

Metrics and Models for Developer Collaboration Analysis in Microservice-Based Systems. A Systematic Mapping Study

Xiaozhou Li¹, Amr S. Abdelfattah³, Ruoyu Su¹, Joseph Lee⁴, Ernesto Aponte⁵, Rachel Koerner³, Tomas Cerny⁶ and Davide Taibi^{1,2}

¹M3S, University of Oulu, Oulu, Finland

²CloudSEA.AI, Tampere University, Tampere, Finland

³Department of Computer Science, Baylor University, Waco, Texas, USA

⁴University of Richmond, Richmond, Virginia, USA

⁵Universidad del Sagrado Corazón, San Juan, Puerto Rico

⁶Systems and Industrial Engineering, University of Arizona, Tucson, Arizona, USA

Abstract

Microservices enable different teams to develop and deploy services independently. Practitioners are frequently mentioning the need for independence between teams and developers, and the need for metrics to measure developer collaboration. To shed light on the existing metrics and models, we conducted a Systematic Mapping Study to identify models for measuring the development activities, the metrics adopted by these methods, and the output produced by the methods themselves. We identified 10 different models proposed in 14 research papers. Results show that a large amount of the existing models adopt qualitative metrics, questionnaires, and surveys to collect the information required while others use information from issue tracking and version systems. The results will enable practitioners and researchers to further validate and extend the research on metrics for evaluating the collaboration and independence among developers.

Keywords

microservice, microservice collaboration metrics, microservice organization models, software metrics,

1. Introduction

Service-based architectures are widely adopted in the industry. In particular, microservice architectures are growing in popularity, having been adopted by 85% of companies that are modernizing their applications [1].


Microservices are an extension of service-based architectures but are designed for smaller, specialized services, which are independently deployed to decrease inter-dependencies of services [2][3]. These microservices can be scaled independently and rapidly. The usage of microservice architecture has grown over recent years as companies aim to improve their

IWSM Mensura'23

✉ xiaozhou.li@oulu.fi (X. Li); amr_elsayed1@baylor.edu (A. S. Abdelfattah); ruoyu.su@student.oulu.fi (R. Su); joseph.lee@richmond.edu (J. Lee); eaponte81@sagrado.edu (E. Aponte); rachel_koerner1@baylor.edu (R. Koerner); tcerny@arizona.edu (T. Cerny); davide.taibi@oulu.fi (D. Taibi)



© 2021 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

competitive abilities. Microservices enable teams to develop and deploy each service independently. Therefore, it is not necessary for teams to synchronize with each other before deploying, which theoretically may improve the velocity of teams. However, teams often still need to synchronize for high-level decisions and for architectural aspects. Examples include: decisions on the adoption of a particular communication mechanism between services and the adoption of a new messaging or tracing platform.

Therefore, collaboration among developers and, in particular, between different teams is of paramount importance when considering microservices. We expect a team to have high collaboration internally, but collaboration between different teams should be reduced as much as possible—maximizing cohesion and minimizing coupling.

Different research groups have investigated collaboration in microservice-based projects [4][5]. There have also been secondary studies published in the context of microservice architectural quality [6][7], adoption benefits and issues [8], metrics for measuring their maintainability [9], as well as tools [10], methods [11][12] and metrics [13] to reconstruct the architectural structure of microservice-based systems. However, there is not yet a clear and widely accepted set of metrics that can be adopted, and there is not yet a review for classifying the models and metrics to evaluate the collaboration and communication among developers.

- A set of **14** studies analyzing (micro)services developer collaboration. Researchers can use these studies as a basis for future investigations into microservice developer collaboration analysis.
- A subset of **10** studies, reporting Metrics and Models for Developer Collaboration Analysis. For each model identified, we considered the availability of tools to support its application as well as existing empirical validations, and we extracted a set of factors, measures, and information that characterize the analyzed models.
- A synthesis and comparison of the Metrics and Models for Developer Collaboration Analysis.

Paper structure: The remainder of this paper is organized as follows: In Section 2, we present the review process and the methodology adopted in this work. Moreover, we define the research questions and describe the criteria we defined to assess whether a study contributes effectively to answering the research questions. In Section 3, we illustrate the information extracted from the reviewed papers. Section 4 discusses the results obtained. In Section 5, we explain the threats to the validity of this work. Finally, in Section 6 we draw conclusions and provide an outlook on future work.

2. Methodology

We aim at identifying and comparing the existing methods used to evaluate collaboration among developers in microservices and the metrics adopted by those methods.

