Knowledge Engineering and Data Science (KEDS)
Vol 4, No 1, July 2021, pp. 29–37

pISSN 2597-4602
eISSN 2597-4637

Do Missing Link Community Smell Affect Developers Productivity: An Empirical Study

Toukir Ahammed [1,*], Sumon Ahmed [2], Mohammed Shafiul Alam Khan [3]

Institute of Information Technology, University of Dhaka
Suhrawardi Udyan Rd, Dhaka, 1000, Bangladesh
*[1] bsse0806@iit.du.ac.bd *; [2] sumon@du.ac.bd; [3] shafiul@du.ac.bd*
** corresponding author*

ARTICLE INFO

Article history:
Received 05 June 2021
Revised 24 June 2021
Accepted 20 July 2021
Published online 17 August 2021

Keywords:
Community Smell
Empirical Study
Missing Link Smell
Productivity

ABSTRACT

Missing link smell occurs when developers contribute to the same source code without communicating with each other. Existing studies have analyzed the relationship of missing link smells with code smell and developer contribution. However, the productivity of developers involved in missing link smell has not been explored yet. This study investigates how productivity differs between smelly and non-smelly developers. For this purpose, the productivity of smelly and non-smelly developers of seven open-source projects are analyzed. The result shows that the developers not involved in missing link smell have more productivity than the developers involved in smells. The observed difference is also found statistically significant.

I. Introduction

Community smells can be referred to as organizational and social anti-patterns in a development community, leading to unforeseen project costs [1]. Although community smells may not be an immediate obstacle for software development, these can affect software maintenance negatively in the long run [2]. The missing link is one of the most common community smells that occurred in the software development community. This smell occurs when developers contribute to the same source code but do not communicate with each other [3].

The productivity of developers is one of the essential factors of software development since it is connected to the cost of the software project. The personnel-related factors are among the ones found to affect productivity most in the literature [4]. Missing link community smell can create the knowledge gap among developers in the development community due to a lack of communication [5]. As a software product can be thought of as the combined effort of all developers, the lack of communication and cooperation can negatively affect mutual awareness and trust among developers [3]. Thus, it can affect the development of software products. This raises the need to understand how missing link smell relates to productivity to manage development productivity more effectively.

The research community has been studied community smells from different perspectives. Some studies worked with the definition [1][6], and detection [3][7] of community smells, while others studied the diffuseness [5] and variability [8] of community smells. A few studies [9][10][11] worked on the prediction of community smells. The effect of community smells on predicting the intensity of code smell [2][12], and bug [13] is also studied. The role of gender diversity on community smells is studied in [14][15]. The refactoring of community smells was investigated in [16]. However, there has been no study investigating the impact of missing link smell on developers' productivity.

In this context, the current study analyzes the productivity of developers involved in missing link smell and who is not. Seven open-source projects such as ActiveMQ and Cassandra are selected for analysis based on several criteria (e.g., availability of developer mailing list). First, missing link smells are identified in each project, finding cases where a collaboration link does not have its communication

https://doi.org/10.17977/um018v4i12021p29-37

counterpart. Then, the developers involved with each smell are identified by extracting the instance of smell. Then, the developers are categorized into smelly and non-smelly developers. Besides, the productivity of individual developers is measured by the number of changes per active day. Finally, statistical analysis is performed on the productivity of smelly and non-smelly developers.

The study results show that there is a significant difference between the productivity of smelly and non-smelly developers. The average productivity of non-smelly developers is significantly higher than smelly developers.

II. Methods

A. Missing Link Community Smell.

Missing link community smell refers to when two developers collaborate in a part of source code but do not communicate with each other [3]. This smell can be detected by finding those collaborations for which no communication is found in the defined communication channel, e.g., mailing list. The occurrence of missing link smell is described below with a sample software development community.

