2012 IEEE Seventh International Conference on Global Software Engineering

Adopting a Social Network Perspective in Global Software Development

Christina Manteli
Dept of Computer Science
VU University Amsterdam
Amsterdam, The Netherlands
Email: cmanteli@cs.vu.nl

Hans van Vliet
Dept of Computer Science
VU University Amsterdam
Amsterdam, The Netherlands
Email: hans@cs.vu.nl

Bart van den Hooff
Faculty of Economics and Business Administration
VU University Amsterdam
Amsterdam, The Netherlands
Email: bhooff@feweb.vu.nl

***Abstract*—In the past few years, software engineering researchers have adopted social network analysis techniques to understand collaboration patterns in global software teams. In this paper, we investigate current research in global software development where social network theory is used as an analysis technique. We do so through a systematic literature review where we collect and analyze previous work that adopt a social network perspective in distributed software development. We use the 3C collaboration model to classify our results based on the communication, coordination and cooperation aspects of global networks. Our results reveal two main coordination structures used in distributed teams, namely the clustering and the core-periphery structure. The analysis of the cooperation activities of the global networks reveal differences between planning and practice. Finally, several tools have been identified that aim to improve communication patterns among distributed team members.**

***Keywords*-global software development; social network analysis; distributed teams;**

I. Introduction

In the past few years, software engineering researchers started to adopt social network analysis (SNA) techniques to understand collaboration patterns in global software teams [1]. According to [2], SNA can be used to reveal collaboration patterns that go beyond the organizational charts and the project planning. Additionally, a social network perspective can help locate expertise, raise communities of practice, promote knowledge sharing, and improve strategic decision making across distributed teams [3]. In previous research, social network analysis has been used to help organizations reveal hidden issues among the global software teams [1].

In this paper, we investigate current research in global software development (GSD) where social network theory is used as an analysis technique. We do so through a systematic literature review study where we collect and analyze previous work that adopt a social network perspective in distributed software development. We explore the new insights that both academics and practitioners can gain, by investigating the collaboration of distributed teams in GSD with the use of social network analysis techniques.

The results reveal new aspects in the coordination, cooperation and communication patterns of the remote team members and the impact of these structures to the team awareness and collaboration. The "default" assumption is that remote teams by and large operate independently. The work is coordinated per site; team members communicate a lot with other team members at the same site, and fairly little with team members at another site. The coordination between sites is delegated to special intermediaries, variously known as boundary spanners or brokers. Social network analysis research in global software development reveals interesting new patterns: the core-periphery structure and the appearance of emergent team members. In a core-periphery structure, people in the core – irrespective of site boundaries – interact frequently with each other, and much less so with team members at the periphery. Emergent team members can be people that are not officially part of the team, but yet heavily participate in the team communication, or people that take on the role of boundary spanner or broker unofficially. In both cases, social network analysis reveals the actual collaboration, as opposed to the one from the organization charts.

In the following section, we provide a theoretical background on the benefits of using SNA in global development environments (Section II). Sections III and IV present our research approach in conducting a systematic literature review and the results of this study. In section V we present the analysis of the results and we conclude in section VI with suggestions for future research in the particular area.

II. Social Network Analysis in Global Software Development

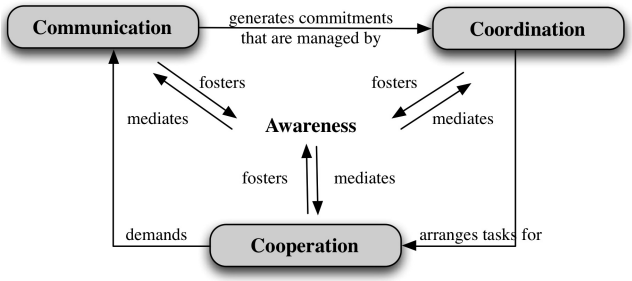
Social network analysis provides a systematic way of analyzing informal networks by mapping and assessing the relationships between people, teams or organizations [4]. According to [5], an important attribute of SNA is that the relationships between the actors of a network are viewed as interdependent rather than as autonomous units. From a knowledge management perspective, in social network analysis the relationships between the actors are the channels through which communication occurs and information "flows" [5]. For example, previous research suggests that people connected with strong ties, through which a broad set of knowledge flows, are more likely to trigger creative ideas [6]. Cross et. al [4] apply social network analysis to

978-0-7695-4787-9/12 $26.00 © 2012 IEEE
DOI 10.1109/ICGSE.2012.10

identify points of knowledge creation and sharing within an organization that hold strategic relevance. Taking a social network perspective to understand how a network of people creates and shares knowledge, makes these interactions within the network visible and as a consequence, related actions and decisions can be taken [4].

The distributed nature of software development and the use of computer-mediated channels for communication, are particularly attractive for the use of social network analysis to investigate software development activities [2]. An important challenge among distributed teams in global software environments is the aspect of awareness, or in other words, to know who knows what, who works in which part of the project and who is who in the remote teams [7]. In global software development, awareness is defined as a means by which team members become familiar with the work of their colleagues, therefore enabling better coordination between the distributed teams [8].