Accordingly, to meet our expectations, we formulated the goal as follows, using the Goal/Question/Metric (GQM) template [14]:

<i>Purpose</i>	Characterize
<i>Object</i>	methods used to evaluate collaboration among developers in microservices
<i>Quality</i>	with respect to the models and metrics adopted
<i>Viewpoint</i>	as reported in scientific literature
<i>Context</i>	concerning the evaluation of developers' collaboration, from 2014 to June 2023.

We formulated the following Research Questions (RQ) based on the GQM defined above and the criteria defined in the Population, Intervention, Comparison and Outcome (PICO) structure [15, 16]. This structure makes it easier to identify the keywords in the next steps.

- Population: published scientific literature reporting evaluations of the collaboration among developers in microservices.
- Intervention: studies reporting models, methodologies, or tools.
- Comparison: comparison of the input and output.
- Outcome: methods, models, and metrics adopted for the evaluation of the collaboration among developers.

Based on these terms, we defined the following general Research Question. In the field of Microservices (Population), what methods can be used for evaluating the collaboration among developers (Intervention), how do they compare (Comparison), and what are the metrics considered in the methods (Outcome)?

We refined this general Research Question into different RQs, whose formulation and motivations are reported in Table 1.

Table 1
Research Questions

	Research Question	Motivations
RQ1	What are the methods used to evaluate collaboration among developers in microservices?	To identify the models and methodologies currently available as well as the effort researchers and practitioners should invest to fill the gaps in this area.
RQ2	What are the input metrics adopted by the selected methods?	To identify the metrics implemented by current methods and compare implementation between method categories.
RQ3	What are the outputs of the selected methods?	To identify the outcomes of models from current methods and their expression of indicating developer collaboration.

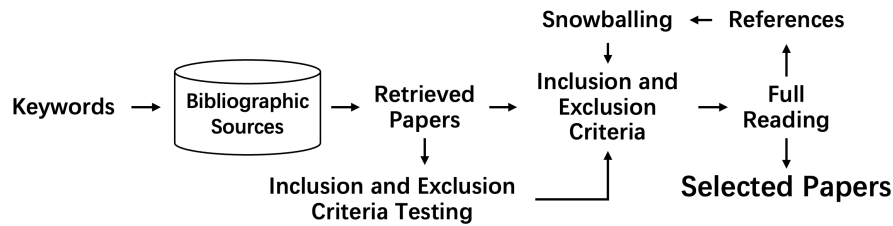


Figure 1: The Search and Selection Process

Table 2

Initial Literature Search by Library

Library	Count
Scopus	647
IEEE	223
ACM	134
Non-duplicates	888

2.1. Search Strategy

The search strategy is given in Fig. 1. First, the search terms of the review were defined. This was the first and most crucial step of our finding studies. After this step, the papers retrieved from the databases were screened by inclusion and exclusion criteria. Then we completed the full reading to obtain the final selected papers. At the same time, we also followed the guidelines by Wohlin, adopting a 'snowball' approach, i.e., by analyzing the references of the primary studies in order to extract more relevant sources for use [17].

2.1.1. Bibliographic Sources Identification

The search process was conducted by query filtration on bibliographic sources. The bibliographic sources we selected were: Scopus, IEEE Xplore, and the Association for Computing Machinery (ACM) digital library. From our experience, these resources produce reputable results that encompass smaller bibliographic source options. The numerical results are displayed in Table 2.

2.1.2. Inclusion and Exclusion Criteria.

The defined selection criteria distinguish relevant articles related to our proposed research questions. The criteria were applied to title and abstract in our initial review and then later to the entire text in our full text review. Our inclusion criteria in Table 3 indicate our specific interest in developer interaction from microservice-related organization structures. The exclusion criteria remove trivial results of extraneous format.

Table 3
Inclusion and Exclusion Criteria

Inclusion	Papers about collaboration, organization, or networks of developers Papers with specified system architecture (microservice, modular, distributed, service-oriented)
Exclusion	Not in English Duplicated (extension has been included) Out of topic (using the terms for other purposes) Not accessible by institution Non peer-reviewed papers Research Plans, roadmaps, vision and position papers

Table 4
Keyword Categorization

Roles	Structures
developer	collaboration
contributor	organization
committer	interaction
team	network