A sample software development community of six developers is illustrated in Figure 1. The example is taken from [17]. Developers are connected through the solid line in the network if they communicate with each other. The dashed lines connect developers to the source code on which they work. The development community can be used to generate two types of Developer Social Network (DSN), such as communication DSN and collaboration DSN. Firstly, the communication DSN can be generated from Figure 1 by considering only communication links, which are displayed in Figure 2. Then, the collaboration network can be generated by linking developers who work in the same part of the source code. Figure 3 represents the collaboration DSN for the considered development community. For example, developer *A* and developer *B* work in the same source code file (Figure 1), so they are connected in the collaboration DSN (Figure 3).

Missing link smell now can be detected by comparing the collaboration network with the communication network. It can be easily observed that one link, *E*–*F*, in the collaboration network (Figure 3) does not have the corresponding counterpart in the communication network (Figure 2). Hence, it represents an instance of a missing link smell between developer *E* and developer *F*.

In recent times, community smells are studied to incorporate the organizational and social aspects of the software development community in software engineering research. Some studies [1][6]

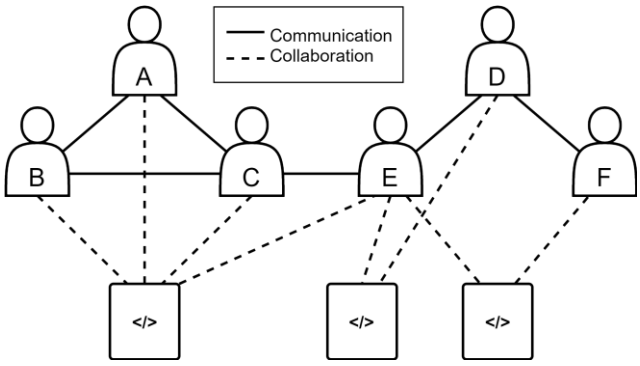

Fig. 1. Software development community

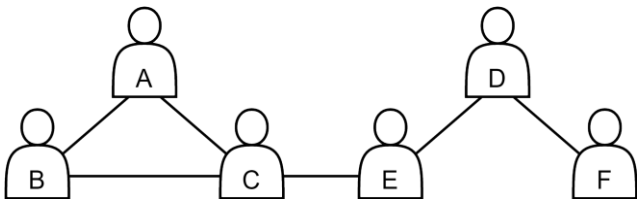

Fig. 2. Communication Network

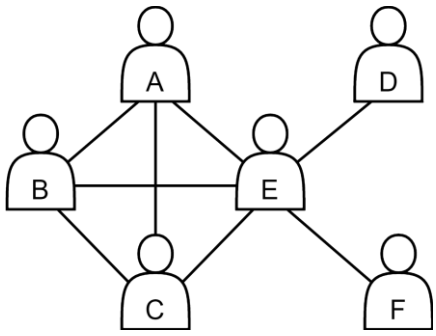

Fig. 3. Collaboration Network

focused on defining different types of community smell, while others focused on identifying [3][5] and predicting [9][10][11] these smells in open-source projects. Besides, a few studies investigated the relationship and the impact of community smells on different software artifacts such as code smell and bug [2][13][18].

The concept of community smell is first introduced in an industrial case study [1]. The authors defined nine different community smells and proposed a list of possible mitigations of these smells, such as learning community, cultural conveyor, stand-up voting, etc. Later, Magnoni [3] proposed the identification pattern of four community smells and developed a tool named *Codeface4Smells* (https://github.com/maelstromdat/CodeFace4Smells), extending an existing socio-technical network analysis tool *Codeface* (http://siemens.github.io/codeface). The enhanced tool detected both communities smells and code smells in an automated approach [7]. Besides detection, a few studies [9][10][11] tried to predict the community smells. Palomba *et al*. [9] worked on the prediction of community smells from socio-technical factors. Almarimi *et al*. [11] also built a model to predict community smells using Ensemble Classifier Chain (ECC) and Genetic Programming (GP) techniques.