Figure 1: The 3C collaboration model (adopted from [9])



In the field of Computer-supported cooperative work (CSCW), awareness is defined as the intermediate between coordination, cooperation and communication (the 3C Collaboration model; see Figure 1) [9]. During communication people negotiate and make decisions, while coordination is the management of these people and their activities. Cooperation is the joint operation in order to execute tasks. Additionally, awareness is the element that offers feedback to the team members about their actions and gives them information about the other participants [10].

In this paper, we use the 3C model to examine how distributed collaboration can be analyzed from a social network perspective and what are the implications in team awareness. We do so by reviewing current social network approaches which focus on the coordination, cooperation and communication structures of the distributed software teams. We argue that a social network perspective can raise new insights into the distributed collaboration and foster team awareness of global software projects.

III. Research Approach

In order to investigate the potential benefits of a social network perspective in GSD, we performed a systematic literature review. According to [11], a systematic review is a way to identify, evaluate and interpret previous research on a particular topic, with specific research questions. In this study, we investigate the current approaches to analyze collaboration in global software development, from a social network perspective.

To frame the research questions, we used the PICO criteria, suggested by [12]. According to these criteria, a research question should include a *Population*, an *Intervention*, a *Comparison*, and an *Outcome*. In this research, a *Population* is defined as the area of global software development. The *Intervention* is the use of social network analysis. The *Comparison* is the evaluation of the social network analysis as an approach to tackle challenges in global software development. And finally, the *Outcome* is the analysis of the current practices of social network analysis in global software development and the identification of the new insights we can gain in the distributed collaboration, by using a social network perspective. Therefore, the scope of our systematic review focuses on the following research questions:

1) *Which aspects in global software development collaboration do the social network analysis approaches address?*
2) *What new insights does social network analysis bring to global software development collaboration?*

A. Research Strategy

According to [11], when starting the search of primary studies, a strategy should be used by defining the keywords, the search query and the digital libraries used as a search source. We have therefore defined a review protocol, based on the research questions (Section III). The review protocol that we used includes two main keywords, namely: **global software development** and **social network analysis** . Based on these search terms, we constructed a list of related keywords, as shown in Table I.

Table I: Search Keywords

Main Terms	Related Terms
Global Software Development	Global Software Engineering Distributed Software Development Distributed Teams
Social Network Analysis	Social Networks

Based on the derived keywords, a search string was constructed. We used the basic boolean operator **OR** to broaden the search and retrieve papers containing *any* of the terms it separates, and the boolean operator **AND** to narrow

the search and retrieve papers containing *all* of the terms it separates.

(social networks OR social network analysis) AND (global software development OR global software engineering OR distributed software development OR distributed teams)

Finally, we applied the search string to the following libraries, as we believe that they provide a confident source for retrieving results in the particular field:

- IEEE Xplore Digital Library
- ACM Digital Library
- Elsevier ScienceDirect
- SpringerLink
- EBSCO
- Wiley Interscience

B. Study Selection

As part of the systematic literature review (SLR) research, a study selection was performed in order to filter the results and to make sure that we include only studies that contain useful information for answering the research questions. For this purpose, we have decided upon some *inclusion* and *exclusion* criteria that will limit biased or irrelevant results. Table II shows all the search criteria used during the study selection.

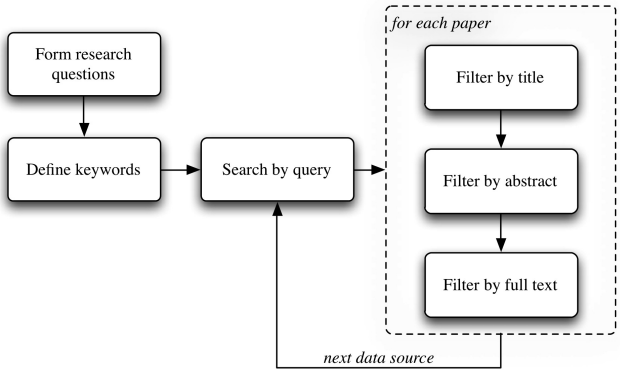
Overall, we are looking for full text papers published in english. We accept papers that describe primary studies, and consequently we reject papers that are a secondary source, that is studies that cite, comment on, or build upon other researches. Additionally, a main inclusion criteria refers to papers that they focus primarily to global software development challenges. Therefore, we include papers that use case studies of software development projects in global/distributed environments.

The evaluation of the final papers to be selected is based first on the relevance of the title of the paper. If the title does not include any of the selected keywords, or it does not clearly reflect the context of the paper, we look at the abstracts which usually present a brief summary of the paper. If no satisfied decisions can be made for the inclusion or exclusion of the publication, based on the abstract then we scan the full text of the paper (Figure 2). These steps were done independently by two researchers. After the filtering by abstracts and by full text, the results were compared and the final papers for inclusion were selected.