2.1.3. Keyword Identification

The search keywords were refined from the structure of our research questions. We started with the concept of "microservice". Finding that our expectation for the definition of a microservice spans multiple terms used in literature today, we expanded the query with alternative spellings and synonyms. Our next goal was to specify developer interaction in software organizations. We recognized that terms such as "organization", "team", or "interaction" on their own would produce excessive amounts of unrelated results. This is because the terms are frequently used beyond the software industry. Therefore, we decided to combine phrases in order to target software systems rather than any other business field. We arrived at two categories: role and structure, with four synonyms for each shown in Table 4. These we paired together so that each role term was combined with each structure term in our string query. In addition, we included the phrases "organizational network" and "collaborator network".

```

( microservice* OR micro-service* OR service-oriented OR distributed OR modular* OR
  peer-to-peer )
AND
( "developer organization" OR "contributor organization" OR "committer organization"
  OR "team organization" OR "developer collaboration" OR "contributor collaboration" OR
  "committer collaboration" OR "team collaboration" OR "developer interaction" OR
  "contributor interaction" OR "committer interaction" OR "team interaction" OR "developer
  network*" OR "contributor network*" OR "committer network*" OR "team network*" OR
  "organizational network*" OR "collaborator network*" )

```

2.1.4. Search and Selection Process

After defining our key elements and crafting our string query, we proceeded with our search process on the selected databases of bibliography sources. Our first run resulted in 888 non-duplicate papers.

Before applying the inclusion and exclusion criteria to every source, we tested a group of 80 papers. The criteria were applied to the title and abstract of these papers. This allowed us to confirm that our query produced results relevant to our research questions.

With the query verified, we applied the criteria to the remaining papers. Each paper was read and evaluated by two authors, and a third author was utilized in cases of disagreement. The authors determined whether to include or exclude the papers. Out of the initial 888 papers, 160 were included by title and abstract.

The second step in our selection process was a full document reading. The inclusion criteria were applied to the full text of the remaining 160 papers. For each paper, two authors determined inclusion, and in 12 instances, a third author was necessary to solve the disagreement. As part of this stage, a record of specific features presented in each paper was maintained. This included terms for the specific system architecture indicated, the team structures presented, as well as the metrics and tools used to identify collaboration. The existence of such specification within the paper is part of the inclusion criteria authors considered. At the end of this stage, 42 documents were included.

The third step of our process was both forward and backward snowballing. In forward snowballing, the references of currently included documents are analyzed by the inclusion criteria. In backward snowballing, references that cite the currently included documents are analyzed by the inclusion criteria. We ran iterations of snowballing until no new studies were founded. This resulted in 10 additional papers added to our set.

For our next step, we went through the existing papers in a round we entitled "Double Check". In this step, authors reviewed the inclusion criteria in the titles and abstract of each paper, adding additional data to our features record. This search and selection process resulted in 11 papers.

The remaining documents were then reviewed for data extraction. We recorded the examples of Method, Category, Input Data Source, Input Metric, and Model Outcome presented in each paper. One paper had no comments on metrics, and so was removed. The remaining documents are the final 10 papers included in this Systematic Mapping Study.

2.2. Data Extraction

To summarize the selected works, we built a table to focus the data on features specific to the research questions. The record developed in the Search and Selection Process was maintained in order to simplify this process. In this stage, we analyzed the Search and Selection record to narrow down the most relevant factors. Five types of information were extracted, as shown in Table 5.

- Name: of method implemented to evaluate collaboration within a system. These names were explicitly extracted from the full text of their paper.

Table 5
Data Extraction Features

RQs	Label	Information
RQ1	Method	Name Category
RQ2	Input	Data Sources Metrics
RQ3	Output	Model Outcomes

- Category: of the method. For instance, decision making theory and thematic analysis fall under the Social Studies Method category
- Input Data Sources: included for the method process.
- Input Metrics: calculations performed to achieve collaboration results. Patterns including measures of centrality, contribution, and modularity were identified.
- Model Outcomes: resulting from method completion. Each of the outcomes help interpret collaboration among developers.

3. Results

In this Section, we report the results to answer the RQs. Therein, the 10 selected papers contain five journal articles, four proceeding papers from conferences, and one from workshops (shown in Figure 2). The earliest two selected papers were published in 2005 and 2011 when their main focuses are distributed systems instead of microservice. More papers were published from 2019 regarding this theme (shown in Figure 3).

