International Journal of Data Mining & Knowledge Management Process (IJDKP) Vol.7, No.5/6, November 2017

On Using Network Science in Mining Developers Collaboration in Software Engineering: A Systematic Literature Review

Mohammed Abufouda and Hadil Abukwaik

University of Kaiserslautern, Department of Computer Science,
Kaiserslautern 67663, Germany
{abufouda,abukwaik}@cs.uni-kl.de

Abstract. *Background*: Network science is the set of mathematical frameworks, models, and measures that are used to understand a complex system modeled as a network composed of nodes and edges. The nodes of a network represent entities and the edges represent relationships between these entities. Network science has been used in many research works for mining human interaction during different phases of software engineering (SE).
Objective: The goal of this study is to identify, review, and analyze the published research works that used network analysis as a tool for understanding the human collaboration on different levels of software development. This study and its findings are expected to be of benefit for software engineering practitioners and researchers who are mining software repositories using tools from network science field.
Method: We conducted a systematic literature review, in which we analyzed a number of selected papers from different digital libraries based on inclusion and exclusion criteria.
Results: We identified 35 primary studies (PSs) from *four* digital libraries, then we extracted data from each PS according to a predefined data extraction sheet. The results of our data analysis showed that not all of the constructed networks used in the PSs were valid as the edges of these networks did not reflect a real relationship between the entities of the network. Additionally, the used measures in the PSs were in many cases not suitable for the used networks. Also, the reported analysis results by the PSs were not, in most cases, validated using any statistical model. Finally, many of the PSs did not provide lessons or guidelines for software practitioners that can improve the software engineering practices.
Conclusion: Although employing network analysis in mining developers' collaboration showed some satisfactory results in some of the PSs, the application of network analysis needs to be conducted more carefully. That is said, the constructed network should be representative and meaningful, the used measure needs to be suitable for the context, and the validation of the results should be considered. More and above, we state some research gaps, in which network science can be applied, with some pointers to recent advances that can be used to mine collaboration networks.

Keywords: Network Analysis, software engineering, developers collaboration, systematic literature review

1 Introduction

Network science is the set of mathematical frameworks, models, and measures that are used to analyze a complex system modeled as a network. Since the seminal works of Watts and Strogatz [54] and Barabási and Albert [6], the field of network science exploded in different directions with a huge amount of measures, models, and applications for network analysis. Fields like Biology, Social science, Physics, and Computer science contribute a lot to the network science field, each from different perspectives. Other domains benefited from the measures and tools provided in the field of network science, and software engineering is one of those

DOI: 10.5121/ijdkp.2017.7601

fields. Software engineering is a complex production system, in which humans are involved in order to produce a software with some certain properties. One face of the complexity of software engineering process is the interaction among developers who develop a software. This complexity can be seen in many phases of the software engineering life cycle including collaborative coding, bug fixing and tracking, communication in mailing lists, and many other interactions. The interactions between developers can be seen as a *network* where the nodes are, in most cases, the developers and the edges between these nodes are the interactions between them. Having these interaction constructed as networks opens the way to use network science as an effective tool to analyze these networks. In this work, we systematically review the application of network analysis for the networks constructed from the collaboration interaction between software developers.

1.1 Research Questions

The course of improving software engineering process, practices, and software quality is hard and it get even harder when considering the human factor. To that end, a lot of work has(is) being done and a lot of venues have been dedicated for mining the human artifacts (like developers' collaboration and communication interactions) and the software artifacts (like the source code, the requirements and specification documentation) in order to gain insights to improve the software engineering. In this research, we aim at reviewing, evaluating, and providing future directions for the use of network science in the area of software engineering specifically for the collaboration between developers. Thus, this work has the following research questions:

- **RQ1**: *How valid and reliable are the constructed networks?* The main criterion for conducting good network analysis is to have a valid network, otherwise, any subsequent analysis may be useless. Thus, this research question investigates the validity and the quality of the constructed networks in the identified studies. Here, we explore the quality of the edges in the constructed networks, like being a real or a proxy relationship, being aggregated over time (longitudinal aggregation) and space (multiplex aggregation).
- **RQ2**: *How valid and reliable is the use of network analysis measures?* Network science is noticeably multidisciplinary, which renders some measures not suitable for every network and in every context. This research question investigates how meaningful the use of certain measures for the constructed networks is. We also shed light on the unstated assumptions of some of the measures that were used intensively in the primary studies.
- **RQ3**: *How valid, reliable, and generalizable are the results reported by the primary studies?* In order to get the confidence regarding the significance of the reported results by any primary study, a validation should be performed. This research question investigates the validation status of the reported results that were based on using network analysis. We also explore the generalizability of the reported results by examining the number of analyzed networks in the study.
- **RQ4**: *How are the results being reflected on the studied context?* Having data is tempting to start analysis, however, any analysis that does not provide actionable and useful insights for the studied context should be avoided. This research question investigates whether the PSs reflected the results on the studied context of software engineering or not.
- **RQ5**: *What are the gaps in the research that can be further researched?* Based on our extracted data and our analysis results, we identify potential research directions to cover existing gaps about where network analysis can be employed. The answer of this question

should help in extending the body of the research in mining developers networks using network analysis. Additionally, we provide some pointers to recent advances in the network science that can be utilized.

1.2 Method

In this systematic literature review (SLR), we followed the guidelines provided by Kitchenham and Charters [32]. They offer the most formal and strict review process we know in empirical software engineering. Figure 1 abstracts the steps that we adapted from Kitchenham and Charters [32] and implemented in this SLR. Based on these guidelines, we built our process model for conducting the review. Figure 2 shows this model in detailed steps. In the following section, we provide details about the model. The rest of this paper is organized as follows. Section 2 provides the definitions that will be used in this paper. Section 3 gives details about the followed steps conducted in this SLR. Section 4 contains the results of the extracted data, while Section 5 provides the answers to the research questions. This paper concludes in Section 6[1].

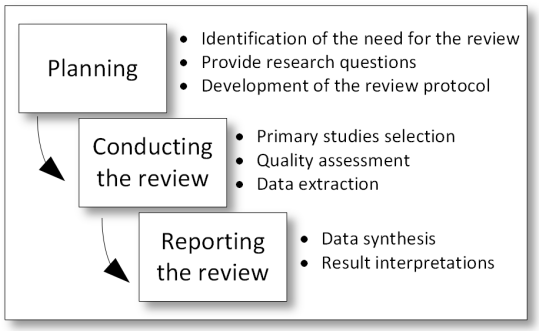

Fig. 1: The systematic review steps(guidelines) adapted from Kitchenham and Charters [32]

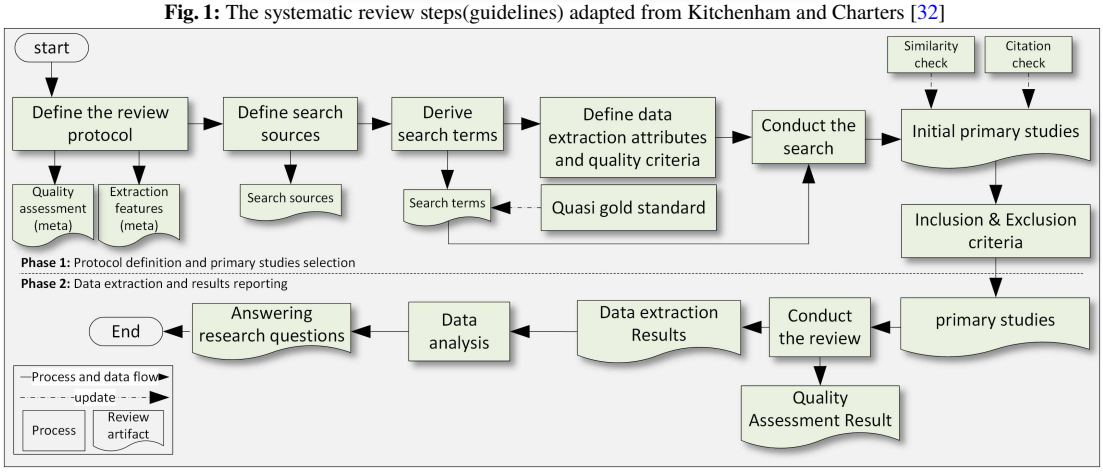

Fig. 2: The systematic review process model followed in this SLR.