Table II: Inclusion and Exclusion criteria

Inclusion criteria	**Exclusion criteria**
Papers that are available online as full texts.	Papers that are not available online as full texts.
Papers that are written in English language.	Papers written in any language other than English.
Papers that describe primary studies.	Papers that are secondary sources, i.e. studies that cite, comment on, or build upon other studies.
Studies that refer primarily to global software development. We include papers that use case studies of software development projects in global environments or otherwise in a distributed manner. We include case studies that refer to development teams that are remotely located.	Studies that do not refer to global software development cases or distributed software teams.
Studies that use a case study from open source software development, in order to address general global software development issues.	Studies that use a case study from open source projects, in order to address issues in OSS communities only, without taking into consideration the general topic of global software development.
Studies that use social network analysis metrics and techniques or studies that they take a qualitative approach to social networks perspective.	Studies that they do not use social network analysis metrics and techniques, but other approaches such as basic statistical analysis metrics and techniques.
	Studies that refer to the use of social media applications (such as Facebook, Twitter and others).

Figure 2: Search Strategy



IV. Overview of the Results

By applying the search strategy described in the previous section (III-A) we concluded in 23 primary studies, in total. Given that we are only looking for primary studies in global software development, and the dedicated attention in IEEE Conferences on global software development (ICSE, ICGSE), it is not a big surprise that there was a 56% acceptance rate from the IEEE results. The overall acceptance rate is calculated at 4% (23 accepted papers out of the 606 total results). The total accepted papers are limited in number due to the particular area in which this systematic literature review study focuses and because of the definite research questions that bound our inclusion criteria.

While applying the search query and filtering the results, we identified the following categories of research papers:

1) Papers that address GSD challenges using a social net-

Table III: Results of primary papers per online source

	Total Query Results	Total Included Papers	Acceptance Rate
IEEE	16	9	56%
SpringerLink	123	2	2%
ACM	149	9	6%
ScienceDirect	121	0	0%
EBSCO	94	1	1%
Wiley	130	2	2%
Total	**606**	**23**	**4%**

work perspective and a case study from a commercial software company.

2) Papers that address GSD challenges using a social network perspective and a case study from open source software company.
3) Papers that use a social network perspective to address challenges in Open Source Software (OSS) community.
4) Papers that use a social network perspective to address challenges in virtual or distributed teams as a general business issue, and they use a commercial or open source GSD case study.

In the third category, we identified a number of studies that use a social network perspective to address challenges in OSS community. These case studies focus on the design and development of open source software, rather than on the collaboration of distributed teams. For this reason, we choose to exclude this type of papers from our analysis and include only those that explicitly center their attention to global software teams and their collaboration, regardless of the nature of the case study project (open source or proprietary). We admit that, without an explicit link to global software development, many of the issues we identify also surface in publications on OSS.

Additionally, we look at the data collection methods that the selected papers use for their case studies. One of the most commonly used practices is to collect data from surveys (48% of the selected papers use this approach). The researchers collect social network data by conducting interviews with the participants or distributing questionnaires. According to [5], this is the most common method for collecting social network data (i.e. data about the relationships or ties between people within a network), where the questionnaire usually contains questions about the respondents' ties to the other people of the network. The second most common data collection method that we observed in our review is from mining various repositories (35% of the selected papers use this approach). In this case, data are collected either from communication repositories such as emails and chat logs, or from software repositories such as change logs and defects libraries. During the analysis, we recognized an essential difference between the two data collection methods; studies that collect data from mining repositories limit the social network participants to the roles of software developers, testers and occasionally architects and/or designers. The reason is that only team members that appear in software documents can be traced as network participants, when data are collected from mining repositories. On the other hand, studies that collect social network data through questionnaires are more broad in the role participation, since more managerial roles of the software development project can be taken into account in the networks analysis, such as project managers, team leaders and others. Finally, we found a small number of papers that used other approaches such as experiments, or a combination of two or more methods to collect data, e.g. questionnaires, documents and observations.

V. Analysis of the Results

As discussed in section II, a popular model to analyze the collaboration between distributed teams is the 3C collaboration model, introduced in the CSCW field. In this research, we use this model to classify whether the selected papers from the systematic literature review, use SNA to tackle challenges in GSD related to communication, coordination and/or cooperation of the remote teams. We choose the 3C model to classify the results because this theory fits the global software development environments where distributed teams need to remotely collaborate through communication, cooperation and coordination activities, and where team awareness remains a great challenge.

Table IV presents the classification of the 23 selected primary studies, based on which aspect of the collaboration model they focus on. In the following paragraphs, we analyze this selection and classification of the primary studies.