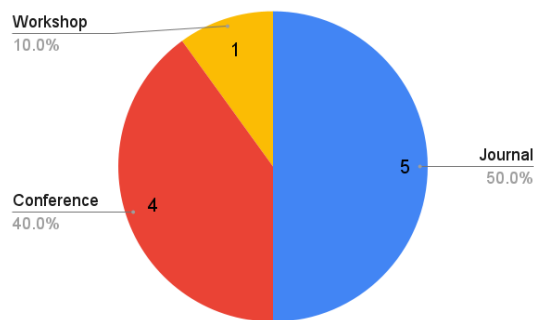


Figure 2: Paper Types

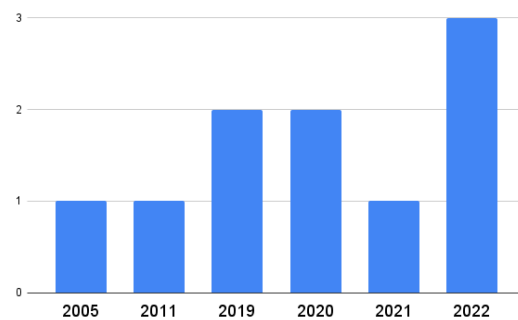


Figure 3: Papers by Years

3.1. RQ1. Method Categories

Summarized from the selected papers, we can observe that half of them adopted social studies methodologies towards empirical analysis on the organizational structures [SP6], [SP7], [SP8], [SP9], [SP10]. Though all these empirical studies utilize qualitative data as the input, the analysis methods and the target output are different.

Table 6
Methods for Organizational Structure Analysis for Microservice Projects

Paper	Method Category	Method Name
[SP1] [SP2] [SP3]	Data-driven Methods	Clustering Community Detection Centrality and Similarity Analysis
[SP4] [SP5]	Customized Methods	OCAM HSB
[SP6] [SP7] [SP8] [SP9] [SP10]	Social Studies Methods	Straussian Grounded Theory Decision Making Theory Logistic Regression Model Collaboration Pattern Analysis Distribution and Correlation Analysis

For example, [SP6] adopted Straussian Grounded Theory on the interview data from three companies to analyze the relation between the factor of adopting DevOps and microservice and that of software teams. [SP7] used Decision Making Theory on the collected decision episode data to identify the different decision-making processes which can be adopted to optimize the organizational structure of software project teams. [SP8] used logistic regression analysis on the qualitative data from two large-scale software projects to investigate the relation between the congruence measures and the failure proneness of source code files. [SP9] used Collaboration Pattern Analysis on questionnaire and interview data regarding a set of finished projects to investigate the relation between work division and team dynamics. [SP10] used Distribution and Correlation Analysis on the relation between the organizational measures and architectural smells. These studies provide approaches to investigate the different key factors that influence the quality of software project organizational structure.

Furthermore, several other studies used data-driven methods on large volumes of data collected from online version control systems, e.g., GitHub to investigate many aspects of software organizational structure [SP1], [SP2], [SP3], [SP4]. For example, [SP1] proposed the method to use interaction frequencies and complete-linkage hierarchical clustering methods on the data of committers number, commits number, number of changed files and number of changed lines to analyze the different developer categories. This method aims to help the practitioners identify the core developers working on different microservices. [SP2] and [SP3] used social network analysis approaches to investigate the organizational structure in the format of networks. Specifically, [SP2] used community detection method on commits, issues and system structure data to identify the latent developer communities in software projects as well as influence of activeness and stability of the organizational structures. [SP3] also used the commits data to construct the developer networks using the centrality analysis and similarity

Table 7

Method categories (RQ1), Method Input (RQ2) and Method Output (RQ3)

Paper	RQ1	RQ2		RQ3
	Method Category	Input Metrics	Data Source	Output
[SP1]	Data-driven	Dev.Con.	GitHub	Dev. Types
[SP2]		Dev.Con., Dev.Int.	GitHub	Org.Str., Org. Factor Impacts
[SP3]		Dev.Con., Dev.Int.	GitHub	Org.Str.
[SP4]	Customized	Dev.Con.	GitHub	Dev. Types, Dev. Ranking
[SP5]		Org.Str.	Survey Data	Org. Optimization
[SP6]	Social Studies	Org.Cha.	Survey Data	Org. Factor Impacts
[SP7]		Dec.Epi.	Survey Data	Org. Optimization
[SP8]		STC, SQa, DP.	Survey Data	Org. Factor Impacts
[SP9]		Dev.Col.Cha.	Survey Data	Org. Factor Impacts
[SP10]		Org.Cha.	Survey Data	Org. Patterns

analysis. [SP4] proposed the Ownership and Contribution Alignment Model (OCAM) to analyze the developer ranking in terms of multiple project metrics using commits, pull requests and code-related data. Specially, [SP5] proposed the method of Human Services Bus (HSB) to facilitate the optimization of organizational structure in terms of the workforce and streamline cross-unit processes.

To summarize, the majority of the studies related to organizational structure focus on using empirical studies methods to analyze the potential factors and their influences, extract organizational structure patterns and optimize the organizational structure in general. Several studies use social network analysis methods, e.g., centrality analysis and community detection, to detect the developers' relation structures, as well as their internal relations, e.g., categories and rankings. Customized methods are also proposed, e.g., OCAM and HSB, to support organizational structure analysis and optimization.

