## Team Performance Management: An International Journal

# A software engineering team research mapping study

Joanna F. DeFranco, Phillip Laplante,

# A software engineering team research mapping study



Joanna F. DeFranco and Phillip Laplante
*The Pennsylvania State University, Malvern, Pennsylvania, USA*

Received 4 August 2017
Revised 5 December 2017
12 December 2017
Accepted 12 December 2017

## Abstract

**Purpose** – The purpose of this mapping study has been performed to identify, critically analyze and synthesize research performed in the area of software engineering teams. Teams, in a general sense, have been studied extensively. But the distinctive processes that need to be executed effectively and efficiently in software engineering require a better understanding of current software engineering team research.

**Design/methodology/approach** – In this work, software engineering team publications were analyzed and the key findings of each paper that met our search inclusion criteria were synthesized. In addition, a keyword content analysis was performed to create a taxonomy to categorize each paper and evaluate the state of software engineering team research.

**Findings** – In software engineering team research, the resulting areas that are the most active are teamwork/collaboration, process/design and coordination. Clear themes of analysis have been determined to help understand how team members collaborate, factors affecting their success and interactions among all project stakeholders. In addition, themes related to tools to support team collaboration, improve the effectiveness of software engineering processes and support team coordination have been found. However, the research gaps determined from the content analysis point toward a need for more research in the area of communication and tools.

**Originality/value** – The goal of this work is to define the span of previous research in this area, create a taxonomy to categorize such research and identify open research areas to provide a clear road map for future research in the area of software engineering teams. These results, along with the key finding themes presented, will help guide future research in an area that touches all parts of the software engineering and development processes.

**Keywords** Teamwork, Mapping study, Software engineering teams, Team research taxonomy, Technical teams

**Paper type** Literature review

## 1. Introduction

Teamwork is a term used to describe the level of synergy and cooperation found within a group of persons striving to achieve a mutual goal. Teamwork itself is a complex topic that has been studied for decades. Salas *et al.* (2005) recognized the vast amount of research in the area of teams and worked to determine the core components (as known as the "Big Five") of teamwork research: team leadership, mutual performance monitoring, backup behavior, adaptability and team orientation. These five components need supporting mechanisms to bring them together: shared mental models, mutual trust and closed-loop communication. All these concepts have been extensively studied – highlighting the complexity of this topic. In another team study, Tannenbaum *et al.* (2012) discussed themes contributing to the growing needs and challenges of modern teams: *dynamic composition*, *technology and distance* and *empowerment and de-layering*. For example, dynamic composition refers to the volatility of a team; however, volatility, in this case, is not referring to typical employee attrition rates, but to volatility as the projects progress. That is, as a projects progresses, team members change due to the team needs. Mathieu *et al.* (2008) also discussed team

Team Performance Management: An International Journal

1352-7592
DOI 10.1108/TPM-08-2017-0040

composition where it has a planned fluidity. For example, a member may be on multiple teams or a team may include temporary contracted employees, or members that are on the team for only a specific task. This complexity is typical in a software project and one reason why the traditional team framework theories are not enough.

The second team challenge according to Tannenbaum is *technology and distance*. As a result of the frequent introduction and improvement of collaboration technologies, anytime–anywhere team member participation is now possible. That technological improvement clearly brings challenges. For example, constant messaging and location information can be intrusive, which fosters team distrust. It can also cause information overload or have the opposite effect of unfulfilled immediate feedback expectations. Finally, Tannenbaum presented *empowerment and de-layering*. Many modern teams operate autonomously where the team control and management went from central team leadership to self-control and self-management. The pressure is on the team to assume various leadership roles, as well as facilitate their own professional development – they also need to self-regulate and self-discipline. All of these areas clearly create new challenges that needed to be addressed as their effects, if not managed properly, will negatively impact project success and facilitate team dysfunction.

Unquestionably, as the complexity of engineering problems grew, a new area of research appeared – *teamwork in technical teams*. Achieving the high level of teamwork is essential in any engineering project and involves a multifaceted approach. Correspondingly, there is an emergence of teamwork research, as it pertains to software engineering. Software engineering, in particular, requires effective inter-team work (e.g. between test, requirements, architecture, construction teams) and effective intra-teamwork (e.g. between project managers, program managers, customers and end users). In other words, software engineering teamwork needs to include all of the necessary stakeholders involved in the complete software engineering process. However, the published literature reviews are focused on teamwork, as it relates to one specific aspect of the software engineering process such as Agile development or distributed/global teams.

Ghobadi and Mathiassen (2015) identified seven team factors that affect knowledge sharing in Agile teams. This research also included a description of the barriers of knowledge sharing among the different Agile team stakeholders (project managers, developers, testers and users). For example, Agile team members with the role of "tester" put a primary emphasis on barriers to the team knowledge-sharing factor involving team capabilities. A "developer" on an Agile team was found to put a primary emphasis on the barriers to project communication in the context of team knowledge sharing.

In another recent study focusing on Agile team development, Dingsoyr *et al.* (2016) identified five factors that affect software team performance: team coordination, goal orientation, team cohesion, shared mental models and team learning. This research also compared the propositions to Agile development practices to begin a dialog on new practices to improve the productivity in Agile teams.

In a systematic literature review on Agile software development by Dyba and Dingsoyr (2008), one of the themes that emerged was the social factors which includes collaborative work and team characteristics, implying it is necessary to focus on the collaborations in an Agile team for the team to succeed.

Another targeted area of team research focuses on distributed/global/virtual software development teams. Virtual teams are typically defined as a team separated by geography, time and/or culture. They are unique and can be beneficial to an organization when the separation factors is managed properly. In other words, the unique managerial and social challenges of these teams require care and need to be further studied (Powell *et al.*, 2004).

A systematic review in the area of global/virtual software engineering teams performed by Smite *et al.* (2010) identifies seven best practices. Three of the best practices involve effective teamwork specifically. The remaining four specified cohesiveness, awareness and communication which are clearly additional factors of effective teamwork. In another study on distributed teams, Agerfalk and Fitzgerald (2006) identified coordination, communication and control as global software development challenges. These challenges need extensive research where the focus is on human-oriented processes. If the challenges are not addressed, the advantages of the global software engineering will be lost (Powell *et al.*, 2004).

In a more recent virtual team study, Gibbs *et al.* (2017) reviewed research pertaining to team type (in any domain) and its impact on the virtual team process, including variation in function, tenure, task, etc. The review concluded that scholars need to consider leadership, cultural composition and technology in a particular team type to guide their research. In addition, Gibbs indicated that a threat to validity of the virtual team research to date is that the majority of investigations use student samples which may bias what we know about virtual teams.

These are all excellent studies, as these topics reflect the reality of modern software development; however, a comprehensive study on teamwork in the area of software development as presented in this paper is necessary instead of piecemeal literature reviews. This will provide researchers and practitioners a complete view of the state of this research domain.

Despite the complexity and multifaceted nature of software team-related research, the ACM CCS classification system lists only the topic "programing teams" in its most recent taxonomy (ACM, 2012). In fact, the classification of the "programming teams" in the ACM CCS is buried under four categories.

In addition to providing a comprehensive summary of research performed in this area, another goal of this study is to classify previous work in a software engineering team research in a more detailed and comprehensive manner. The challenge is that a simple search to envelope the body of software team research is virtually impossible. Thus, a mapping study is necessary to create an objective aggregation and summary overview of this topic.

We used a rigorous, unbiased process to collect, evaluate and synthesize relevant research that has occurred in the area of software engineering teams. A systematic mapping study is a derivative of a systematic literature review except that the purpose of a mapping study is to classify the research identified and identify where the research is lacking (Kitchenham *et al.*, 2016). In other words, this type of review is performed to learn about a topic, synthesize new knowledge on the topic or address a deficiency in existing literature by being a facilitator for further research.

We fully describe the mapping study process that was used and present the resulting papers on software engineering team research. Our review resulted in a taxonomy to categorize and analyze the software engineering team research. We then use the taxonomy to categorize the papers as a result of a keyword analysis. Finally, we present an analysis and synthesis of the key findings of the top research areas. The goal is for researchers to use these results as a springboard and road map for future software engineering team research.

## 2. Systematic mapping study

A mapping study is a methodical procedure with a goal of synthesizing existing research to create a generalization that will promote future research. The process, very similar to a systematic literature review, provides a means to create an unbiased evaluation of a particular research topic in a reliable, rigorous and repeatable manner. Kitchenham (2004) described a systematic review process that was further refined by Brereton *et al.* (2007).

We have summarized the process with a three-phase illustration, namely, *define*, *organize* and *execute*, as shown in Figure 1. In the *define* stage, the type, need and contribution of the review are described. The "type" of review essentially describes the domain under investigation. The "need" for the review may describe an insufficiency in that research domain. The "contribution" of the review provides a road map to address the "need." The *organize* stage includes conceptual structuring, which refers to guiding theories or a point of view about the topic, and it describes the paper review procedure. Finally, in the *execute* stage, the literature collected is analyzed, synthesized and reported. The analysis consists of critically reviewing the literature for inclusion and synthesis with the other papers included. The synthesis can result in a taxonomy with which to categorize the research or result in a future research agenda. The final reporting may include practical questions for future research.

We now discuss how we used this process to evaluate the software engineering team research. We provide sufficient detail so that the process may be replicated by other researchers.

**Define:** *review type, need and contribution.*

Software engineering is certainly a complex and critical activity. The complexity is due to the constant evolution of process and technology and the field of software engineering is largely a team-oriented activity, which adds another type of complexity that needs to be effectively studied. Accordingly, there is a need to organize this research and to determine any omissions or deficiencies in the research to facilitate future research, a contribution of this study.

The software engineering team research can help practitioners understand *soft skills*, such as effective communication and project management, and *team factors*, such as cultural differences, team size, trust, distance and team member personality. For example, in a paper reviewing literature on global team challenges, cultural differences were ranked second with 31 papers and trust was ranked fifth with 13 papers (da Silva *et al.*, 2010). This ranking is not surprising, as research in global teams has shown that the development of trust can be hindered by distance, which will affect team cooperation (Casey, 2010).