A. Coordination

In global software development, coordination among the distributed teams is a challenge due to time, geographical and cultural differences [16]. The difficulties of knowing who to contact about what and communicating effectively across sites, leads to a number of coordination problems [35]. From the analysis, we observe that a social network perspective can bring new insights into the coordination issues in GSD. More specifically, we see that previous researches use social network analysis to investigate the way people are structured within the team and the role that they play in the coordination of the remote groups. Summarizing the results, we identified two main coordination patterns; the core-periphery pattern and the clusters.

Core-Periphery: In a core-periphery structure, "*a particular group of developers are at the center of the coordination activities. The rest of the developers (in the periphery) seem to rely solely on interactions with the centrally positioned developers, for coordinating their tasks.*" [24]. Core-periphery coordination indicates that there is a group of active and interconnected members, independent of geographic location

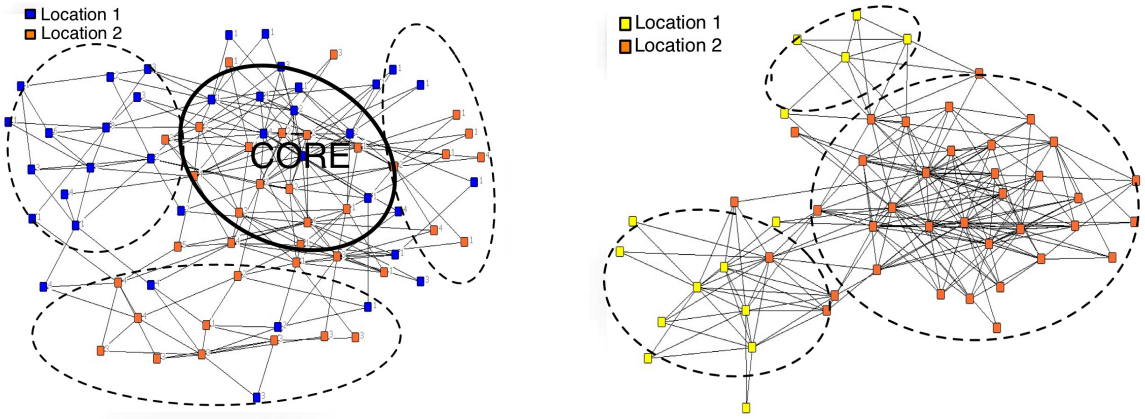
Table IV: SLR Results based on the collaboration model

	Coordination	Cooperation	Communication
Herbsleb & Mockus, 2003 [13]			x
Damian et. al, 2007 [8]	x	x	
Damian, Marczak & Kwan, 2007 [14]		x	
Damian, Kwan & Marczak, 2010 [15]		x	
Avritzer et. al, 2010 [16]	x		x
Urdangarin et. al, 2008 [17]	x	x	x
Fonseca, de Souza & Redmiles, 2006 [18]			x
Nguyen, Wolf & Damian, 2008 [19]	x		
Wolf, Nguyen & Damian, 2008 [20]	x		
Ehrlich & Chang, 2006 [21]		x	
Sarker, Kirkeby & Chakraborty, 2007 [22]	x		
de Souza, Hildenbrand & Redmiles, 2007 [23]			x
Cataldo & Herbsleb, 2008 [24]	x		
Cataldo & Herbsleb, 2008 [25]		x	
Boden & Avram, 2009 [26]	x		
Mikawa, Cunnington & Gaskins, 2009 [27]		x	
Trainer, Al-Ani & Redmiles, 2011 [28]			x
de Souza et. al, 2007 [29]			x
Sarma et. al, 2009 [30]			x
de Souza, Froehlich & Dourish, 2005 [31]	x		x
Chang & Ehrlich, 2007 [32]	x	x	
Sarker et. al, 2011 [33]		x	
Milewski et. al, 2008 [34]	x	x	

[19]. Figure 3a illustrates an example of the core-periphery structure. This coordination structure appears to be in contradiction with the current theory of global software development where remote locations often work as independent groups and there is less cohesion between the teams, due to proximity. Furthermore, the core structure indicates a dense network which in turn, suggests increased collaboration ties and better awareness. A core-periphery structure may also be related to better performance because such structures hold the potential to improve the speed and flexibility with which information diffuses within a group [36]. Finally, the identification and analysis of such coordination structures can help manage the dependencies between the work of core members and that of their peripheral colleagues, and vice versa [31].

