Knowledge Management in Distributed Agile Software Development Projects

Mohammad Razzak, Touhid Bhuiyan, Rajib Ahmed

Noname manuscript No.
(will be inserted by the editor)

Knowledge Management in Distributed Agile Software Development Projects: Techniques, Strategies and Challenges

Mohammad Abdur Razzak · Touhid Bhuiyan · Rajib Ahmed

Received: date / Accepted: date

Abstract Knowledge management (KM) is essential for success in global software development. Software organizations are now managing knowledge in innovative ways to increase productivity. In agile software development, collaboration and coordination depend on the communication, which is the key to success. To maintain effective collaboration and coordination in distributed agile projects, practitioners need to adopt different types of knowledge sharing techniques and strategies. There are also few studies that focus on knowledge sharing in distributed agile projects. This research investigates the knowledge sharing techniques and strategies applied by the practitioners in distributed agile projects. In addition to that, challenges faced by the practitioners during knowledge sharing in distributed agile projects are also identified and discussed.

Keywords Knowledge management · knowledge sharing · distributed · agile · global software development

1 Introduction

Software engineering is a knowledge intensive area. This forces software organizations to manage their knowledge and later use it in smarter, innovative ways to solve problems [Schneider, 2009]. It helps software development organizations to acquire and maintain a competitive advantage. KM is crucial for success in global software development [Richardson et al., 2009].

Mohammad Abdur Razzak · Touhid Bhuiyan
Department of Software Engineering
Daffodil International University-Bangladesh
E-mail: (arazzak, t.bhuiyan)@diu.edu.bd

Rajib Ahmed
Exertis Ztorm, Stockholm, Sweden
E-mail: l.rajibahmed@gmail.com

Global software development can be described as "software work which is attempted in different geographical locations across the national boundaries in a coordinated fashion, to involve synchronous and asynchronous interaction" [Sahay et al., 2003]. Software developers work with knowledge and are dependent on each other's work. In global software development this synchronization is dependent on KM. Some studies have identified that knowledge sharing is difficult in distributed agile project due to the lack of face-to-face communication between team members [Boden and Avram, 2009, Holz and Maurer, 2003]. In the agile software development collaboration and coordination depends on communication, which is crucial to successful software development [Šmite et al., 2010]. One of the major objectives of KM is to improve productivity through effective knowledge sharing and transfer [Kavitha and Ahmed, 2011]. So, the success of agile projects relies on effective knowledge sharing among teams.

This research focuses on exploring knowledge sharing in distributed agile projects. More specifically, this research attempts to identify knowledge sharing techniques, strategies and practices that take place between locally and globally distributed agile teams, and the challenges faced by the practitioners in a distributed agile environment. We are driven by the following research questions:

RQ1: How do team members contribute to knowledge creation in a distributed agile project?

RQ2: How do team members share knowledge in a distributed agile project?

RQ3: What are the challenges faced by the practitioners when sharing knowledge in a distributed agile project?

2 Related Work

Software development is considered to be a complex, knowledge intensive and rapidly changing activity, where a number of individuals, teams and organizations are involved infulfilling common goals, interests and responsibilities [Curtis et al., 1988, Nicholson and Sahay, 2004]. Technological and strategic knowledge helps developers to communicate; so it is essential to keep the knowledge stored in the organization for the future reuse. Davenport and Prusak [Davenport and Prusak, 2000] define it as "a method that simplifies the process of sharing, distributing, creating, capturing and understanding the company's knowledge". As the size of the organization grows rapidly, it becomes harder to find out where the knowledge resides. Research shows that if the companies manage their knowledge in a better way, they can increase quality, and decrease the time and development costs[Rus et al., 2002]. To improve the organizational performance, it is important to manage knowledge in a structured way which will help to convey the right knowledge to the right people at the right time. O'Dell and Grayson [O'Dell and Grayson, 1998] discussed that, knowledge management is not a vital methodology; it is a framework, a management mind-set which is based on past experiences and the creation of new wheels for exchanging knowledge.

To foster dynamic knowledge sharing, improve productivity and coordination in software development teams, agile approaches were introduced. Agile teams share knowledge through several practices [Chau and Maurer, 2004]: pair programming, release and sprint planning, customer collaboration, cross-functional teams, daily scrum meetings and project retrospectives. But, the authors [Chau and Maurer, 2004] argue that, these practices are team-oriented and rely on face-to-face interaction between team members. These practices do not facilitate knowledge sharing in distributed agile teams but are effective for collocated and small teams. In one study Dorairaj *et al.* [Dorairaj et al., 2012] reported that in distributed agile project, team members practice sprint planning, daily scrums, sprint reviews and project retrospective meetings. Distributed agile team members share knowledge through effective use of knowledge management tools like *Wiki*, pair-programming and video-conferencing.

Michael Earl [Earl, 2001] has classified knowledge management into three categories: technocratic, economic and behavioral. Earl also divided these three categories into seven schools, Technocratic: *Systems, Cartographic and Engineering*, Economic: *Commercial* and Behavioral: *Organizational, Spatial and Strategic*. Both *codification*[Hansen et al., 2005] strategy and *systems school* practice depend on the technology which applies Nonaka's [Nonaka, 1994] *externalization* conversion technique to convert tacit knowledge into explicit knowledge. Research shows that the technocratic school is closely related with traditional software development and those who are developing software through traditional approaches they are probably benefiting from the technocratic schools [Dingsøyr et al., 2009]. On the other hand, behavioral schools are more related with the agile approaches and agile teams are more benefit more from the behavioral school. A survey in traditional and agile companies shows that agile companies seem to be more satisfied with their knowledge management approaches compared to traditional companies [Bjørnson and Dingsøyr, 2009]. In agile software development, knowledge sharing happens through the interaction. Developers share knowledge by working together and through close interaction with customers; and more specifically, pair programming, extreme programming, daily scrum meetings, and sprint retrospectives in Scrum. In traditional software development, knowledge management relied primarily on explicit knowledge but in the agile software development KM relies on tacit knowledge [Nerur et al., 2005]. In agile software development, information radiators and collocating teams are related with the spatial school [Bjørnson and Dingsøyr, 2009].

In traditional software development, knowledge stored explicitly in the documentation, but in the agile development methodology the knowledge is tacit [Kavitha and Ahmed, 2011]. Extracting tacit knowledge to create explicit knowledge is one of the greatest challenges of knowledge organization [Nonaka and Konno, 1998]. Due to the absence of explicit knowledge in the agile software development, experts need to spend much of their time on repeatedly answering the same questions, knowledge is lost when experienced developers leave project, there is less support for re-usability and there is less contribution to organizational knowledge [Kavitha and Ahmed, 2011]. In the agile collocated development, informal communication is the key enabler for knowledge sharing but when an agile project is distributed, informal communication and knowledge sharing is a challenge due to low communication bandwidth as well as social and cultural distance [Maalej and Happel, 2008]. Due to spatial, temporal and cultural factors, communication also becomes aggravated in the distributed settings [Hildenbrand et al., 2008]. Several studies [Holz and Maurer, 2003, Boden et al., 2009] also point out that, knowledge sharing in the distributed agile projects is difficult due the challenges in communication, especially face-to-face interaction between team members in different geographical locations. To address these problems, we investigated how shared knowledge creation and transfer activities performed in the distributed agile projects. Along with that, we also investigated what challenges are faced by the practitioners when sharing knowledge among globally distributed agile team members.

