Proceedings of the 7th World Congress on Electrical Engineering and Computer Systems and Sciences (EECSS'21)
~~Prague, Czech Republic~~ Virtual Conference - July, 2021
Paper No. CIST 111
DOI: 10.11159/cist21.111

Analysing the Sentiments in a Hybrid FLOSS Community based on Commits

Luigi Benedicenti[1], Tommi Mikkonen[2], Jurka Rahikkala[3]
[1]University of New Brunswick
Fredericton, New Brunswick, Canada
Luigi.Benedicenti@unb.ca
[2]University of Helsinki
Helsinki, Finland
tommi.mikkonen@helsinki.fi
[3]Vaadin
Turku, Finland
jurka.rahikkala@vaadin.com

Abstract **-** Open source repositories, by their nature, are open for study. This has inspired researchers to study them using various tools, leading to an established body of knowledge regarding how developers execute their work, communicate, and coordinate the effort. One flavor of open source community, which has received less attention, is hybrid open source, where developers and other stakeholders with various backgrounds exist. In this paper, we perform a sentiment analysis to a hybrid open source community, based on comments given when making commits. The goal is to understand if the community members behave differently, depending on their background in the company that owns the community IPR, being an independent contributor, or representing a collaborating organization. Based on the results, we find that there indeed is a clear difference in the sentiments, with in-house developers having a tendency to be more negative than independent developers, who in contrast express positive sentiments in their commit comments.

Keywords**:** Hybrid open source · hybrid communities · sentiment analysis · open source software · OSS · open source communities · free/libre/open source software · FLOSS.

1. Introduction

By their nature, free/libre/open source software (FLOSS) and the repositories that store their code are open for study. This has inspired researchers to study them using various tools, and this in turn has added a lot to the body of knowledge regarding how developers execute their work, communicate, and coordinate the effort. Various metrics focusing on code (e.g. [4, 8]) as well as methods such as sentiment analysis [9] have been applied.

To a large degree, existing studies focus on understanding how the software is composed, and how developers work together. This has resulted in concepts discussing this relation such as socio-technical congruence [17], and expanding them to other interaction between individuals, organization and processes [11]. A dimension that has received less attention is the fact that participants FLOSS communities have different backgrounds, and that participating developers can also be characterized by factors other than their direct contributions. In fact, it is acknowledged that depending on the background and role of the developer [14], developer motivations and other aspects can be different.

One flavour of open source community where developers and other stakeholders with various backgrounds exist is hybrid open source [14]. Hybrid open source refers to a development and governance model that balances between open source and closed source software, seeking to gain the advantages of both models. While there is no strict definition for the term, common forms of hybrid open source include dual licencing, gated communities where everything is open once you enter the community but not necessarily to the outside, and relying on open source in development but on a single authority for licencing and intellectual property rights (IPR).

In this paper, we perform a study to analyse the polarity of developers contributions in a hybrid project and determine whether or not there is a difference between the polarity of comments originating from internal developers in emotional contagion episodes compared to that of external developers. In the study, the data set that we analyse consists of comments

that the contributors write when committing new code. The goal is to understand if the community members behave differently, depending on their background in the company that owns the community IPR, being an independent contributor, or representing a collaborating organization. To achieve this goal, we make use of a sentiment analysis classifier and appropriate statistical tools to extract and classify the information on emotional contagion episodes. Given the large number of developers contributions, the analysis is statistically significant, within the limits highlighted in the section on threats to validity.

This paper provides two novel contributions: the first is to detect emotional contagion episodes in a large hybrid development project. The second is to determine the origin of such emotional contagion episodes and analyse the difference between the proportion of positive, neutral, and negative contributions from internal developers and external developers.

The rest of the paper is structured as follows. In Section 2, we present the background of the paper. In Section 3, we introduce our case company and case data set. In Section 4, we present the analysis done for the project, and in Section 5, we discuss our key findings. In Section 6, we present some threats to validity of this work. Towards the end of the paper, in Section 7, we draw some conclusions.

