pp. 135-144, 2001.

[17] J. Mathieu, M. T. Maynard, T. Rapp, and L. Gilson, "Team effectiveness 1997-2007: A review of recent advancements and a glimpse into the future," *Journal of Management,* vol. 34, pp. 410-476, 2008.

[18] J. E. Mathieu, T. S. Heffner, G. F. Goodwin, E. Salas, and J. A. Cannon-Bowers, "The influence of shared mental models on team process and performance," *Journal of Applied Psychology,* vol. 85, pp. 273-283, Apr 2000.

[19] J. E. Mathieu, M. T. Maynard, S. R. Taylor, L. L. Gilson, and T. M. Ruddy, "An examination of the effects of organizational district and team contexts on team processes and performance: A meso-mediational model," *Journal of Organizational Behavior,* vol. 28, pp. 891-910, 2007.

[20] S. Mohammed, L. Ferzandi, and K. Hamilton, "Metaphor no more: A 15-year review of the team mental model construct," *Journal of Management,* vol. 36, pp. 876-910, 2010.

[21] C. Monaghan, B. Bizumic, K. Reynolds, M. Smithson, L. Johns-Boast, and D. van Rooy, "Performance of student software development teams: The influence of personality and identifying as team members," *European Journal of Engineering Education,* 2015.

[22] S. Nerur, A. Cannon, V. Balijepally, and P. Bond, "Towards an understanding of the conceptual underpinnings of agile development methodologies," in *Agile Software Development: Current Research and Future Directions*, T. Dingsøyr, T. Dybå, and N. B. Moe, Eds., ed Berlin Heidelberg: Springer-Verlag, 2010, pp. 15-29.

[23] N. Ramasubbu, C. F. Kemerer, and J. Hong, "Structural complexity and programmer team strategy: An experimental test," *IEEE Transactions on Software Engineering,* vol. 38, pp. 1054-1068, 2012.

[24] D. Smite, N. B. Moe, and P. J. Ågerfalk, "Fundamentals of agile distributed software development," in *Agility across time and space: Implementing agile methods in global software projects*, D. Smite, N. B. Moe, and P. J. Ågerfalk, Eds., ed Berlin Heidelberg: Springer-Verlag, 2010, pp. 3-7.

[25] R. J. Stout, J. A. Cannon-Bowers, E. Salas, and D. M. Milanovich, "Planning, shared mental models, and coordinated performance: An empirical link is established," *Human Factors,* vol. 41, pp. 61-71, 1999.

[26] V. Stray, D. I. K. Sjøberg, and T. Dybå, "The daily stand-up meeting: A grounded theory study," *Journal of Systems and Software,* vol. 114, pp. 101-124, 2016.

[27] A. Teh, E. Baniassad, D. van Rooy, and C. Boughton, "Social Psychology and Software Teams: Establishing Task-Effective Group Norms," *IEEE Software,* vol. 29, pp. 53-58, 2012.

[28] D. M. Wegner, "Transactive memory: A contemporary analysis of the group mind," in *Theories of Group Behavior*, B. Mullen and G. Goethals, Eds., ed: Springer New York, 1987, pp. 185-208.

[29] J. L. Wildman, A. L. Thayer, D. Pavlas, E. Salas, and W. R. Howse, "Team knowledge research: Emerging trends and critical needs," *Human Factors,* vol. 54, pp. 84-111, 2012.

[30] H. D. Yang, H. R. Kang, and R. M. Mason, "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.

[31] X. Yu and S. Petter, "Understanding agile software development practices using shared mental models theory," *Information and Software Technology,* 2014.