2 Definitions

In this section, we provide the definitions [2] required to proceed in this review. We also provide information about how network analysis should be properly applied in order to get meaningful

[1] The data of this SLR is available upon request.

[2] We will stick to the terminology used in network science field to help researchers in software engineering in the future in finding the related work more easily.

results. The data extraction fields presented in Section 3.4 are based on the definitions in this section.

An undirected network $G = (V, E)$ is a tuple that is composed of two sets V and E, where V is the set of nodes and E is the set of edges such that an undirected edge e is defined as $e = \{u, v\} \in E$ where $u, v \in V$. For a directed network $\overrightarrow{G} = (V, \overrightarrow{E})$, a directed edge $\overrightarrow{e}$ is defined as $\overrightarrow{e} = (u, v)$ where the node u is the *source* and the node v is the *target*. For undirected networks, the *Degree Centrality* of a node w, $deg(w)$, is defined as the number of nodes that are connected to it, while for directed networks the *in-degree* and the *out-degree* are defined as the number of edges in the network where the node w is the target and the source node, respectively. The set of all neighbors of a node v is denoted as $N(v)$. Graphs can be weighted, i.e., edges are associated with a weight that reflects the intensity, frequency, or the distance between two nodes. Thus, a weighted undirected network is defined as: $G = (V, E, \omega)$, where $\omega : E \rightarrow \mathbb{R}$. Networks whose nodes and edges are fixed over time are called *static* networks. Networks that consider the temporality of nodes and edges are called *dynamic* or *temporal* networks, and normally denoted as $G_t(V, E)$, e.g, the network G at time point t.

2.1 Centrality measures

A set of measures is defined based on the definition of a network. The *distance* (reachability) between two nodes u and v is defined as: $d(u, v)$ which is the shortest path between the two nodes. The *diameter* of a network is the maximal distance between any two nodes in the network. The *clustering coefficient* of a node v is defined as: $cc(v) = \frac{e(v)}{deg(v)(deg(v)-1)/2}$ for directed networks and as: $cc(v) = \frac{e(v)}{deg(v)(deg(v)-1)}$ for undirected networks. The *network density* measures how many edges there are in the network compared to the maximum possible number of edges, and it is defined as: $\eta(G) = \frac{2m}{n(n-1)}$ for undirected networks, where $m = |E|$ and $n = |V|$. The networks with high density are called *dense* networks and those with low density are called *sparse* networks. The *betweenness centrality* of a node v is defined as: $B(v) = \sum_{s \in V(G)} \sum_{t \in V(G)} \frac{\sigma_{st}(v)}{\sigma_{st}}$, where $\sigma_{st}(v)$ is the number of shortest paths between the nodes s and t that includes the node v, while σ_{st} is the number of all shortest paths between the nodes s and t. The *closeness centrality* of a node v is defined as: $C(v) = (\sum_{w \in V(G)} d(v, w))^{-1}$.

2.2 Bipartite networks

A special case of networks is the *Bipartite* networks (2-mode networks or affiliation networks), which is defined as $G(V_L, V_R, E)$, where $V = V_L \cup V_R$, $V_L \cap V_R = \phi$, and $E : V_L \rightarrow V_R$. The bipartite networks are appropriate model when we want to model the interactions between two disjoint entities, like developer and source code. A method called *One Mode Projection* (OMP) is classically used to convert the bipartite networks into a unipartite network, where the nodes in the projected network consists of only one type of nodes.

2.3 Multiplex

Networks can be *Multiplex* (Multi-layer, multilateral, etc.), where the interactions between the nodes have many types, each can be represented as a different network. The formal definition of this type of networks is: $\mathcal{M} = (\mathcal{G}, \mathcal{C})$ where $\mathcal{G} = \{G_\alpha; \alpha \in 1, \cdots, M\}$ is the set of all layers,

each of them $G_\alpha \in \mathcal{G}, G_\alpha = (V_\alpha, E_\alpha)$ represents one type α of interactions. The inter-layer edge $\mathcal{C}$ is defined as: $\mathcal{C} = \{E_{\alpha\beta} \subseteq V_\alpha \times V_\beta;\ \alpha, \beta \in \{1, \cdots, M\},\ \alpha \neq \beta\}$ represents the number of edges between the nodes from two different layers. Normally, when $\mathcal{C} = \phi$, the networks are called multiplex, otherwise, it is called multilayer.

2.4 Null models and network motifs

Null models are graph models that are used to test the statistical significance of the results obtained from an observed (constructed) network. Using a null model gives more confidence and creditability to the conclusions of any network-based measures or models. *Random Graphs* with the same degree sequence of the observed network are wildly used null model. Network *motif* [49] is a subgraph that exists significantly higher than the corresponding null model. Finding those motifs gives insights about the patterns that occur frequently in a network.

3 The procedures of the review

In this section, we provide a full description of the SLR process, which we followed in this study. This process is illustrated in Figure 2. We started by defining the goal of the review, which is a row version of the presented research questions in Section 1.1. Accordingly, we defined the *Quality Assessment* (QA) items of the PSs and the *Extraction Features* (EF) items for the analysis of the PSs. The search sources were then defined and the *search terms* where identified. To add more strictness to the review, we followed the *Quasi-Gold Standard* method presented by Zhang et al. [59] to select the appropriate *search terms* semi-systematically as full-systematic term identification is hard to achieve and to reproduce. Having our search terms identified, the search process was conducted on the defined search sources producing the *raw primary studies*. The raw primary studies were checked by snowballing , which is a forward and backward citation check. We did also a similarity check to find similar papers for each paper in the primary studies. The similarity check is provided in some libraries' search engine that enabled a library user to see similar and related papers. Having the raw primary studies, we applied the inclusion and exclusion criteria to get the final set of the *primary studies*.

3.1 Quasi-Gold Standard search

One challenge in performing a rigor SLR is to identify the search terms systematically. To the best of our knowledge, the most rigorous method to attain high reproducibility is the Quasi-Gold Standard (QGS) provided by Zhang et al. [59]. This method starts with manually identifying a set of studies that are relevant to the SLR and a set of search terms. Then, the researchers run the automatic search in the search libraries many times, in each run then, the quality of the retrieved papers is evaluated. The evaluation is done by calculating the *sensitivity* of the search result (i.e., the number of related studies found by the search divided by the number of QGS studies). In each run, the search terms are updated by introducing new relevant terms until we reach the targeted sensitivity, which should be 100% in the best situation. In our study, we did manual identification of 50 related work as a QGS [3] from 8 related venues by searching their proceedings in the last 10 years. The first run of the QGS search resulted in a sensitivity of

[3] Please note that this number includes papers for using network analysis in minable software artifacts too as the plan was to review the application of the network science in software engineering in general. However, we found that splitting the results of the primary studies into two groups: (1) minable human aspects and (2) minable software artifacts would be more appropriate for intensive analysis.

