IFKAD 2021: Managing Knowledge in Uncertain Times
Rome, Italy 1-3 September 2021
ISBN 978-88-96687-14-7 || ISSN 2280-787X

Transform Knowledge Assets into Reality: How the Purposeful Combination of Knowledge Activities Enables Organisations to Channelise the Knowledge Flow in Software Engineering Development

Martin Dietze *

Dermalog Identification Systems
Hamburg, Germany

Marion Kahrens

European College of Business and Management
London, UK

* *Corresponding author*

Abstract

Purpose — Knowledge Activities like acquiring, codifying, storing, maintaining, transferring and creating knowledge, can be seen as what constitutes Knowledge Management in an organisation. Executing knowledge activities should ideally come at no additional cost and blend well with the respective organisation's business. Being a highly knowledge-intensive discipline, software engineering becomes an interesting testbed for this approach. This study aims to identify existing connections between knowledge activities and some frequently used software team practices or measures, establish their relations and investigate their suitability as tools for implementing knowledge management.
Methodology and approach — In accordance with the research questions, this study used a qualitative research approach. Using a questionnaire, practitioners such as software developers and team leaders were asked to provide feedback on a set of previously selected team practices and measures typically used in software engineering projects and to assess their relation to the activities of acquiring, codifying, storing, maintaining, transferring and creating knowledge. The obtained results were analysed using frequency analysis and the comparison of summarised agreement vs. disagreement related to typical software development team practices and measures.
Findings — The results show that meetings and team processes like code reviews or pair programming come with intensive knowledge transfer while knowledge is acquired through activities involving client contact. New knowledge is created in a large number of team processes, often along with other knowledge activities. In addition, the inquiry revealed that the selection of team practices and measures are be both relevant and considered important for managing knowledge in software teams. This indicated that

ISBN 978-88-96687-14-7 || ISSN 2280-787X

activities that are already part of the software engineering process in many organisations can be facilitated to trigger particular knowledge activities.
Practical implications — The study clearly shows the relevance of practices and measures well-accepted in software engineering for implementing knowledge management in software development teams. Based on a systematic analysis it establishes their correspondence with particular knowledge activities thus allowing practitioners to assess or create the prerequisites for knowledge creation in their teams along with existing workflows. Project and team leaders, and ultimately the whole organisation benefit from a tool enabling them to improve both flow and creation of knowledge within and between teams strengthening expertise and sustainability in the particularly knowledge-intensive software market.

Keywords – Knowledge Creation, Knowledge Activities, SECI Model, Software Engineering, Development Team Practices

Paper type – Academic Research Paper

1 Introduction

Due to the nature of what it produces and the work involved to create it the software industry had to find ways of dealing with knowledge at a time when knowledge management as a scientific discipline had not yet evolved. After theories on systematic knowledge creation and knowledge management, Nonaka's SECI model (Nonaka and Takeuchi, 1995) and first notions of knowledge activities (Pentland, 1995; Andersen, 1996) surfaced in the 1990s, blending practitioners' best practice and researchers' findings became possible. This led to considerable research focusing on the particularities of knowledge in software engineering as well as the introduction of some of the newly established terminology and methodologies such as knowledge transfer and storage. Some researchers focused on technology commonly used in the software industry like Wikis and issue trackers to support knowledge transfer, maintenance and storage (Dingsøyr and Conradi, 2002; C. Khalil and S. Khalil, 2019; Sungkur and Ramasawmy, 2014; Abdur *et al.* 2015). Others, like Dissanayake *et al.* (2013), conclude that knowledge management key concepts as described by Nonaka and Konno (1998), Nonaka, Toyama, and Byosière (2003), and Nonaka and von Krogh (2009) are already present in agile software development methodologies.

However, while the combination of knowledge management and software engineering as a field of interest for researchers has now been around for at least two decades, a separation between the two camps is still noticeable and calls for

ISBN 978-88-96687-14-7 || ISSN 2280-787X

a more holistic approach trying to connect knowledge management to existing elements of software development teams' work. This research aims to fill this gap by investigating a selection of software development team practices, measuring their relevance to team knowledge and establishing their relation to six knowledge activities as proposed by Heavin and Adam (2012) thus enabling software project leaders to more consciously involve the concept of knowledge when designing team processes.

The paper is organised as follows: In section 2 relevant existing research is reviewed, followed by two research questions. The research method used in this study is outlined in section 3 The results are presented in section 4 followed by the conclusion in section 5.

2 Literature Review

2.1 Knowledge Creation

The fundamental concepts of what is known today as Knowledge Management and Knowledge Creation have been created and further developed since the 1990s, and a vast amount of relevant research on how to implement KM in — usually large scale — enterprises exists (Nonaka, 1991; von Krogh, Ichijo, *et al.* 2000; Ichijo and Nonaka, 2008). Nonaka (1994), defines knowledge as 'justified true belief' held by individuals. In order to qualify as knowledge rather than just information a "clear understanding of information and their associated patterns" is an additional important requirement (Bierly *et al.* 2000, p. 600). Knowledge is also the actuality of skilful action and/or the potentiality of creating situations to permit (skilful) action (Nonaka and von Krogh, 2009).