The results of this mapping study will answer the following research questions:

*RQ1*. How much research activity pertaining to *software engineering teams* has been published?

*RQ1a*. What is the geographical and chronological distribution of the research in the software engineering teams?

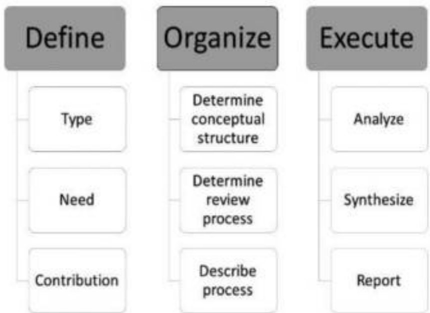


**Figure 1.**
Summarized mapping study process

*RQ1b*. What are the research communities that conducted the research in the software engineering teams?

*RQ2*. Which categories of the software engineering team process or dynamics are being addressed by researchers?

*RQ2a*. Which category of software engineering team research is the most prevalent?

*RQ2b*. What types of teams are the focus of the software engineering team researchers?

*RQ3*. What are the major, common findings and themes in the software engineering team research?

*RQ4*. Where are the gaps in the current state of the software engineering team research?

**Organize:** *Determine the conceptual structure and determine and describe the review process*.

The current state of the software engineering team literature spans the many facets of the software engineering process. This is reasonable, as *the team* is involved in every aspect of the software engineering process. However, the span of research may cause difficulty for a researcher to determine the key findings in software engineering team research if they are studying a particular facet of the software engineering process. For example, if the literature in global software engineering teams is studied, the research findings that focus on the software engineering team effectiveness will be omitted from that review unless it also covered global teams. To address this deficiency, we propose a conceptual structure via a taxonomy to organize the software engineering team research. We created this taxonomy by performing a keyword content analysis to be discussed in the results section of this mapping study.

A detailed plan to search for the literature to include in this study is described below. The research protocol defines the detailed plan of the review. Having a detailed plan to follow during the mapping study process attempts to mitigate any researcher bias (Kitchenham, 2004). For the review described herein, we planned first to determine the targeted databases in which to conduct literature search, then to decide the search terms and search criteria that would produce the most effective and complete results of this literature review.

*2.1 Databases*

We considered the largest professional organizations that capture the state-of-the-art software engineering research in their digital libraries: IEEE Xplore (digital library for the Institute of Electrical and Electronics Engineers) and the ACM (digital library for the Association for Computing Machinery). We also searched Google Scholar (https://scholar.google.com/) because it contains search metadata provided by other major publishers (e.g. Elsevier, John Wiley) and harvested other research resources via Web crawling.

*2.2 Search terms and criteria*

The differences or inconsistencies from one database to another required several search terms and search engine criteria combinations to be piloted. The search terms are to be retrieved from the research questions outlined in the *Define* phase. We constructed several search strings and criteria to be utilized in the automated search to determine which terms would be the most effective. For example, when using the IEEE Xplore database, the term “teams” resulted in 214,216 papers. This result reduced to 56,532 when changing the search criteria to only include scholarly journal publications. IEEE Xplore also had many search criteria options, such as the option to conduct the search with only metadata versus searching within the full text of the document. However, we did not want to limit the years or the research venue to only journal publications or to only searching metadata as opposed to the full text. Thus, the pilot exercise revealed we needed to explore search terms with justification from this research community to target this search.

To most effectively target the appropriate research, we modeled our search term criteria after the work of Hoonlor *et al.* (2013), which was a literature review on computer science trends. In that review, the researchers utilized the ACM Computing Classification System or CCS (ACM, 2012). We used the CCS category terms in bold shown here:

```
Software and its engineering
    software creation and management
        collaboration in software development
            open source model
               programming teams
```

We used the same search terms in IEEE Xplore, ACM and Google Scholar. Again, with no other search engine restrictions. We reviewed all resulting periodicals and reviewed conferences based on a manual search of the following venues suggested by subject matter experts who reviewed a preliminary version of this work: Cooperative and Human Aspects of Software Engineering (CHASE), International Conference on Software Engineering (ICSE), International Conference on Global Software Engineering (ICGSE), Computer Supported Cooperative Work (CSCW), European Conference on Computer-Supported Cooperative Work (ECSCW), Conference for Human-Computer Interaction (CHI), International Symposium on the Foundations of Software Engineering (FSE). In total, 502 publications were reviewed for this study. Table I shows the breakdown of the number of results from each database utilized.

The co-authors of this paper further validated the review protocol and results described.

**Execute:** *Analyze, synthesize and report.*

One of the goals in this part of the review is to select the appropriate studies. In the selection process, the quality of each publication, as well as its relevance in answering the research questions, is assessed. In addition, the data to answer the research questions will be extracted and synthesized.

As noted in Brereton *et al.* (2007), "keywords and abstracts are too poor quality for researchers to determine whether papers are relevant to specific research questions." It is for this reason that more than the title, keywords and abstract need to be read to determine the quality and relevance of the paper. As we found papers that contained specific research related to software engineering teams, we selected them for this study. Next, we assessed the quality of the paper in matching the inclusion criteria of the literature search. Following are the inclusion/exclusion criteria for this study.

### *2.3 Inclusion conditions*

- Papers that describe research on software engineering teams and the collaborative process of software engineering. For example, the study must have included

**Table I.**
Database results

| Venue | IEEE | ACM | Google Scholar |
|---|---|---|---|
| Journals | 8 | 100 | 42 |
| Conferences | 28 | 219 | 105 |

analysis of either teams or individuals on teams that were working on some aspect of a software development or software engineering project.
- Papers that had complete data – not preliminary studies or research proposals. Therefore, if the study used empirical research methods, the data description needed to be complete. Note: a few papers were included in this study that were not of empirical nature, and instead were summaries of tools utilized by software teams or provided value regarding facets of the software engineering team process.
- Papers that studied open source projects if the focus was studying some aspect of the people working on the project.
- Papers that studied a software development process that involves collaboration such as pair programming.

*2.4 Exclusion conditions*
- Papers focusing on software engineering education (e.g. teaching teamwork to students in a capstone course or assessing individual contributions for course team projects).
- Non-referred publications (e.g. newsletters, bulletins).
- Papers focusing on a specific part of a software engineering process where the goal of the research is not related to improving the effectiveness of the team.
- Papers focusing on tools to support a part of the software engineering process that does not present or focus on data supporting team collaboration or improving the teamwork outcome.
- Papers that present only preliminary results (e.g. overview of one case study or an incomplete study where the preliminary results were a precursor to the goal of the study not reached yet).
- Short papers describing panel sessions, preliminary ideas, idea or theories not tested or one-page summaries of research submitted to a conference.

To begin the synthesis process and answer the research questions, we extracted the following data from each study:
- the keywords to be used in a content analysis to ultimately determine the taxonomy/categories of software engineering team research;
- the type of team evaluated (traditional, Agile, global, open source, pair programmers;
- research demographics (publication venue, researcher affiliation, country, publication year); and
- information to report the major findings of each study.

The results of the data synthesis are used to determine the main themes within software development team research and categorize each of the papers selected. To determine the categories of research, we performed a content analysis on the keywords of each paper. Performing content analysis examines the meaning of words or concepts and "goes beyond merely counting words to examine language intensely for the purpose of classifying large amounts of text into an efficient number of categories of similar meanings" (Hsieh and Shannon, 2005).

The next step is the reporting process. The details of the papers and data extraction will be presented in Section 3 of this paper. Note that the data and analysis of this mapping study were validated throughout by the authors of this paper as part of the review process.

## 3. Results and analysis

This section presents the results and analysis of the resulting data collected. Each of the five research questions (RQ) presented in Section 2 will be addressed.

### *3.1 How much research activity pertaining to software engineering teams has been published? What is the geographical and chronological distribution of the research in software engineering teams? What are the research communities that conducted the research in the software engineering teams?*

The Appendix contains a full citation list of the 103 papers (journal and conference) included in this mapping study. As previously mentioned, there was no search year restriction in this study. When considering the papers that met the inclusion criteria, the initial journal publication year was 1981 and the initial conference publication year was 1989. In addition, from 1981 to 1998, there were only two conference papers and three journal papers published; therefore, to improve the visual scale of Figure 2, only data from 1999 to 2016 were graphed. The chronological distribution of papers shows that the publications peaked from 2006 to 2011, as 77 per cent of the conference papers and 54 per cent of the journal papers were published during that time frame. However, a peak again appeared in 2015 but with no overlap in research topics for that year.

We also identified the country of author institution affiliation of each paper. Shown in Figure 3, is the combination of both journal and conference publications; there were 29 countries represented overall. In total, 42 per cent of the papers were investigated or co-investigated by researchers working at organizations located in the USA, followed by 11 and 9 per cent of the included papers from researchers affiliated with organizations in Canada and Germany, respectively. The remaining papers were distributed among the remainder of the countries shown in the bar graph.

Table II shows the distribution among research affiliation. The table shows that 72 per cent of the research was conducted strictly by researchers from an academic institution, 7 per cent from researchers associated with a company and 22 per cent of research had researchers from both affiliations.

Each paper addressed various aspects of the software engineering team research that will be analyzed and summarized in the next section.