Clusters: In this coordination pattern, the network is divided in sub-groups (clusters) which are often based on the geographic location of the teams [32]. Figure 3b illustrates an example of a networks of clusters. A coordination structure of clusters appears to be more in line with the current global software development theories, indicating that members in co-located teams are more connected with each other than they are with their remote colleagues [17]. This coordination structure has a negative effect on the team awareness, since members of a team don't know their remote colleagues, they don't know who works on which part of the project and they often don't know where to find important information, when it is needed. In order to overcome the difficulties that a cluster coordination brings, literature suggests the positioning of a "bridging" role between the remote teams [34]. This intermediate role is the focus of attention of several studies, described in various terms such as "boundary spanners", "brokers", "liaisons", "gatekeepers" and others. Bridging is perceived as a good coordination strategy in the management of distributed collaboration, facilitating team awareness and knowledge management [26]. Bridging roles are also important within a network because they cover the structural holes that might exist between people within the network, or between sub-groups. Global software collaboration is, by its nature, a situation where structural holes may emerge between groups that are geographically, temporally and cultural distant [34]. Chang & Ehrlich [32] conclude that "*individuals who are more central can exert more influence by virtue of being connected with other powerful individuals in the network, and have access to more resources than less central counterparts.*"

Another role positioning that gains attention in the analysis of distributed collaboration, both in the core-periphery and the cluster coordination pattern, is the "star" which is defined as "the individual who stands in the center of attention" [22]. According to previous studies, a star plays an essential role in the team coordination because he/she is in the center of a network of distributed teams, coordinating activities across the various geographical locations and facilitating knowledge transfer [22]. It was also observed that the role an individual plays in the network can highly benefit his or her performance, mainly because that person can have "*ample access to information, rather than controlling the flow of information*" [24]. Researching the role positioning of the people that participate in a distributed software development project can help us understand who are those individuals that are more likely to coordinate the distributed activities, how do they perform and what can be done to foster their role within the network.



(a) Core-periphery (b) Clusters

Figure 3: Coordination structures in Global Software Development

Coordination is analyzed here from a network perspective. Based on this perspective, we can examine the coordination aspect based on the network structure of the teams, i.e. the way people are organized within and across teams and the various roles they can play within the network. Additionally, new insight into the coordination patterns in global software development provide a means for comparison between the different collaboration modes. In our study, we find that the core-periphery is a relatively new structure. Current literature suggests that a core-periphery as a coordination structure has a positive effect in the collaboration between distributed teams. Finally, the clustering structure is a more common structure that conforms to the known disadvantages and challenges of distributed collaboration. Nevertheless, with the right strategic positioning of the team members and the existence of "bridges" or "stars", these challenges can be surpassed [34].

B. Cooperation

Previous studies examine how social factors, such as organizational culture and history of the relationship between the remote sites, play an important role in awareness of changes in requirements and code that require effective cooperation across sites [37]. Additionally, in another study different approaches to support awareness between teams' cooperation have been identified, by examining improvements brought to the "shared space" between team members or how they interact using shared artifacts [38]. From a social network perspective, we examine how the analysis of the distributed collaboration networks can bring new insights into the cooperation aspects of the distributed teams.

A main observation in the teams' cooperation, from a social networking perspective, is the appearance of "emergent" team members [15]. According to literature, emergent members can be of two types; first, they can be people who were not initially planned to participate in the development project but in practice they emerge as contributing members. Second is when a person emerges as the "bridge" between remote teams, even though that position was planned for someone else. A reason for this phenomenon is e.g. the role of the specific person, or his/her position within the team, which might constitute him/her as a better intermediate for the cooperation among the distributed teams. Milewski et. al. [34] note that "*most of the bridges we have found in our research and in the literature are not ones that were developed intentionally as part of organization strategy, but units that have grown out of organizational need, or of the team positioning themselves in this role.*". As an example in a study conducted among several distributed networks of people working on requirements engineering, results showed that on average, about one third of the team members were emergent roles [14]. In another study, it was observed that although teams were designed to communicate "freely", i.e. all members from the remote teams could communicate with all of their colleagues in the central locations, in practice only few members emerged as the intermediates between teams at different locations [17].

Another observation is how cooperation between the distributed teams is facilitated through familiarity and trust [33]. People are more likely to cooperate with each other if they have previously worked together in another project [21], [32]. Furthermore, familiarity can also refer to the awareness of each others' cultural differences. Previous research in team trust in GSD suggests that when people are familiar with each others' cultural differences, trust factors between them are higher and consequently cooperation is facilitated; "*Developing positive shared experiences across geographic sites leads to stronger team dynamics for distributed software development teams, and stronger team dynamics build trust.*" [27]. Trust has also been researched as an influential factor on the member's cooperation performance. Previous

work revealed that when individual has a low trustworthiness (indicated by low trust centrality), even if that person is communicating a lot, he/she is not cooperating efficiently with his/her colleagues, as opposed to individuals who communicate a lot but at the same time, they also have high trustworthiness [33].

The implication on team awareness that these cooperation strategies have, varies. In some cases, it was observed that the emergent members appearing in the projects, despite the initial planning and documentation, negatively impact team awareness. People within the team do not know who they work with, with whom they need to communicate and collaborate and consequently they are not aware of who knows what in the project. On the other hand, case studies were reported, where despite the appearance of those emergent members, team awareness was high and team members recognized the emergent members and efficiently worked with them. It was also observed that people who are central in the cooperation between the remote teams tend to perceive that the teams are cooperating effectively, which in turn creates a positive environment for cooperation among the distributed team members [32].