Nonaka (1991) points out that there is a distinction between explicit and tacit knowledge. Tacit knowledge comprises information that is difficult to express, formalize or share. It resides within people and according to Stenmark (2000) it can neither be documented in a manual nor be explained in words to others. Hence this kind of knowledge requires involving the knowing one and transferring the knowledge via periods of practices where experiences are shared through actions. In other words, knowledge needs to be transmitted from person to person (Grover and Davenport, 2017). Nonaka and Toyama (2003) reinforce Stenmark's idea of tacit knowledge by stating that the only way to acquire tacit knowledge is through shared direct experience by spending time together.

ISBN 978-88-96687-14-7 || ISSN 2280-787X

Thereby, tacit knowledge often takes the form of narratives, analogies or metaphors in order to give insight into the 'why' and 'how' something is done the way it is (Holste and Fields, 2010). From a practical standpoint this means after writing down the narrative, it is read by another group of people in the company and should be discussed afterwards. The analysis of the narrative will help making better decisions in the future. However, the ability to amplify tacit knowledge is limited.

When focusing on an entrepreneurial environment, making tacit knowledge explicit carries the risk of losing sustainable competitive advantage. Tacit knowledge transformed to explicit and having left the original company may become industry best practice (Lubit, 2001). In opposition to tacit knowledge, explicit knowledge can be transmitted by using strings, i.e. interaction between physical objects, in the right circumstances (Collins, 2010). It can be codified and/or verbalized (Nonaka and Toyama, 2003; Grover and Davenport, 2017) and expressed by data numbers, formulas, pictures, manuals, patents (Holste and Fields, 2010). Both explicit and tacit knowledge can be passed on and converted using appropriate methods and with varying degrees of difficulty or required effort. The resulting four possible transitions — tacit-to-tacit (Socialisation), tacit-to-explicit (Externalisation), explicit-to-explicit (Combination) and explicit-to-tacit (Internalisation) are the pillars of the SECI model, according to which organisational knowledge creation evolves from a spiral out of continuously transforming and transferring explicit and tacit knowledge between individuals (Nonaka and Takeuchi, 1995).

Since the beginning of the SECI model's development, a broad academic debate has arisen, mainly around the distinctions between the different conversion processes, the relationship between the explicit and tacit levels and its possibilities related to cultural differences. The adoption and application of the SECI model is under continuous consideration and development (von Krogh, Nonaka, *et al.* 2012; C. S. Lee and Kelkar, 2013; Nezafati *et al.* 2009; Tee and S. S. Lee, 2013). KM practitioners soon learnt that a lot of important knowledge is not explicit. Only discussion, probing, reflection and conversion of tacit knowledge can bring out valuable explicit knowledge. The four conversion modes of tacit in tacit, tacit in explicit, explicit in explicit and explicit reversed in the form of new tacit knowledge constitute an approach to support the exchange and creation of knowledge. The SECI model is widely accepted but varying contents and perceptions regarding the importance of particular aspects of the knowledge

ISBN 978-88-96687-14-7 || ISSN 2280-787X

creation model exist, such as cultural aspects, the practical implications of the transformation of knowledge and the role of management (Kahrens and Früauff, 2018).

An important extension, the concept of Ba (Japanese for 'space'), introduced by Nonaka and Konno (1998) and Nonaka and Toyama (2005), stating that in order for the SECI model's transitions to take place an enabling context is necessary. The four pillars are complemented by four types of Ba: Originating Ba (face to face) for Socialisation, Interacting Ba (peer to peer) for Externalisation, Cyber Ba (group to group) for Combination and Exercising Ba (on the site) for Internalisation. The concept of Ba stresses how important relationships and interactions between peers are for the process of knowledge creation, and the environment needs to provide psychological safety and care (von Krogh, Ichijo, *et al.* 2000). Besides the already mentioned creation of the appropriate enabling Ba, fostering organisational knowledge creation can be supported by targeted management decisions. Knowledge transfer and knowledge creation best prosper in micro-communities of small to medium size and changing membership over the years, and sources of innovation, like e.g. R&D departments need to be organised (Ichijo and Nonaka, 2008) accordingly. Teece (2010) points out that in tech companies (e.g. Intel), innovation already originates from development and production without even the need for dedicated R&D departments.

Knowledge sharing, depending on the flow of information, plays a central role here. Inside an organisation, decentralised decision-making and reduced formalisation are preferred as they encourage initiative and participation (Saifi *et al.* 2016). When dealing with peers outside the respective organisation, similar principles apply. Building communities with them enables tacit knowledge sharing through co-experience and shared praxis (Sapir *et al.* 2016; von Krogh, Ichijo, *et al.* 2000; Nonaka and Toyama, 2005; Ichijo and Nonaka, 2008). While the SECI model starts from the point where individuals already carry knowledge to enter the knowledge creation spiral, the idea of distinguishing between embodied and not-yet embodied ('self-transcending') tacit knowledge is introduced in Scharmer (2001). Scharmer address the question for the driving force behind the knowledge spiral, stressing the importance of appropriate enabling contexts and the necessary conversational complexity in communication. In the future, the application and extension of the SECI model are expected to be leaner in structures and more digitised concerning knowledge creation and sharing (Kahrens and Früauff, 2018).