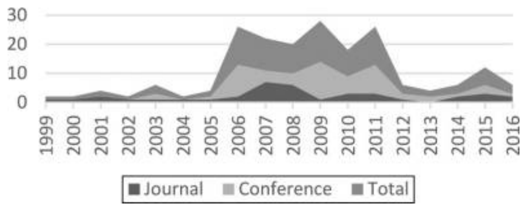


**Figure 2.** Chronological distribution of publications

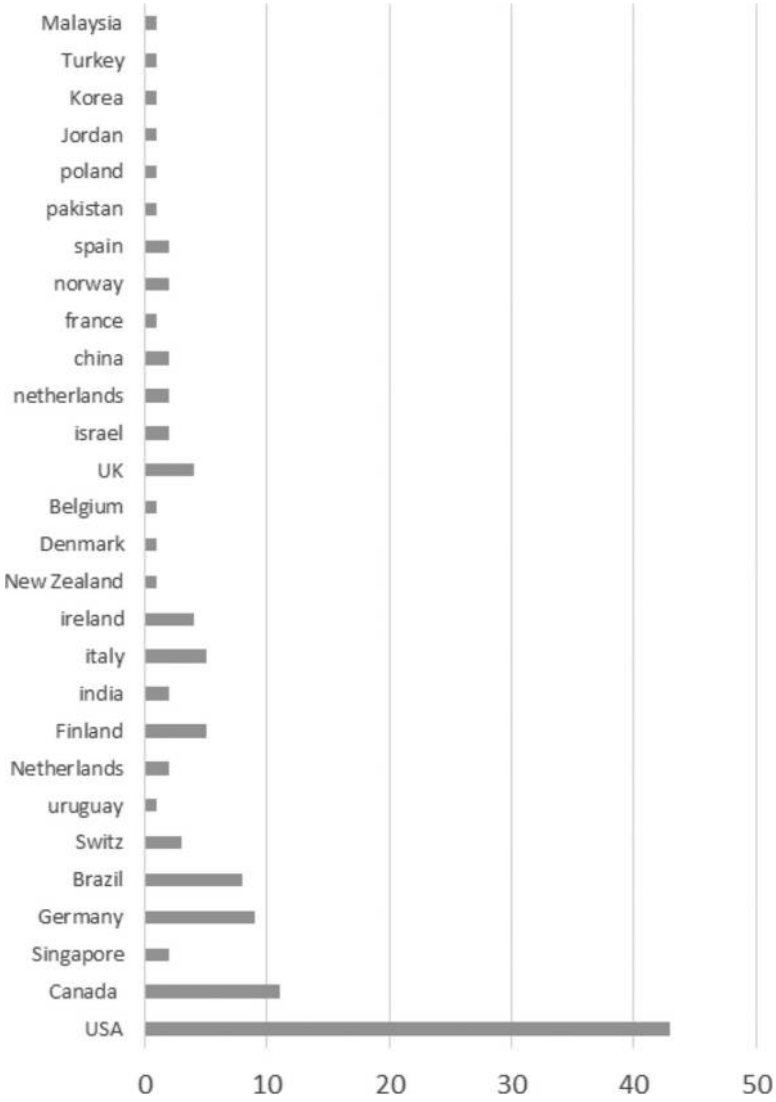


**Figure 3.**
Country distribution of researcher organization affiliation

| Venue | Academic (%) | Industry (%) | Both (%) |
|---|---|---|---|
| Journals | 73 | 5 | 20 |
| Conferences | 69 | 8 | 23 |
| Total | 72 | 7 | 22 |

**Table II.**
Researcher affiliation distribution

*3.2 Which categories of the software engineering team process or dynamic is being addressed by researchers? Which category of software engineering team research is the most prevalent? What types of teams are the focus of the software engineering team researchers?*
Keywords are descriptive words provided by authors to indicate the content of a paper. These words assist readers and archiving systems in determining a document relevance when searching a particular domain of research.

A keyword content analysis of the papers included in this study was performed. Categories emerged from this process and are described in Table III. These categories created a taxonomy to classify software engineering research. The designator letter will be used to show the categories into which each paper fell. The Appendix includes the classification of the each paper included in this study.

When reviewing the categories in Table III, there are a few things to consider. First, the process to categorize the keywords is not a straightforward one, as the keywords chosen by some authors are ambiguous. In those cases, the findings of the paper needed to be reviewed again to determine the meaning behind certain keywords. In addition, if a keyword seemed to apply to two categories, then the paper was scanned again to determine category in which to place the keyword. For example, "programming," a relatively frequent keyword, may have been referring to the research domain (e.g. software development or software engineering) or may have been referring to improving an activity within software engineering (e.g. process/design).

Next, there were also papers that describing research in Agile software processes in general (e.g. how Agile practices effect the team's well-being) or papers that focused on a specific part of a specific Agile process (e.g. evaluating pair programming with respect to system complexity). In the latter case, many of the authors would not even use the word "Agile" as a keyword or even use it in the main text. Given these facts, we decided to create two separate categories to classify each of those types of papers.

**Table III.**
Software engineering research taxonomy for classification

| Paper Classification | Designator | Description (Research Included) |
|---|---|---|
| Agile process | A | Agile development practices in general, i.e. no specific type of Agile process was the focus |
| Global software development | G | The effects of globally distributed software development teams. This also includes research conducted on open source development teams |
| Process/design | P | Includes software engineering procedures and processes (e.g. maintenance, testing etc.). Also papers that focus on specific Agile process such as pair programming |
| Teamwork/ collaboration | T | Includes factors to improve team effectiveness and human aspects of the software engineering process. For example, team climate, motivation, team diversity, shared understanding, attitude etc. |
| Communication | C | Factors to improve the effectiveness of communication, barriers to communication etc. |
| Coordination | O | Includes project management tasks and goals. Also, task coordination among actors, etc. |
| Project improvement/ effectiveness/quality | Q | Includes factors that improve project efficiency, performance, productivity, etc. |
| Tools | L | Includes research on collaborative tools for communication, knowledge management, etc. |

And finally, the generic categories that came from keywords describing the research domain (e.g. software engineering) or keywords describing the type of research performed (e.g. empirical studies, controlled experiments, etc.) were not included in the taxonomy.

Figure 4 shows the prevalence of the resulting categories of research that emerged from the keyword analysis. Author keywords that represented teamwork or collaboration appeared the most at 67 per cent, followed by keywords representing process and design at 49 per cent and those that represent coordination at 40 per cent.

During the content analysis of the key findings of each paper, specific team structure categories emerged. Upon further analysis, there were five resulting structures in which the researchers focused: open source software, Agile, traditional, pair programming and global teams. Clearly, some of these categories overlap, but the categorization is based off of the research focus. For example, a global team may be using an Agile method, but the research focused on the effects of team location on project success – thus, the paper was categorized as a global team paper.

Figure 5 shows the team-type distribution. The bars in the graph show the team structure focus of the research separating journal and conference papers, as well as an overall focus by combining both paper venues. The overall result is a research focus of traditional teams at 41 per cent and global teams at 33 per cent.

Further analysis of the publication year showed that the global team research peaked from 2005 to 2007, and the team research that focused on traditional teams peaked from 2008 to 2010 (Figure 6).

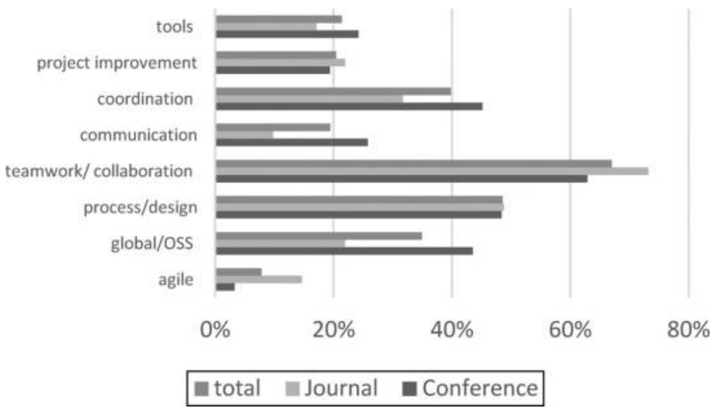


**Figure. 4.** Category rankings from keyword analysis

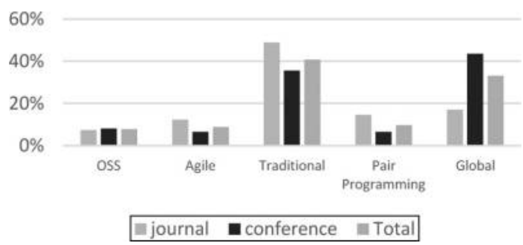


**Figure 5.** Research distribution of team type

Separating journal from conference papers can sometimes show emerging topics. For example, having more global papers at a conference may indicate a trend of research that will appear in journals later. In this case, for the analysis separating journal and conference papers, as shown in Figure 7, the journal papers peaked later in 2011 for global teams and at the same time for traditional teams in 2010.

### *3.3 What are the major, common findings and themes in software engineering team research?*

The Appendix contains a summary of the key findings for each paper included in this study. Next, themes from those key findings of the top three research areas are presented. These are organized with each research area in isolation (if a paper was categorized in only that research area) and in combination with the other research areas.

*3.3.1 Teamwork/collaboration (T).* In total, 22 papers were classified with a team/collaboration focus. They may have had some other classifications, but they did not have a process or coordination focus. A content analysis of the findings resulted in four themes:

(1) tools to improve collaboration and team productivity;

(2) Agile and team effectives, attitude and leadership;

(3) factors effecting team success; and

(4) general team communication analysis.

*3.3.2 Process/design (P).* In total, 14 papers were classified with a process focus. They may have had some other classifications, but they did not a have a teamwork/

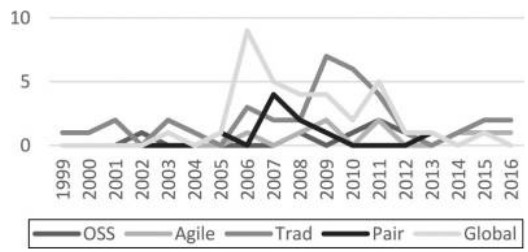


**Figure 6.**
Team type trend

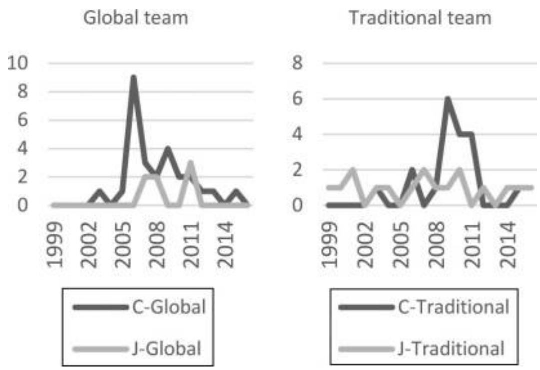


**Figure 7.**
Research distribution of global and traditional teams

collaboration or coordination focus. A content analysis of the findings resulted in three themes:

(1) effects of Agile (emphasis on pair programming) on a process;
(2) aspects of a process to improve team effectiveness; and
(3) understanding team working processes.