68% with 16 missed papers out of the 50. After inspecting those missed papers, we updated the search terms and reached 100% sensitivity from the second (and the last) run of the QGS search method. The used search queries are in the provided data, and here is a sample of the query used in the IEEE search library:

("Document Title":software OR "Document Title":program OR "Document Title":evolution OR "Document Title":maintenance OR "Document Title":architecture OR "Document Title":bug OR "Document Title":maintainability OR "Document Title":requirements OR "Document Title":testing OR "Document Title":clone OR "Document Title":opensource) and ("Document Title":graph OR "Document Title":network OR "Document Title":tree OR "Document Title":analysis OR "Document Title":collaboration OR "Document Title":call OR "Document Title":interaction)

Libraries search engine	Row primary studies	Initial primary studies	Primary studies
ACM	1992	50	15
IEEEXplor	2500	124	15
Springer digital library	1553	15	3
Elsevier ScienceDirect	113	8	2
Wiley open access	35	2	0
Sum	**6080**	**199**	**35**

Table 1: The selection process of the primary studies across the selected libraries. The row primary studies were title-abstract sifted, then the inclusion and exclusion criteria were applied on the initial primary studies resulting in the PSs.

3.2 Search libraries and In(Ex)clusion criteria

We identified the following search libraries as a basis for the search process. The libraries are: (1) The *ACM* digital library, (2) The *IEEExplore*, (3) The *Springer*, and (4) The *ScienceDirect*. The initial studies resulted from the search process was 6080 that we sifted based on the title and the abstract content. The result was 199 related initial primary studies that we read in order to apply the inclusion and exclusion criteria. Table 1 and Figure 3 show more details about the number of selected studies in each step across the search libraries.

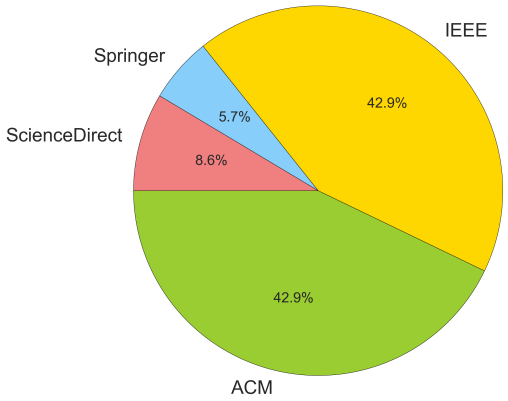

Fig. 3: The percentages of the number of primary studies across the used libraries.

Table 2 shows the inclusion criteria. Any included study should satisfy all of the inclusion criteria in the table. Table 3 shows the exclusion criteria along with the number of excluded studies due to each exclusion criterion.

Figure 4 shows the distribution of the number of primary studies and their publication year. The

Inclusion item	Description
I_1	Research that uses network analysis in studying software engineering practices or artifacts or processes.
I_2	Empirical study that provides quantitative results.
I_3	It covers minable human aspects and/or minable software artifacts
I_4	It is a peer reviewed publications.

Table 2: The inclusion criteria applied in this SLR.

earlier primary study was published in 2005, a few years after the emergence of network science papers.

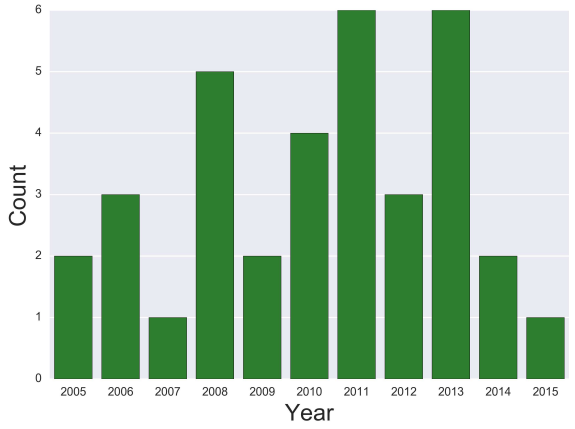

Fig. 4: The distribution of the number of primary studies over years 2005-2015.

Exclusion item	Description
E_1	Any study that is not in English. (1)
E_2	Replication studies, invited papers, lessons learned, technical papers, and position papers. (5)
E_3	Visualization tools, process engineering, and controlled experiment papers. (14)
E_4	Theoretical models that have not been validated empirically with real systems. (10)
E_5	Papers that do not cover the minable human aspects in software engineering. (41)
E_6	Duplicated papers or initial work that has been extended later. (4)
E_7	Papers that only construct networks from data without analysis. (8)

Table 3: The exclusion criteria along with the number of excluded studies due to each exclusion criterion.

After applying the inclusion and exclusion criteria, we got the following primary studies. For the *ACM digital library* and the *IEEE*, we had 15 studies selected from each. For the *SciencDirect* and the *Springer*, we had 3 and 2 primary studies, respectively. The included primary studies in this SLR are shown in Table 4 along with their venues. Figure 5 shows the number of primary studies per venue. The *Others* represents all venues with only one primary study. From Figure 5 and from Table 4, we noticed that major software engineering conferences like *ICSE* and *FSE* have the highest number of primary studies. Also, from the information in the table and the figure, we see that there is 31 primary studies published in conference proceedings and the rest, 4 studies, are published in journals.

3.3 Quality assessment

Performing quality assessment is important in SLRs in order to get insights regarding the quality of the included PSs and to perform a proper analysis for them. We followed the method presented in [22] for the quality assessment task. We restricted our quality assessment to the

Digital library	PS	Year	Venue[4]
ACM	[28]	2005	SIGSOFT Softw. Eng. Notes
	[42]	2005	SIGSOFT Softw. Eng. Notes
	[10]	2006	MSR
	[12]	2008	FSE
	[46]	2008	CHASE
	[45]	2008	FSE
	[37]	2008	FSE
	[55]	2009	ICSE
	[20]	2010	ISEC
	[17]	2011	MSR
	[11]	2011	FSE
	[51]	2011	ISEC
	[29]	2011	CHASE
	[34]	2013	ISEC
	[44]	2014	ICPC
IEEE	[48]	2008	ECECE
	[9]	2009	ERE
	[7]	2010	ICPC
	[52]	2010	CRE
	[27]	2011	ICSM
	[36]	2011	ICSE
	[25]	2012	ICGCC
	[57]	2012	ICSE
	[58]	2012	ICSI
	[5]	2013	ASONAM
	[16]	2013	CHSAE
	[38]	2013	ICSM
	[60]	2013	ICCSC
	[8]	2014	ICSME
	[30]	2015	ICSE
ScienceDirect	[50]	2006	EOSSD
	[56]	2006	IST
	[53]	2010	IST
Springer	[24]	2007	OSDAI
	[21]	2013	Complex networks

Table 4: A summary of the identified primary studies in this SLR.

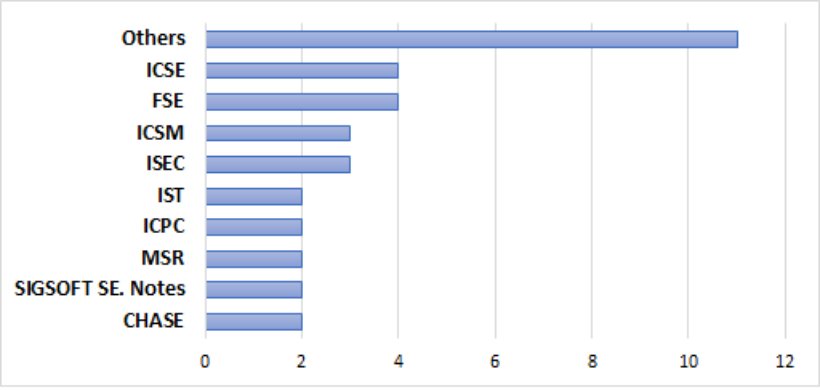

Fig. 5: The distribution of the number of primary studies over the venues.

research quality of the PSs as the other aspects were covered in the data extraction in order to answer the research questions in Section 1.1. We did not apply any exclusion based on the quality of the PSs as this will limit our ability to answer the research question properly.