ISBN 978-88-96687-14-7 || ISSN 2280-787X

The SECI model has also been subject to criticism, a.o. for being too vague in its recommendations and its perception of both, tacit and explicit knowledge (Schreyögg and Geiger, 2003; Schreyögg and Geiger, 2004; Lin, 2010), also for being deeply rooted in Japanese culture and hence not sufficiently considering cultural differences as possible obstacles to its application (Andreeva and Ikhilchik, 2011). Nevertheless, this research will consider the basic concept of the SECI model when assessing typical meeting structures in software development projects to improve the exchange among team members and thereby increase learning and knowledge.

2.2 Knowledge Management in Software Development

It is a common understanding that software engineering and development is a highly knowledge-intensive discipline, and various aspects of it have attracted the interest of researchers in knowledge management (Rus and Lindvall, 2002; Kavitha and Irfan, 2011; Vasanthapriyan *et al.* 2015; Shongwe, 2017). In terms of the taxonomy proposed by Earl (2001), knowledge management strategies are usually closely related to the technocratic schools for traditional software development and more to the behavioural schools for agile approaches (Bjørnson and Dingsøyr, 2008). Rus and Lindvall (2002) exemplify the need for knowledge management in software engineering by the different dimensions of knowledge involved in it:

- Process and product knowledge is concerned with how the development of a particular project should be approached.
- Domain knowledge is related to what a software system is developed for, like e.g. accounting or space shuttles.
- Technological knowledge includes programming languages, tools, libraries selected and used for implementation.
- Knowledge of local policies covers amongst others conventions existing in development teams.
- Also, it is important to know who the bearer of specific knowledge in the organisation is.

Hegde and Walia (2014) note that generally the different involved forms and dimensions of knowledge have a significant impact on the developers' creativity.

An early study based on 'lessons learnt' reports from eight software companies (Dingsøyr and Conradi, 2002) shows that employees are interested in this field

ISBN 978-88-96687-14-7 || ISSN 2280-787X

and also willing to invest in it, e.g. by developing supporting tools. From analysing knowledge flows involved when creating source code Sandhawalia and Dalcher (2010) conclude that the activities of knowledge creation, learning and reflection clearly play an important role in uncovering and reviewing mistakes and mismatches. Similarly, Ward and Aurum (2004) find that knowledge management helps appreciate the challenges and complexities inherent in software development with employees being able to identify activities related to knowledge management in their organisations. However, the same research also reveals that employees found the helpers, techniques and methodologies in use for the software development process inadequate to support the intended knowledge management.

A significant number of studies stresses the importance of software tool support. The use of knowledge repositories (e.g. Wikis, databases etc.) is considered a central element of knowledge management in software organisations (Dingsøyr and Conradi, 2002; C. Khalil and S. Khalil, 2019). However, this term is a bit misleading as persisted knowledge becomes *information* and needs to be transferred back to *knowledge* when consumed by others. Bjørnson and Dingsøyr (2008) confirm a tendency to over-emphasise the role of technology by finding that some software engineering organisations focus on storage and retrieval of knowledge at the cost of knowledge transfer, application and creation. In particular collaborative software tools already popular in software development environments, like issue trackers and the already mentioned Wikis, are useful, e.g. for externalisation of knowledge (Sungkur and Ramasawmy, 2014; Abdur *et al.* 2015).

Because of their strong reliance on team processes, agile methods already contain various elements of what is known from the SECI model as is shown in the mapping of agile practices to characteristics of Ba in Dissanayake *et al.* (2013). Similarly, Abdur *et al.* (2015) find processes and tools common in agile methods beneficial for knowledge creation. These findings are confirmed by an investigation in large French companies conducted by C. Khalil and S. Khalil (2019) who based on a theoretical model of knowledge management in agile environments and interviews with employees from 12 French companies conclude that agile practices contribute to the creation of a knowledge-intensive culture. But what makes agile methods so interesting from the knowledge management perspective is their team dynamics and the flow of information they bring along. These contribute to competitiveness and more or less come automatically in small

ISBN 978-88-96687-14-7 || ISSN 2280-787X

or medium-sized enterprises (Basri and O'Connor, 2011; Shongwe, 2017; Heavin and Adam, 2014). Even larger organisations can benefit from it since agile teams are usually kept small on purpose: e.g. Scrum recommends teams of three to nine people (Sutherland and Schwaber, 2010).

Elements fostering open discussion or collaboration have a strong influence on team knowledge. One is the Retrospective in Scrum where team members gather after each sprint reflecting on what did not go well in the finished iteration and which measures should be taken to prevent it from happening again (Viana *et al.* 2013). Similarly in projects based on the waterfall model lessons learnt meetings take place after milestones. Another example is Pair Programming where two developers share the same screen while composing source code together (Kavitha and Irfan, 2011).

2.3 Knowledge Activities

While knowledge clearly plays a crucial role in software engineering, still some measures are required in order to achieve or design a systematic process of knowledge creation. For this, researchers propose the definition of Knowledge Activities (KA) — transactions or manipulations of knowledge where the knowledge is the object, not the result (Jetter *et al.* 2006; Kraaijenbrink *et al.* 2006). The term Knowledge Activity is fairly generic, hence several different interpretations are in use. Table 1 provides a selection of variations to the concept.