2. Background and Motivation

2.1. Hybrid Open Source

There are numerous ways how open source communities and companies can inter- act. Terms such as sponsored community [18], OSS 2.0 [2], and commercial open source [13] are used to describe this interaction. Hybrid open source software is a term that is commonly used to a model where an open source community and a company jointly work on a piece of software, under a governance model that is acceptable by both.

One commonly adopted hybrid open source model is that a company establishes an open source project around its software [16]. Then, it is possible to work so that the IPR of the software remains at the hands of one single actor, who can accept contributions if they comply with IPR terms set by the company. This way, the company can release software as open source, but at the same time, keep the option to dual-licence the software with other terms, or to change the licence later on, if so desired.

An obvious challenge in a hybrid open source setup is that the developers might be treated unequally, with in-house developers' views easily gaining more weight than those of community members. At the same time, the motivations between two groups may be diverse, so they may view the situations differently, leading to different sentiments towards the project.

3. Case Study

3.1. Case Company and Software

The case company has been developing their main Open Source product and its predecessors since 2000. Their customer base is global ranging across various industries who build applications of their own on top the framework. The business model is to offer a set of services from planning and implementation to sub-contracting to complement the FLOSS product.

The software used in the case study a web development framework that is licensed under a liberal Apache 2 license [15]. It has an active community of over 100.000 members, consisting of developers themselves using the framework to produce their own products and services. Being fundamentally FLOSS, the development of the framework takes place in the open, and it is possible to contribute to the project as an independent developer.

However, while there are some actively contributing developers in the community, the development has been historically focused on in-house activities of the case company. In addition to bug reporting, a significant community contribution is typically discussion and helping other community members with their problems.

3.2. Data Set

The data set provided by the case company consists of two files. One contains in- formation on the contributors to the project identified with a randomly assigned unique identifier. The information associated with the contributor ID is a list of

pull requests, commit IDs, whether the contributor is part of the company or an external contributor, and commit comments. The second file contains a list of records, each of which contains a pull request identifier, a contributor identifier, the date and time of contribution, and a comment.

We used the information on the first file to augment the records in the second file and create a data set that contains a field on whether or not the contributor is internal or external to the organization. The processing was performed using Mathematica, a symbolic and numerical processing system [19]. The resulting data set of 11,576 records was further processed to evaluate each comment using the sentiment analysis classifier integrated in Mathematica. The result of the classifier was encoded so that +1 corresponded to a positive result, 0 corresponded to a neutral result, and -1 corresponded to a negative result.

4. Analysis

The analysis we conduct in this paper follows a protocol previously published in [1] to define and determine the occurrence of emotional contagion, which is defined as the manner in which an affect demonstrated by a contributor is transmitted to other contributors. In this research, we limit our scope to the emotional contagion episodes that can be traced by analysing the online interaction of contributors to the case company's main product repository. Given that most contributors work remotely, and all use the repository to record their contributions to the product, it is reasonable that such interactions constitute the vast majority of the interactions among contributors. Therefore, in our analysis, we will need to determine if a comment generates a trend of comments with similar sentiment.

As all contributions are timestamped, it is reasonable to consider them as a single sequence. However, time locality is only one of the locality principles that we can adopt to conduct an analysis. A more sophisticated locality principle is clustering by pull request identifier and then sorting each cluster by timestamp. This produces a series of comment sequences, where each sequence is related to a single issue (pull request). Both analyses are presented here.

The first step in the analysis of the data set, however, is more general, and consists in the creation of a chart plotting the overall trend of the cumulative sentiment ("Project Mood") in chronological sequence (Fig. 1). Overall, the project mood appears to show a large amount of small changes and also a more general trend that looks like a slow, noisy oscillation. It also appears that the last comments were mostly negative, which will be further discussed in the analysis section. In total, there are 749 positive comments, 9,679 neutral comments, and 1,148 negative comments.

As each record identifies whether the contributor is internal or external to the case company, it is possible to calculate the proportion of internal and external contributions for each type of emotional contagion (Table 2). Note that the matching process resulted in a small number of contributors of unknown origin, which are reported as well in the table. Values are rounded to the first digit after the decimal point.