*3.3.3 Coordination (O).* In total, six papers were classified with a coordination focus. They may have had some other classifications, but they did not have process or coordination. A content analysis of the findings resulted in one theme:

(1) Team dispersion and its effects on productivity and communication management.

*3.3.4 Teamwork/collaboration (T) & process/design (P).* In total, 20 papers were classified with a teamwork/collaboration and process/design focus. They may have had some other classifications, but they did not have a coordination focus. A content analysis of the findings resulted in four themes:

(1) tools supporting teams;
(2) Agile team behavior, management and effectives;
(3) new processes or team structure to improve team outcomes; and
(4) analysis of team members and their interaction processes.

*3.3.5 Teamwork/collaboration (T) and coordination (O).* In total, 20 papers were classified with a teamwork/collaboration and coordination focus. They may have had some other classifications, but they did not have a process/design focus. A content analysis of the findings resulted in three themes:

(1) tools to improve coordination;
(2) global and virtual team management; and
(3) human aspects of teamwork.

*3.3.6 Process/design (P) and coordination (O).* In total, nine papers were classified with a coordination and process/design focus. They may have had some other classifications, but they did not have a teamwork/collaboration focus. A content analysis of the findings resulted in three themes:

(1) analysis of coordination in distributed teams;
(2) analysis of general developer processes; and
(3) analysis of globally created systems to determine best practices.

*3.3.7 Teamwork/collaboration (T), process/design (P) and coordination (O).* Six papers fell in all three of these categories, resulting in two themes:

(1) processes for improved collaboration (tagging, distributed inspection); and
(2) observations of team effectiveness factors (culture, pragmatics, schedules).

### *3.4 Where are the gaps in the current state of software engineering team research?*

Another goal of this study was to determine the research gaps. A logical way to determine these gaps was to evaluate the least active research areas resulting from the keyword analysis. As such, it would appear from Figure 4 that the general Agile category would be considered a research gap; however, given the keywords related to processes such as pair programming were included in the process/design category, and pair programming is considered an Agile process, more accurate research gaps would be *communication*, *project improvement* and *tools* categories. Considering all the papers have an inherent goal of project improvement but may have not chosen a keyword to represent project improvement, the real research gap appears to point toward research in the areas of *communication* and *tools*. In addition, in the top research areas, tool research is mentioned several times as a major finding.

The goal of many of the tools that were presented had a focus of improvement of collaboration/communication, but obviously communication can be a complex research area in itself, as it is a part of the entire software engineering process, i.e. analysis, requirements, design, etc. Given that fact, tools specifically built for the software engineering process need to address each facet from the analysis of the problem to the delivery of the system.

That said, we looked more deeply into the key findings of the papers classified in the tool category which revealed that software engineering tool research is tightly coupled with the papers in the top research areas. Specifically, 72 per cent of the tools papers were also categorized as teamwork research; 50 per cent of the tools papers were also categorized as process and design research; and finally 45 per cent of papers categorized as both tools and coordination research.

Interestingly, only 27 per cent of the tool papers were coupled with papers categorized as global software engineering research. This is concerning because as mentioned earlier, although technology made global software engineering possible, it is also a challenge. We need to understand how technology affects a team's progress and effectiveness (Tannenbaum *et al.*, 2012). This is especially true, as technology is a critical component to the global software engineering process.

Going one level deeper into the findings of the papers categorized as both global and tool research, we found that 36 per cent of them presented the development of a new tools to support the software engineering process and 64 per cent of the research were tool reviews, investigations of tools to support a specific task such as software artifact inspection, investigation of tools to aid in communication and collaboration or investigations to find more productive ways to utilize tools (e.g. anonymity and "tagging"). Thus, the specific gap we are highlighting here are tools to support the global software engineering effort as the research in this area seemed disparate. We also want to point out a deficiency determined during the mapping study process that should be considered. This deficiency is the formality of the keyword selection by researchers. As mentioned earlier, some keywords that should have been included by the authors were not; thus, to determine the themes presented in Section 3.3, we used both the keywords and the findings determined from reading the full text of the papers. Therefore, another gap could be construed as improving the keyword choices by researchers. Publisher categories such as the ACM CCS is one solution but also can be limiting if used in isolation. A possible solution is a combination of both a publisher category list and author keywords not included in the publisher list.

## 4. Threats to validity and limitations

In all studies, the potential threats to validity must be considered. In systematic literature reviews or mapping studies in particular, the limitation with the most focus is the publication selection and inaccurate data extraction (Dyba and Dingsoyr, 2008; Kitchenham *et al.*, 2011). For example, Kitchenham *et al.* (2011) highlighted a data collection bias when researchers favor a type of study. In the case of that particular study, systematic mapping studies were favored and it may have caused the presentation of benefits rather than problems.

Because we favor the benefits of teamwork in software engineering, a bias may be introduced in our data reporting of teamwork – focusing on benefits rather than problems. Thus, we made every attempt to remove that bias and any other bias in this mapping study by using the systematic process described in Section I of this paper. For instance, in addition to the paper selection criteria outlined earlier in this study, we used the study research questions to assist in the study selection, in an attempt to remove the paper selection bias (Dyba and Dingsoyr, 2008). We also made an attempt to remove bias from the keyword content analysis results by using a co-author peer validation process described earlier and a systematic process in the analysis including abstract and full text reviews.

## 5. Conclusion and roadmap for future research

In this study, 103 papers met the inclusion criteria for this mapping study. We presented an analysis and synthesis of the key findings of each paper and a taxonomy highlighting the research areas software engineering team research infiltrates. Using the resulting taxonomy (shown in Table III), we categorized each paper using a keyword content analysis process. The resulting categorizations are included in the Appendix. We then used the paper categorization results to rank the research areas. The Appendix also includes the major key findings of each paper and a complete citation list of the papers selected for this study. We also presented key finding themes in the top three taxonomy categories in Sections 3.3 and 3.4.

To summarize, the top three research areas within the software engineering teams are teamwork/collaboration, process/design and coordination. Individually and in every combination of those top areas, there are clear themes of analysis to understand how collaboration among team members, factors affecting their success and interactions among all project stakeholders. This theme is an obvious input to the second clear theme which includes tools to support team collaboration, improve the effectiveness of software engineering processes and support the coordination among and within teams. Tool research (e.g. new tools developed and tool evaluation) was in fact coupled with many of the papers that fell in the top research areas. In other words, effective tools would be able to integrate knowledge and facilitate software team collaboration.

Another noticeable theme appeared among all combinations of the coordination categories which was research in the area of dispersed or virtual team interaction. Interestingly, tool papers were also tightly coupled with global software engineering research. Understandably, dispersed development is more feasible with effective collaboration tools. However, even though this type of development is feasible, the tools available may not be effective. This fact may indicate the reason for the research theme of project management issues in areas such as team structure, coordination and communication process improvement in the area of dispersed teams.

The two gaps determined by the analysis conducted in this mapping study as presented in Section 3.4 are tools to support all communication efforts involved in the global software engineering process and a more rigorous keyword selection process. These results along with the key finding themes presented earlier will help guide future research in an area that touches all parts of the software engineering and development processes.

## References


ACM (2012), available at: www.acm.org/about/class/2012

Agerfalk, P. and Fitzgerald, B. (2006), "Flexible and distributed software process: old pentunias in new bowls?", *Communications of the ACM*, Vol. 49 No. 10, pp. 27-34.

Brereton, P., Kitchenham, B., Budgen, D., Turner, M. and Khalil, M. (2007), "Lessons from applying a systematic literature review process within the software engineering domain", *The Journal of Systems and Software*, Vol. 80 No. 4, pp. 571-583.

Casey, V. (2010), "Developing trust in virtual software development teams", *Journal of Theoretical and Applied Electronic Commerce Research*, Vol. 5 No. 2, pp. 41-58.

da Silva, F., Costa, C., Franca, C. and Prikladinicki, R. (2010), "Challenges and solutions in distributed software development project management: a systematic literature review," *International Conference on Global Software Engineering*, *IEEE*, *Princeton, NJ*, pp. 87-96.

Dingsoyr, T., Faegri, T.E., Dyba, T., Haugset, B. and Lindsjorn, Y. (2016), "Team performance in software development: research results versus Agile principles", *IEEE Software*, Vol. 33 No. 4, pp. 106-110.

Dyba, T. and Dingsoyr, T. (2008), "Empirical studies of Agile software development: a systematic review", *Information and Software Technology*, Vol. 50 Nos 9/10, pp. 833-859.

Ghobadi, S. and Mathiassen, L. (2015), "Perceived barriers to effective knowledge sharing in Agile software teams", *Journal of Information Systems*, Vol. 26 No. 2.

Gibbs, J., Sivunen, A. and Boyraz, M. (2017), "Investigating the impacts of team type and design on virtual team processes", *Human Resource Management Review*, Vol. 27 No. 4, pp. 590-603.

Hoonlor, A., Szymanski, B. and Zaki, M. (2013), "Trends in computer science research", *Communications of the ACM*, Vol. 56 No. 10, pp. 74-83.

Hsieh, H.-F. and Shannon, S. (2005), "Three approaches to qualitative content analysis", *Qualitative Health Research*, Vol. 15 No. 9, pp. 1277-1288.