3 Method & Data Analysis

Because this research addresses an issue *"How can we retain the benefits that agile practices provide with respect to KM in distributed agile projects"* which is rather under-investigated, this study takes an explorative approach. Exploratory research helps to find out what is happening, seeking new insights and gathering ideas [Marczyk et al., 2010, Runeson and Höst, 2009]. In some qualitative research, data collected through observation or interviews are exploratory in nature. So, extensive interviews are helpful to handle this type of situation [Sekaran, 2006]. This type of exploratory research was also helpful in achieving our goal through analysis of similarities and differences among the cases [Creswell, 2008]. The primary focus of this study was to discover the knowledge sharing activities in distributed agile projects in order to identify techniques, strategies and challenges.

3.1 Sampling

The selection criteria for these interviewees were based on the kind of company they work at, the experience of the company in distributed agile development (more than 2 years), interviewee role in the distributed team as well as in the company, project duration and project distribution. The participants of this research were project managers, team leaders, software architects, line managers, senior software developers, system developers and Scrum masters in different countries involved in distributed agile projects, located in different countries i.e. Sweden, Norway, Germany, Ukraine, China, India, Bangladesh, USA, and Latvia. To get the rounded perspective of this research phenomenon we included different roles from the agile team.

3.2 Data Collection

There are three types of interview techniques namely structured, semi-structured and unstructured [Flick, 2009]. Due to the qualitative nature of this study we used semi-structured interviews for conducting a series of interviews in software industries involved in distributed agile projects. According to Robson [Robson, 2002], an in-depth semi-structured interview is helpful in *finding out what is happening and seeking new insights*. *Seventeen* semi-structured interviews were conducted from *seven* teams in order to identify how practitioners are creating, storing and sharing knowledge related to software development among geographically distributed agile teams. These semi-structured interviews were a combination of both open and focused questions. It helps both interviewer and interviewee to discuss a topic in more details. Before the interviews started, we discussed about overall goal of this research to interviewee. The interview questions were *descriptive* and with the base questions there were follow up questions asked based on the discussion. We were concern about some key terms: *shared knowledge creation, knowledge transfer, strategies and challenges* which later helped us for data analysis and those terms which also evolve with interview questions. We conducted seventeen semi-structured interviews from six different companies. The selected companies are involved with software product development, have different organizational settings and structure and are located in different countries. The duration of these interviews averaged 60 minutes and the interview sessions were tape recorded. Among the seventeen semi-structured interviews, nine were conducted through Skype and eight were face-to-face, depending on distance between interviewer and interviewee.

3.3 Analysis and Synthesis

In qualitative research, data analysis is the most difficult and crucial aspect due to raw data sets. According to Basit [Basit, 2003], raw data can not help the reader to understand the social world or the participants view unless such data is systematically analyzed. To organize collected data we adopted *thematic analysis* [Braun and Clarke, 2006] technique during analysis. Thematic analysis is used to identify, analyze and report patterns or themes within data. It minimally organizes and describes data set in detail. In thematic analysis a theme captures data with relation to research questions and represents them in a pattern within the data set [Braun and Clarke, 2006]. This analysis is performed through a process which maintain six phases to establish meaningful patterns of the data set. Braun and Clarke [Braun and Clarke, 2006] provides an outline through the six phases of analysis. These phases are: familiarization with data, generating initial codes, searching for themes among code, reviewing themes, defining and naming themes, and producing the final report.

Table 1 Overview of the stuided distributed Agile projects

Projects	*Project Distribution*	*Team Size*	*Team Types*	*Agile Position/Roles*
Alpha	Sweden-Germany	6-7*	Dispersed	Team Leader Developer
Beta	Norway-Bangladesh	5-6*	Dispersed	Project Manager Developer
Gamma	USA-Bangladesh	12-16**	Distributed	Head of Engineering Senior Developer Developer
Delta	Sweden-Bangladesh	16-18**	Dispersed	Software Architect Developer
Epsilon	Latvia- Ukraine	11-15**	Distributed	Project Manager Developer
Zeta	Sweden-China	26-35***	Distributed	Line Manager Software Developers System Developer
Eta	Sweden-India	45-55***	Hybrid	System Developer Scrum Master

*In Table 1 *,**,*** indicates Small, Medium and Large scale teams respectively*

In the *first* stage, we transcribed all the collected interview data into written form in order to conduct a thematic analysis. It helped us to identify possible themes, patterns and to develop potential codes [Guest et al., 2011]. *Second* phase started with initial codes from the extracted data. There are different types of Coding techniques suggested in different studies such as; *open, axial, selective, descriptive/topic and pattern or analytic*[Miles and Huberman, 1994, Punch, 2009, Shull et al., 2007]. In our case, we applied open coding technique and went through all transcribed textual data by highlighting sections of the selected codes. That also helped us to relate coded data with research theme and research questions. In *third* stage, we analyzed broader level of theme rather than codes that helps to sort different codes into potential themes [Braun and Clarke, 2006].As Braun and Clarke suggested coding as many potential themes/patterns as possible because initially some themes seems to be insignificant, but later they may be important in the analysis process. Later, mind mapping tools were used to represent them into theme-piles. This stage gave us a sense of the significance of individual themes. Stage *four* is reviewing themes. In this stage we identified irrelevant (not enough or diverse) data with relate to different themes and broken down into separate themes. After refining all themes we identified "essence" of each theme and different aspects of the data each theme captures in stage *five*. At the end, in stage *six*, we provided extract data with relate to research questions and present some dialog that connected with different themes in support of results and discussion sections.

4 Validity Threats

To handle validity threats it is important to identify all possible factors that might affect the accuracy or dependability of the results.

4.1 Internal Validity

Internal validity for qualitative research mostly relates to the researchers biasness and interpretation of data [Bleijenbergh et al., 2011].For finding a similar knowledge level for our interviewees, we went on interviewee profiles on *Linkedin* and their years of experience. After finding out the basic information, the interviewer sent a formal email to the interviewee with an invitation letter about becoming involved with this research. To mitigate the threat of following our own bias, interview questions were designed to have a majority of open ended questions. Every interview started with a similar introduction and some clarification questions. Then the recorded interview was transcribed immediately afterward to reduce the risk of missing some information. Furthermore, researchers sent an interview report to the interviewee in order to check whether interview data was correctly transcribed and to confirm the content indicated participants thoughts, viewpoints, feelings and experiences. In qualitative research it is important to understand the interviewee's inner meaning words. To maintain reliability during data analysis we used a thematic, qualitative data analysis technique, that helped to identify, analyze and report themes within data. The extracted data from the transcribed data was checked twice for any discrepancy by two researchers.

4.2 External Validity

External validity threat is more applicable to research that are quantitative and which tries to generalize outcome of the research. However, our findings can be generalized only for the agile software development teams which are involved in the development of a shared project from distributed locations.

5 Results

In this section, we describe different findings (techniques, strategies and challenges) from the *seven* cases, that promote effective knowledge creation and sharing activities in distributed agile projects.

5.1 Knowledge Creation: Locally and Globally

We have found that distributed agile project teams practice different types of techniques for both local and global shared knowledge creation. *Pair programming*, *customer collaboration*, *Scrum/Kanban* boards and *community of practice* are explicit practices used by the teams to perform both local and global shared knowledge (see in Table 2).