The next step in the analysis is to identify every emotional contagion sequence in the data set sorted by timestamp. The number and length of sequences are shown in Table 1. There are a large number of short sequences, but the number of long sequences is small regardless of the type of contagion. The longest positive contagion episode is 13 contributions long, versus 9 for the longest neutral and negative contagion.

Further analysis of the timestamped sequence shows that there are 37 sequences of positive contagion with external originators, with an average length of 1.51 steps per sequence, and 77 sequences of positive contagion with inter- nal originators, with an average length of 1.91 steps per sequence. Moreover, the longest positive contagion sequences are originated by internal contributors. The longest externally generated positive contagion sequence is 4 commits, whereas the longest internally originated positive contagion sequence is 12 commits. All positive contagion sequences longer than 4 commits (6, 10, and 12 commits in fact) have an internal originator. There are 18 unique internal originators of positive contagion, and 7 external ones.

Moving to neutral contagion, there are 147 sequences of neutral contagion with external originators, with an average length of 11.48 steps per sequence, and 454 sequences of neutral contagion with internal originators, with an aver- age length of 15.57 steps per sequence. Moreover, the longest neutral contagion sequences are originated by internal contributors. The longest externally generated neutral contagion sequence is 70 commits, whereas the longest internally originated neutral

contagion sequence is 171 commits. All neutral contagion sequences longer than 70 commits have an internal originator. There are 24 unique internal originators of neutral contagion, and 14 external ones.

Finally, turning to negative contagion, there are 31 sequences of negative contagion with external originators, with an average length of 1.65 steps per sequence, and 202 sequences of negative contagion with internal originators, with an average length of 1.92 steps per sequence. Moreover, the longest negative contagion sequences are originated by internal contributors. The longest externally generated negative contagion sequence is 4 commits, whereas the longest internally originated negative contagion sequence is 8 commits. All negative contagion sequences longer than 4 commits have an internal originator. There are 24 unique internal originators of negative contagion, and 14 external ones.

A similar analysis of the contagion sequences clustered by pull requests reveals some additional information. There are 2,801 pull requests, which correspond to the same number of clusters. Each of these clusters may contain positive, negative, and/or neutral sequences. This leads to a shorter sequence of comments per cluster. The average length of such a sequence is 4.13 comments. For brevity, the results of the analysis are shown in Table 3.

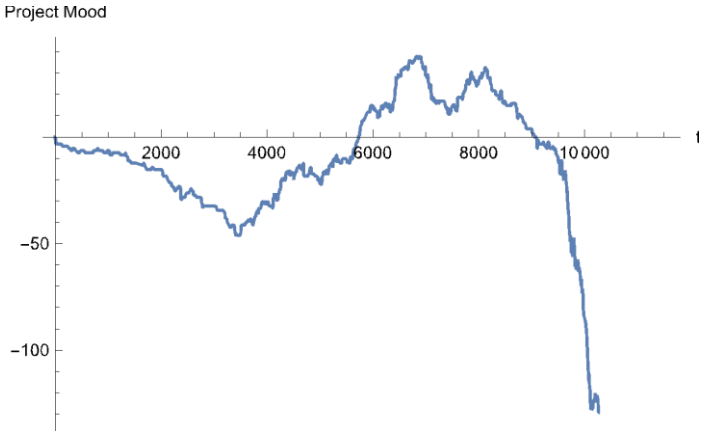

Fig. 1: Cumulative sentiment plot for all the time-sorted records in the data set.

Table 1: Emotional contagion episodes. Positive, Neutral, and Negative episodes shown respectively.

Positive		Neutral		Negative	
Length	Times Occurring	Length	Times Occurring	Length	Times Occurring
2	77	2	177	2	132
3	22	3	75	3	63
4	11	4	34	4	22
5	2	5	15	5	13
7	2	6	13	6	9
11	2	7	7	7	6
13	1	8	1	8	1
		9	2	9	1

Table 2: Proportion of contributions by contributor category and type of emotional contagion.