Code	Addresses	Description
QA_1	Research Design	The study provides a clear research design that includes research goals, hypotheses and other aspects of research design.
QA_2	Data Collection	The study provides a clear information about the data used in the analysis and how it was collected and validated.
QA_3	Threats to Validity (or Limitation)	The study provides the threats that affect the validation of the study including internal, external, and construction threats.
QA_4	Empirical Results Interpretation	The study provides sufficient and thorough interpretation of the results.
QA_5	Reflections on SE	The study reflects and links the results to the studied aspects of software engineering.
QA_6	Reproducibility	The study provides sufficient information to reproduce the experiments. E.g., links to the datasets, the steps in details...etc.

Table 5: The quality assessment items. The evaluation of each item can take a descriptive value from Explicit, Implicit, or None with numeric values 1, 0.5, or 0.

Based on the defined quality criteria shown in Table 5, we got the quality scores for all of the primary studies as shown in Table 6. The average quality score for all primary studies is 3.9 out of 5. The retrieved primary studies from the $IEEE$ library have, on average, higher quality than the studies from the other libraries. The average quality scores for the primary studies retrieved from the *ACM*, *ScienceDirect*, and the Springer libraries are 3.9, 3.2, and 2, respectively. The results of quality assessment show that there are 5 PSs with quality score 6, which is the maximum, out of the 35 primary studies. The lowest quality score was 0.5 for only one primary study. We also calculated the average score for each quality aspect shown in Table 5. The table shows that the average value for the quality assessment item QA_6 was the lowest among the other quality assessment items, while the highest average value was for QA_4. We found a 0.4 positive correlation, using the R^2 coefficient, between the quality of the primary studies published in the *IEEE* over time. This correlation was insignificant for the other libraries. Figure 6 shows the distribution of the quality scores of the primary studies.

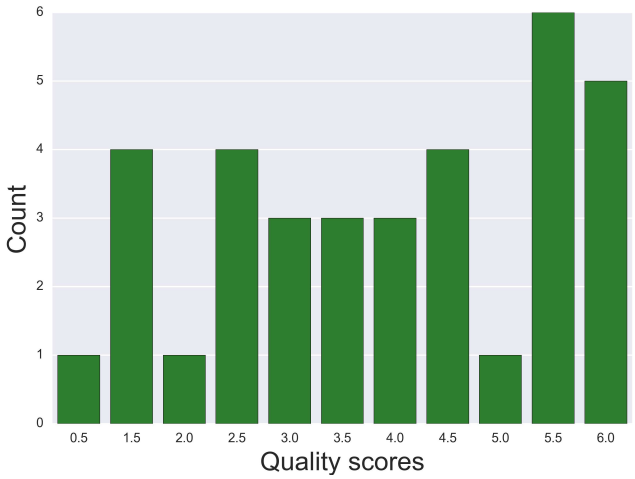

Fig. 6: The distribution of the number of primary studies over the quality scores.

[4] The full name of the venue can be found in the references.

PS	Quality Aspect						Quality score
	QA_1	QA_2	QA_3	QA_4	QA_5	QA_6	
[55]	1	1	1	1	1	0.5	5.5
[51]	0.5	0	0	0	0	0	0.5
[46]	0.5	0.5	0	0.5	0	0	1.5
[45]	1	0.5	1	1	1	0	4.5
[44]	1	1	1	1	1	1	6
[42]	0	1	0	0.5	1	0	2.5
[37]	0	1	1	0.5	0.5	1	4
[34]	0.5	1	1	1	1	0	4.5
[29]	0.5	1	1	0.5	0	0	3
[28]	0.5	0	0	0.5	0.5	0	1.5
[20]	1	1	0	1	1	0.5	4.5
[17]	1	1	1	1	1	1	6
[11]	1	1	0	1	1	0.5	4.5
[12]	1	1	1	1	1	0.5	5.5
[10]	1	1	0	1	0.5	0	3.5
[60]	1	1	1	1	1	1	6
[58]	0	1	0	1	1	0.5	3.5
[57]	1	1	1	1	1	0.5	5.5
[52]	1	1	0	0	0	0.5	2.5
[48]	0.5	0	0	1	0	0	1.5
[38]	0.5	0.5	0	1	0.5	0.5	3
[36]	1	1	1	1	1	1	6
[30]	1	1	1	1	1	1	6
[27]	1	0	1	1	0.5	0	3.5
[25]	1	0.5	1	1	1	0.5	5
[16]	1	0	1	1	1	0	4
[9]	1	0.5	0	0.5	0.5	0	2.5
[8]	1	1	1	1	1	0.5	5.5
[7]	0.5	1	1	1	1	1	5.5
[5]	1	1	0	1	0.5	0.5	4
[53]	0.5	0.5	0	1	1	0	3
[50]	0.5	1	1	1	1	1	5.5
[56]	0	0.5	0	1	0	0.5	2
[24]	0	0	0	1	0.5	0	1.5
[21]	0	1	0.5	0.5	0	0.5	2.5
Avg.	0.69	0.7	0.53	0.84	0.7	0.4	3.9

Table 6: The quality assessment of the primary studies.

3.4 Data extraction and synthesis

Data synthesis for heterogeneous studies is always hard, particularity when the primary studies cover the same aspect with different measurements and present the results in different ways. Also, the primary studies in most cases use a mixed-methods for the analysis that makes a universal comparison impossible. Thus, we followed a meta-ethnography method [15], as advised by Silva et al. in [19], which translates the quantitative findings of the PSs into comparable features that can be used for a proper analysis. Thus, we invested a lot of time during the execution of this SLR in designing a good set of features to be extracted from the primary studies in order to answer the research questions. Table 7 shows the set of features extracted from each primary study. Features in category "A" are documentary features that capture the general goal of each primary study. For space limitation, we will discuss few of them in the following. Feature "A1" in this review has the same value (i.e., *Human Aspects* that is the main topic of this review). Feature

"A2" represents the SE context that the proposed network analysis method serves, for example, how can we do bug assignment by analyzing the developers collaboration network?. Feature "A3" captures the network being used in each PS. Features of category "B" address the network analysis aspects of the PSs, and features of category "C" address the other aspects. Features "B1" to "B6" address questions about the constructed network type. This is an important step towards understating how valid the constructed networks are, which is covered in features "B7" to "B10". Feature "B7" reflects how good an edge is. Real edges are real connections between the nodes. A proxy edge is used when the real edge is hard to obtain. For example, changing a file, a module, or any software artifact may not yield a real relationship. Instead, we call this interaction a proxy. An example of a real edge is the edges constructed in the communication email networks, in which the edge and its direction are explicitly found in reality. Having clear idea about the constructed network, we can check the validity of the used measures and their meaningfulness. Features "B11" to "B13" address the used measures in each PS. Feature "B12" is important and it reflects whether the meaning of the used measures is sufficiently explained or not, and if the explanation exists, whether it is meaningful or not. The feature "B14" tests whether the results in each PS are validated or not.

The extraction features in Table 7 are designed to be mapped to the research questions in Section 1.1. Extraction features "B1" to "B10" give us rich information that enable obtaining strong evidence regarding the quality of the constructed networks addressed in **RQ1**. Features "B11" to "B13" answer **RQ2**, while the feature "B14" is mapped to **RQ3**. Features "C1" to "C4" answer **RQ4**. The answers of the first four research questions enable us to answer **RQ5**.

Key	Description
A1	Human aspects or software artifact aspects
A2	The addressed context. Examples: bug tracking and team formation.
A3	What is the constructed network? Examples: Developer-Module network and social networks of developers.
B1	What is a node?
B2	What is an edge?
B3	Is the network multiplex (Multilayer)?
B4	Is the network bipartite (2-mode networks)?
B5	Is the network dynamic or static?
B6	Is the network directed or undirected ?
B7	Edge quality.
B8	Is the used network aggregated from multiplex interactions?
B9	Is the used network aggregated from longitudinal networks?
B10	Are edge weights (like collaboration intensity/frequency) considered?
B11	What are the used measures?
B12	Are the meaning of the used measures explained?
B13	Does the work contribute new measures, tools, models, and algorithms that are based on network analysis?
B14	Does the work use any Null Model to validate the results?
C1	Data source project. Open source and/or industry
C2	The analyzed project(s)
C3	Link the results and interpret them in the context of SE, i.e., A2
C4	Practical implications, for examples, lessons or advice for practitioners

Table 7: The extraction features of the SLR.