Table 1: Variants to the concept of Knowledge Activities

Authors	Name	Activities
Holsapple and Joshi (2004)	Knowledge Manipulation Activities	Acquiring, Selecting, Internalising, Using, Externalising, Generating
Ward and Aurum (2004)	KM Activities	Application, Distribution, Organisation, Adaptation, Identification, Acquisition, Creation
Costa and Monteiro (2017)	Knowledge Management Processes	Acquisition, Storage, Codification, Sharing, Application, Creation
Kraaijenbrink *et al.* (2006)	Knowledge Activities	Elicitation, Codification, Detection, Assessment, Transfer (of knowledge), Transfer (of knowledge holder), Nurturing, Motivation
Heavin and Adam (2012)	Knowledge Activities	Acquire, Codify, Store, Maintain, Transfer, Create

ISBN 978-88-96687-14-7 || ISSN 2280-787X

Heavin and Adam (2012) derive a list of six Knowledge Activities — acquire, codify, store, maintain, transfer, create — from the definition provided by Holsapple and Joshi (2004) conducting an investigation of KA application in five Irish software SMEs finding a dominance of knowledge and an underrepresentation of knowledge creation (Heavin and Adam, 2012; Heavin and Adam, 2014). Ward and Aurum (2004) focus on seven separate activities — knowledge application, distribution, organisation, adaptation, identification, acquisition and creation — finding them used and evolving with technology, organisational culture and practises. In a subsequent study, the same group of researchers investigate how and whether all these activities are executed by looking at four projects in two software companies and linking them to typical project lifecycle stages: Requirements Gathering, Analysis & Design, Building & Development, Testing and Deployment (Aurum *et al.* 2008). Their findings indicate that while the KAs are executed, this often happens implicitly, i.e. without having been identified as KAs by developers. Although there clearly is a common understanding of the concept behind knowledge activities, there are differences in detail e.g. Costa and Monteiro (2017) refer to knowledge sharing and application instead of transferring and maintaining.

Hence it can be concluded that the concept of knowledge activities (or KM activities or knowledge management processes) is recognised as a relevant element in implementing knowledge management. To a far lesser extent, the question has been addressed which software development practices actually result in executing which KAs. Sungkur and Ramasawmy (2014) refer to such activities as building blocks for knowledge management processes to which technical tools are linked in order to show which aspects of knowledge management they support. In their survey on knowledge management in software engineering, Vasanthapriyan *et al.* (2015) go through the software development life cycle and analyse its stages for their relation to the teams' knowledge, while employees from small South African software companies comment on their experience regarding KAs in Shongwe (2017). Examples for such mappings, while not actually focused on, can be found in some other works, like Sandhawalia and Dalcher (2010) and Abdur *et al.* (2015).

Mapping of some team processes and measures to knowledge activities can be found as side products in some of the abovementioned studies, like e.g. Pair Programming, Daily Scrum, rotation/visit in Abdur *et al.* (2015), or design, test, implementation, document, development in Aurum *et al.* (2008). However, a

ISBN 978-88-96687-14-7 || ISSN 2280-787X

systematic analysis of how elements of software development workflows and management measures relate to knowledge activities would be of value to both researchers and practitioners, which is why this research is dedicated to this aspect.

2.4 Research Questions

The purpose of this study was to develop a holistic concept of implementing knowledge activities in the context of software engineering organisation.

Therefore, the overarching research question for this qualitative exploratory study was:

> ***RQ1.*** *Which knowledge activities are related to which team practices and measures?*
>
> *A subordinate research question to support the analysis and interpretation of primary data was:*
>
> ***RQ2.*** *How should team practices and measures be combined in order to support a mixture of knowledge activities suitable for effective knowledge management in software engineering organisations?*

3 Research Method

3.1 Qualitative Research

In accordance with the proposed research questions, this study used a qualitative research approach. Although such a kind of qualitative research allows for observing a phenomenon in-depth (Eisenhardt and Graebner, 2007) and case study research seems to be appropriate, the authors of this paper decided to take a broader view that is not bound to a single case organization.

Specific contextual problems are related to software development and delivery that create general challenges in relation to knowledge management in projects. In terms of complexity, these challenges are related to the complexity of software development projects consisting of complex interactions among project members internally and externally. Qualitative research methods are helpful to analyse abstract constructs of observable phenomenon (Creswell, 2014).

This study is concerned with analysing how knowledge activities and their structured and purposeful implementation benefit software engineering

ISBN 978-88-96687-14-7 || ISSN 2280-787X

organizations. The underlying assumptions of this research were based on a partially phenomenological stance. According to van Manen (2007), a context-sensitive form of interpretive inquiry, the phenomenology of practice, is well suited to serve practitioners who in their day-to-day practice may be unaware of the depths of people's experiences. By constructing the realities related to knowledge activities in developing and programming software solutions in projects with multiple perspectives this qualitative empirical method composed summaries of lived experience and asked a group of experts to assess these practices. Thereby, the underlying patterns and structures of meaning can be drawn when a central feature of the phenomenon is considered (van Manen, 2016). The interpretive findings of this research may serve as the foundation for future empirically conducted researches aiming to measure the degree to which knowledge activities applied in software engineering projects contribute to the organizational knowledge base.