Kitchenham, B. (2004), "Procedures for performing systematic reviews," Keele University Technical Report, TR/SE-0401, available at: www.inf.ufsc.br/~awangenh/kitchenham.pdf (accessed 24 January 2016).

Kitchenham, B., Budgen, D. and Brereton, P. (2011), "Using mapping studies as the basis for further research", *Information and Software Technology*, Vol. 53 No. 6, pp. 638-651.

Kitchenham, B., Budgen, D. and Brereton, P. (2016), *Evidence-Based Software Engineering and Systematic Reviews*, CRC Press, Boca Raton.

Mathieu, J., Maynard, M.T., Rapp, T. and Gilson, L. (2008), "Team effectiveness 1997-2007: a review of recent advancements and a glimpse into the future", *Journal of Management*, Vol. 34 No. 3, pp. 410-476.

Powell, A., Piccoli, G. and Ives, B. (2004), "Virtual teams: a review of current literature and directions for future research", *Advances in Information Systems*, Vol. 35 No. 1, pp. 6-36.

Salas, E., Sims, D. and Burke, C.S. (2005), "Is there a 'big five' in teamwork?", *Small Group Research*, Vol. 36 No. 5, pp. 555-599.

Smite, D., Wohlin, C., Gorschek, T. and Feldt, R. (2010), "Empirical evidence in global software engineering: a systematic review", *Empirical Software Engineering*, Vol. 15 No. 1, pp. 91-118.

Tannenbaum, S., Mathieu, J., Salas, E. and Cohen, D. (2012), "Teams are challenging: are research and practice evolving fast enough?", *Industrial and Organizational Psychology*, Vol. 5 No. 1, pp. 2-24.

## Appendix

**Table AI.** Mapping study data

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 1 | S.T. Acuna and N. Juristo | Assigning People to Roles in Software Projects | 675-696 | *Softw. Pract. Exper.* | 34 | 7 | 2004 | Assigning people to roles according to their capabilities and the capabilities demanded by the role improves software development | PT |
| 2 | E. Arisholm and H. Gallis and T. Dyba and D.I.K. Sjoberg | Evaluating Pair Programming with Respect to System Complexity and Programmer Expertise | 65-86 | *IEEE Trans. Softw. Eng.* | 33 | 2 | 2007 | If a complex task there was a 48% increase in the proportion of correct solutions but no reduction in time required. For a simpler system, 20% decrease in time required but no difference in correctness | APQ |
| 3 | J. Arsenyan, G. Buyukozkan | Modeling collaborative software development using axiomatic design principles | 6 | *International Journal of Computer Science* | 36 | 3 | 2009 | Developed guidelines based on the Axiomatic Design (related to customer needs) principles to increase success and effectiveness of collaborative software development | PT |
| 4 | H. Bani-Salameh, C. Jeffery | collaborative and social development environments: a literature review | 89-103 | *International Journal of Computer Applications and Technology* | 49 | 2 | 2014 | Reviews the state of the art in collaborative development environments as well as their evolution | TL |
| 5 | A. Begel and N. Nagappan and C. Poile and L. Layman | Coordination in Large-scale Software Teams | 1-7 | Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering | | | 2009 | Presented the most common objects of coordination among large software development teams: schedules and features – not code or interfaces | GPTO |

(*continued*)

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 6 | S. Bendifallah and W. Scacchi | Work Structures and Shifts: An Empirical Analysis of Software Specification Teamwork | 260-270 | Proceedings of the 11th International Conference on Software Engineering | | | 1989 | Data showing the recurrence of various kinds of shifts in the teams' work structures. For example, separative work structures were associated with improved specification teamwork efficiency, whereas integrative work structures were associated with improved specification product quality | PT |
| 7 | B. Berenbach and M. Gall | Toward a Unified Model for Requirements Engineering | 237-238 | Proceedings of the IEEE International Conference on Global Software Engineering | | | 2006 | Proposed an extension of UML for requirements engineering to improve clarity and facilitate communication when teams are at different locations and/or are lacking in domain expertise | GPC |
| 8 | T. Bipp and A. Lepper and D. Schmedding | Pair Programming in Software Development Teams - An Empirical Study of Its Benefits | 231-240 | Inf. Softw. Technol. | 50 | 3 | 2008 | Pair programming teams produced nearly as much code as the teams of individual workers at the same time. In addition, the code produced by the paired teams was easier to read and to understand. This facilitates finding errors and maintenance | PQ |

(*continued*)

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 9 | C. Bird and N. Nagappan and P. Devanbu and H. Gall and B. Murphy | Does Distributed Development Affect Software Quality? An Empirical Case Study of Windows Vista | 518-528 | Proceedings of the 31st International Conference on Software Engineering | | | 2009 | In evaluation of Windows Vista development data, were able to refute conventional beliefs that distributed software development is riskier and more challenging than co-located development | GPCOQ |
| 10 | C. Bird and D. Pattison and R. D'Souza and V. Filkov and P. Devanbu | Latent Social Structure in Open Source Projects | 24-35 | Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering | | | 2008 | Sub-communities spontaneously arise within OSS projects as the projects evolve | GT |
| 11 | A. Boden and G. Avram | Bridging Knowledge Distribution - The Role of Knowledge Brokers in Distributed Software Development Teams | 8-11 | Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering | | | 2009 | Case study showing how small software companies rely on the commitment of particular team members and informal knowledge management practices | GPCQ |
| 12 | G. Borchers | The Software Engineering Impacts of Cultural Factors on Multi-cultural Software Development Teams | 540-545 | Proceedings of the 25th International Conference on Software Engineering | | | 2003 | Describes how cultural factors (Japan, India, the USA) impacted software engineering techniques used on the projects | PTO |
| 13 | B. Bruegge; A.H. Dutoit; T. Wolf | Sysiphus: Enabling informal collaboration in global software development | 139-148 | 2006 IEEE International Conference on Global Software Engineering (ICGSE) | | | 2006 | Developed a tool that can provide a uniform framework for system models, collaboration artifacts, and organizational models | TCOL |

*(continued)*

**Table AI.**

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 14 | G. Canfora and A. Cimitile and F. Garcia and M. Piattini and C.A. Visaggio | Evaluating Performances of Pair Designing in Industry | 1317-1327 | J. Syst. Softw. | 80 | 8 | 2007 | Pair programming used in the design phase of software development - improves the outcome but slows the process | P |
| 15 | G. Canfora and M. Di Penta and R. Oliveto and S. Panichella | Who is Going to Mentor Newcomers in Open Source Projects? | 1-14 | Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering | | | 2012 | Developed a system that identifies and recommends mentors in software projects by mining data from mailing lists and versioning systems. The mentors are identified among experienced developers who actively interact with newcomers | G |
| 16 | V. Casey and I. Richardson | Project Management Within Virtual Software Teams | 33-42 | Proceedings of the IEEE International Conference on Global Software Engineering | | | 2006 | Presented six specific project management related areas that need to be addressed to facilitate successful virtual team operation | GTO |
| 17 | V. Casey and I. Richardson | Virtual Teams: Understanding the Impact of Fear | 511-526 | Softw. Process | 13 | 6 | 2008 | Highlighted the importance and negative impact fear (e.g. job loss due to remote colleagues) played and the consequences this had for the success of the location of the collaborating teams | GTO |

(*continued*)

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 18 | M. Cataldo and J.D. Herbsleb | Communication Patterns in Geographically Distributed Software Development and Engineers' Contributions to the Development Effort | 25-28 | Proceedings of the 2008 International Workshop on Cooperative and Human Aspects of Software Engineering | | | 2008 | Showed the communication impact of defined formal roles on a geographically distributed software development team | GOL |
| 19 | M. Cataldo and J.D. Herbsleb | Factors Leading to Integration Failures in Global Feature-oriented Development: An Empirical Analysis | 161-170 | Proceedings of the 33rd International Conference on Software Engineering | | | 2011 | Showed that in feature driven development (a light-weight Agile process) technical factors such as nature of component dependences and organizational factors such as the geographic dispersion of the feature teams and the role of the feature owners had an effect on software quality | GPQ |
| 20 | S. Cherry and P. N. Robillard | Audio-video Recording of Ad Hoc Software Development Team Interactions | 13-21 | Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering | | | 2009 | Presented data to quantify interactions of team members with respect to the various roles. For example, the relationship between the roles people play within a team and the time they spend in ad hoc (interrupting someone else's work) collaboration activities | TO |

(*continued*)

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 21 | J. Chong and T. Hurlbutt | The Social Dynamics of Pair Programming | 354-363 | Proceedings of the 29th International Conference on Software Engineering | | | 2007 | Pair programming did not conform to the roles of "driver" and "navigator". Instead they moved together through different phases of the task – discussing issues at the same strategic level of abstraction and in the same role | PO |
| 22 | E.O. Conchuir and H. Holmstrom and P.J. Agerfalk and B. Fitzgerald | Exploring the Assumed Benefits of Global Software Development | 159-168 | Proceedings of the IEEE International Conference on Global Software Engineering | | | 2006 | Analyzed the benefits of global software development. For example, although there is an eightfold developer salary differential between USA and India – there is significant overhead in communication, coordination, and control – thus this benefit is only partially realized | TCOQ |
| 23 | E.J. da Silva; C.A. Tacla; J. P.A. Barthes; M.P. Ramos; E.C. Paraiso | A collaborative virtual workspace for software development | 24-29 | 2015 IEEE 19th International Conference on Computer Supported Cooperative Work in Design | | | 2015 | Developed a tool to improve collaboration among participants, providing semantic information within the context of software development | PTL |