4 Results

In this section, we will provide the synthesized results, while we provide an evidence-based answers to the research questions in Section 5.

4.1 From SE to a network (Fields A)

In this section, we provide the results for the extraction features "A1" to "A3". As per the inclusion and exclusion criteria described in Section 3.2, all of the included PSs were handling *Human Aspects* of software engineering. Thus, the 35 PSs had that value in that field. Feature $A2$ had different values that handled different contexts of SE. There are 8 PSs that handled *Bug* related issues like bug assignment, bug tracking and bug prediction. Other PSs addressed different contexts like *Build Failure*, *Code Ownership*, *Team Organization*, *Failure Prediction*, *Code Quality*, and *Team Collaboration*.
Figure 7 shows the used networks in the PSs. The most used network, in 15 PS, was the *Collaboration Network* where the nodes were developers and an edge appeared between any two developers if they worked together on the same file, line of code, module, or project. If the collaboration network was based on more than one type of interaction between the developers, then this was a one-mode projection from a multiplex bipartite network. The *Communication network* was constructed by modeling the developers as nodes and the edges were communication messages between developers. The *Developer-project network* was a bipartite network where the two sets V_L and V_R were the developers and the projects, respectively, and the edges represented developers taking part in a project. Similarly, the *Developer-module network* was a bipartite network between developers and the modules. Those two networks were found in 7 primary studies, [45,42,11,9,56,24,21]. The *Follow network* was the social network of the Github platform where a developer could explicitly follow another one, like Twitter. The other two networks, the *Hierarchical* and *Repository* networks were devised and contributed networks of the works in [5,38], respectively.

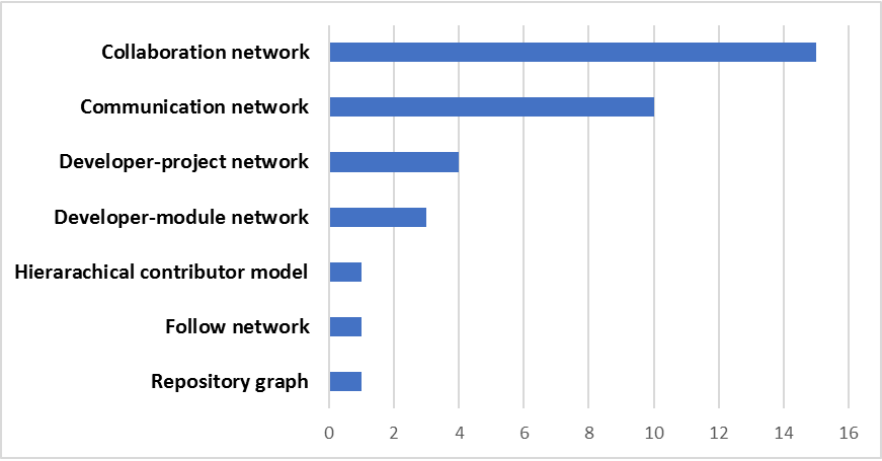

Fig. 7: The constructed networks in the PSs and their frequencies.

4.2 The constructed networks, their validity, and measures validation (Features B)

Not surprisingly, the nodes in the majority of the networks of the PSs were *Contributors*, which means any human who contributed to a software repository, like a developer, a tester, an architect, etc. An exception to this was the work in [38,9,56,24,21] where the nodes were two sets of a bipartite network and the analysis was done on the constructed bipartite network, not on a one-mode projected network.

For feature "B3", there were 3 PSs, [11,60,8], where the constructed network was a multiplex network. Additionally, only 5 PSs, [46,34,58,25,53], considered the temporality of a network and provided a *dynamic* network. The rest of the primary studies considered a *static* network. The PSs that considered directed networks were 13 PSs, [55,17,10,60,57,48,38,30,9,8,5,53,50] and the rest PSs considered an undirected network. We found that 60% of the PSs, 21 PSs, used a *proxy* edge. Only 5 PSs, [55,46,44,8,56], used a multiplex representation of the network. The rest used a unipelx network. Nearly 89% of the PSs aggregated the networks over time and used a single static network, only 4 primary studies, [46,34,58,25] considered different networks of the same nodes over time and performed analysis based on this situation. For feature "B10", only 13 PSs considered the weights of the edges, while the other 22 PSs did not consider the edge weights.

The used measures in the PSs are shown in Figure 8. The figure shows that the *Degree* and the *Betweenness* centrality measures were dominant among the used measures with 21 and 17 times used in the PSs, respectively. Motif analysis was used only once in the work [52]. The measures in "Others" are measures that were used only one time across the PSs. For the feature "B12", we

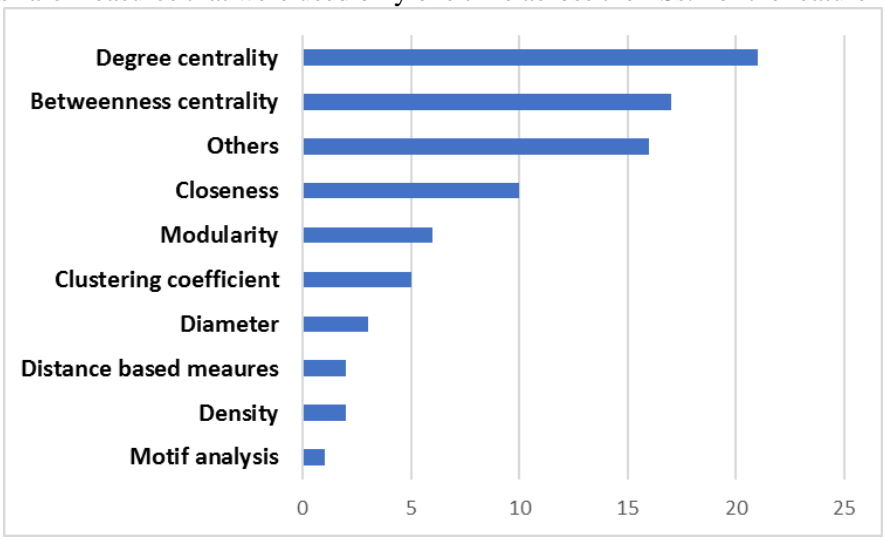

Fig. 8: The used measures in the PSs and their frequencies.

found that 11 PSs did not provide sufficient definition and explanation for the used measures, and only 9 PSs, [45,44,11,60,58,38,30,16,8], contributed (devised) new measures and(or) models for the constructed networks. Surprisingly, we found only 14%, 5 PSs out of the 35, that used a *Null model* to validate the results. The rest of the PSs did not use any statistical validation of the results.

4.3 Data, results, and interpretation (Feature C)

The used data set in the PSs was distributed as follows: (1) 6 PSs used data from industry, (2) 27 PSs used data from open source platforms, and (3) 2 PSs, [9,29], used both industry and open source data in the experiments. From the open source project, the projects *Mozilla*, *Apache*, *Eclipse*, and *Linux* were used much often. For the industry projects, the *MS-Vista binaries* were used 2 times by the same author in [11,45]. The MS-Vista binaries was used also in addition to an open source project (i.e., the *Eclipse*) in the work [9]. Two PSs did not provide information about the used industrial data [20,16]. The projects Eclipse, Firefox, and Apache were used the most in the PSs with frequencies 7, 6, and 4 times, respectively. The answer to feature "C3" found as follows. Only 34% of the PSs, 12 PSs, provided an *Explicit* interpretation of the results in the context of the studied context of SE, feature "A2". Other 14 studies provided *Implicit* interpretation of the results, which means it was not clear in the primary study how the context in "A2" was related to the results *directly*. The rest of the results, 9 PSs, did not provide any information about connecting the results to feature "A2".
The PSs that provided *Explicit* lessons and guidelines for the practitioners based on the results of the study were 5 studies only, [55,45,44,11,12]. Also, 4 studies [42,17,7,53] provided *Implicit* lessons and guidelines for the practitioners, and the rest did not provide any information.