Tamburri *et al*. [5] explored the diffuseness of community smells and developer's perception about the presence and effect of community smells. The authors found that the diffuseness of community smells high in open-source projects, and developers recognized community smells as an obstacle that may hinder software evolution. The authors also analyzed the relationship between community smells and different socio-technical factors, such as socio-technical congruence, turnover, and truck factor.

Catolino *et al*. [14] investigated the role of gender diversity and women's participation in community smells. The authors found that gender-diverse teams had fewer community smells than non-gender-diverse teams, and the involvement of women in teams can reduce the number of community smells. In another study, Catolino *et al*. [16] suggested some refactoring strategies to deal with community smells in practice, such as mentoring, creating communication plans, and restructuring the development community. In a recent study, Catolino *et al*. [8] investigated the impact of socio-technical factors on community smells and found that communicability is essential in most cases to prevent the increase of community smells.

Ahammed *et al*. [18] investigated how missing link community smell was related to the introduction of bugs, i.e., Fix-Inducing Changes (FIC) in the system. The authors found that the number of smelly commits (developers involved in community smells) and FIC commits are positively correlated. The authors also found that the severity of bugs was most significant that were introduced by developers involved in missing link smells. In another study [17], the same authors made an exploratory study on seven projects from Apache on the engagement of developers in missing link community smell. They found that the contribution activities of developers are positively correlated with their involvement in missing link smell.

The existing studies investigated the impact of community smell on technical artifacts such as code smell intensity [2] or bug [13] by employing a community-aware prediction model. Palomba *et al*. [2] conducted an empirical study on nine open-source projects. They also measured how community smells impact the code smell intensity by proposing a code of smell intensity prediction model. They found that community smells contribute to the intensity of code smell. Eken *et al*. [13] conducted an empirical investigation on ten open-source projects to find how community smells can predict bugs.

The authors found the impact of community smells as a contributing factor in predicting bug-prone classes. The current study aims at understanding the impact of community smell from the perspective of developers on how they perform in the software project. The study performs an empirical investigation on 1004 developers from 7 open-source projects where the projects are divided into a six-month window. The study reveals how missing link community smell affects the productivity of developers in open-source projects by measuring the productivity in terms of the number of changes per active day.

B. Proposed Framework

This study aims to understand how missing link smell affects the productivity of developers. First, missing link smells are detected from the project repository and mailing list. Then, the developers were involved with extracted missing link smells. Thus, the developers of the project can be divided into two categories: smelly and non-smelly developers. Next, the number of changes made by individual developers to the repository is computed. The productivity of individual developers is calculated as the number of changes per active day. Finally, the productivity of smelly and non-smelly developers is compared to identify the effect of missing link smell. The overview of the methodology is illustrated in Figure 4.

1) Data collection

The data is collected from 7 open-source projects for the analysis of the study. The choice of these projects is guided by the availability of source code and developer mailing list archive. The source code of the selected projects is available in *Github*, and the development mailing list archive is available in *Gmane*, a mailing list archive. The name of the projects, source code repository, number of commits, number of files, lines of code, analyzed periods, project ages, number of developers are reported in Table 1. The analyzed projects have different sizes in terms of KLOC (ranging from 483 to 1392 KLOCs) and different community sizes (from 44 to 438 developers).

2) Missing link smell detection

Missing link smells are detected in the projects according to the identification pattern introduced by [3]. First, the source code repository of a project is cloned locally from *Github* (https://github.com/), and the mailing list archive is downloaded from *Gmane* (http://gmane.io/). The projects are analyzed using a six-month window. For each window, a collaboration DSN is generated by analyzing the project's repository. All commits are analyzed. Developers who contribute to the same part of source code within that window are connected through an edge. Next, a communication DSN is constructed analyzing the mailing list of the project. All emails in the mailing list are analyzed, and developers who replied in the same email within a given window are connected. Finally, collaboration DSN and communication DSN are compared to find missing link smell. For each edge in the collaboration network, the corresponding communication part is searched in the communication DSN. Any edge that is present in collaboration DSN but absent in communication DSN is identified as missing link smell.