*(continued)*

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 24 | R.C. de Boer and H. van Vliet | Writing and Reading Software Documentation: How the Development Process May Affect Understanding | 40-47 | Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering | | | 2009 | There are different levels of shared understanding – which appears to be tied to the development process employed | PTC |
| 25 | A. De Lucia and F. Fasano and G. Tortora and G. Scanniello | Assessing the Effectiveness of a Distributed Method for Code Inspection: A Controlled Experiment | 252-261 | Proceedings of the International Conference on Global Software Engineering | | | 2007 | Proposed a distributed inspection method that minimizes synchronous collaboration among team members by identifying defects in software artefacts that can be dealt with asynchronously | PTOQL |
| 26 | P. Dewan and P. Agarwal and G. Shroff and R. Hegde | Distributed Side-by-side Programming | 48-55 | Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering | | | 2009 | Analyzed side-by-side programming, a variation of pair programming. Discussed the three modes: coupled programming, concurrent browsing programming and concurrent browsing | PTOQ |
| 27 | T.A. dos Santos, R.M. de Araujo, A.M. Magdaleno | Identifying collaboration patterns in software development social networks | 51-60 | *Journal of Computer Science* | | | 2010 | Proposed a SNA approach to ID collaboration patterns in software development | PT |
| 28 | Y. Dubinsky and O. Hazzan | Using a Role Scheme to Derive Software Project Metrics | 693-699 | *J. Syst. Archit.* | 52 | 11 | 2006 | Developed a role scheme in order to raise team member accountability in software development projects | TO |

(*continued*)

**Table AI.**

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 29 | K. Dullemond; B. van Gameren | What Distributed Software Teams Need to Know and When: An Empirical Study | 61-70 | 2013 IEEE 8th International Conference on Global Software Engineering (ICGSE) | | | 2013 | Reported on the prioritization of information for distributed software engineers based on their current activity and status | GTC |
| 30 | T. Dyba; and E. Arisholm and D.I.K. Sjoberg and J.E. Hannay and F. Shull | Are Two Heads Better Than One? On the Effectiveness of Pair Programming | 12-15 | IEEE Softw. | 24 | 6 | 2007 | Discussed the findings of 15 pair programming studies that were conducted to determine pair programming effectiveness | PTQ |
| 31 | K. Ehrlich and K. Chang | Leveraging Expertise in Global Software Teams: Going Outside Boundaries | 149-158 | Proceedings of the IEEE International Conference on Global Software Engineering | | | 2006 | Presented how people on a global software team acquire and used available expertise. For example, information was acquired from people outside their software team | GPCO |
| 32 | A.C.C. Franca and F.Q.B. da Silva | Designing Motivation Strategies for Software Engineering Teams: An Empirical Study | 84-91 | Proceedings of the 2010 ICSE Workshop on Cooperative and Human Aspects of Software Engineering | | | 2010 | Determined a set of motivators influences the software engineers' motivation | TO |
| 33 | R. Frost | Jazz and the Eclipse Way of Collaboration | 114-117 | IEEE Software | 24 | 6 | 2007 | Created a tool to make teams and teams of teams more productive | GTL |
| 34 | T. Gandomani, H. Zulzalil, A.A.A. Ghani, A.B. Md. Sultan, K.Y. Sharif | How human aspects impress Agile software development transitions and adoption | 129-148 | *International Journal of Software Engineering and its Applications* | 8 | 1 | 2014 | Identified and classified diverse human aspects of Agile transformation processes | ATO |

(*continued*)

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 35 | R.G. Gowda and D.J. Polzella | Comparison of Selected Survey Instruments for Software Team Communication Research | 43-54 | Proceedings of the IEEE International Conference on Global Software Engineering | | | 2006 | Presented a survey of validated team communication instruments that could be tailored for software team research | T |
| 36 | U. Hahn and M. Jarke and T. Rose | Teamwork Support in a Knowledge-Based Information Systems Environment | 467-482 | *IEEE Trans. Softw. Eng.* | 17 | 5 | 1991 | Presented a tool called CoNeX. It integrates different tasks encountered in software projects via a conceptual modeling strategy. The tool has an emphasis on integrating the semantics of software development with aspects of group work | PL |
| 37 | L. Hattori | Enhancing Collaboration of Multi-developer Projects with Synchronous Changes | 377-380 | Proceedings of the 32Nd ACM/IEEE International Conference on Software Engineering - Volume 2 | | | 2010 | Developed a modeling tool to enhance team awareness and collaboration. The tool provides relevant information to developers in a non-intrusive manner | TL |
| 38 | L. Hattori and M. Lanza | Syde: A Tool for Collaborative Software Development | 235-238 | Proceedings of the 32Nd ACM/IEEE International Conference on Software Engineering – Volume 2 | | | 2010 | Developed a tool to reestablish team awareness by sharing change and conflict information across developer workspace | TOL |
| 39 | S.M. Henry and K.T. Stevens | Using Belbin's Leadership Role to Improve Team Effectiveness: An Empirical Investigation | 241-250 | *J. Syst. Softw.* | 44 | 3 | 1999 | A single software engineering team leader improved team performance over teams with multiple leaders or no leader | T |

(*continued*)

**Table AI.**

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 40 | J.D. Herbsleb and H. Klein and G.M. Olson and H. Brunner and J.S. Olson and J. Harding | Object-oriented Analysis and Design in Software Project Teams | 249-292 | *Hum.-Comput. Interact.* | 10 | 2 | 1995 | OOD reduced the need for clarification in design discussions, differences in participation, and how meeting time is spent | PL |
| 41 | J.D. Herbsleb and A. Mockus | Formulation and Preliminary Test of an Empirical Theory of Coordination in Software Engineering | 138-137 | Proceedings of the 9th European Software Engineering Conference Held Jointly with 11th ACM SIGSOFT International Symposium on Foundations of Software Engineering | | | 2003 | Identified an empirical theory of coordination in software engineering, where the problem of coordination is ensuring the mutual constraints among engineering decisions are recognized and correctly acted upon as the engineering work proceeds | CO |
| 42 | J.D. Herbsleb and D.J. Paulish and M. Bass | Global Software Development at Siemens: Experience from Nine Projects | 524-533 | Proceedings of the 27th International Conference on Software Engineering | | | 2005 | Present lessons learned on project management, division of labor, ongoing coordination of technical work and communication of nine global software development projects | G |
| 43 | H. Holmstrom and E.O. Conchuir and P.J. Agerfalk and B. Fitzgerald | Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-Cultural Distance | 3-11 | Proceedings of the IEEE International Conference on Global Software Engineering | | | 2006 | Present a case study highlighting challenges and solutions associated with managing global software development of three US-based companies operating in Ireland | GTCOL |

(*continued*)

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 44 | C. Jergensen and A. Sarma and P. Wagstrom | The Onion Patch: Migration in Open Source Ecosystems | 70-80 | Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering | | | 2011 | Showed that prior experience in an OSS project does not have a huge effect on the overall centrality of a developer's contribution. This contradicts existing assumptions of the onion model. Onion model in an OSS project is when a newcomer joins by first contributing at the periphery through mailing list discussions and bug trackers | GPT |
| 45 | A. Jermakovics and A. Sillitti and G. Succi | Mining and Visualizing Developer Networks from Version Control Systems | 24-31 | Proceedings of the 4th International Workshop on Cooperative and Human Aspects of Software Engineering | | | 2011 | Presented an approach to reveal the structure of OSS development teams - can be used to improve the modularity in visualization of developer networks | GP |
| 46 | I. Kwan and D. Damian | The Hidden Experts in Software-engineering Communication (NIER Track) | 800-803 | Proceedings of the 33rd International Conference on Software Engineering | | | 2011 | Determined why people emerge in discussions: when crisis occurs, response to explicit requests, response to forwarded announcements, and response to a follow up on a previous meeting | T |

(*continued*)

**Table AI.**

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 47 | T.D. LaToza and G. Venolia and R. DeLine | Maintaining Mental Models: A Study of Developer Work Habits | 492-501 | Proceedings of the 28th International Conference on Software Engineering | | | 2006 | Analyzed the work habits of developers. For example, one finding was that developers were forced to invest a lot of effort recovering implicit knowledge by exploring code and interrupting teammates as design documents were inadequate | APCO |
| 48 | M. Levy and O. Hazzan | Knowledge Management in Practice: The Case of Agile Software Development | 60-65 | Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering | | | 2009 | Examined the integration of knowledge management and Agile software development | PT |
| 49 | T.L. Lewis and W.J. Smith | Creating High Performing Software Engineering Teams: The Impact of Problem Solving Style Dominance on Group Conflict and Performance | 12-129 | *J. Comput. Sci. Coll.* | 24 | 2 | 2008 | Negative team outcomes are increased when a team is formed with one dominant problem solving preference (measured by MBTI) of its members | PT |
| 50 | S. Licorish and A. Philpott and S.G. MacDonell | Supporting Agile Team Composition: A Prototype Tool for Identifying Personality (In)Compatibilities | 66-73 | Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering | | | 2009 | Developed and preliminarily evaluated a tool intended to assist software engineers and project managers in forming Agile teams using team member personality information | TOL |

(*continued*)

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 51 | T. Lima, R.P. dos Santos, J. Oliveria, C. Werner | The importance of socio-technical resources for software ecosystems management | 98-113 | *Journal of Innovation in Digital Ecosystems* | 3 | 2 | 2016 | Socio-technical resources have aided collaboration in software development for software ecosystems. Coordination of teams based on more knowledge of actor's tasks and interactions, and monitoring of quality of software ecosystems platforms through the orchestration of the contributions developed by external actors | POL |
| 52 | K. Liukkunen; K. Lindberg; J. Hyysalo; J. Markkula | Supporting Collaboration in the Geographically Distributed Work with Communication Tools in the Remote District SME's | 155-164 | 2010 5th IEEE International Conference on Global Software Engineering | | | 2010 | Determined categories of collaboration challenges. Communication tools are needed to provide an infrastructure for collaborative sessions, to interact with colleagues and customers | GTCL |
| 53 | K.M. Lui and K.C.C. Chan and J. Nosek | The Effect of Pairs in Program Design Tasks | 197-211 | *IEEE Trans. Softw. Eng.* | 34 | 2 | 2008 | Pairs outperformed individuals in program design-related tasks | T |
| 54 | A.M. Magdaleno and M. de Oliveira Barros and C. M. Lima Werner and R.M. de Araujo and C.F.A. Batista | Collaboration Optimization in Software Process Composition | 452-466 | *J. Syst. Softw.* | 103 | C | 2015 | Utilizing a genetic algorithm to navigate the solution space for the software problem will maximize collaboration among team members | PT |