3.2 Sampling and data collection

Several factors contribute to the success of knowledge management activities within organisations including the firm's objectives, the enabling conditions and the environment, the knowledge processes including the dialogue between individuals and the individual's experiences (Nonaka and von Krogh, 2009). The implication here is that knowledge activities are best examined within the context where they take place. A key advantage of qualitative research is that the researchers are free to choose from multiple sources while still incorporating one's know-how to extract understanding.

In this research, data was collected through questionnaires. The study population included managers, project managers, developers, and other experts involved in software development projects for internal organizational purposes or for external client-related purposes.

The sampling strategy for the data collection followed the principle of convenient purposive sampling that requires recruiting from a group or organisations with a close perspective on the research phenomenon of interest from a group or organization (Creswell, 2014). In the case of the questionnaire, the convenient purposive sampling took place by posting the questionnaire on online research platforms (SurveyCircle) and social media (LinkedIn) approaching special expert group related to software project management, agile

ISBN 978-88-96687-14-7 || ISSN 2280-787X

methodologies. Although convenience sampling strategies do not allow generalising the findings due to the unknown population, it plays a prominent role in the field of business and management (Bryman and Bell, 2007). The expectation was to reach 50 participants with a self-administrated internet-mediated questionnaire. Table 2 illustrates the composition of the participants.

Table 2: Composition of the participants

Role	**Team Lead**	**Developer**	**Architect**	**Other**	
	40.48%	30.95%	5.95%	22.62%	
Organisation employees	**1 … 10**	**11 … 50**	**51 … 250**	**251 … 500**	**over 500**
	10.71%	7.14%	30.36%	12.50%	39.29%
Organisation core business	**Standard Software**	**Individual Software**	**Products incl. software**	**Services incl. software**	**Others**
	3.57%	19.64%	26.79%	26.79%	23.21%

In this research, RQ1 and RQ2 served as a guide for the review of the questionnaire data which was analysed by frequency analysis and the comparison of summarised agreement vs. disagreement to statements related to typical software development team practices and measures.

4 Results and Interpretation

4.1 Various companies' approaches to software development

Through the questionnaire responses were obtained from practitioners working for companies of 5 different scales. The vast majority of these companies develop individual software, software as part of a service or software as part of a product. Almost all of them use some mixture of software development methodologies while a significantly smaller number focuses on a single dominant one (used by more than 75%). While still 45% use the waterfall model to some degree, a vast majority uses an agile methodology (58%) or a mix of agile and traditional (54%).

4.2 How Team Practices and Measures correspond with Knowledge Activities

In the questionnaire, practitioners were asked to 'tag' team practices and measures with knowledge activities executed or fostered by them. The result can be seen in figure 3 where the respective bar lengths indicate relevance.

ISBN 978-88-96687-14-7 || ISSN 2280-787X

Part 1: Meetings	Acquire	Codify	Store	Maintain	Transfer	Create
Daily standups						
Recurring longer meetings						
Client Meetings						
Retrospectives						
Part 2: Development	Acquire	Codify	Store	Maintain	Transfer	Create
Design software						
Write Code						
Code Reviews						
Pair Programming						
Test each others' features						
Part 3: Planning	Acquire	Codify	Store	Maintain	Transfer	Create
Collaborative Release Planning						
Epics/Stories Reviews						
Joint estimate software features						
Part 4: Organisation	Acquire	Codify	Store	Maintain	Transfer	Create
Bring in new staff						
Onboarding / Mentoring						
Team assignment changes						
Rotate responsibilities in the team						
Cross-disciplinary task forces						
Part 5: Learning & Documentation	Acquire	Codify	Store	Maintain	Transfer	Create
Contribute to documentation						
Give presentations or training						
Engage in Special Interest Groups						
Attend Conferences						
Part 6: Other	Acquire	Codify	Store	Maintain	Transfer	Create
Informal exchange in the kitchen						
Informal gathering after work						
Part.in Open Source projects						
Part. in cross-disciplinary exchange						

Figure 3: Team practices and Knowledge Activities

Generally, knowledge activities were found to correspond with the 6 parts categorising practices and measures:

- The four kinds of meetings show relevance in maintaining and transferring knowledge while client meetings also help to acquire and create knowledge. Retrospectives even relate significantly to all knowledge activities with the only exception of acquiring knowledge.
- Development exhibits an emphasis on codifying, transferring and creating knowledge.
- Collaborative planning helps in transferring, also to a lesser degree maintaining and creating knowledge.
- Organisational measures leading to permanent or temporary changes in team composition emphasise knowledge transfer, acquisition and creation.
- Learning measures and documentation cover all 6 knowledge activities to different degrees.
- Informal exchange and participation in external activities are helpful for acquiring, transferring and creating knowledge.

ISBN 978-88-96687-14-7 || ISSN 2280-787X

The above results, in particular the findings displayed in figure 3, answer RQ1.