3.2. RQ2. Method Input Metrics

Herein, to answer RQ2, we also summarize what input metrics, as well as the data sources, are adopted by the academia to study the organizational structure in microservice-related projects (shown in Table 7).

By summarizing the selected papers, we can observe that the papers that used data-driven approaches (shown in Table 6) mainly adopted developer contribution metrics, e.g., Number of Commits, Code Churns, Code Complexity, Ticket Complexity, Pull Request Complexity, Number of Pull Requests, Number of Issues, etc., as the main input metrics [SP1], [SP2], [SP3]. Especially, the papers that contribute to the identification of organizational structures (*Org.Str.*) use developer contribution (*Dev.Con.*) together with developer interaction (*Dev.Int.*) as input metrics [SP2], [SP3]. All these three studies use the data collected from GitHub. [SP4] also adopted Github as data-source when their customized method (i.e., OCAM) utilizes also contribution as input. For [SP5], the HSB method they proposed uses documented or extracted organization structure information which is collected from qualitative data (surveys).

On the other hand, all the studies that adopt social studies methodologies (shown in Table 6)

uses survey data as the source but with different input metrics [SP6], [SP7], [SP8], [SP9], [SP10]. For example, both [SP6] and [SP10] consider the organizational characteristics (*Org.Cha.*), e.g., company culture, working environments, meetings, communications, etc. as the input metrics. These metrics can be obtained through surveys and analyzed by social studies methods. [SP7] uses decision episodes as input metric, which refer to "a sequence of messages that begins with a decision trigger that presents an opportunity or a problem that individuals need to decide on, that includes at least more than one message, and that possibly ends with a decision announcement" [18]. [SP8] adopts the Socio-technical congruence (STC), software quality (SQa), and development productivity (DP) as input metrics when [SP9] uses developer collaboration characteristics (*Dev.Col.Cha.*), e.g., Collaboration Intensity, collaboration density (overall, cross-location, cross-role), work division, etc.

3.3. RQ3. Method Outputs

To answer RQ3, the different outputs, and targets of the methods adopted by the selected papers are also summarized in Table 7. Therein, the methods proposed by [SP1] and [SP4] target providing developer (abbreviated as *Dev.*) types as outputs. In addition, [SP4] also provides the developer ranking as the output, where their visualization also shows the comparison between different teams with various focuses. On the other hand, [SP2] and [SP3] adopt social network analysis as a method and provide the network-shape organization (abbreviated as *Org.*) structure as output. Especially, [SP2] also provides the latent communities of the organizational structure networks as the output together with the analysis of the impact of organizational factors impact, e.g., organization stability. Both [SP5] and [SP7] provide organizational structure optimization as output, where [SP5] focuses on the workforce and streamlines cross-unit processes to leverage the new IT systems when [SP7] focuses on decision-making improvement. [SP6], [SP8] and [SP10] provide empirical analysis output targeting the impact of certain organizational factors on other factors. For example, [SP6] focuses on the analysis of the adoption of DevOps in the organization as a factor and its impact on the benefits and issues of the team. [SP8] focuses on the influential relations amongst the three factors, i.e., the Socio-technical congruence (STC), software quality (SQa), and development productivity (DP). On the other hand, [SP9] focuses on the location-related factors as the factors with their impacts on the team collaboration as output. Different from the other studies mentioned above, [SP10] provide organizational structure patterns as the output.

4. Discussion

The analysis of the literature shows that, despite the community of practitioners frequently mentioning the need for developer collaboration metrics and models, there is only a limited amount of works investigating such metrics and models.

As shown in Figure 4, all the data-driven methods consider metrics on developer contribution and developers' interactions while Social Studies methods, adopt a large number of metrics. While social studies are commonly based on survey data, it is interesting to note that in one case they also consider developers' interaction.