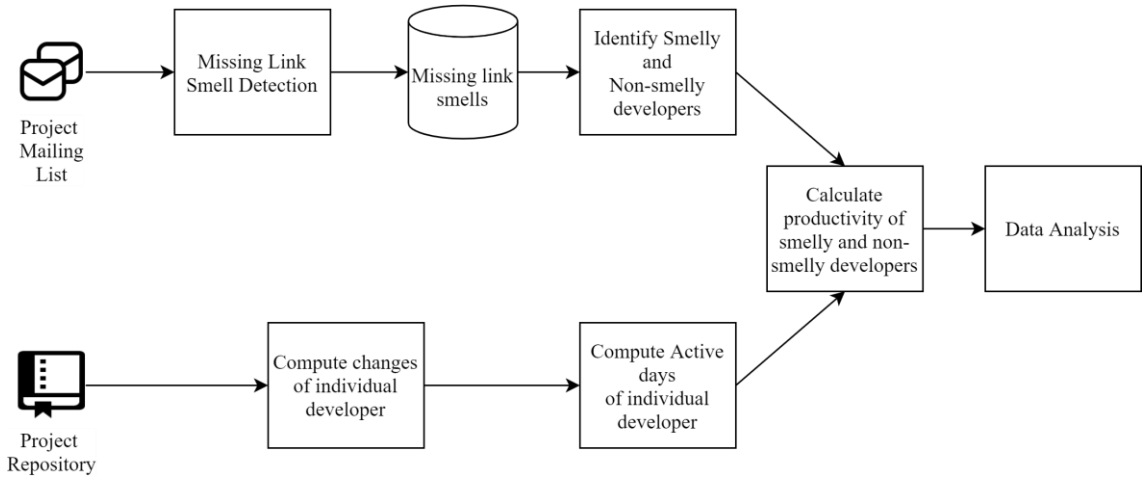

Fig. 4. Proposed framework

Table 1. List of analyzed projects

#	Project	Repository	Commits	Files	KLOC	Analysis Period	Authors	Age (year)
1	ActiveMQ	github.com/apache/activemq	10771	5454	970	Apr 2006 to Jan 2021	143	15
2	Cassandra	github.com/apache/cassandra	25896	3989	989	Oct 2009 to Sep 2020	438	11
3	Cayenne	github.com/apache/cayenne	6644	5093	539	Nov 2007 to Aug 2020	62	13
4	CXF	github.com/apache/cxf	16080	11701	1392	Nov 2010 to Sep 2020	203	10
5	Jackrabbit	github.com/apache/jackrabbit	8848	3610	660	Dec 2005 to Aep 2020	50	15
6	Mahout	github.com/apache/mahout	4480	2095	483	Oct 2008 to Aug 2020	64	12
7	Pig	github.com/apache/pig	3696	2458	591	Oct 2010 to Aug 2020	44	10
	Average		10916	4914	803		143	12