4.3 Most effective Team Practices and Measures

Finally, RQ2 asks how team practices and measures should be combined in order to achieve a mixture suitable for effective knowledge management in software engineering organisations. While figure 3 already gives a good indication of how particular activities can be fostered, some team practices and measures were found to be of particular value because of their high relevance for some specific knowledge activities, like onboarding and mentoring (acquire and transfer knowledge), team membership changes (transfer and create knowledge). Others are attractive because of their versatility, like contributing to documentation, Retrospectives and Pair Programming.

This indicates that for instance Retrospectives should take place because of their correspondence with 5 out of 6 knowledge activities. While they are usually compulsory for Scrum teams this is not the case for many others, hence introducing them and making sure they are always held could be helpful in this context. Pair programming can give a boost when needed, even if there are reasons not to use it by default. Finally, in general, teamwork is beneficial not only for knowledge transfer, ideally while making changes to the team composition from time to time.

Of course, it needs to be understood that just introducing the above will hardly trigger an automatism not requiring any further steering or management: once a tool's potential is understood it should be introduced and then applied in a purposeful way allowing it to fulfil its potential.

5 Discussion and Conclusion

Based on the responses obtained from practitioners using an online questionnaire, team practices and measures were investigated on their relation to six knowledge activities. The clearness with which connections to knowledge activities were identified indicates their applicability as a tool for decision making by project leaders in software engineering.

Most of the practices and measures originate from agile software development methodologies and are widely in use in the software industry, even in organisations not considering themselves as 'agile'. This is significant because it shows that development teams will usually not need to start from scratch or

ISBN 978-88-96687-14-7 || ISSN 2280-787X

sacrifice productivity for the sake of knowledge management. On a side note, this also corresponds with other researchers' findings of agile software development methodologies being more suitable for knowledge management because of their focus on communication and collaboration (Dissanayake *et al.* 2013; Heavin and Adam, 2014; C. Khalil and S. Khalil, 2019).

The findings highlight some particular practices' and measures' versatility in terms of knowledge management indicating their potential to add value to software projects. Hence team leaders do not even need to rely on a large number of them in order to achieve the desired mixture of knowledge activities. An organisation does not have to be 'agile' in order to benefit from this: elements like Daily Scrums or Retrospectives either have their equivalents in traditional methodologies or can be easily integrated.

Software project work is frequently subject to enormous time pressure. Teams dealing with knowledge efficiently have an advantage here, while unfortunately systematic knowledge management usually does not promise immediate return on investment. This is why blending well with existing work procedures and not coming at an additional cost are critical requirements. Relying on commonly-used procedures with some direct value to the software development process while understanding their potential in knowledge management fills this gap.

These outcomes may already sound familiar to experienced practitioners, however, increased attention, a more substantiated understanding of processes and measures and their relationship to knowledge-related aspects can contribute significantly to more sustainable team development with little or no overhead.

Future research will continue exploring the application of knowledge management in software engineering by investigating the here presented team practices and measures on their importance and software teams' proficiency in applying them and how this relates to organisations' characteristics like scale or preferred development methodology.

References

Abdur, R., T. Bhuiyan, and R. Ahmed, (2015), "Knowledge management in distributed agile software development projects", in Proceedings of the AIKM 2014 - Artificial Intelligence for Knowledge Management, Vol. 469, pp. 107–131.

Andersen, A., (1996), The knowledge management tool: External benchmarking version, American Productivity and Quality Center, Houston.

Andreeva, T. and I. Ikhilchik, (2011), "Applicability of the SECI Model of knowledge creation in Russian cultural context: Theoretical analysis", Scholarly Articles at Russian Management Journal, Vol. 18, No. 1, doi: 10.1002/kpm.351.

Aurum, A., F Daneshgar, and J. Ward, (2008), "Investigating knowledge management practices in software development organisations - an Australian experience", Information and Software Technology, Vol. 50, No. 6, pp. 511–533, doi: 10.1016/j.infsof.2007.05.005.

Basri, S. and R. O'Connor, (2011), "The impact of software development team dynamics on the knowledge management process(S).", in Proceedings of the 23rd International Conference on Software Engineering and Knowledge Engineering, pp. 339–342.

Bierly, P., E. Kessler, and E. Christensen, (2000), "Organizational learning, knowledge and wisdom", Journal of Organizational Change Management, Vol. 13, No. 6, pp. 595–618.

Bjørnson, F. O. and T. Dingsøyr, (2008), "Knowledge management in software engineering: A systematic review of studied concepts, findings and research methods used", Information and Software Technology, Vol. 50, No. 11, pp. 1055–1068.

Bryman, A. and E. Bell, (2007), Business research methods, 2nd ed., Oxford University Press, Oxford.

Collins, Harry, (2010), "Tacit & explicit knowledge", Bibliovault OAI Repository, the University of Chicago Press, doi: 10.7208/chicago/9780226113821.001.0001.

Costa, Vítor and Samuel Monteiro, (2017), "Key knowledge management processes for innovation: A systematic literature review", VINE Journal of Information and Knowledge Management Systems, Vol. 46, No. 3, pp. 386–410, doi: 10.1108/VJIKMS-02-2015-0017.