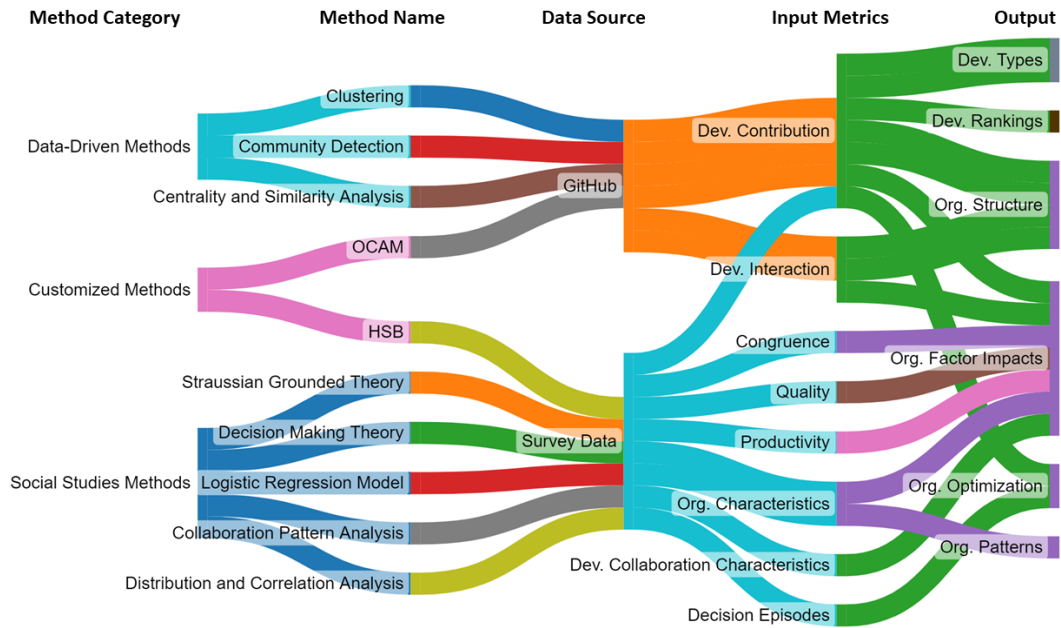


Figure 4: The Search and Selection Process

Based on the aforementioned results, we suggest researchers consider organizational metrics also in data-driven collaboration measurement models. In particular, several organizational metrics could be derived from issue trackers and from source control management tools. Examples of these metrics could be the organizational structure of the teams, or the development teams collaborating among them via pull-requests, and issues.

It is important to note that only a limited amount of models are directly applicable to the industry (e.g. in the CI/CD pipeline). To be more exact, models must be using quantitative metrics with easily accessible data sources (e.g. issue trackers, and version control systems).

4.1. Future Research Directions

As a result of the above discussion of the RQs, we propose the following directions for future research in this area:

- Focus on the validation of existing models and metrics. This could increase the validity of the models and thus their dissemination in the industry. While a limited amount of models already exist, according to the results of our work, they have not been strongly validated and, as a consequence, application has been limited.
- Center the models around quality factors that are of real interest to stakeholders, and in particular try to identify models using inputs from APIs. Several models identified in this work adopt interviews or surveys to assess the metrics. However, these collection methods require a lot of effort to be conducted and are not completely automated.

- Develop tools that support the research directions listed above (i.e., tools able to apply the methods and calculate the collaboration metrics automatically). Most of the tools developed in the primary studies are prototypes and a majority are not available or maintained anymore.

4.2. Threats to Validity

Considering that the results may be subject to validity threats, we follow the three categories of threats to validity proposed by Ampatzoglou et al. [19], which contain Study Selection Validity, Data Validity, and Research Validity. Different from the guideline proposed by Wohlin et al. [20], this categorization is more suitable for secondary studies in the field of software engineering.

Study Selection Validity. In the initial search process, we diversified the keywords identified in the research questions using synonyms. The terms adopted in our study are sufficiently stable to be used as search strings. To ensure we retrieved all papers on the selected topic, we used various bibliographic sources. Although this would have covered the vast majority of publications, there could have been potential problems or limitations which may have caused us to miss some relevant research, so we used a snowballing approach as a supplement. This criteria is consistent with the aims and RQs of this paper, and follow the guidelines recommended by Petersen et al. [21].

Data Validity. The data extraction process was done by two or more researchers independently, and if discrepancies arose, a third author was involved in the discussion to eliminate any disagreements. In addition, we checked the quality of each selected paper carefully. Regarding the data analysis process, we summarised the extracted results, based on the defined categories, and presented them in the form of graphs and tables.

Research Validity. Before the start of this study, several discussions were organized to determine the research method. The decision to use the Systematic Mapping Study was agreed upon by all authors, and this would reduce the threat of research method bias. After the research method was chosen, all the authors also worked together to define the research questions through multiple iterations of discussions. The papers selected for the review are listed in the appendix of this study and can be easily replicated by scholars.

5. Conclusion

In this work, we summarized and compared the existing metrics and models for analyzing the Developer Collaboration Analysis Microservice-Based Systems.

We surveyed 888 scientific papers, identifying 10 community measurement models adopted in 14 different studies in relevance to microservices.