	Positive	Neutral	Negative
Internal	64.5%	81.3%	81.7%
External	35.5%	18.4%	13.2%
Unknown	0%	0.3%	5.1%

Table 3: Analysis of the contagion sequences clustered by pull requests.

	Positive	Neutral	Negative
Sequences with external originators	97	502	59
Average length (external)	1.28	2.98	1.34
Sequences with internal originators	198	2132	350
Average length (internal)	1.45	3.51	1.54
Longest contagion origin	internal	equal	internal
Longest external sequence	5	30	4
Longest internal sequence	13	30	8
Unique external originators	24	30	29
Unique internal originators	12	20	20
Unique unknown originators	0	3	2

5. Discussion

In the following, we propose an interpretation of the results presented in the previous section. A preliminary consideration is that in Fig. 1 the cumulative sentiment plot seems to suggest some macro trends. In particular, the overall mood of the development effort, measured as cumulative sentiment from the sentiment analysis, appears to be negative and decreasing for the first part of the plot. Then, a positive influx turns the tide and pushes the mood in the positive region, until in the final part of the plot, there is a rapid decrease toward the lowest mood value. Keeping in mind that the cumulative sentiment plot is only one way to look at the development process, and that the time interval between two consecutive points is not necessarily constant, which affects the horizontal scale of the plot, the macro trend displayed is consistent with a development effort that goes through a discovery phase, then subsequently a creative and productive phase, and finally a compressed delivery phase (often called ”crush” in certain development efforts). Further analysis would be needed to confirm this progression.

It is also important to note that both the timestamped sequence and the pull request cluster approaches are valid for the analysis: repositories like GitHub offer both perspectives to a developer, and although the timestamped sequence perspective is offered by default, it is simple to shift to the pull request cluster. Because of this ease of transition between perspectives, there is a strong probability that the two perspectives influence each other in terms of emotional contagion. This means that when looking for the cause for a certain contagion sequence, that may be justified partly by the local timestamped sequence trend and partly by the local trend of the specific pull request.

Another consideration involves the abundance of neutral contagion episodes. This can be explained by several factors. One that immediately comes to mind is the professional behaviour of contributors. It could be argued that in the ma- jority of cases, contributors will attempt to adopt a technical language with neutral emotional value in an effort to provide as unbiased a contribution as possible. Technical language is also difficult to classify by natural language pro- cessing engines, which might also be a factor in the observed number of neutral contributions. Additionally, it could be argued that neutral contagion is not a contagion at all, but rather, the absence of a polarized response; but this is less likely as we know that there are known ways to exert a calming effect on an emotional display, especially in professional environments.

When looking at the originators of contagion, in both the timestamped se- quence and the pull request clusters, internal originators appear to have greater influence on the contagion sequence than external developers, as shown by the longer sequences they generate. This result appears to not be influenced by the number of internal vs. external contributors. Conversely, the higher number of negative sequences originated by internal vs. external contributors may well depend on the fact that there is a higher proportion of internal vs. external contributors. Nonetheless, as Table 2 shows, much as the proportion of neutral and negative comments is roughly the same in the timestamped sequence (especially if we assume that the unknown contributions are external, as the internal contributors are likely to be more stable), the proportion of positive comments is skewed toward the external contributors. This may reflect a bias that external contributors have toward making positive comments in an effort to seek acceptance in the community.

6. Threats to Validity

Perhaps the most important consideration for threats to validity concerns the external validity, i.e., the extent to which the data presented here can be extrapolated and applied to other situations. Given that the number of contributors in- volved in the development effort being analysed is small compared to the overall population of FLOSS developers, it is prudent to assume that the data presented in this paper is most relevant for this development alone, and may be extended to other developments in the same case company. A more general model could be obtained by repeating the analysis for a large number of projects, similarly to other empirical software engineering research efforts.