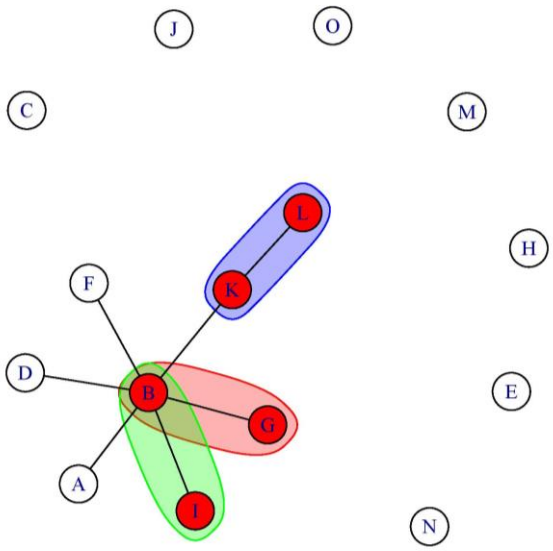

Fig. 5. Instances of Missing Link Smells in a window of Mahout Project

The steps mentioned above are performed on selected projects using *Codeface4Smells* tool. The tool preprocesses the provided artifacts, i.e., source code repository and mailing list, and generates developers' collaboration and communication network [3]. The generated networks are then used to detect the occurrence of missing link smells. The tool returns the list of missing link smells along with the corresponding developers involved with these smells for each evaluated project. The developers involved in at least one missing link smell are identified as smelly developers, and the rest are considered non-smelly developers.

Figure 5 illustrates the collaboration network and the instances of missing link smell for a six-month window of *Mahout* Project. There are 15 developers in the collaboration network for this specific window. The original name of the developers is not disclosed due to privacy reasons. The instances of missing links are marked in the network, and the developers involved with missing link smells are marked with red. There are three instances of missing link smell, i.e., *B-I*, *B-G*, *K-L*. There are five developers involved with these smells, i.e., developer *B*, *I*, *G*, *K*, *L*. These five developers are considered smelly developers.

3) Measuring productivity

The productivity of an individual can be measured as the amount of output generated per unit time [19]. The most straightforward approach to measure the contribution of a developer is to count the number of commits. However, assessing the contribution of developers using the number of commits is not a viable measurement because all commits are not equal in size. Therefore, the size of commits should be taken into account while measuring the developer's contribution. The total of modified lines in a commit is used to measure the size of that commit. The previous study also used a similar approach to measure the developer's contribution [20].

The contribution of a developer is extracted from the project repository. First, all the commits of an individual developer and all the files modified in these commits are identified. Then the number of changes, i.e., the sum of added and deleted lines, in the modified files are calculated. Then, the total number of changes is computed as the sum of all changes of a developer. Next, the number of active days of the individual developer is measured by analyzing the commit history of that developer. The number of active days is the count of days the developer made at least one commit in the repository. Then the productivity is calculated as the number of changes per active day by a developer. Equation (1) shows how productivity is measured.

$$Productivity = NumberOfTotalChanges / ActiveDays \quad (1)$$

4) Data analysis

This study aims at understanding whether smelly developers exhibit different productivity compared to non-smelly developers. The following null hypothesis is formulated to investigate the impact of missing link smell on developers productivity:

H_0: The productivity of smelly and non-smelly developers is not significantly different.

To attempt rejecting H_0, Wilcoxon Rank Sum Test, a non-parametric statistical test, is used. This test can determine whether the difference of two ordinal or interval non-parametric distributions is significantly different. The test statistic (W) indicates a significant difference between two sample sets if the ranks of the two sets significantly differ. The test is used to assess whether the productivity of developers differs between smelly and non-smelly developer groups. The test will also reveal whether the observed difference between the productivity of smelly developers and non-smelly developers is statistically significant. The result is considered significant if the p-value is less than 0.01.

III. Results and Discussions

This section presents and discusses the results obtained through the experimentation on the selected projects. The experimentation is performed according to the methodology stated above. The resulting dataset consists of 1004 developers from seven different projects. The number of smelly and non-smelly developers of all evaluated projects is reported in Table 2. The total number of smelly developers is 468, and the number of non-smelly developers is 536 in the evaluated projects. Figure 6 illustrates the project-wise ratio of smelly and non-smelly developers.