Results show that, despite the practitioner community asking for independence amongst developers and ways to evaluate such independence, only a limited amount of measurement models have been proposed at this time. Even the available models are not yet validated or widely adopted in industry, calling for the need for thorough industrial validations.

Future works could aim for developing more extensive measurement models, as well as validate the existing ones (e.g. validating the relationship between the logical coupling [22]

and the structural coupling [23], the organizational structure, and the collaborations between developers [24])

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1854049 and Grant No. 2245287, and a grant from the Academy of Finland (grant n. 349488 - MuFAAno).

References

- [1] solo.io, 2022 Service Mesh Adoption Survey, <https://lp.solo.io/service-mesh-adoption-survey>, 2022.
- [2] D. Taibi, K. Systä, A decomposition and metric-based evaluation framework for microservices, in: *Cloud Computing and Services Science*, Springer International Publishing, Cham, 2020, pp. 133–149.
- [3] D. Taibi, K. Systä, From monolithic systems to microservices: A decomposition framework based on process mining, in: *Proceedings of the 9th International Conference on Cloud Computing and Services Science - Volume 1: CLOSER, INSTICC, SciTePress*, 2019, pp. 153–164. doi:10.5220/0007755901530164.
- [4] J. Sorgalla, F. Rademacher, S. Sachweh, A. Zündorf, On collaborative model-driven development of microservices, in: M. Mazzara, I. Ober, G. Salaün (Eds.), *Software Technologies: Applications and Foundations*, Springer International Publishing, Cham, 2018, pp. 596–603.
- [5] V. Lenarduzzi, O. Sievi-Korte, On the negative impact of team independence in microservices software development, in: *Proceedings of the 19th International Conference on Agile Software Development: Companion, XP '18*, 2018. doi:10.1145/3234152.3234191.
- [6] D. Taibi, V. Lenarduzzi, C. Pahl, Architectural patterns for microservices: A systematic mapping study, in: *Proceedings of the 8th International Conference on Cloud Computing and Services Science - Volume 1: CLOSER, INSTICC, SciTePress*, 2018, pp. 221–232. doi:10.5220/0006798302210232.
- [7] D. Taibi, V. Lenarduzzi, C. Pahl, *Microservices Anti-patterns: A Taxonomy*, Springer International Publishing, Cham, 2020, pp. 111–128. doi:10.1007/978-3-030-31646-4_5.
- [8] J. Soldani, D. A. Tamburri, W.-J. Van Den Heuvel, The pains and gains of microservices: A systematic grey literature review, *Journal of Systems and Software* 146 (2018) 215–232. doi:<https://doi.org/10.1016/j.jss.2018.09.082>.
- [9] J. Bogner, S. Wagner, A. Zimmermann, Automatically measuring the maintainability of service- and microservice-based systems: A literature review, in: *Proceedings of the 27th International Workshop on Software Measurement and 12th International Conference on Software Process and Product Measurement, IWSM Mensura '17*, 2017, p. 107–115. doi:10.1145/3143434.3143443.
- [10] A. Bakhtin, X. Li, J. Soldani, A. Brogi, C. Tomas, D. Taibi, Tools reconstructing microservice architecture: A systematic mapping study, in: *Agility with Microservices Programming, co-located with ECSA 2023*, 2023.