5.1.1 Pair Programming

Pair programming is used for both local and global knowledge creation. From the series of semi-structured interviews we have found that both local and remote team members work together in one workstation to solve specific problems. They help each other to share their thoughts and create knowledge through discussion. In two cases, we have found that teams do not perform pair programming for shared knowledge creation among remote team members. In the Epsilon(ε) project, all development team members are in one site, and for that reason they do not need to perform pair programming for global shared knowledge creation. However, Zeta(ζ) project is a collaboration with a Chinese team on the same product, but the development team does not have any dependency. The development teams working on different modules and later core developers merge all modules together for specific release. But the local teams in Zeta (ζ) project perform pair programming.

5.1.2 Pre-planning game/customer collaboration

In the de- velopment cycle the customer has an important role. Customer collaboration helps teams to build up technical-business col- laboration on a project and also helps to set the direction of the project. In agile software development customers are always involved with the development teams by providing project requirements and performing acceptance testing. Through customer collaboration agile teams participate in creating local knowledge. Evidence was also found from different cases that customers are also involved with the remote development teams to create knowledge through continuous discussion and features feedback. We have also found that customers are involved in issue tracking systems, which helps both the project manager and the developers towards early iteration. In two cases (δandε), we found that customer collaboration performed only in the local sites for shared knowledge creation.

5.1.3 Scrum/Kanban boards

The are two types of boards used by the office to create knowledge and common understanding. A *Scrum board* is used for teams that plan their work in sprint. A *Kanban board* is used to manage and construct team work in progress. In table 2 it is shown that, teams use *Scrum* and *Kanban* boards for shared knowledge creation among both local and globally distributed team members. In two cases (γandη), we found that teams are using boards both locally and globally. In Gamma (γ) project, the remote team has a sub-Scrum board, which is replica of the main Scrum board. Along with that, the local team (in γproject) upload pictures of the main Scrum board into a repository every day. But in Eta (η) project, teams use a visual Scrum board to perform shared knowledge creation among distributed team members.

5.1.4 Innovation boards

Most innovative ideas are kept in the human mind as tacit knowledge. Due to continuous work loads, sometimes it is impossible to have a discussion with a team member, or other knowledgeable person. So, rather than talking with someone, people share their ideas through the innovation board in an explicit way. In one interview the researchers found that teams are using innovation boards to share their ideas with both collocated and remote team members.

5.1.5 Workshop/Seminars

Weekly or monthly workshops and seminars are arranged through collaboration between business teams, technical teams and customers, in order to share knowledge about projects and the latest technologies. This

Table 2 Knowledge creation techniques: Locally and Globally

Techniques	α	β	γ	δ	ε	ζ	η
Pair programming	L,G	L,G	L,G	L,G	—	L	L,G
Customer collaboration	L,G	—	L,G	L	L	L,G	L,G
Scrum/Kanban boards	L	—	L,G	L	—	L	L,G
Innovation boards	—	—	—	—	—	—	L,G
Workshops/Seminars	—	—	—	—	—	L	L
Community of practice	—	—	—	L,G	L,G	L,G	L,G
Technical presentation	—	—	L	—	—	L	L
Technical forum	—	—	—	—	—	L,G	L,G

**In Table 2, L indicates Locally, G— Globally and "—" not in practice*
Dispersed teams- α, β, δ; Distributed teams- γ, ε, ζ; Hybrid team- η

kind of workshop facilitates common understanding and communication between different team members. Workshops also help to facilitate tacit knowledge sharing through *socialization.* In the studied cases, we only observed *large-scale* teams practicing these techniques locally, to create shared knowledge. Later, the theme of the workshops/seminars was shared among remote team members through repositories.

5.1.6 Community of practice

To succeed in agile projects, learning is an important asset for agile teams. Agile teams practice two modes of learning: *peer learning* and *community learning.* In *peer learning*, team members start learning through interacting and collaborating with team members. *Community learning* is accessing and conceiving information that is available in knowledge archives or in discussion forums. We found community of practice within different projects, where it performed to share knowledge creation among local and remote team members.

We also found that to create shared knowledge, teams perform *technical presentations.* But these activities are only performed in the local site and later slides or documents are shared among remote team members. Technical forums are also in practice to perform shared knowledge creation between local and remote team members.

5.2 Knowledge Sharing: Locally and Globally

Knowledge exchange is always challenging in distributed agile teams due to a lack of face-to-face interaction among team members. Practitioners and researchers are trying to mitigate these challenges by initiating different kinds of techniques and tools. From the studied cases we observed that practitioners maintain different types of tools and techniques to share knowledge among globally distributed teams. Based on the findings, these knowledge sharing techniques are listed in Table 3.

All studied projects are concerned with using repositories to share knowledge between local and remote team members. Most of the task and product related knowledge is kept in the repositories, which are easy to access by the remote team members. Different teams also depend on daily scrum, weekly sprints status, discussion forums, online conferences and common chat rooms to share knowledge between local and remote team members. Electronic boards are helpful for sharing knowledge across remote teams. Only one case was found where knowledge is shared between both collocated and distributed teams through electronic boards.

5.2.1 Repositories

To share knowledge among distributed sites, local teams used different types of repositories like *Wiki, JIRA, Redmine, Confluence and GitHub etc.* These types of repos- itories provide efficient mechanisms to access codified knowledge. From the gathered data it is evident that (see in Table 3) practitioners are most dependent on repositories to share knowledge among both local and distributed team members.

Wiki, according to Ulrike Cress [Cress and Kimmerle, 2008], provides new opportu- nities to learn and use collaborative knowledge building and sharing, through social interaction and individual learning. In different cases we found that *wikis* are helpful for starting new threads and discussing issues with other team members. It is also helpful for new team members as it (the *wiki*) provides detailed information about features, documents and so forth.

Project and Issue tracking: Nowadays, almost all medium and large distributed or dispersed agile teams are using *JIRA/Redmine* to track issues, bugs, tasks, deadlines, codes and hours. As collaboration and content sharing tools practitioners used *Confluence* to share docs, files, ideas, specifications, diagrams and mockups. During the interview one project leader said,

...Most of the time we share tacit knowledge between both local and global teams. After that, the in-

Table 3 Knowledge sharing techniques among different sites

Techniques	α	β	γ	δ	ε	ζ	η
Repositories	L,G	L,G	L,G	L,G	L,G	L,G	L,G
Pair programming	L,G	L,G	L,G	L,G	—	L	L,G
Version control	—	—	—	—	L,G	—	—
Screen sharing	G	G	G	G	—	—	—
Daily scrum	L,G	—	L,G	—	L,G	L	L,G
Weekly sprint status	L,G	—	L,G	L,G	G	L	L,G
Common chat room	—	—	L,G	L,G	L,G	L,G	L,G
Technical forum	—	—	—	—	—	L,G	L,G
Discussion forum	—	—	L,G	L,G	—	L,G	L,G
Electronic board	—	—	—	—	—	—	L,G
Online conference	G	—	G	G	G	L,G	L,G
Rotation/Visit	—	—	—	—	G	G	G

**In Table 3, L indicates Locally, G— Globally and "—" not in practice*
*Dispersed teams- **α, β, δ**; Distributed teams- **γ, ε, ζ**; Hybrid team- **η***

formation is converted through Redmine to make it explicit... **Project Leader - Alpha project**.

It is also evident that, in one case we found local teams up- load *Scrum* board pictures, slides and workshop information in the repositories, which is codified and easy to access by the remote team members.