The productivity of both smelly and non-smelly developers is measured; the number of changes per active day. Thus, the dataset contains two developer groups, i.e., smelly and non-smelly, with their

Table 2. Number of Smelly and Non-Smelly Developers

#	Project	#Committers	#Non-smelly	#Smelly
1	ActiveMQ	143	74	69
2	Cassandra	438	233	205
3	Cayenne	62	25	37
4	CXF	203	116	87
5	Jackrabbit	50	26	24
6	Mahout	64	40	24
7	Pig	44	22	22
	Average	143	536	468

corresponding productivity value. Then the Wilcoxon Rank Sum Test is performed to assess the null hypothesis, H_0, which states the productivity does not differ between these two groups. The p-value obtained from the test is used to accept or reject the null hypothesis. The mean productivity of these two groups is also calculated.

The productivity of smelly and non-smelly developers is reported in Table 3. The mean productivity of smelly developers is 333.90, whereas the mean productivity of non-smelly developers is 445.84. The observed difference is identified significant from Wilcoxon Rank Sum Test (W = 72374, p-value < 0.01). The p-value indicates that the null hypothesis H_0 can be rejected. Thus, the result implies that the productivity of smelly developers and non-smelly developers is significantly different. The productivity (mean) of non-smelly developers is significantly higher than smelly developers.

Table 3. Mean Productivity of Smelly and Non-Smelly Developers

Developer	Productivity (Mean)	p-value	Decision
Smelly	333.90	< 0.01	Effect Exist
Non-smelly	445.84		