5 Discussions

In this section, we discuss the results presented in Section 4, and provide answers to the research questions in Section 1.1.

5.1 Discussion around RQ1:

To answer this question, we need to test the quality of the constructed network. To that end, we will use the results of features "B1" to "B10" to answer this question. The quality of the network stems from the quality of its components (i.e., the nodes and the edges) and on the other aspects we mentioned in the definition section like being weighted, multiplex, or bipartite. Based on the results, we found that 60% of the PSs used a proxy edge. A proxy edge is obtained by doing one (or more) of the following:

- Classical OMP: which was described in the Section 2. The PSs [21,24,56,8,42,53] did classical OMP, which does not respect the edge weights. That means, the links between any two developers, for example, in the network who collaborated 1 time or 1000 times are treated equally. The resulted network is not informative and a lot of measure, like the degree centrality, do not provide meaningful results for the projected networks if the weights are ignored.
- Multiplex(space) aggregation: In this type of aggregation, like PSs [8,12,53,55,44,56], the edges are aggregated from different types of relationships(interactions). This results in a dense disguised network, in which the link do not reflect any special interaction that can be meaningfully quantified. Edge aggregation from different types of interactions is an oversimplification of reality, which yields wrong conclusions as shown by Cardillo et al. [18].
- Temporal(time) aggregation: In this type of aggregation, like the PSs [42,37,29,28,20,12,48,38,36,30,27,8,7,53,50,56,24,21], the edges are aggregated over time into a single network.

The problem of the previous classical OMP and the aggregations is that it generates a very dense graphs, where every node is almost connected to every other node in the network. This renders many measures, like the degree centrality, not of much benefit. Some researchers were aware of this problem and included this as a threat to validity in their work, like the PS [29]. Additionally, the bipartite networks, the dynamic networks, and the multiplex networks have their special characteristic that differs from the unipartite static network. Thus, converting any non-unipartite, non-static, or multiplex network into a unipartite static network should be done with extreme care if required, and generally we recommend keeping the network in its nature. Additionally, the collaboration intensity (or frequency) was not considered when it should be. There are 22 PSs that did not consider the weights of the edges in the graph where the weights are crucial. Thus, our answer to the *RQ1* is: *In* 60% *of the PSs, the constructed networks are not of sufficient quality to be used as a model for the studied context. Accordingly, we believe that rigorous validation need to take place for the subsequent analysis results of those PSs.*
To address the problems found in the PSs for this research question, we provide here some pointers to extend the space of the used methods. For example, the work by Zweig [62] provides a systematic method to do a proper one mode projection, the works by Kivelä et al. [33] and by Boccaletti et al. [13] provide a good resource for the methods, the models, and the measures that can be used to analyze the multiplex networks, and the seminal work by Holme and Saramäki [26] provide frameworks, measures, and methods for the temporal networks (dynamic networks).

5.2 Discussion around RQ2:

To answer this question, we will provide a discussion on the used measures in the PSs, and extensively talk about one of them, which is the betweenness centrality that was used too much. Borgatti [14] showed that many of the centrality measures embraced unstated assumptions that once were not satisfied, the results of the measures could not be reliable and interpretable. For space limitation, we will discuss the betweenness centrality that was used 16 times in the PSs. This centrality measure was introduced by Freeman [23] and implicitly assumed that: (1) there is a process going on top of the network, (2) the process is based on the shortest paths, (3) the process takes place among all pairs of nodes with the same frequency, and (4) the process is sequential [61]. Thus, any network where those assumptions are not met can not benefit from the betweenness centrality as a measure for identifying central nodes. Based on this information and the information provided in Section 5.1, none of the networks in the PSs is suitable to apply the betweenness centrality on it. Any obtained results based on this measure and these networks are not meaningful. Similar situations to this are also found with other measures. For example, the diameter of the network and the closeness centrality do not give an informative value in the analysis of a social interaction. Those measures are informative in networks where the cost of establishing an edge is high, like adding new street in the street networks and adding new DNS server in the internet network. However, for communication and social networks where an edge is a sent email, and for the collaboration networks where the edge is a repository fork or a comment added to the code, the diameter, for example, is just meaningless and does not provide a value to the analysis. In addition, only 9 PSs, [45,44,11,60,58,38,30,16,8], contributed a new model or a new measure based on the specifics of the used networks, the other 26 PSs only used an existing measure in the literature. Also, 11 primary studies did not provide an explanation and interpretation over the used measure. Thus, our answer to *RQ2* is: *In at least* 25 *PSs, the used measures were not suitable and were not reliable. Also, in* 11 *PSs, the provided measures were*

not sufficiently explained in the context of used network. Thus, the results of those PSs need a proper justification for their usage validity. To tackle this issue, we recommend designing models and devising measures for each constructed network if the existing methods and measures are not suitable. In the primary studies covered in this SLR, there are 9 PSs, [45,44,11,60,58,38,30,16,8], that provided new measures and models specifically for the constructed network that respected the nature of the modeled system.

5.3 Discussion around RQ3:

Classically, the random graphs null models are used to validate the significance of the results obtained from networks. For our PSs, only 14% of them, 5 PSs,[11,12,30,27,16], out of the 35, used a *Null model* to validate the results. The rest of the PSs used no validation method at all. We noticed that those studies used the random graphs null model to validate the quality of the found clusters (community). Thus, our answer to the *RQ3* is: *The results of* 86% *of the PSs in this SLR were not validated at all. Accordingly, the reported results of these studies may not be reliable.* To overcome this issue, we provide here some pointers for the null models used to validate the significance of the results. Newman [40] provided a description of using random graphs as a null model. The work by Schlauch et al. [47] provided a comprehensive comparison between different null models.

5.4 Discussion around RQ4:

We noticed that 37% of the PSs, 3 out of 8, used industry data sets and 24% of the PSs, 7 out of 29, used open source data sets provided *Explicit* lessons and guidelines for the practitioners. Based on that and on the results provided in Section 4.3, our answer to the *RQ4* is: *Translating the results of the analysis into the studied context of SE is poor. Also, the number of PSs that provided lessons and guidelines for practitioners is very limited.*

5.5 Discussion around RQ5:

RQ5 has been partially answered after answering *RQ1*, *RQ2*, *RQ3*, and *RQ4*. Here we elaborate more on the answer for *RQ5* based on the results in Section 4 and based on the answers to the research questions in Sections 5.1 to 5.4. Another contribution in this paper is the following items that may help the researchers to further extend the body of the work in mining software repositories using network science.

1. *Temporal measures:* As we saw in the previous sections, the constructed networks were in many times aggregated over time and the static measures were applied, which weakens the validity of the results or simply turns them not meaningful [31]. Thus, we recommend using the dynamic networks and their corresponding measures as provided by Kim et al. [31] and by Holme et al. [26]. We strongly think that utilizing temporal networks as a model to understand the dynamics of collaboration and its effect would give more actionable insights.
2. *Weighted networks:* The collaboration between developers and the communication between them can hardly be imagined without considering the intensity (or frequency) of this collaboration or communication, which is a rich information that should be utilized. The work by Opsahl [43] and by Newman [41] provide methods for bipartite and weighted networks and their corresponding centrality measures.
3. *Community detection*: The PSs that handle community detection were based on the work of Newman [39]. A recent work by Ahn et al. [4] showed promising results for finding

structural properties in networks based on the link communities. This method could provide more robust team organizations with the collaborative development.