5.2.2 Pair programming

Pair programming plays an impor- tant role in creating and sharing developers knowledge in both locally and globally distributed project. In pair programming, two developers work together at one computer with a common goal [Palmieri, 2002]. In the studied projects, we found teams are using pair programming techniques to share knowledge among remote team members. Team members use Skype to share screens among remote team members. Along with that we also found that teams use *TeamViewer* and *VPN* services to share the same computer screen with remote team members, in order to perform pair programming.

5.2.3 Daily Scrum/weekly sprints status/online conferences

Scrum meetings are a source for sharing project progress information among team members. Usually a Scrum standup meeting is held in collocation. From the gathered data we found that distributed teams practice Scrum standup meetings with Internet Relay Chat (IRC), Skype or other group chatting software. Through daily Scrum weekly sprints status/online conferences local teams share knowledge with distributed team members. In one case (ζ), we found that due to less dependency, the development team does not need to perform Scrum meetings/weekly sprint status. But team (ζ) maintain online conferences in order to share knowledge among remote team members (if needed). However, apart from Beta(β) project, other projects explicitly maintain *online conferences* globally for knowledge sharing among distributed team members.

5.2.4 Common chat room

Common chat rooms are useful for exchanging knowledge among distributed teams. From the empirical findings we observed that for faster and quicker communication among distributed team members, *medium-* and *large-scale* teams maintain common chat rooms.

In one case, a software architect said that *...the Sprint management system handles all task related knowledge but for the domain related knowledge sharing we maintain a common chat room, which helps us to resolve specific problems within a short time* **- Software Architect, Delta project**

But, in another case we found that, it is not an efficient way to communicate among distributed teams due to language barriers, common understanding, technological factors and so forth. Frequently misunderstandings occur and things go wrong. To mitigate these types of problem, practitioners also suggested different types of mitigation techniques.

5.2.5 Technical forum

The idea behind a technical forum is *learning through sharing knowledge.* Technical forums are like communities of practice which create a network between technical team members. They are self-organizing groups that consist of individuals who share information, experience and technical skill on a specialized discipline [McDermott, 1999]. Technical forums assist distributed teams in quick problem solving and reduce development time since team members do not get stuck on recurring issues. Building trust between team members in the distributed environment is challenging; so knowledge sharing through technical forums can build trust between developers. Technical forums help to create and share both local and distributed knowledge. We have found that large-scale team members practice *technical forum* techniques to share knowledge among remote team members.

5.2.6 Electronic board

The office boards hold a lot of knowledge which is difficult to share among distributed teams. The interviews revealed that practitioners are using electronic boards to share and access knowledge both locally and globally. Electronic boards hold the tasks list to perform , latest information and along with that necessary technical and business information are regularly updated in a wiki. Electronic boards help to decrease the communication overhead.

In one case an interviewee said *...I am not satisfied with the current tools; Its tough to describe designs to new team members. Visual aids are helpful during discussions* **- Project Manager, Beta project.**

5.2.7 Rotation/Visits

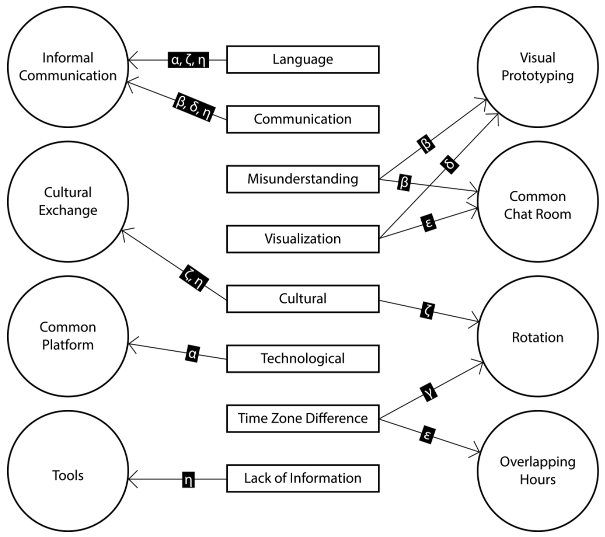
The primary intention of team member rotation between different sites is knowledge sharing. Due to frequent face-to-face interaction with product owners, onsite team members get more business and domain related information than offshore team members [Sureshchandra and Shrinivasavadhani, 2008]. A lack of face-to-face meetings and poor socialization also causes a lack of trust among distributed team members [Moe and Šmite, 2008]. Rotation between on-site and distributed team members promotes the sharing of business and domain related knowledge across the teams. From the data gathered we found that, both *distributed* and *hybrid* teams visit remote sites and rotate team members to increase the trust and communication bandwidth between team members. But the studied *dispersed* teams never visit and rotate with remote team members.

One of the distributed teams line managers said, *Visits to remote sites are highly costly. So, we rotate team members and mostly, the duration of the rotation between team members is 3-6 months.* **- Line Manager, Zeta project**

We have also found that teams practice *version control, screen sharing and discussion forums* to maintain knowledge sharing among both local and remote team members.

5.3 Challenges faced by practitioners during knowledge sharing among distributed teams

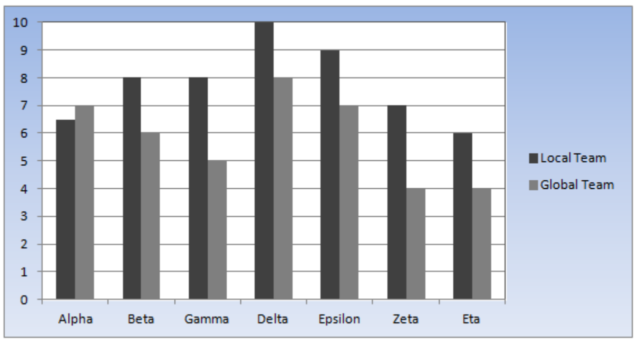
In agile software development most of the knowledge is tacit, which resides in the human mind rather than documentation. This codified tacit knowledge is shared among between locally and globally distributed team members through tools. The knowledge sharing approach varies between team members due to experience levels. The types of problem this leads to are search availability and difficulty finding the right knowledge at the right time. We have also found that to share tacit knowledge between remote team members, teams maintain a *common chat room* and *online conference* (see in Table 3). In one project (β), we found that the team does not share tacit knowledge among dispersed team members. But, based on the situation, sometimes the team performs pair programming through screen sharing, to resolve problems. Challenges faced by the practitioners during knowledge sharing among distributed team members are shown in Figure1. Mitigation techniques applied by practitioners are also shown in the same Figure 1.



Dispersed teams- **α, β, δ**; *Distributed teams-* **γ, ε, ζ**; *Hybrid team-* **η**

Fig. 1 Knowledge sharing challenges and mitigation techniques

In Figure 1, arrows indicate the mitigation techniques ap- plied by practitioners for a specific challenge. Based on the severity of *communication*, *language* and *cultural* challenges frequently faced by practitioners during knowledge sharing in distributed agile projects (see Figure 1). Distanced teams are also struggling with *misunderstanding* and *visualization* challenges.