Creswell, J.W., (2014), Research design: Qualitative, quantitative, and mixed methods approaches, 4th ed., Sage Publications, Los Angeles.

Dingsøyr, T. and R. Conradi, (2002), "A survey of case studies of the use of knowledge management in software engineering", International Journal of Software Engineering and Knowledge Engineering, Vol. 12, No. 4, pp. 391–414, doi: 10.1142/S0218194002000962.

Dissanayake, I., Ramakrishna Dantu, and S. Nerur, (2013), "Knowledge management in software development: The case of agile software", in Proceedings of the 19th Americas conference on information systems, AMCIS 2013 - hyperconnected world: anything, anywhere, anytime, Vol. 3, pp. 2298–2304.

Earl, M., (2001), "Knowledge management strategies: Toward a taxonomy", Journal of Management Information Systems, Vol. 18, No. 1, pp. 215–242.

Eisenhardt, K.M. and M.E. Graebner, (2007), "Theory building from cases: Opportunities and challenges", The Academy of Management Journal, Vol. 50, No. 1, pp. 25–32, doi: 10.5465/AMJ.2007.24160888.

Grover, V. and T. H. Davenport, (2017), "General perspectives on knowledge management: Fostering a research agenda", Journal of Management Information Systems, Vol. 18, No. 1, pp. 5–21.

Heavin, C. and F. Adam, (2012), "Characterising the knowledge approach of a firm: An investigation of knowledge activities in five software SMEs", Electronic Journal of Knowledge Management, Vol. 10, No. 11, pp. 48–63.

Heavin, C. and F. Adam, (2014), "From knowledge activities to knowledge scenarios: Cases in five irish software SMEs", International Journal of Management and Enterprise Development, Vol. 13, No. 1, pp. 37–61, doi: 10.1504/IJMED.2014.059852.

Hegde, R. and G. Walia, (2014), "How to enhance the creativity of software developers: A systematic literature review", in Proceedings of the 26th International Conference on Software Engineering and Knowledge Engineering.

Holsapple, C. and K. D. Joshi, (2004), "A knowledge management ontology", in Holsapple, C., Handbook on Knowledge Management 1: Knowledge Matters, Springer, Berlin, Heidelberg, pp. 89–124.

Holste, J. S. and D. Fields, (2010), "Trust and tacit knowledge sharing and use", Journal of Knowledge Management, Vol. 14, No. 1, pp. 128–140.

Ichijo, K. and I. Nonaka, (2008), Knowledge creation and management: New challenges for managers, Oxford University Press, New York.

Jetter, A. et al. (2006), Knowledge integration: The practice of knowledge management in small and medium enterprises, Physica, Heidelberg, pp. 1–203.

Kahrens, M. and D. H. Früauff, (2018), "Critical evaluation of nonaka's SECI model", in Syed, J. et al. The Palgrave Handbook of Knowledge Management, Springer International Publishing, Cham, pp. 53–83.

Kavitha, R. and M. S. Irfan, (2011), "A knowledge management framework for agile software development teams", doi: 10.1109/PACC.2011.5978877.

Khalil, C. and S. Khalil, (2019), "Exploring knowledge management in agile software development organizations", International Entrepreneurship and Management Journal, doi: 10.1007/s11365-019- 00582-9.

Kraaijenbrink, J., D. Faran, and A. Hauptman, (2006), "Knowledge integration by SMEs — framework", in, Knowledge Integration, Springer, Berlin, Heidelberg, chap. 2, pp. 17–28, doi: 10.1007/3-7908-1681-7_2.

Lee, C. S. and R. S. Kelkar, (2013), "ICT and knowledge management: Perspectives from the SECI model", The Electronic Library, Vol. 31, No. 2, pp. 226–243.

Lin, D., (n.d.), "Wissensmanagement Reloaded - ein Ordnungsrahmen für den systemischen Umgang mit Wissen im Enterprise 2.0", ger, MA Thesis, Technische Universität Dresden, 2010, MA thesis, Kiel, Hamburg, url: http://hdl.handle.net/10419/36700.

Lubit, R., (2001), "Tacit knowledge and knowledge management: The keys to sustainable competitive advantage", Organizational Dynamics, Vol. 29, No. 3, pp. 164–178.

Nezafati, N., A. Afrazeh, and S. M. J. Jalali, (2009), "A dynamic model for measuring knowledge level of organizations based on Nonaka and Takeuchi model (SECI)", Scientific Research and Essays, pp. 531–542.

Nonaka, I., (1991), "The knowledge-creating company", Harvard Business Review, Accessed on 17.01.2021.

Nonaka, I., (1994), "A dynamic theory of organizational knowledge creation", Organization Science, Vol. 5, No. 1, pp. 14–37.

Nonaka, I. and N. Konno, (1998), "The concept of "ba": Building a foundation for knowledge creation", California Management Review, Vol. 40, No. 3, pp. 40–54.

Nonaka, I. and H. Takeuchi, (1995), The knowledge-creating company: How Japanese companies create the dynamics of innovation, Oxford University Press, New York.

Nonaka, I. and R. Toyama, (2003), "The knowledge-creating theory revisited: Knowledge creation as a synthesizing process", Knowledge Management Research & Practice, Vol. 1, No. 1, pp. 2–10.