C. Communication

According to the 3C Collaboration model, communication is related to the exchange of messages and information among people [9]. In the field of software engineering, recent studies have turned their focus on the "fit" of the communication patterns between the members of software development teams (distributed or co-located) and the dependencies between the modules/artifacts that each team develops [39]. This "fit" is also referred to as the socio-technical congruence i.e. the match between the social and technical dependencies in a software development project [40]. In global software development, the socio-technical congruence is often studied as a means to analyze and improve the communication and awareness between the remote members, and therefore to improve their collaboration.

During the SLR analysis, we identified studies that use social network analysis to examine the communication patterns of team members, and subsequently analyze and improve the socio-technical congruence of the distributed software development projects. Consequently, we recognized several tools that use SNA techniques as a means to facilitate communication between distributed team members, ease the expertise identification, and increase team awareness. The following paragraphs describe those tools:

Ariadne: This tool identifies automatically dependent pieces of code and their authors, and it creates a social network of developers. In other words, it can visualize and present which developer depends on the code of another developer. "*The goal of the tool is to automatically identify situations where there is a "mismatch" between the dependency and the communication networks.*" [18]. From the experiments conducted using Ariadne, it was noticed that this tool can promote social network awareness among virtual teams and consequently increase familiarity among team members; "*Ariadne provides developers an enhanced awareness of interdependencies using a visual approach. Interdependencies are one example of collaborative traces; they describe relationships between developers based on the source code they implement.*" [28]. Additionally, it was observed that Ariadne can support managers in monitoring interactions among developers and taking the necessary decisions [28].

Travis: Travis is a tool that creates traceability networks by linking artifacts, activities and users. Travis also offers different views of the communication network in role-based criteria, i.e. based on whether the user is a developer, a designer, a project manager etc. This option can help eliminate the number of notifications that team members receive, creating an overload of sent messages and avoid over-communication among team members; "*The goal here is to reduce the number of notifications that software developers receive because a common problem was the overwhelming flood of notification messages initiated by other software developers or software tools due to changes in the artifacts.*" [23]. This has proven particularly useful for the people acting as bridges between teams, or otherwise acting as information brokers. Travis can help these information brokers to manage the large amounts of information they receive from the communication between the collaborating teams; "*TraVis provides increased awareness within offshore software development projects based on a broad range of traceability and rationale visualizations that are created with information extracted from the collaborative development platform.*" [23].

Tesseract: It is a socio-technical dependency browser specifically constructed to show "*the social as well as the technical relationships among the different project entities, such as developers, source code, bugs etc*" [30]. Tesseract also calculates the communication behavior of the teams, which is the social network of developers determined by the communication records and it monitors progress over time. Additionally, the tool "*can highlight the "mismatches" between technical dependencies and the communication patterns of the developers*" [30]. This approach can help developers become aware of the other project activities and improve communication with their colleagues; "*Some interviewees suggested that the developer-to-developer linkages could serve as a means of creating an awareness of which developers work closely- information that is missing in their distributed work settings.*" [30].

Augur: Augur "*draws views of the network of contributors to a project, relating them according to patterns in their development activity*" [31]. Additionally, this network view property can use different visualizations to indicate different features of the relationship between individuals. Similarly

to the above mentioned tools, Augur provides a way to examine the communication patterns of the project participants, through the analysis of the technical dependencies of the project; "*It provides the technical means to explore the extend to which software artifacts have inscribed into them patterns of interaction and participation.*" [31].

Communication relationships have proven important for knowledge sharing and team awareness [22]. An essential feature of the socio-technical congruence tools is the visualization of the communication patterns and the relationships between project participants. Previous work suggests that this type of communication visualizations can help developers as well as managers to keep track of the project status, be aware of the code dependencies and take the necessary decisions and actions [29].

VI. Conclusion

In this paper, we examined how social network analysis can provide researchers and practitioners with new insights into the current global software development challenges. Two different coordination structures of distributed teams in a GSD context have been identified in this SLR study: the core-periphery and the clusters. In terms of team awareness, core-periphery appears the more beneficial structure, yet it does not show up very often in GSD projects. Clusters come with certain disadvantages due to the sub-grouping and the independent work of the teams based on their geographic location. Through the current SNA approaches, researchers have identified strategies that can help tackle these disadvantages, such as the positioning of bridges between the clusters.

Analyzing the cooperation structures from a social network perspective we distinguished two new insights in the current GSD practices: the emergent members and the role of familiarity and trust in distributed cooperation. We observed that through the use of social network analysis new cooperation patterns were identified in projects that were initially planned otherwise.

A social network perspective can help us reveal differences between what is planned and documented within a project, and what happens in practice. These differences can be studied over time, and the findings related to other metrics, such as the number of bugs found, the size of backlogs, and the like, may lead to corresponding interventions.