Not satisfied (0-3), Satisfied (4-6) and Highly satisfied (7-10) Dispersed teams- **α, β, δ**; *Distributed teams-* **γ, ε, ζ**; *Hybrid team-* **η**

Fig. 2 Success of knowledge sharing

Though teams face different types of challenges during knowledge sharing among distributed team members, we identified successful knowledge sharing in both locally and globally distributed agile teams from the seven cases studied. Based on the seven cases the above graph (see Figure 2) has been drawn. An *ordinal scale* is used to map the interviewee satisfaction with their KM activities in both local and global teams. Figure 2 depicts that in project (*Alpha*) interviewees think knowledge sharing activities are more successful among distributed team members than in the local team, due to language barriers (*non-native English speaker*) between local team members. It is also evident from the gathered data that *dispersed* feature teams are more successful in their knowledge sharing among distributed team members than *distributed* and *hybrid* teams.

6 Lesson Learned

6.1 Codification of Knowledge

According to Polanyi [Polanyi, 2009], *Individuals know more than they can say.* Polanyi classified human knowledge into two categories. *Tacit knowledge*, which is very difficult to describe or express: this type of knowledge is transferred through demonstration. Tacit knowledge has an important cognitive dimension which consists of mental models, beliefs and perspectives [Nonaka, 1994, Bennet and Tomblin, 2006, Nonaka, 2007]. So it cannot be easily characterized by clear expressive language. *Explicit knowledge*, is easily written down and codified. It is easily possible to characterize explicit knowledge in textual or symbolic forms. This kind of knowledge resides in textbooks, memos and technical documents. Codification of knowledge is the conversion of tacit knowledge into explicit knowledge in a written, verbal or visual format. The extraction process of tacit knowledge into explicit is called *externalization.* Tacit knowledge cannot be interpreted fully even by an expert [Ahmed et al., 2012]. This type of knowledge is more deeply placed in action and is hard to express in words [Hislop, 2002]. Nelson *et al.* [Nelson and Winter, 1982] conclude that it is impossible to describe all the necessary aspects of organizational tacit knowledge for successful performance. In organizations, most of the tacit knowledge is work related, which is learned informally as the team works [Wagner and Sternberg, 1987]. Codification extract tacit knowledge into explicit ; it is a challenging task, so an expert needs to understand the essence of the tacit knowledge in order to increase the degree of explicitness of knowledge. Surprisingly, we found from our results that all studied cases are concerned about knowledge codification. To codify tacit knowledge, teams are using Wiki, JIRA, Confluence etc. In local sites, technical presentations and discussion forums are also taken into account as knowledge codification strategies. Later, teams share codified knowledge among remote team members through repositories and that is helpful for the remote team members to reuse codified stored knowledge.

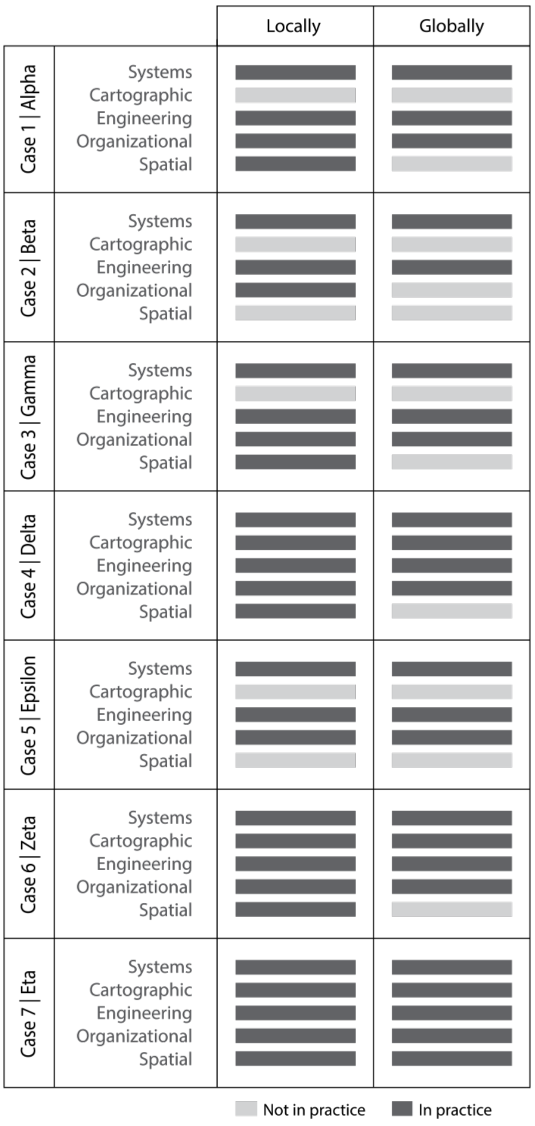
6.2 Knowledge Management Strategies in Practices

We found that knowledge management schools are in practice, according to the results in Table 2 and 3. We used Earls [Earl, 2001] framework to select types of strategies, "schools" or practices in the different projects that applied to managing knowledge locally and globally.

Based on the evidence from the different cases, we found that knowledge management schools were in use to manage knowledge both locally and globally. It is also evident that *systems*, *cartographic*, *engineering*, *organizational* and *spatial* schools are practiced in distributed agile projects to manage knowledge both locally and globally. Both *commercial* and *strategic* schools are focused on a business perspective (*patent*, *copyright*, *trademark*, *know-how* and *intellectual assets* [Earl, 2001]) and there is also no evidence found within gathered data sets that indicates those schools (*commercial* and *strategic*) are in practice. For that reason those schools are not taken into account in this research.

6.2.1 Systems school

This schools philosophy is to codifying knowledge with the help of technology. Organizations use repositories for storing and sharing knowledge. These knowledge repositories usually store domain specific information. The codification of knowledge can be compared with



		Locally	Globally
Case 1 | Alpha	Systems	■	■
	Cartographic	□	□
	Engineering	■	■
	Organizational	■	■
	Spatial	■	□
Case 2 | Beta	Systems	■	■
	Cartographic	□	□
	Engineering	■	■
	Organizational	■	□
	Spatial	□	□
Case 3 | Gamma	Systems	■	■
	Cartographic	□	□
	Engineering	■	■
	Organizational	■	■
	Spatial	■	□
Case 4 | Delta	Systems	■	■
	Cartographic	■	■
	Engineering	■	■
	Organizational	■	■
	Spatial	■	□
Case 5 | Epsilon	Systems	■	■
	Cartographic	□	□
	Engineering	■	■
	Organizational	■	■
	Spatial	□	□
Case 6 | Zeta	Systems	■	■
	Cartographic	■	■
	Engineering	■	■
	Organizational	■	■
	Spatial	■	□
Case 7 | Eta	Systems	■	■
	Cartographic	■	■
	Engineering	■	■
	Organizational	■	■
	Spatial	■	■

□ Not in practice ■ In practice

Fig. 3 Knowledge sharing strategies in practice

the *externalization* of knowledge by Nonaka [Dingsøyr et al., 2009]. It is easy to realize the benefits of knowledge bases and the systems school is the most researched school [Bjørnson and Dingsøyr, 2008]. These knowledge bases become richer and more useful over time. As shown in the Figure 2, the *systems* school is in practice in all cases to manage knowledge locally and globally. Though search functions are a difficult issue in the systems school, practitioners depend on it because across distances this school effectively perform knowledge sharing activities using repositories.