(*continued*)

**Table AI.**

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 55 | M. Mantei | The Effect of Programming Team Structures on Programming Tasks | 106-113 | *Commun. ACM* | 24 | 3 | 1981 | Choosing one of three team structures (Democratic Decentralized, Controlled Decentralized, and Controlled Centralized) depending on the properties of the task (difficulty, size, duration, modularity, reliability, time, sociability) | TO |
| 56 | G. Matturro and Florencia Raschetti and Carina Fontan | Soft Skills in Software Development Teams: A Survey of the Points of View of Team Leaders and Team Members | 101-104 | Proceedings of the Eighth International Workshop on Cooperative and Human Aspects of Software Engineering | | | 2015 | Determined the soft skills valued by team leaders are: communication skills, customer orientation, interpersonal skills, and teamwork. The soft skills valued for team members: analytic, problem solving, commitment, responsibility, eagerness to learn, motivation and teamwork | T |
| 57 | A.E. Milewski | Global and Task Effects in Information-seeking Among Software Engineers | 311-326 | *Empirical Softw. Engg.* | 12 | 3 | 2007 | Software engineer's information seeking preferences: they prefer non-social sources (documentation) when seeking factual information. They prefer social sources (ask friend/coworker) for diagnostic/problem-solving questions | GPT |

(*continued*)

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 58 | A.E. Milewski and M. Tremaine and F. Kobler and R. Egan and S. Zhang and P. O'Sullivan | Guidelines for Effective Eridging in Global Software Engineering | 477-492 | *Softw. Process* | 13 | 6 | 2008 | Based on empirical findings, proposed a set of guidelines for the bridging tactic (location to facilitate collaboration and coordination across locations) expectations | GT |
| 59 | A. Mockus | Organizational Volatility and Its Effects on Software Defects | 117-126 | Proceedings of the Eighteenth ACM SIGSOFT International Symposium on Foundations of Software Engineering | | | 2010 | Determined that the proximity to an organizational change is significantly associated with reductions in software quality. For example, organizational volatility involves overhead for an existing team to train new developers and gaps in tacit knowledge produced by the departure of experienced developers | TQ |
| 60 | A. Mockus | Succession: Measuring Transfer of Code and Developer Productivity | 67-77 | Proceedings of the 31st International Conference on Software Engineering | | | 2009 | Investigated the best code ownership transfer or succession measure to pinpoint the most likely mentors | GPOQ |

(*continued*)

**Table AI.**

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 61 | A. Mockus and R.T. Fielding and J.D. Herbsleb | Two Case Studies of Open Source Software Development: Apache and Mozilla | 309-346 | *ACM Trans. Softw. Eng. Methodol.* | 11 | 3 | 2002 | Presented an analysis of the comparison of Apache and several commercial products then use results to analyze Mozilla. Based on the analysis the research concludes with thoughts about the prospects for high performance commercial/OSS hybrids | GPOQ |
| 62 | M. Mohtashami, T. Marlowe, V. Kirova, F. Deek | risk-driven management contingency policies in collaborative software development | 247-271 | *International Journal of Information Technology and Management* | 10 | #2-4 | 2011 | Presented a collaborative risk management framework, and focus on a set of governing policies and management contingency processes required to support the framework and to manage the interactions with other critical project activities and goals | TCOQ |

(*continued*)

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 63 | M. Mohtashami, C. Sylla, I. Im, F. Deek | A study of information processing effectiveness in inter-organizational collaboration in software development | 233-246 | *International Journal of Information Technology and Management* | 10 | #2-4 | 2011 | This paper confirms that management profile, deployed information technology, and suitable organizational background are all important for the success of inter-organizational communication. In addition, inter-organizational communication has a strong positive effect on the success of SD processes and the quality of the software product | PT |
| 64 | C.V.F. Monteiro and F.Q. B. da Silva and I.R.M. dos Santos and F. Farias and E.S.F. Cardozo and A.R.G. do A. Leitao and D. N.M. Neto and M. J.A.P. Filho | A Qualitative Study of the Determinants of Self-managing Team Effectiveness in a Scrum Team | 16-23 | Proceedings of the 4th International Workshop on Cooperative and Human Aspects of Software Engineering | | | 2011 | Certain determinants (e.g. self-efficacy, self-observation/evaluation, and self-goal setting) can support the adoption of Agile methods while the use of Scrum may affect other determinants. Findings indicate that the adoption of a management method has a complex interplay with the determinants of Agile team effectiveness | AT |

(*continued*)

**Table AI.**

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 65 | M.M. Muller | Two Controlled Experiments Concerning the Comparison of Pair Programming to Peer Review | 166-179 | *J. Syst. Softw.* | 78 | 2 | 2005 | When requiring an additional review phase and forced to produce programs of similar correctness, single developers and pairs of developers become interchangeable in terms of development cost | P |
| 66 | S. Narayanan and S. Mazumder and R. Raju | Success of Offshore Relationships: Engineering Team Structures | 73-82 | Proceedings of the IEEE International Conference on Global Software Engineering | | | 2006 | A comparison of two different offshoring structuring models | GTCO |
| 67 | L.D. Panjer and D. Damian and M. Storey | Cooperation and Coordination Concerns in a Distributed Software Development Project | 77-80 | Proceedings of the 2008 International Workshop on Cooperative and Human Aspects of Software Engineering | | | 2008 | In distributed software development, how are interdependent activities and issues coordinated and recorded. For example, one finding was that over geographic distances, either team members working on tightly coupled work or dependent differing work, coordination was difficult over geographic distances | GPO |

(*continued*)

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 68 | D.E. Perry and H.P. Siy and L.G. Votta | Parallel Changes in Large-scale Software Development: An Observational Case Study | 308-337 | *ACM Trans. Softw. Eng. Methodol.* | 10 | 3 | 2001 | Presented an analysis of the problems encountered in parallel development by teams of developers. For example, there is a significant correlation between the degree of parallel work on a given component and the number of quality problems it has | PQ |
| 69 | A.J. Phuwanartnurak | Interdisciplinary collaboration through wikis in software development | 82-90 | 2009 ICSE Workshop on Wikis for Software Engineering | | | 2009 | Use of Wikis encourages more information sharing during the software development process | PTCL |
| 70 | M. Pikkarainen and J. Haikara and O. Salo and P. Abrahamsson and J. Still | The Impact of Agile Practices on Communication in Software Development | 303-337 | *Empirical Softw. Engg.* | 13 | 3 | 2008 | Agile practices improve both informal and formal communication. However, it further indicates that, in larger development situations involving multiple external stakeholders, a mismatch of adequate communication mechanisms can sometimes even hinder the communication | ACO |

(*continued*)

**Table AI.**

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 71 | R. Prikladnicki and J.L.N. Audy and R. Evaristo | A Reference Model for Global Software Development: Findings from a Case Study | 18-28 | Proceedings of the IEEE International Conference on Global Software Engineering | | | 2006 | Presented a reference model for global software development. Included a description and factors that enable multinational corporations to operate successfully across geographic and cultural boundaries | GPOQ |
| 72 | M. Rao, T. Earls, G. Sanchez | International collaboration in transorganizational systems development: the challenges of global insourcing | 52-69 | *Journal of Global Information Technology Management* | 10 | 3 | 2007 | Presents lessons learned and best practices for companies doing global software development | GTO |
| 73 | N. Ramasubbu and R.K. Balan | Globally Distributed Software Development Project Performance: An Empirical Analysis | 125-134 | Proceedings of the 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering | | | 2007 | Team dispersion significantly reduces development productivity and has effects on conformance quality but can be significantly mitigated through structured software engineering processes | GOQ |

(*continued*)

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 74 | N. Ramasubbu and M. Cataldo and R.K. Balan and J.D. Herbsleb | Configuring Global Software Teams: A Multi-company Analysis of Project Productivity, Quality, and Profits | 261-270 | Proceedings of the 33rd International Conference on Software Engineering | | | 2011 | Presented the impacts of project-level configurational choices of global software teams. For example, achieving higher levels of productivity and quality require diametrically opposed configurational choices | GOQ |
| 75 | J. Rasheed, F. Azam, M.A. Iqbal | Most effective communication management techniques for geographically distributed project team members | 530-541 | *International Journal of Advancements in Technology* | 2 | 4 | 2011 | Describes communication management using data mining and knowledge based tools to extract data and knowledge of social structures | GCO |
| 76 | M. Ressin and J. Abdelnour-Nocera and A. Smith | Of Code and Context: Collaboration Between Developers and Translators | 50-52 | Proceedings of the 4th International Workshop on Cooperative and Human Aspects of Software Engineering | | | 2011 | For software developed for international customers the researchers suggest that a translator should be integrated into the development team to understand each other's work better | GT |
| 77 | P.N. Robillard and M.P. Robillard | Types of Collaborative Work in Software Engineering | 219-224 | *J. Syst. Softw.* | 53 | 3 | 2000 | The characteristics of four types of collaborative activities are described and a quantitative breakdown of how people spend their time in collaboration within a single software engineering project is presented | T |

(*continued*)