Finally, communication patterns have been studied through the lenses of socio-technical congruence, in an effort to analyze communication and collaboration between project participants and software artifacts dependencies. We identified several tools that have been introduced in the current literature, with the aim of improving team communication and at the same time to support team awareness and expertise identification.

We conclude that more research is needed in the field of global software development by taking a social network perspective. We argue that this kind of approach may bring new insights into the distributed teams collaboration and promote new solutions and collaboration processes or techniques, to tackle the current global challenges. Social network analysis can help researchers and practitioners examine the coordination, cooperation and communication structures of the remote and co-located teams, identify potential weak points and support the organization and the decision-making processes of the distributed activities.

References

[1] S. Marczak, I. Kwan, and D. Damian, "Social networks in the study of collaboration in global software teams," *International Conference on Global Software Engineering (ICGSE '07)*, August 2007.

[2] C. Del Rosso, "Comprehend and analyze knowledge networks to improve software evolution," *Journal of Software Maintenance and Evolution: Research and Practice*, vol. 21, no. 3, pp. 189–215, 2009.

[3] K. Ehrlich and I. Carboni, "Inside social network analysis," IBM Research, Tech. Rep., 2005.

[4] R. Cross, A. Parker, L. Prusak, and S. P. Borgatti, "Knowing what we know: Supporting knowledge creation and sharing in social networks." *Organizational Dynamics*, vol. 30, no. 2, pp. 100 – 120, 2001.

[5] S. Wasserman and K. Faust, *Social network analysis: Methods and applications*. Cambridge University Press, 1994.

[6] M. E. Sosa, "Where do creative interactions come from? the role of tie content and social networks," *Organization Science*, vol. 22, pp. 1–21, January 2011.

[7] C. Manteli, B. van den Hooff, A. Tang, and H. van Vliet, "The impact of multi-site software governance on knowledge management," in *Global Software Engineering (ICGSE), 2011 6th IEEE International Conference on*, 2011, pp. 40 –49.

[8] D. Damian, L. Izquierdo, J. Singer, and I. Kwan, "Awareness in the wild: Why communication breakdowns occur," in *Global Software Engineering, 2007. ICGSE 2007. Second IEEE International Conference on*, 2007, pp. 81 –90.

[9] N. Kock, *Encyclopedia of E-collaboration*. Information Science Reference, 2008.

[10] H. Fuks, A. Raposo, M. Gerosa, and C. Lucena, "Applying the 3c model to groupware development," *International Journal of Cooperative Information Systems*, vol. 14, no. 2-3, p. 299, 2005.

[11] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," *Engineering*, vol. 2, no. EBSE 2007-001, 2007.

[12] M. Petticrew and H. Roberts, *Systematic reviews in the social sciences: A practical guide*. Oxford, 2006.

[13] J. Herbsleb and A. Mockus, "An empirical study of speed and communication in globally distributed software development," *Software Engineering, IEEE Transactions on*, vol. 29, no. 6, pp. 481 – 494, 2003.

[14] D. Damian, S. Marczak, and I. Kwan, "Collaboration patterns and the impact of distance on awareness in requirements-centred social networks," in *Requirements Engineering Conference, 2007. RE '07. 15th IEEE International*, 2007, pp. 59 –68.

[15] D. Damian, I. Kwan, and S. Marczak, "Requirements-driven collaboration: Leveraging the invisible relationships between requirements and people," in *Collaborative Software Engineering*, I. Mistrík, A. van der Hoek, J. Grundy, and J. Whitehead, Eds. Springer Berlin Heidelberg, 2010, pp. 57–76.

[16] A. Avritzer, D. Paulish, Y. Cai, and K. Sethi, "Coordination implications of software architecture in a global software development project," *Journal of Systems and Software*, vol. 83, no. 10, pp. 1881 – 1895, 2010.

[17] R. Urdangarin, P. Fernandes, A. Avritzer, and D. Paulish, "Experiences with agile practices in the global studio project," in *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on*, 2008, pp. 77 –86.

[18] S. B. Fonseca, C. R. B. de Souza, and D. F. Redmiles, "Exploring the relationship between dependencies and coordination to support global software development projects," in *Global Software Engineering, 2006. ICGSE '06. International Conference on*, oct. 2006, p. 243.

[19] T. Nguyen, T. Wolf, and D. Damian, "Global software development and delay: Does distance still matter?" in *Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference on*, 2008, pp. 45 –54.

[20] T. Wolf, T. Nguyen, and D. Damian, "Does distance still matter?" *Software Process: Improvement and Practice*, vol. 13, no. 6, pp. 493–510, 2008.

[21] K. Ehrlich and K. Chang, "Leveraging expertise in global software teams: Going outside boundaries," in *Global Software Engineering, 2006. ICGSE '06. International Conference on*, 2006, pp. 149 –158.

[22] S. Sarker, S. Kirkeby, and S. Chakraborty, "Path to stardom in globally distributed teams: An examination of a knowledge-centered perspective using social network analysis," in *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on*. IEEE, 2007, pp. 189c–189c.