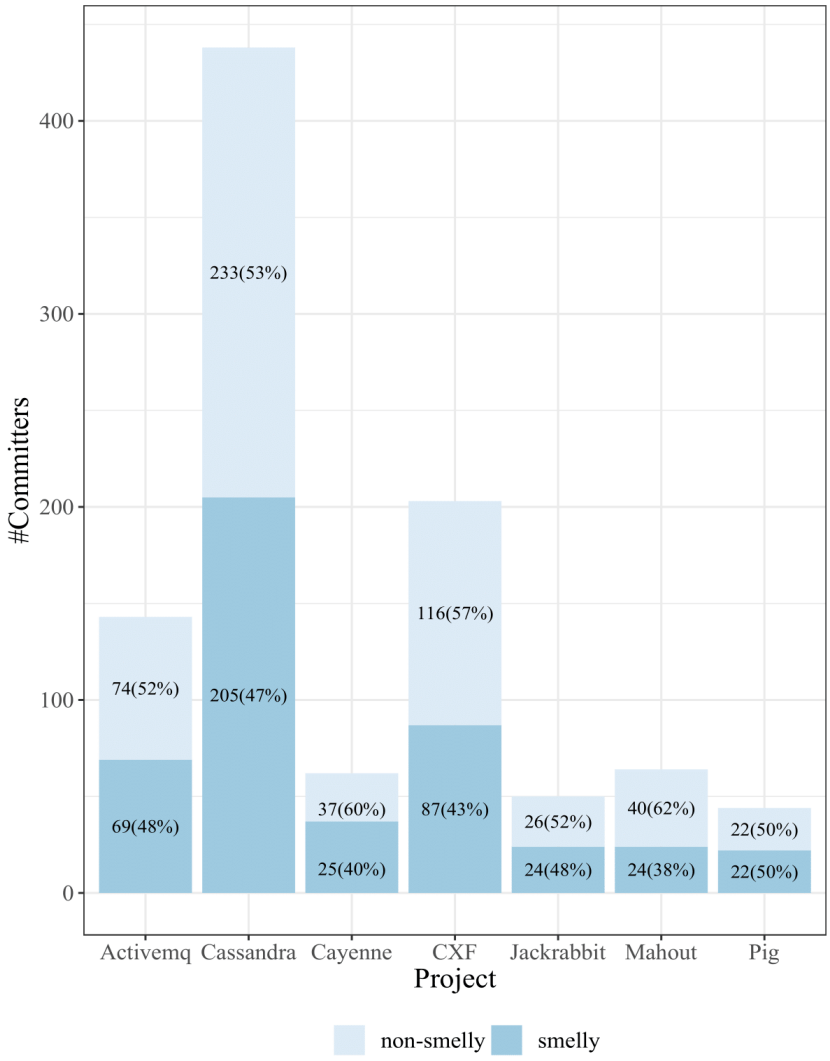

Fig. 6. Number of Smelly and Non-Smelly Developers

The result suggests that the developers involved in missing link smell show lower productivity in terms of the number of changes per active day than the developers who are not involved in missing link smell. These results indicate that missing link smell affects the productivity of developers negatively. The lower productivity of developers can increase the cost of the software project. Hence, missing links should be monitored carefully, and steps are taken to mitigate these smells if necessary.

IV. Conclusion

This study investigates the effect of missing link smell on developers' productivity. The productivity of 1004 developers from seven open-source projects is analyzed. Missing link smells are identified in these projects, and the developers are categorized into two groups, i.e., smelly and non-smelly. Productivity is measured as the number of changes performed by a developer per active day.

The Wilcoxon Rank Sum Test result shows that the productivity differs significantly between smelly and non-smelly developers. The developers who are not involved in any missing link smell show higher productivity than the developers involved in smell. The result suggests that missing link smells should be taken care of to manage development productivity effectively. Missing link smell should be monitored, and necessary steps should be taken to mitigate this smell to maintain productivity and software cost.

The missing link smells detected by *Codeface4Smells* are directly included in the study without further verification. Moreover, this tool uses a mailing list to generate the communication network as the source of communication data. The result can be different if other communication channels exist, such as Skype and Slack. However, according to contribution guidelines of evaluated projects, a mailing list is the primary communication channel in these communities.

In the future, more open-source projects can be analyzed to generalize the result. Moreover, other types of community smell such as Organizational Silo, Radio Silence can also be considered to see their effect on productivity.

Acknowledgment

Bangladesh Research and Education Network (BdREN) provides the virtual machine facility used in this research.

Declarations

Author contribution

All authors contributed equally as the main contributor of this paper. All authors read and approved the final paper.

Funding statement

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Conflict of interest

The authors declare no known conflict of financial interest or personal relationships that could have appeared to influence the work reported in this paper.

Additional information

Reprints and permission information is available at http://journal2.um.ac.id/index.php/keds.

Publisher's Note: Department of Electrical Engineering - Unversitas Negeri Malang remains neutral with regard to jurisdictional claims and institutional affiliations.

References

[1] D. A. Tamburri, P. Kruchten, P. Lago, and H. Van Vliet, "Social debt in software engineering: insights from industry," *J. Internet Serv. Appl.*, vol. 6, no. 1, pp. 1–17, 2015.

[2] F. Palomba, D. Andrew Tamburri, F. Arcelli Fontana, R. Oliveto, A. Zaidman, and A. Serebrenik, "Beyond Technical Aspects: How Do Community Smells Influence the Intensity of Code Smells?," *IEEE Trans. Softw. Eng.*, vol. 47, no. 1, pp. 108–129, 2018, doi: 10.1109/TSE.2018.2883603.

[3] S. Magnoni, "An approach to measure community smellsin software development communities," Politecnico di Milano, Italy, 2016.

[4] A. Trendowicz and J. Münch, "Factors Influencing Software Development Productivity-State-of-the-Art and Industrial Experiences," *Advances in Computers*, vol. 77. Elsevier, pp. 185–241, 2009, doi: 10.1016/S0065-2458(09)01206-6.

[5] D. A. Tamburri, F. Palomba, and R. Kazman, "Exploring Community Smells in Open-Source: An Automated Approach," *IEEE Trans. Softw. Eng.*, vol. 47, no. 3, pp. 630–652, 2021, doi: 10.1109/TSE.2019.2901490.

[6] D. A. Tamburri, "Software Architecture Social Debt: Managing the Incommunicability Factor," *IEEE Trans. Comput. Soc. Syst.*, vol. 6, no. 1, pp. 20–37, 2019, doi: 10.1109/TCSS.2018.2886433.