6.2.2 Cartographic school

This school focuses on the mapping of organizational knowledge and aims to build knowledge directories by disclosing who knows what [Earl, 2001]. This is sometimes achieved by yellow-pages, which ensure the accessibility to others of a knowledgeable person within the organization for knowledge exchange. Though knowledge maps and directories on company intranets might be helpful for distributed team members to have an idea of who knows what, in distributed projects it seems challenging to put into practice. This is because it needs joint effort and commitment from both local and remote team members. In collocation, it seems easier to find a knowledgeable or experienced person because they knew each other well. In globally distributed projects who knows what and what is where are important issues for effective knowledge exchange. We have found that the *cartographic* school is practiced by different projects (δ,ζ,η) to exchange knowledge both locally and globally. This strategy is also practiced in different companies by introducing the idea of knowledge brokers: this helps other developers to consult with knowledgeable and experienced software engineers [Schneider, 2009]. Knowledge brokers are knowledgeable and experienced software engineers who will communicate with other developers, provide them with information or listen to them.

6.2.3 Engineering school

This school of knowledge management focuses on business process re-engineering [Bjørnson and Dingsøyr, 2008] and knowledge flows in organizations. This school has a more empirical attention than other schools, which focuses on managing knowledge about software development processes and improvement of software development processes. More specifically, this school focuses on formal routines, mapping of knowledge flows, project reviews, and social interactions. Software process improvements like CMMI can be regarded as a stimulus for knowledge flow throughout the organization. This school supports explicit knowledge sharing and in distributed projects, temporal distance does not affect this school. In globally distributed projects coordination is one the major challenges and the engineering school focuses on the coordination process and aims to ensure knowledge flows within the organization though shared databases. The processes of using tools *(i.e the installation manual for GitHub or SVN with eclipse)*, quality code writing techniques, testing and reviews are all documented in repositories to share among distributed teams. Practice of this school is found in the studied projects.

6.2.4 Organizational school

The philosophy of the *organizational* school is to create a network by collaborating between communities to share or pool knowledge. This school of knowledge management focuses on organizational structure. These structures are often referred as "knowledge communities" [Bjørnson and Dingsøyr, 2008]. This is a networking approach for people to communicate and share knowledge. Based on the seven cases, this school is in practice for knowledge sharing both locally and globally.

6.2.5 Spatial school

The intention of the spatial school is to encourage socialization (tacit to tacit knowledge) as a means of knowledge exchange [Lloria, 2008]. The spatial school is more concerned with the development and utilization of the social capital which develops from people interactions, formal or informal, repeatedly over time [Earl, 2001]. The spatial school focuses on designing office space to promote knowledge sharing [Dingsøyr et al., 2009]. Organizations use different office settings to promote communication between people. For example, in the case of software organizations agile methodologies may use boards, charts or other tools to create spatial knowledge. Sometimes, even common spaces like conference rooms, dining rooms or places for refreshment and activities are also places where knowledge can be shared. Five cases were found that practice the *spatial* school to manage knowledge locally and only one case (*hybrid team*,η) was found that to practice the *spatial* school to manage knowledge both locally and globally. The *hybrid team* uses visual boards to communicate among collocated and distributed team members.

6.3 The Process of Organizational Knowledge Creation

Knowledge creation is continuous and dynamic process; and people are benefited from that through sharing and interacting with each other [Ernst and Kim, 2002]. Based on *explicit* and *tacit* knowledge, Nonaka *et al.* proposed dynamic knowledge creation model [Nonaka, 1994]. In this model, the authors discussed about four types of knowledge conversion modes. The conversion mode between tacit to tacit called *socialization*, between tacit to explicit it is called *externalization*, between explicit to explicit it is called *combination* and between explicit to tacit it is called *internalization.* Each modes can independently create knowledge.

6.3.1 Socialization- tacit to tacit

Socialization facilitate to share tacit knowledge through shared experience. In this context, the learning process start through observation, interaction and practice. So, as a result of this process technical skills and mental

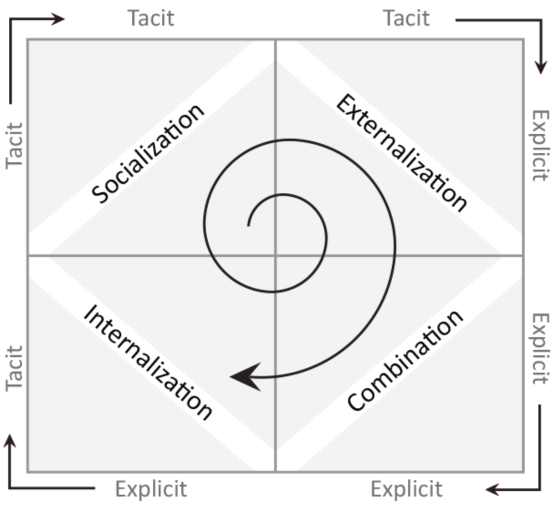


Fig. 4 Nonaka's dynamic knowledge creation model [Nonaka, 1994]

models are created and shared. Trust is the main ingredient in order to foster socialization (*social interaction*) in the organization [Nonaka and Takeuchi, 1995]. The main intension of the tacit knowledge is to learn and gain experience from others; so knowledge is created by learning during the discussion in front of coffee machine, meeting, training and team work [Nonaka, 1994] [Bolisani and Scarso, 1999][Haldin-Herrgard, 2000][Marwick, 2001]. In this research, we have found, team members share theirs knowledge infront of coffee machine. All studied projects share tacit knowledge among both collocated and distributed teambers.

6.3.2 Externalization- tacit to explicit

The extraction process of tacit knowledge into explicit is called *externalization.* In this book [Ahmed et al., 2012] the authors claimed that, tacit knowledge cannot be interpret fully even by an expert. This type of knowledge is more deeply placed in action and stiff to express in word [Hislop, 2002]. Nelson *et al.* [Nelson and Winter, 1982] conclude, it is impossible to describe all necessary aspects of organizational tacit knowledge for successful performance. In another research Wanger *et al.* [Wagner and Sternberg, 1987] mentioned, in the organization most of the tacit knowledge is work related that is learned informally during the team works. *Codification* is challenging to extract tacit knowledge into explicit; so an expert needs to understand essence of the tacit knowledge to increase degree of explicitness of knowledge. Through this study we have found, nowadays, almost all small, medium and large scale software development firms start using tools[3] to facilitate explicit knowledge sharing throughout the both collocated and distributed teams. Experience team members helping each others indirectly[4] with their experience is the distributed environment using those tools, by the problem solving, manual creation and individuals internalize what they experience.

6.3.3 Combination- explicit to explicit

The conversion strategy of existing explicit knowledge into new explicit knowledge is called *combination.* In the organization it is happen through meeting and conversation. The main intension of combination process is to manage all unstructured knowledge into one place by sorting, adding, categorizing and defining context. This process will help the team members to get right information at the right time. In the distributed environment team members might benefited from this process.

6.3.4 Internalization- explicit to tacit

Express explicit knowledge and convert it into tacit knowledge is called *internalization.* People read documents previously stored knowledge from the repositories and try to learn from that to enhance their existing knowledge. By applying this process explicit knowledge become tacit and it helps if knowledge is verbalized in documents and expressed through oral stories.

6.4 A Relation with the concept of "Ba" [Nonaka and Konno, 1998]

Japanese philosopher Kitro Nishida originally proposed this *"ba"* concept, "Ba" means *place* [Nonaka and Konno, 1998]. This space can be physical (i.e. office, dispersed or distributed team), virtual(i.e. email, teleconference), mental (i.e., shared experiences, idea), or any combination of them. *Ba* consider as a shared space that servers as a foundation for knowledge creation. In this research, we have found that, practitioners also apply ba through spatial school. The intention of the spatial school is to encourage socialization (tacit to tacit knowledge) as a means of knowledge exchange.