[23] C. de Souza, T. Hildenbrand, and D. Redmiles, "Toward visualization and analysis of traceability relationships in distributed and offshore software development projects," in *Software Engineering Approaches for Offshore and Outsourced Development*, ser. Lecture Notes in Computer Science, B. Meyer and M. Joseph, Eds. Springer Berlin / Heidelberg, 2007, vol. 4716, pp. 182–199.

[24] M. Cataldo and J. D. Herbsleb, "Communication patterns in geographically distributed software development and engineers' contributions to the development effort," in *Proceedings of the 2008 international workshop on Cooperative and human aspects of software engineering*, ser. CHASE '08. New York, NY, USA: ACM, 2008, pp. 25–28.

[25] M. Cataldo and J. Herbsleb, "Communication networks in geographically distributed software development," in *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, ser. CSCW '08. ACM, 2008, pp. 579–588.

[26] A. Boden and G. Avram, "Bridging knowledge distribution - the role of knowledge brokers in distributed software development teams," in *Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering*, ser. CHASE '09. IEEE Computer Society, 2009, pp. 8–11.

[27] S. P. Mikawa, S. K. Cunnington, and S. A. Gaskins, "Removing barriers to trust in distributed teams: understanding cultural differences and strengthening social ties," in *Proceeding of the 2009 international workshop on Intercultural collaboration*, ser. IWIC '09. ACM, 2009, pp. 273–276.

[28] E. H. Trainer, B. Al-Ani, and D. F. Redmiles, "Impact of collaborative traces on trustworthiness," in *Proceeding of the 4th international workshop on Cooperative and human aspects of software engineering*, ser. CHASE '11. ACM, 2011, pp. 40–47.

[29] C. R. de Souza, S. Quirk, E. Trainer, and D. F. Redmiles, "Supporting collaborative software development through the visualization of socio-technical dependencies," in *Proceedings of the 2007 international ACM conference on Supporting group work*, ser. GROUP '07. ACM, 2007, pp. 147–156.

[30] A. Sarma, L. Maccherone, P. Wagstrom, and J. Herbsleb, "Tesseract: Interactive visual exploration of socio-technical relationships in software development," in *Proceedings of the 31st International Conference on Software Engineering*, ser. ICSE '09. IEEE Computer Society, 2009, pp. 23–33.

[31] C. de Souza, J. Froehlich, and P. Dourish, "Seeking the source: software source code as a social and technical artifact," in *Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work*, no. 10. New York, NY, USA: ACM, 2005, pp. 197–206.

[32] K. T. Chang and K. Ehrlich, "Out of sight but not out of mind?: Informal networks, communication and media use in global software teams," in *Proceedings of the 2007 conference of the center for advanced studies on Collaborative research*. ACM, 2007, pp. 86–97.

[33] S. Sarker, M. Ahuja, S. Sarker, and S. Kirkeby, "The role of communication and trust in global virtual teams: A social network perspective," *Journal of Management Information Systems*, vol. 28, no. 1, pp. 273–310, 2011.

[34] A. E. Milewski, M. Tremaine, F. Köbler, R. Egan, S. Zhang, and P. O'Sullivan, "Guidelines for effective bridging in global software engineering," *Software Process: Improvement and Practice*, vol. 13, no. 6, pp. 477–492, 2008.

[35] J. D. Herbsleb, A. Mockus, T. A. Finholt, and R. E. Grinter, "An empirical study of global software development: distance and speed," in *Proceedings of the 23rd International Conference on Software Engineering*. IEEE Computer Society, 2001, pp. 81–90.

[36] J. N. Cummings and R. Cross, "Structural properties of work groups and their consequences for performance," *Social Networks*, vol. 25, no. 3, pp. 197 – 210, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0378873302000497

[37] P. Dourish and V. Bellotti, "Awareness and coordination in shared workspaces," in *Proceedings of the 1992 ACM conference on Computer-supported cooperative work*. ACM, 1992, pp. 107–114.

[38] I. Steinmacher, A. Chaves, and M. Gerosa, "Awareness support in global software development: A systematic review based on the 3c collaboration model," in *Collaboration and Technology*, ser. Lecture Notes in Computer Science, G. Kolfschoten, T. Herrmann, and S. Lukosch, Eds. Springer Berlin / Heidelberg, 2010, vol. 6257, pp. 185–201.

[39] I. Kwan, A. Schroter, and D. Damian, "Does socio-technical congruence have an effect on software build success? a study of coordination in a software project," *Software Engineering, IEEE Transactions on*, vol. 37, no. 3, pp. 307 –324, 2011.

[40] M. Cataldo, P. A. Wagstrom, J. D. Herbsleb, and K. M. Carley, "Identification of coordination requirements: implications for the design of collaboration and awareness tools," in *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, ser. CSCW '06. ACM, 2006, pp. 353–362.