[7] F. Giarola, "Detecting code and community smells in open-source: an automated approach," Politecnico di Milano, Italy, 2018.

[8] G. Catolino, F. Palomba, D. A. Tamburri, and A. Serebrenik, "Understanding Community Smells Variability: A Statistical Approach," in *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Society (ICSE-SEIS)*, 2021, pp. 77–86, doi: 10.1109/ICSE-SEIS52602.2021.00017.

[9] F. Palomba and D. A. Tamburri, "Predicting the emergence of community smells using socio-technical metrics: A machine-learning approach," *J. Syst. Softw.*, vol. 171, p. 110847, 2021, doi: 10.1016/j.jss.2020.110847.

[10] N. Almarimi, A. Ouni, and M. W. Mkaouer, "Learning to detect community smells in open source software projects," *Knowledge-Based Syst.*, vol. 204, p. 106201, 2020, doi: 10.1016/j.knosys.2020.106201.

[11] N. Almarimi, A. Ouni, M. Chouchen, I. Saidani, and M. W. Mkaouer, "On the detection of community smells using genetic programming-based ensemble classifier chain," in *Proceedings - 2020 ACM/IEEE 15th International Conference on Global Software Engineering, ICGSE 2020*, 2020, pp. 43–54, doi: 10.1145/3372787.3390439.

[12] F. Palomba, D. A. Tamburri, A. Serebrenik, A. Zaidman, F. A. Fontana, and R. Oliveto, "How do community smells influence code smells?," in *Proceedings - International Conference on Software Engineering*, 2018, pp. 240–241, doi: 10.1145/3183440.3194950.

[13] B. Eken, F. Palma, B. Ayşe, and T. Ayşe, "An empirical study on the effect of community smells on bug prediction," *Softw. Qual. J.*, vol. 29, no. 1, pp. 159–194, 2021.

[14] G. Catolino, F. Palomba, D. A. Tamburri, A. Serebrenik, and F. Ferrucci, "Gender diversity and women in software teams: How do they affect community smells?," in *Proceedings - 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Society, ICSE-SEIS 2019*, 2019, pp. 11–20, doi: 10.1109/ICSE-SEIS.2019.00010.

[15] G. Catolino, F. Palomba, D. A. Tamburri, A. Serebrenik, and F. Ferrucci, "Gender Diversity and Community Smells: Insights from the Trenches," *IEEE Softw.*, vol. 37, no. 1, pp. 10–16, 2020, doi: 10.1109/MS.2019.2944594.

[16] G. Catolino, F. Palomba, D. A. Tamburri, A. Serebrenik, and F. Ferrucci, "Refactoring Community Smells in the Wild: The Practitioner's Field Manual," in *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering in Society*, 2020, pp. 25–34.

[17] T. Ahammed, M. Asad, and K. Sakib, "Understanding the Involvement of Developers in Missing Link CommunitySmell: An exploratory Study on Apache Projects," in *Proceedings of the 8th International Workshop on Quantitative Approachesto Software Quality co-located with APSEC 2020, Singapore (virtual)*, 2020, pp. 64–70.

[18] T. Ahammed., M. Asad., and K. Sakib., "Understanding the Relationship between Missing Link Community Smell and Fix-inducing Changes," in *Proceedings of the 16th International Conference on Evaluation of Novel Approaches to Software Engineering - ENASE*, 2021, pp. 469–475, doi: 10.5220/0010500604690475.

[19] S. Wagner and F. Deissenboeck, "Defining productivity in software engineering," in *Rethinking Productivity in Software Engineering*, Springer, 2019, pp. 29–38.

[20] G. Gousios, E. Kalliamvakou, and D. Spinellis, "Measuring developer contribution from software repository data," in *Proceedings - International Conference on Software Engineering*, 2008, pp. 129–132, doi: 10.1145/1370750.1370781.