Nonaka, I. and R. Toyama, (2005), "The theory of the knowledge-creating firm: Subjectivity, objectivity and synthesis", Industrial and Corporate Change, Vol. 14, No. 3, pp. 419–436.

Nonaka, I., R. Toyama, and P. Byosière, (2003), "A theory of organizational knowledge creation: Understanding the dynamic process of creating knowledge", in Handbook of Organizational Learning and Knowledge, Oxford University Press, Oxford, pp. 491–517.

Nonaka, I. and G. von Krogh, (2009), "Tacit knowledge and knowledge conversion: Controversy and advancement in organizational knowledge creation theory", Organization Science, Vol. 20, No. 3, pp. 635–652.

Pentland, B. T., (1995), "Information systems and organizational learning: The social epistemology of organizational knowledge systems", Accounting, Management and Information Technologies, Vol. 5, No. 1, Special Issue Information Technology and Organizational Learning, pp. 1–21, issn: 0959-8022, doi: 10.1016/0959-8022(95)90011-X.

Rus, I. and M. Lindvall, (2002), "Knowledge management in software engineering", IEEE Software, Vol. 19, No. 3, pp. 26–38.

Saifi, S., S. Dillon, and R. Mcqueen, (2016), "The relationship between management support and knowledge sharing: An exploratory study of manufacturing firms: Management support and knowledge sharing", Knowledge and Process Management, Vol. 23, No. 2, pp. 124–135, doi: 10.1002/kpm.1506.

Sandhawalia, B. and D. Dalcher, (2010), "Knowledge flows in software projects: An empirical investigation", Knowledge and Process Management, Vol. 17, No. 4, pp. 205–220, doi: 10.1002/kpm.357.

Sapir, A., I. Drori, and S. Ellis, (2016), "The practices of knowledge creation: Collaboration between peripheral and core occupational communities", European Management Review, Vol. 13, No. 1, pp. 19–36, doi: 10.1111/emre.12064.

Scharmer, C. O., (2001), "Self-transcending knowledge: Sensing and organizing around emerging opportunities", Journal of Knowledge Management, Vol. 5, No. 2, pp. 137–151.

Schreyögg, G. and D. Geiger, (2003), "Kann die Wissensspirale Grundlage des Wissensmanagements sein?", Diskussionsbeiträge des Instituts für Management, Nr. 20, Berlin.

Schreyögg, G. and D. Geiger, (2004), "Kann man implizites in explizites Wissen konvertieren? Die Wissensspirale auf dem Prüfstand", in Frank, U., Wissenschaftstheorie in Ökonomie und Wirtschaftsinformatik, pp. 269–288.

Shongwe, M. M., (2017), "Knowledge management in small software development organisations: A South African perspective", SA Journal of Information Management, Vol. 19, No. 1.

Stenmark, D., (2000), "Leveraging tacit organizational knowledge", Journal of Management Information Systems, Vol. 17, No. 3, pp. 9–24.

Sungkur, R. and M. Ramasawmy, (2014), "Knowledge4Scrum, a novel knowledge management tool for agile distributed teams", VINE, Vol. 44, No. 3, pp. 394–419, doi: 10.1108/VINE-12-2013-0068.

Sutherland, J. and K. Schwaber, (2010), "Scrum guide", Retrieved 06-02-2010 from scrumguides.org, url: https://scrumguides.org/scrum-guide.html.

Tee, M. Y. and S. S. Lee, (2013), "Advancing understanding using Nonaka's model of knowledge creation and problem-based learning", International Journal of Computer-Supported Collaborative Learning, Vol. 8, No. 3, pp. 313–331.

Teece, D., (2010), "Technological innovation and the theory of the firm: The role of enterprise-level knowledge, complementarities, and (dynamic) capabilities", Handbook of the Economics of Innovation, Vol. 1, pp. 679–730.

van Manen, M., (2007), "Phenomenology of practice", Phenomenology & Practice, Vol. 1, No. 1, pp. 11–30, doi: 10.29173/pandpr19803.

van Manen, M., (2016), Researching lived experience: Human science for an action sensitive pedagogy, 2nd ed., Routledge, New York.

Vasanthapriyan, S., J. Tian, and J. Xiang, (2015), "A survey on knowledge management in software engineering", in Proceedings of the 2015 IEEE International Conference on Software Quality, Reliability and Security - Companion, pp. 237–244.

Viana, D. et al. (2013), "A qualitative study about the life cycle of lessons learned", in Proceedings of the 2013 6th International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE 2013, pp. 73–76.

von Krogh, G., K. Ichijo, and I. Nonaka, (2000), Enabling knowledge creation: How to unlock the mystery of tacit knowledge and release the power of innovation, Oxford University Press, New York.

von Krogh, G., I. Nonaka, and L. Rechsteiner, (2012), "Leadership in organizational knowledge creation: A review and framework", Journal of Management Studies, Vol. 49, No. 1, pp. 240–277.

Ward, J. and A. Aurum, (2004), "Knowledge management in software engineering - describing the process", in Proceedings of the 2004 Australian Software Engineering Conference, pp. 137–146.