[3] wiki's, redmine, JIRA etc.

[4] *Reading and listening to success stories make people feel the truth and root of the story[Nonaka and Takeuchi, 1995]*

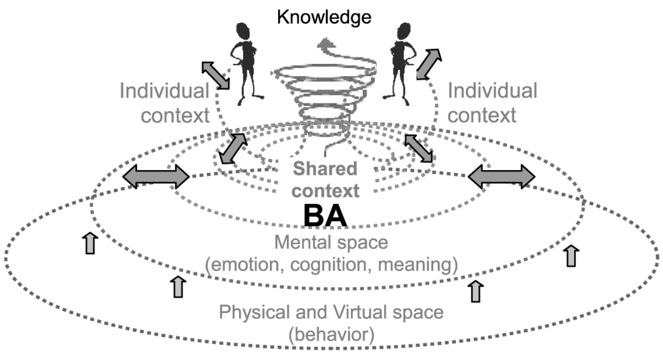

Knowledge
Individual context
Individual context
Shared context
BA
Mental space (emotion, cognition, meaning)
Physical and Virtual space (behavior)

Fig. 5 Ba and Knowledge Conversion [Nonaka and Konno, 1998]

6.5 Shared Understanding between team members

Sucessful development and deployment depends on the shared understanding between stackholders and software engineers. According to Martin et al. [Glinz and Fricker, 2014], there are two types of facts among group of people during shared understanding: explicit shared understanding(ESU) which denotes requirements, design documents, and manuals. Implicit shared understanding denotes the common understanding of non-specified knowledge, such as assumotions, opinions, and values. Organization that facilitate values, structures, and practices that helps to coordinate and communicate with team members effectively. During this study, we have found, studied projects apply *"Pair programming, customer collaboration, technical presentation and so forth* to establish shared understanding among distributed team members. In addition to that, we have also observed that, team also apply both *tacit* and *explicit* knowledge sharing strategies to build shared understanding among distributed team membres as in software engineering both strategies implies shared understanding.

Martin et al. [Glinz and Fricker, 2013] identified the enablers and obstacles for shared understanding. Domain knowledge, Previous joint work or collaboration, Existence of reference systems, Culture and Values, Geographic distance, and Trust are enable and foster shared understanding. Contractual situation, Outsourcing, Regulatory constraints, Normal vs. radical design, Team size and diversity, and Fluctuation are an obstacle to both implicit and explicit shared understanding. Achivinging shared understanding among distributed team members is not easy though. To achive shared understanding among distributed team members its required to overcome obstracles and enable the knowledge shareing and creation techniques to establish shared understanding.

7 Conclusion and Future Work

7.1 Summary

The aim of this research was to discover the knowledge sharing techniques, strategies applied and challenges faced by the practitioners in distributed agile projects. To perform knowledge management activities in a distributed agile project, different teams practice different types of approaches. But, in general, we found that different types of knowledge creation and sharing techniques are applied by practitioners to perform knowledge management activities in distributed agile projects. Along with that, we also found different types of strategies practiced by the team members to manage knowledge both locally and globally.

7.2 Contributions

For the first research question, we found that:

– To perform shared knowledge creation in a distributed agile project, team members practice: pair programming, customer collaboration, Scrum/Kanban boards, innova- tion boards, workshops/seminars, learning, technical pre- sentation and technical discussion techniques.
– In globally distributed agile projects, teams practice dif- ferent types of strategies to perform shared knowledge creation such as: *systems, engineering, organizational and cartographic* schools. We observed that the spatial school is in practice for local knowledge creation but when the project is distributed, this school is used less due to expensive tools.

For the second research question, we found that:

– To share knowledge among distributed sites, team mem- bers practice different type of techniques: repositories, pair programming, version control, screen sharing, daily scrums, weekly sprint status, common chat rooms, technical forums, discussion forums, electronic boards, online conferences, rotations/visits etc.
– *Systems*, *engineering* and *organizational* school strategies are explicitly in practice to share knowledge among distributed team members. These strategies foster effective knowledge sharing activities for team members in distributed agile projects. In distributed development, who knows what and what is where need to be known by employees, for effective knowledge sharing: this is associated with the cartographic school. But, in distributed agile projects, this school has is used less due to social-cultural distances. The spatial school facilitates knowledge sharing by using office space but in distributed agile projects this strategy is not explicitly in practice to share knowledge among remote team members.

For the third research question, we found that:

– During knowledge sharing among distributed team mem- bers, practitioners faced different types of challenges, such as: language, communication, misunderstanding, vi- sualization, cultural, technological, time zone difference and lack of information.
– To mitigate those challenges, practitioners also apply different types of mitigation techniques, such as: informal communication, cultural exchange, common platform, tools, visual prototyping, common chat rooms, rotation, and overlapping hours.

7.3 Future Work

Through a series of semi-structured interviews from agile practitioners, we investigated knowledge sharing activities in distributed agile projects. Communication, coordination and collaboration are the keys to fostering knowledge sharing between team members in agile software development. However, we have seen knowledge sharing in distributed agile projects is challenging, due to factors such as communication difficulties, language barriers and cultural barriers. To mitigate those challenges and succeed in knowledge sharing within and across borders, practitioners adopt different types of techniques to manage knowledge both locally and globally. Along with these techniques, we have also noticed that, practitioners adopt different types of strategies to manage knowledge both locally and globally. Though systems, engineering and organizational schools are explicitly in practice, the spatial school has less concern with managing knowledge in distributed agile projects. With closer observation between software engineering and schools Bjørnson and Dingsøyr found that there is a heavy focus on the systems and engineering schools [Bjørnson and Dingsøyr, 2008]. There are also limited number of studies focusing on the organizational school, but no studies in software engineering were found, that focus on the spatial aspect [Bjørnson and Dingsøyr, 2008]. Agile software development is more related to *socialization*, which includes the spatial schools concepts of knowledge sharing strategies. There is a lot of knowledge residing in the office space and office space fosters knowledge sharing through spatial knowledge management strategy. In the future, it will be interesting to find the spatial school being practiced in distributed agile projects.

8 Acknowledgement

We are thankful to our dear supervisor Dr. Darja Šmite, Associate Professor and this paper also published as a Blekinge Institute of Technology's Master thesis of 1st and 3rd authors in 2013.

References

K. Schneider. *Experience and knowledge management in software engineering*. Springer, 2009.

I. Richardson, M. O'Riordan, V. Casey, B. Meehan, and I. Mistrik. Knowledge management in the global software engineering environment. In *Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on*, pages 367–369. IEEE, 2009.

S. Sahay, B. Nicholson, and S. Krishna. *Global IT outsourcing: software development across borders*. Cambridge University Press, 2003.

A. Boden and G. Avram. Bridging knowledge distribution-the role of knowledge brokers in distributed software development teams. In *Cooperative and Human Aspects on Software Engineering, 2009. CHASE'09. ICSE Workshop on*, pages 8–11. IEEE, 2009.

H. Holz and F. Maurer. Knowledge management support for distributed agile software processes. *Advances in Learning Software Organizations*, pages 60–80, 2003.

D. Šmite, N.B. Moe, and P. Ågerfalk. Agility across time and space: Implementing agile methods in global software projects. 2010.