4. *Link prediction and assessment*: The prediction of collaboration or communication between developers was, surprisngly, not found in the PSs. In network science there is a plethora of work in link prediction started by Liben-Nowell and Kleinberg [35]. The link prediction is an active area in network science and can be utilized in mining software repositories, especially when incorporating external information [1,2,3]. An application of the link prediction in this context could be built as recommendation system for bringing developers to projects in open source.

5.6 Closing thoughts

- *The collaboration networks*: in the PSs were projected from different interactions. They were proxy networks from the one mode projection. In most cases, we could not retrieve any information about the procedures done by the researchers in order to get the OMP.
- *The communication networks*: they were in some cases projected from bipartite network of thread-developer network, where a thread was a message that was sent to all developers and any one could reply it. This results in a very dense network that was not really a communication network.
- *Dynamic vs Static*: Should be considered carefully. While static networks give some insights regarding the modeled network, a dynamic analysis of a network gives richer insights regarding the process that is being done over this network.
- *Bipartite networks*: should be utilized as bipartite networks when possible. This requires extra measures and more models that are context dependent.
- *In-applicability of some measures*: not every measure can provide meaningful results for every network. A measure should be first understood well in order to use it and get useful insights.

5.7 Threats to validity

- *Completeness*: The completeness of any SLR is hard to attain. We described the steps we followed to cover as relevant studies as possible.
- *Reproducibility*: We incorporated the QGS in order to guarantee the reproducibility of the results in the SLR. All of the data used for this review is available upon request.

6 Conclusion

In this work, we have presented a systematic literature review to identify and evaluate the use of network science as a tool to understand the collaboration of developers in software engineering. We followed the most rigor steps we know in performing a systematic review. We identified 35 primary studies, assessed their quality, and extracted the data from them. The data extracted from the primary studies was used to answer 5 research questions. Our answers showed that the primary studies used networks that lack the required quality, they used in most cases unsuitable measures, and in most cases they do not validate the results. We provided some pointers that can be utilized in future research works in order to do a good network analysis with actionable insights.

References

1. M. Abufouda and K. A. Zweig. Interactions around social networks matter: Predicting the social network from associated interaction networks. In *2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014)*, pages 142–145, Aug 2014.
2. M. Abufouda and K. A. Zweig. Are we really friends?: Link assessment in social networks using multiple associated interaction networks. In *Proceedings of the 24th International Conference on World Wide Web*, pages 771–776. ACM, 2015.
3. M. Abufouda and K. A. Zweig. Link classification and tie strength ranking in online social networks with exogenous interaction networks. *arXiv preprint arXiv:1708.04030*, 2017.
4. Yong-Yeol Ahn, James P Bagrow, and Sune Lehmann. Link communities reveal multiscale complexity in networks. *Nature*, 466(7307):761–764, 2010.
5. M.Y. Allaho and Wang-Chien Lee. Analyzing the social ties and structure of contributors in open source software community. In *Advances in Social Networks Analysis and Mining (ASONAM), 2013 IEEE/ACM International Conference on*, pages 56–60, Aug 2013.
6. Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.
7. Nicolas Bettenburg and A.E. Hassan. Studying the impact of social structures on software quality. In *Program Comprehension (ICPC), 2010 IEEE 18th International Conference on*, pages 124–133, June 2010.
8. P. Bhattacharya, I. Neamtiu, and M. Faloutsos. Determining developers' expertise and role: A graph hierarchy-based approach. In *Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on*, pages 11–20, Sept 2014.
9. C. Bird, N. Nagappan, H. Gall, B. Murphy, and P. Devanbu. Putting it all together: Using socio-technical networks to predict failures. In *Software Reliability Engineering, 2009. ISSRE '09. 20th International Symposium on*, pages 109–119, Nov 2009.
10. Christian Bird, Alex Gourley, Prem Devanbu, Michael Gertz, and Anand Swaminathan. Mining email social networks. In *Proceedings of the 2006 International Workshop on Mining Software Repositories*, MSR '06, pages 137–143, New York, NY, USA, 2006. ACM.
11. Christian Bird, Nachiappan Nagappan, Brendan Murphy, Harald Gall, and Premkumar Devanbu. Don't touch my code!: Examining the effects of ownership on software quality. In *Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering*, ESEC/FSE '11, pages 4–14, New York, NY, USA, 2011. ACM.
12. Christian Bird, David Pattison, Raissa D'Souza, Vladimir Filkov, and Premkumar Devanbu. Latent social structure in open source projects. In *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, SIGSOFT '08/FSE-16, pages 24–35, New York, NY, USA, 2008. ACM.
13. Stefano Boccaletti, Ginestra Bianconi, Regino Criado, Charo I Del Genio, Jesús Gómez-Gardenes, Miguel Romance, Irene Sendina-Nadal, Zhen Wang, and Massimiliano Zanin. The structure and dynamics of multilayer networks. *Physics Reports*, 544(1):1–122, 2014.
14. Stephen P Borgatti. Centrality and network flow. *Social networks*, 27(1):55–71, 2005.
15. Nicky Britten, Rona Campbell, Catherine Pope, Jenny Donovan, Myfanwy Morgan, and Roisin Pill. Using meta ethnography to synthesise qualitative research: a worked example. *Journal of health services research & policy*, 7(4):209–215, 2002.
16. B. Caglayan, A.B. Bener, and A. Miranskyy. Emergence of developer teams in the collaboration network. In *Cooperative and Human Aspects of Software Engineering (CHASE), 2013 6th International Workshop on*, pages 33–40, May 2013.
17. Gerardo Canfora, Luigi Cerulo, Marta Cimitile, and Massimiliano Di Penta. Social interactions around cross-system bug fixings: The case of freebsd and openbsd. In *Proceedings of the 8th Working Conference on Mining Software Repositories*, MSR '11, pages 143–152, New York, NY, USA, 2011. ACM.
18. Alessio Cardillo, Massimiliano Zanin, Jesús Gómez-Gardenes, Miguel Romance, Alejandro J García del Amo, and Stefano Boccaletti. Modeling the multi-layer nature of the european air transport network: Resilience and passengers re-scheduling under random failures. *The European Physical Journal Special Topics*, 215(1):23–33, 2013.
19. Fabio QB Da Silva, Shirley SJO Cruz, Tatiana B Gouveia, and Luiz Fernando Capretz. Using meta-ethnography to synthesize research: A worked example of the relations between personality and software team processes. In *Empirical Software Engineering and Measurement, 2013 ACM/IEEE International Symposium on*, pages 153–162. IEEE, 2013.