**Table AI.**

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 78 | P. Rola, D. Kuchta, D. Kopczyk | Conceptual model of working space for Agile (scrum) project team | 49-63 | *Journal of Systems and Software* | 118 | | 2016 | Describe a validated office space arrangement for teams using the Agile methodology | AT |
| 79 | S. Salinger and F. Zieris and L. Prechelt | Liberating Pair Programming Research from the Oppressive Driver/Observer Regime | 1201-1204 | Proceedings of the 2013 International Conference on Software Engineering | | | 2013 | Present a more realistic pair programming role model where multiple roles can be held by one person at the same time, and some of their facets are subtle | P |
| 80 | A. Sarma; D. Redmiles; A. v.d. Hoek | Categorizing the Spectrum of Coordination Technology | 61-67 | Computer | 43 | 6 | 2010 | Created a collaboration tool classification system (organized according to five coordination paradigms)– serving as a road map for improving an organization's coordination practices | TOL |
| 81 | S. Sawyer | Effects of intra-group conflict on packaged software development team performance | 155-178 | *Information Systems Journal* | 11 | 2 | 2001 | A combination of a team's characteristics (social pressures, interdependence, resource sharing), team member characteristics (behaviors and individual differentiation) and existing levels of intragroup conflict account for one half of the variance between the best and worst performing teams | TQ |

(*continued*)

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 82 | V.S. Sharma and V. Kaulgud | Studying Team Evolution During Software Testing | 72-75 | Proceedings of the 4th International Workshop on Cooperative and Human Aspects of Software Engineering | | | 2011 | Provides insights in the changes in team interactions and individual roles during the testing phase | PT |
| 83 | V. Shipp and P. Johnson | Supporting Collaboration in the Development of Complex Engineering Software | 84-87 | Proceedings of the 4th International Workshop on Cooperative and Human Aspects of Software Engineering | | | 2011 | The frequent informal design discussions that occur between users and developers have inadequate support provided by formal documentation | T |
| 84 | A. Sillitti and G. Succi and J. Vlasenko | Toward a Better Understanding of Tool Usage (NIER Track) | 832-835 | Proceedings of the 33rd International Conference on Software Engineering | | | 2011 | Modeled developer tool use data to better understand how tools are used individually and collectively | L |
| 85 | S.E. Sim and R.C. Holt | The Ramp-up Problem in Software Projects: A Case Study of How Software Immigrants Naturalize | 361-370 | Proceedings of the 20th International Conference on Software Engineering | | | 1998 | Present lessons learned regarding joining an established software development team. For example, a concentration on pragmatics (information that is immediately useful) enabled the new comer to start working with source code quickly | PTOL |

(*continued*)

**Table AI.**

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 86 | K. Smolander, M. Rossi, S. Pekkola | Collaboration Change in Enterprise Software Development | 68-74 | Proceedings of the 9th International Workshop on Cooperative and Human Aspects of Software Engineering | | | 2016 | Illustrate four different modes of collaboration (contract, cooperation, personified, and process) that are each a main driver of development under different circumstances | TO |
| 87 | D. Socha and K. Sutanto | The Pair As a Problematic Unit of Analysis for Pair Programming | 64-70 | Proceedings of the Eighth International Workshop on Cooperative and Human Aspects of Software Engineering | | | 2015 | Pair programmers spend 20% of their pairing time interacting with people outside the pair | N/A |
| 88 | H. Spanjers and M. ter Huurne and B. Graaf and M. Lormans and D. Bendas and R. van Solingen | Tool Support for Distributed Software Engineering | 187-198 | Proceedings of the IEEE International Conference on Global Software Engineering | | | 2006 | Presented an investigation of a software process system and define a list of features for systems that support distributed software engineering | GPTL |
| 89 | M.A. Storey; J. Ryall; R.I. Bull; D. Myers; J. Singer | TODO or to bug | 251-260 | 2008 ACM/IEEE 30th International Conference on Software Engineering (ICSE) | | | 2008 | Task annotations embedded within the source code support a variety of activities that are fundamental to articulation work within software development task management | P |

(*continued*)

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 90 | S. Syed-Abdullah and M. Holcombe and M. Gheorge | The Impact of an Agile Methodology on the Well Being of Development Teams | 143-167 | *Empirical Softw. Engg.* | 11 | 1 | 2006 | Agile methodology has a positive effect on the enthusiasm of software developers. It showed to be related to the cognitive, affective and managerial properties of the XP practices | AT |
| 91 | J. Teixeira, G. Robles, J. Gonzalez-Barahona | lessons learned from applying social network analysis on an industrial free/libre/open source software ecosystem | 27 | *Journal of Internet Services and Applications* | 6 | 14 | 2015 | Present how rival firms collaborate on OSS projects. In addition present how heterogeneous actors (individuals, startups, established firms and public organizations) can collaborate on a complex infrastructure for big-data in the open source arena | GTO |
| 92 | A. Thurimella, T. Wolf | Issue-based Variability Modeling | 11-22 | International Conference on Global Software Engineering | | | 2007 | Tested a tool to be used for a software product line variability. The tool supports information collaboration for variability management. Instantiation and evolution of variability models are supported | N/A |

(*continued*)

**Table AI.**

**Table AI.**

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 93 | C. Treude | The Role of Emergent Knowledge Structures in Collaborative Software Development | 389-392 | Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering | | | 2010 | Studied emergent knowledge structures created use lightweight technologies in software development tools. For example, lifecycle management, identification of cross-cutting concerns, categorization of work and several idiosyncratic processes were supported by tagging | T |
| 94 | C. Treude and M. Storey | How Tagging Helps Bridge the Gap Between Social and Technical Aspects in Software Development | 12-22 | Proceedings of the 31st International Conference on Software Engineering | | | 2009 | Developed "tagging" - a lightweight social computing mechanism, to bridge the gap between social the technical and social aspects of software development teams | PTOL |
| 95 | C. Treude; M.A. Storey | Work Item Tagging: Communicating Concerns in Collaborative Software Development | 19-34 | IEEE Transactions on Software Engineering | 38 | 1 | 2012 | Creating a tagging mechanism which may play a role in improving team-based software development practices | PT |

(*continued*)

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 96 | H. Unphon and Y. Dittrich and A. Hubaux | Taking Care of Cooperation when Evolving Socially Embedded Systems: The PloneMeeting Case | 96-103 | Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering | | | 2009 | Confirmed understanding of what is relevant when estimating the evolvability of socially embedded systems. For example, using their framework they were able to understand the impact of changes in socially embedded systems on the cooperation among developers and between users and developers | GPL |
| 97 | P. Vir Singh | The Small-world Effect: The Influence of Macro-level Properties of Developer Collaboration Networks on Open-source Project Success | 1-27 | *ACM Trans. Softw. Eng. Methodol.* | 20 | 2 | 2010 | There is a relationship between small-world (a specific form of global network structure) properties of a community and the technical and commercial success of the software produced by its members. In other words the network size influence project success | GTCQ |
| 98 | P. Vitharana and K. Ramamurthy | Computer-Mediated Group Support, Anonymity, and the Software Inspection Process: An Empirical Investigation | 167-180 | *IEEE Trans. Softw. Eng.* | 29 | 2 | 2003 | Teams with support for anonymity are more effective in identifying the seeded errors in the more complex task. They also express a more positive attitude toward code inspection tasks | PTQL |

(*continued*)

**Table AI.**

Table AI.

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 99 | J. Vlietland, R. van Solingen, H. van Vliet | Aligning codependent scrum teams to enable fast business value delivery: a governance framework and set of intervention actions | 418-429 | *Journal of Systems and Software* | 113 | | 2015 | Identified validated interventions that mitigate collaboration issues between codependent scrum teams. The interventions showed to facilitate faster delivery as well | ATO |
| 100 | G.M.P. Wanderley; M.P. Ramos; C.A. Tacla; G.Y. Sato; E.J.d. Silva; E.C. Paraiso | MODUS-SD: User modeling in collaborative software development | 372-378 | Proceedings of the 2012 IEEE 16th International Conference on Computer Supported Cooperative Work in Design (CSCWD) | | | 2012 | Created a tool to model a user's knowledge and preferences. The tool follows each user during daily work | PT |
| 101 | T. Wolf and A. Schroter and D. Damian and T. Nguyen | Predicting Build Failures Using Social Network Analysis on Developer Communication | 1-11 | Proceedings of the 31st International Conference on Software Engineering | | | 2009 | Communication structure measures cannot individually predict failed builds. A set of communication structure measures can be combined into a model that predicts filed integrations builds | TCQ |
| 102 | Q. Xuan, V. Filkov | Building it Together: Synchronous Development in OSS | 222-233 | Proceedings of the 36th International Conference on Software Engineering | | | 2014 | The goal was to identify synchronized activities in OSS projects. For example, there is a strong correlation between synchronized co-commits and communication | GPTC |

(*continued*)

| # | Author | Title | Page | Journal or Conference name | Vol. | No. | Year | Key finding | Class |
|---|---|---|---|---|---|---|---|---|---|
| 103 | L. Yilmaz and J. Phillips | The Impact of Turbulence on the Effectiveness and Efficiency of Software Development Teams in Small Organizations: Research Sections | 247-265 | Softw. Process | 12 | 3 | 2007 | Explored the effects of team behavior on the efficiency and effectiveness of software development organizations that pursue incremental and iterative processes such as RUP. They observed that "agility" via incremental and iterative development is valid and usual as well as the autonomous and concurrent team archetypes are better suited for large projects | PT |

**Table AI.**

## About the authors

Joanna F. DeFranco earned her PhD in Computer and Information Science from New Jersey Institute of Technology, MS in Computer Engineering from Villanova University and BS in Electrical Engineering and Math from Penn State University. She is an Assistant Professor of Software Engineering with the Pennsylvania State University. She has worked as an Electronics Engineer for the Navy and as a Software Engineer at Motorola. Her research interests include software engineering teams, project management, and effective collaboration. Joanna F. DeFranco is the corresponding author and can be contacted at: jfd104@psu.edu

Phillip Laplante (M'86–SM'90–F'08) received BS in Systems Planning and Management, MEng in Electrical Engineering and PhD in Computer Science from the Stevens Institute of Technology, Hoboken, NJ, in 1983, 1986 and 1990, respectively, and MBA from the University of Colorado, in 1999. He is a Professor of Software and Systems Engineering at Penn State. His research interests include requirements engineering, the internet of things, safety critical software systems and software quality.