RK Kavitha and I. Ahmed. A knowledge management framework for agile software development teams. In *Process Automation, Control and Computing (PACC), 2011 International Conference on*, pages 1–5. IEEE, 2011.

B. Curtis, H. Krasner, and N. Iscoe. A field study of the software design process for large systems. *Communications of the ACM*, 31(11):1268–1287, 1988.

B. Nicholson and S. Sahay. Embedded knowledge and offshore software development. *Information and organization*, 14(4):329–365, 2004.

T.H. Davenport and L. Prusak. *Working knowledge: How organizations manage what they know*. Harvard Business Press, 2000.

I. Rus, M. Lindvall, and S. Sinha. Knowledge management in software engineering. *IEEE software*, 19(3): 26–38, 2002.

C. O'Dell and C.J. Grayson. If only we knew what we know: identification and transfer of internal best practices. *California management review*, 40:154–174, 1998.

T. Chau and F. Maurer. Knowledge sharing in agile software teams. *Logic versus approximation*, pages 173–183, 2004.

S. Dorairaj, J. Noble, and P. Malik. Knowledge management in distributed agile software development. In *Agile Conference (AGILE), 2012*, pages 64–73. IEEE, 2012.

M. Earl. Knowledge management strategies: Toward a taxonomy. *Journal of management information systems*, 18(1):215–233, 2001.

M T Hansen, N Nohria, and T Tierney. Whats your strategy for managing knowledge? *Knowledge Management Critical Perspectives on Business and Management*, 77(2):322, 2005.

I. Nonaka. A dynamic theory of organizational knowledge creation. *Organization science*, 5(1):14–37, 1994.

T. Dingsøyr, F.O. Bjørnson, and F. Shull. What do we know about knowledge management? practical implications for software engineering. *Software, IEEE*, 26 (3):100–103, 2009.

F.O. Bjørnson and T. Dingsøyr. A survey of perceptions on knowledge management schools in agile and traditional software development environments. *Agile Processes in Software Engineering and Extreme Programming*, pages 94–103, 2009.

S. Nerur, R.K. Mahapatra, and G. Mangalaraj. Challenges of migrating to agile methodologies. *Communications of the ACM*, 48(5):72–78, 2005.

Ikujiro Nonaka and Noboru Konno. The concept of ba : Building a foundation for knowledge creation. *California Management Review*, 40(3):40–54, 1998.

W. Maalej and H.J. Happel. A lightweight approach for knowledge sharing in distributed software teams. *Practical Aspects of Knowledge Management*, pages 14–25, 2008.

T. Hildenbrand, M. Geisser, T. Kude, D. Bruch, and T. Acker. Agile methodologies for distributed collaborative development of enterprise applications. In *Complex, Intelligent and Software Intensive Systems, 2008. CISIS 2008. International Conference on*, pages 540–545. IEEE, 2008.

A. Boden, G. Avram, L. Bannon, and V. Wulf. Knowledge management in distributed software development teams-does culture matter? In *Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on*, pages 18–27. IEEE, 2009.

G.R. Marczyk, D. DeMatteo, and D. Festinger. *Essentials of research design and methodology*, volume 2. Wiley, 2010.

P. Runeson and M. Höst. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2):131–164, 2009.

U. Sekaran. *Research methods for business: A skill building approach.* John Wiley & Sons, 2006.

J.W. Creswell. *Research design: Qualitative, quantitative, and mixed methods approaches.* Sage Publications, Incorporated, 2008.

U. Flick. *An introduction to qualitative research.* Sage Publications Limited, 2009.

C. Robson. *Real world research: a resource for social scientists and practitioner-researchers*, volume 2. Blackwell Oxford, 2002.

T. Basit. Manual or electronic? the role of coding in qualitative data analysis. *Educational Research*, 45 (2):143–154, 2003.

V. Braun and V. Clarke. Using thematic analysis in psychology. *Qualitative research in psychology*, 3(2): 77–101, 2006.

G. Guest, K.M. MacQueen, and E.E. Namey. *Applied thematic analysis.* Sage Publications, Incorporated, 2011.

M.B. Miles and A.M. Huberman. *Qualitative data analysis: An expanded sourcebook.* Sage Publications, Incorporated, 1994.

K.F. Punch. *Introduction to research methods in education.* Sage Publications Limited, 2009.

F. Shull, J. Singer, and D.I.K. Sjøberg. *Guide to advanced empirical software engineering.* Springer, 2007.

I. Bleijenbergh, H. Korzilius, and P. Verschuren. Methodological criteria for the internal validity and utility of practice oriented research. *Quality & Quantity*, 45(1):145–156, 2011.

U. Cress and J. Kimmerle. A systemic and cognitive view on collaborative knowledge building with wikis. *International Journal of Computer-Supported Collaborative Learning*, 3(2):105–122, 2008.

D.W. Palmieri. Knowledge management through pair programming. 2002.

R. McDermott. Learning across teams. *Knowledge Management Review*, 8(3):32–36, 1999.

K. Sureshchandra and J. Shrinivasavadhani. Adopting agile in distributed development. In *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on*, pages 217–221. IEEE, 2008.

N.B. Moe and D. Šmite. Understanding a lack of trust in global software teams: a multiple-case study. *Software Process: Improvement and Practice*, 13(3):217–231, 2008.

M. Polanyi. *The tacit dimension.* University of Chicago press, 2009.

A. Bennet and M.S. Tomblin. A learning network framework for modern organizations: Organizational learning, knowledge management and ict support. *VINE*, 36(3):289–303, 2006.

Ikujiro Nonaka. The knowledge-creating company. *Harvard Business Review*, 26(4-5):598–600, 2007.

P.K.K. Ahmed, K.K.K. Lim, and A.Y. Loh. *Learning through knowledge management.* Routledge, 2012.

D. Hislop. Mission impossible? communicating and sharing knowledge via information technology. *Journal of Information Technology*, 17(3):165–177, 2002.

R.R. Nelson and S.G. Winter. *An evolutionary theory of economic change.* Belknap press, 1982.

R.K. Wagner and R.J. Sternberg. Tacit knowledge in managerial success. *Journal of Business and Psychology*, 1(4):301–312, 1987.

F.O. Bjørnson and T. Dingsøyr. Knowledge management in software engineering: A systematic review of studied concepts, findings and research methods used. *Information and Software Technology*, 50(11): 1055–1068, 2008.

M.B. Lloria. A review of the main approaches to knowledge management. *Knowledge Management Research & Practice*, 6(1):77–89, 2008.

Dieter Ernst and Linsu Kim. Global production networks, knowledge diffusion, and local capability formation. *Research policy*, 31(8):1417–1429, 2002.

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

E. Bolisani and E. Scarso. Information technology management: a knowledge-based perspective. *Technovation*, 19(4):209–217, 1999.

T. Haldin-Herrgard. Difficulties in diffusion of tacit knowledge in organizations. *Journal of Intellectual capital*, 1(4):357–365, 2000.

A.D. Marwick. Knowledge management technology. *IBM systems journal*, 40(4):814–830, 2001.

Martin Glinz and Samuel A Fricker. On shared understanding in software engineering: an essay. *Computer Science-Research and Development*, pages 1–14, 2014.

Martin Glinz and Samuel Fricker. On shared understanding in software engineering. In *Software Engineering*, pages 19–35. Citeseer, 2013.