20. Subhajit Datta, Vikrant Kaulgud, Vibhu Saujanya Sharma, and Nishant Kumar. A social network based study of software team dynamics. In *Proceedings of the 3rd India Software Engineering Conference*, ISEC '10, pages 33–42, New York, NY, USA, 2010. ACM.
21. Andrew Dittrich, Mehmet Hadi Gunes, and Sergiu Dascalu. Network analysis of software repositories: identifying subject matter experts. In *Complex Networks*, pages 187–198. Springer, 2013.
22. Tore Dyba, Torgeir Dingsoyr, and Geir K. Hanssen. Applying systematic reviews to diverse study types: An experience report. In *Proceedings of the First International Symposium on Empirical Software Engineering and Measurement*, ESEM '07, pages 225–234, Washington, DC, USA, 2007. IEEE Computer Society.
23. Linton C Freeman. Centrality in social networks conceptual clarification. *Social Networks*, 1(1978/79):215–239.
24. Yongqin Gao and Greg Madey. Network analysis of the sourceforge. net community. *Open Source Development, Adoption and Innovation*, pages 187–200, 2007.
25. Peng He, Bing Li, and Yuan Huang. Applying centrality measures to the behavior analysis of developers in open source software community. In *Cloud and Green Computing (CGC), 2012 Second International Conference on*, pages 418–423, Nov 2012.
26. Petter Holme and Jari Saramäki. Temporal networks. *Physics reports*, 519(3):97–125, 2012.
27. Qiaona Hong, Sunghun Kim, S.C. Cheung, and Christian Bird. Understanding a developer social network and its evolution. *2013 IEEE International Conference on Software Maintenance*, 0:323–332, 2011.
28. Shih-Kun Huang and Kang-min Liu. Mining version histories to verify the learning process of legitimate peripheral participants. *SIGSOFT Softw. Eng. Notes*, 30(4):1–5, May 2005.
29. Andrejs Jermakovics, Alberto Sillitti, and Giancarlo Succi. Mining and visualizing developer networks from version control systems. In *Proceedings of the 4th International Workshop on Cooperative and Human Aspects of Software Engineering*, CHASE '11, pages 24–31, New York, NY, USA, 2011. ACM.
30. M. Joblin, W. Mauerer, S. Apel, J. Siegmund, and D. Riehle. From developer networks to verified communities: A fine-grained approach. In *Software Engineering (ICSE), 2015 IEEE/ACM 37th IEEE International Conference on*, volume 1, pages 563–573, May 2015.
31. Hyoungshick Kim and Ross Anderson. Temporal node centrality in complex networks. *Physical Review E*, 85(2):026107, 2012.
32. Barbara Kitchenham and S Charters. Guidelines for performing systematic literature reviews in software engineering. In *Technical report, Ver. 2.3 EBSE Technical Report. EBSE*. sn, 2007.
33. Mikko Kivelä, Alex Arenas, Marc Barthelemy, James P Gleeson, Yamir Moreno, and Mason A Porter. Multilayer networks. *Journal of complex networks*, 2(3):203–271, 2014.
34. Amit Kumar and Avdhesh Gupta. Evolution of developer social network and its impact on bug fixing process. In *Proceedings of the 6th India Software Engineering Conference*, ISEC '13, pages 63–72, New York, NY, USA, 2013. ACM.
35. David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *journal of the Association for Information Science and Technology*, 58(7):1019–1031, 2007.
36. A. Meneely and L. Williams. Socio-technical developer networks: should we trust our measurements? In *Software Engineering (ICSE), 2011 33rd International Conference on*, pages 281–290, May 2011.
37. Andrew Meneely, Laurie Williams, Will Snipes, and Jason Osborne. Predicting failures with developer networks and social network analysis. In *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, SIGSOFT '08/FSE-16, pages 13–23, New York, NY, USA, 2008. ACM.
38. Xiaozhu Meng, B.P. Miller, W.R. Williams, and A.R. Bernat. Mining software repositories for accurate authorship. In *Software Maintenance (ICSM), 2013 29th IEEE International Conference on*, pages 250–259, Sept 2013.
39. M. E. J. Newman. Detecting community structure in networks. *The European Physical Journal B*, 38(2):321–330, Mar 2004.
40. Mark EJ Newman. Random graphs as models of networks. *arXiv preprint cond-mat/0202208*, 2002.
41. Mark EJ Newman. Analysis of weighted networks. *Physical review E*, 70(5):056131, 2004.
42. Masao Ohira, Naoki Ohsugi, Tetsuya Ohoka, and Ken-ichi Matsumoto. Accelerating cross-project knowledge collaboration using collaborative filtering and social networks. *SIGSOFT Softw. Eng. Notes*, 30(4):1–5, May 2005.
43. Tore Opsahl, Filip Agneessens, and John Skvoretz. Node centrality in weighted networks: Generalizing degree and shortest paths. *Social networks*, 32(3):245–251, 2010.
44. Sebastiano Panichella, Gerardo Canfora, Massimiliano Di Penta, and Rocco Oliveto. How the evolution of emerging collaborations relates to code changes: An empirical study. In *Proceedings of the 22Nd International Conference on Program Comprehension*, ICPC 2014, pages 177–188, New York, NY, USA, 2014. ACM.

45. Martin Pinzger, Nachiappan Nagappan, and Brendan Murphy. Can developer-module networks predict failures? In *Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, SIGSOFT '08/FSE-16, pages 2–12, New York, NY, USA, 2008. ACM.
46. Mathias Pohl and Stephan Diehl. What dynamic network metrics can tell us about developer roles. In *Proceedings of the 2008 International Workshop on Cooperative and Human Aspects of Software Engineering*, pages 81–84. ACM, 2008.
47. Wolfgang E Schlauch, Emőke Ágnes Horvát, and Katharina A Zweig. Different flavors of randomness: comparing random graph models with fixed degree sequences. *Social Network Analysis and Mining*, 5(1):1–14, 2015.
48. M. Schwind and C. Wegmann. Svnnat: Measuring collaboration in software development networks. In *E-Commerce Technology and the Fifth IEEE Conference on Enterprise Computing, E-Commerce and E-Services, 2008 10th IEEE Conference on*, pages 97–104, July 2008.
49. Shai S Shen-Orr, Ron Milo, Shmoolik Mangan, and Uri Alon. Network motifs in the transcriptional regulation network of escherichia coli. *Nature genetics*, 31(1):64–68, 2002.
50. Sulayman Sowe, Ioannis Stamelos, and Lefteris Angelis. Identifying knowledge brokers that yield software engineering knowledge in {OSS} projects. *Information and Software Technology*, 48(11):1025 – 1033, 2006.
51. Ashish Sureka, Atul Goyal, and Ayushi Rastogi. Using social network analysis for mining collaboration data in a defect tracking system for risk and vulnerability analysis. In *Proceedings of the 4th India Software Engineering Conference*, ISEC '11, pages 195–204, New York, NY, USA, 2011. ACM.
52. D. Surian, D. Lo, and Ee-Peng Lim. Mining collaboration patterns from a large developer network. In *Reverse Engineering (WCRE), 2010 17th Working Conference on*, pages 269–273, Oct 2010.
53. Sergio L Toral, María del Rocío Martínez-Torres, and Federico Barrero. Analysis of virtual communities supporting oss projects using social network analysis. *Information and Software Technology*, 52(3):296–303, 2010.
54. Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world' networks. *nature*, 393(6684):440–442, 1998.
55. Timo Wolf, Adrian Schroter, Daniela Damian, and Thanh Nguyen. Predicting build failures using social network analysis on developer communication. In *Proceedings of the 31st International Conference on Software Engineering*, ICSE '09, pages 1–11, Washington, DC, USA, 2009. ACM.
56. Jin Xu, Scott Christley, and Gregory Madey. Application of social network analysis to the study of open source software. In *The Economics of Open Source Software Development*, pages 247 – 269. Elsevier, Amsterdam, 2006.
57. Jifeng Xuan, He Jiang, Zhilei Ren, and Weiqin Zou. Developer prioritization in bug repositories. In *Software Engineering (ICSE), 2012 34th International Conference on*, pages 25–35, June 2012.
58. Qi Xuan, M. Gharehyazie, P.T. Devanbu, and V. Filkov. Measuring the effect of social communications on individual working rhythms: A case study of open source software. In *Social Informatics (SocialInformatics), 2012 International Conference on*, pages 78–85, Dec 2012.
59. He Zhang and Muhammad Ali Babar. On searching relevant studies in software engineering. 2010.
60. Wen Zhang, Song Wang, Ye Yang, and Qing Wang. Heterogeneous network analysis of developer contribution in bug repositories. In *Cloud and Service Computing (CSC), 2013 International Conference on*, pages 98–105, Nov 2013.
61. Katharina Anna Zweig. *Network Analysis Literacy: A Practical Approach to Network Analysis Project Design*. Springer Publishing Company, Incorporated, 2014.
62. Katharina Anna Zweig and Michael Kaufmann. A systematic approach to the one-mode projection of bipartite graphs. *Social Network Analysis and Mining*, 1(3):187–218, 2011.