- [11] M. E. Gortney, P. E. Harris, T. Cerny, A. A. Maruf, M. Bures, D. Taibi, P. Tisnovsky, Visualizing microservice architecture in the dynamic perspective: A systematic mapping study, *IEEE Access* 10 (2022) 119999–120012. doi:10.1109/ACCESS.2022.3221130.
- [12] T. Cerny, A. S. Abdelfattah, V. Bushong, A. Al Maruf, D. Taibi, Microservice architecture reconstruction and visualization techniques: A review, in: *2022 IEEE International Conference on Service-Oriented System Engineering (SOSE)*, 2022, pp. 39–48. doi:10.1109/SOSE55356.2022.00011.
- [13] L. Lelovic, M. Mathews, A. Elsayed, T. Cerny, K. Frajtak, P. Tisnovsky, D. Taibi, Architectural languages in the microservice era: A systematic mapping study, in: *Proceedings of the Conference on Research in Adaptive and Convergent Systems, RACS '22*, 2022, p. 39–46. doi:10.1145/3538641.3561486.
- [14] V. R. Basili, G. Caldiera, H. D. Rombach, *The Goal Question Metric Approach*, Wiley, 1994.
- [15] B. Kitchenham, S. Charters, *Guidelines for performing Systematic Literature Reviews in Software Engineering*, 2007.
- [16] B. Kitchenham, P. Brereton, A systematic review of systematic review process research in software engineering, *Information & Software Technology* 55 (2013) 2049–2075.
- [17] C. Wohlin, Guidelines for snowballing in systematic literature studies and a replication in software engineering, in: *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, 2014, pp. 1–10.
- [18] H. Annabi, K. Crowston, R. Heckman, *Depicting what really matters: Using episodes to study latent phenomenon* (2008).
- [19] A. Ampatzoglou, S. Bibi, P. Avgeriou, M. Verbeek, A. Chatzigeorgiou, Identifying, categorizing and mitigating threats to validity in software engineering secondary studies, *Information and Software Technology* 106 (2019) 201–230.
- [20] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, *Experimentation in Software Engineering*, Springer, 2012.
- [21] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, Systematic mapping studies in software engineering, in: *12th International Conference on Evaluation and Assessment in Software Engineering (EASE) 12*, 2008, pp. 1–10.
- [22] D. Amoroso D’Aragona, L. Pascarella, A. Janes, V. Lenarduzzi, D. Taibi, Microservice logical coupling: A preliminary validation, in: *2023 IEEE 20th International Conference on Software Architecture Companion (ICSA-C)*, 2023, pp. 81–85. doi:10.1109/ICSA-C57050.2023.00028.
- [23] S. Panichella, M. Rahman, D. Taibi, Structural coupling for microservices, in: *Proceedings of the 11th International Conference on Cloud Computing and Services Science - Volume 1: CLOSER, INSTICC, SciTePress*, 2021, pp. 280–287. doi:10.5220/0010481902800287.
- [24] D. Amoroso d’Aragona, X. Li, T. Cerny, A. Janes, V. Lenarduzzi, D. Taibi, One microservice per developer: Is this the trend in oss?, in: *European Conference On Service-Oriented And Cloud Computing(ESOCC)*, 2023.

Appendix A: The Selected Papers (SPs)

- [SP1] Gustafsson, T., & Saltan, A. (2019). Managing Open-source Microservices Projects. Joint Proceedings of the Inforte Summer School on Software Maintenance and Evolution Tampere, Finland, September 2-4, 2019.
- [SP2] Ashraf, U., Mayr-Dorn, C., Mashkoo, A., Egyed, A., & Panichella, S. (2021, May). Do communities in developer interaction networks align with subsystem developer teams? An empirical study of open source systems. In 2021 IEEE/ACM Joint 15th International Conference on Software and System Processes (ICSSP) and 16th ACM/IEEE International Conference on Global Software Engineering (ICGSE) (pp. 61-71). IEEE.
- [SP3] Jermakovics, A., Sillitti, A., & Succi, G. (2011, May). Mining and visualizing developer networks from version control systems. In Proceedings of the 4th International Workshop on Cooperative and Human Aspects of Software Engineering (pp. 24-31).
- [SP4] Zabardast, E., Gonzalez-Huerta, J., & Tanveer, B. (2022, March). Ownership vs Contribution: Investigating the Alignment Between Ownership and Contribution. In 2022 IEEE 19th International Conference on Software Architecture Companion (ICSA-C) (pp. 30-34). IEEE.
- [SP5] Bieberstein, N., Bose, S., Walker, L., & Lynch, A. (2005). Impact of service-oriented architecture on enterprise systems, organizational structures, and individuals. *IBM systems journal*, 44(4), 691-708.
- [SP6] Zhou, X., Huang, H., Zhang, H., Huang, X., Shao, D., & Zhong, C. (2022, May). A cross-company ethnographic study on software teams for DevOps and microservices: organization, benefits, and issues. In Proceedings of the 44th International Conference on Software Engineering: Software Engineering in Practice (pp. 1-10).
- [SP7] Eseryel, U. Y., Wei, K., & Crowston, K. (2020). Decision-making processes in community-based free/libre open source software-development teams with internal governance: An extension to decision-making theory. *Communications of the Association for Information Systems*, 46(1), 20.
- [SP8] Cataldo, M., & Herbsleb, J. D. (2012). Coordination breakdowns and their impact on development productivity and software failures. *IEEE Transactions on Software Engineering*, 39(3), 343-360.
- [SP9] Bosnić, I., & Čavrak, I. (2019, May). Project work division in agile distributed student teams-who develops what?. In 2019 ACM/IEEE 14th International Conference on Global Software Engineering (ICGSE) (pp. 162-171). IEEE.
- [SP10] Tamburri, D. A., Kazman, R., & Fahimi, H. (2022). On the Relationship Between Organizational Structure Patterns and Architecture in Agile Teams. *IEEE Transactions on Software Engineering*, 49(1), 325-347.