In terms of internal validity, although the tools employed in the analysis are considered robust, it is important to note that the reliability of sentiment analysis tools is not perfect because to the authors' knowledge, there is no tool that is able to fully process natural language. The concept of detecting emotional contagion through sentiment analysis is also relatively new, and thus subject to further validation. A mitigating factor in this threat to validity is that the pull request comments are the main way in which contributors communicate among themselves, as internal and external contributors alike only share a common repository and must therefore exchange information through that repository alone. Nevertheless, the possibility exists that some interactions, especially for internal contributors, have not been recorded in the repository.

For the clustered analysis, it is also important to note that although the number of pull requests is large enough to suggest a good level of statistical significance, the contagion sequences for each cluster are relatively short, and thus their significance may be lower unless the analysis is combined, like we did here.

7. Conclusions

Open source comes in many forms. One of the forms is hybrid open source, where commercial and community interests coexist. This setup has several forms, and, depending on the business model, the different types of participants may assume different roles. This in turn can have an effect on how they regard the community as a whole as well as the software under development.

In this paper, we have studied if there is a difference in sentiment between in-house developers and individual contributors in a hybrid FLOSS project, that is controlled by a company. The study is based on comments that the developers have made when committing code to the common code base.

The analysis shows that external contributors in general seem to have more positive comments than the internal ones. This trend is visible in terms of proportion of contributions per contributor category as well as in contagion sequences clustered by pull requests. In this work, we did not seek explanation for the observation, but this is left for future work.

References

[1] Benedicenti, L.: Emotional contagion in open software collaborations. In: Ivanov, V., Kruglov, A., Masyagin, S., Sillitti, A., Succi, G. (eds.) Open Source Systems. pp. 47–54. Springer International Publishing, Cham (2020)

[2] Fitzgerald, B.: The transformation of open source software. MIS quarterly pp. 587–598 (2006)

[3] Gleeson, P.: Keeping coders happy. In: Working with Coders, pp. 191–204. Springer (2017)

[4] Gousios, G., Kalliamvakou, E., Spinellis, D.: Measuring developer contribution from software repository data. In: Proceedings of the 2008 international working conference on Mining software repositories. pp. 129–132 (2008)

[5] Graziotin, D., Wang, X., Abrahamsson, P.: Are happy developers more productive? In: International Conference on Product Focused Software Process Improvement. pp. 50–64. Springer (2013)

[6] Graziotin, D., Wang, X., Abrahamsson, P.: Happy software developers solve prob- lems better: psychological measurements in empirical software engineering. PeerJ 2, e289 (2014)

[7] Guzman, E., Azo´car, D., Li, Y.: Sentiment analysis of commit comments in github: an empirical study. In: Proceedings of the 11th Working Conference on Mining Software Repositories. pp. 352–355 (2014)

[8] Herraiz, I., Izquierdo-Cortazar, D., Rivas-Herna´ndez, F., Gonzalez-Barahona, J., Robles, G., Duenas-Dominguez, S., Garcia-Campos, C., Gato, J.F., Tovar, L.: Flossmetrics: Free/libre/open source software metrics. In: 2009 13th European Conference on Software Maintenance and Reengineering. pp. 281–284. IEEE (2009)

[9] Islam, M.R., Zibran, M.F.: Leveraging automated sentiment analysis in software engineering. In: 2017 IEEE/ACM 14th International Conference on Mining Soft- ware Repositories (MSR). pp. 203–214. IEEE (2017)

[10] Jurado, F., Rodriguez, P.: Sentiment analysis in monitoring software development processes: An exploratory case study on github's project issues. Journal of Systems and Software 104, 82–89 (2015)
[11] Outi Sievi-Korte, Fabian Fagerholm, K.S., Mikkonen, T.: Dimensions of con- sistency in gsd: Social factors, structures and interactions. In: PROFES 2020). Springer (2020)
[12] Qu, Y., Shanahan, J., Wiebe, J.: Exploring attitude and affect in text: Theories and applications. In: Technical Report SS-04-07. Published by The AAAI Press, Menlo Park, California. (2004)
[13] Riehle, D.: The commercial open source business model. In: SIGeBIZ track of the Americas Conference on Information Systems. pp. 18–30. Springer (2009)
[14] Shah, S.K.: Motivation, governance, and the viability of hybrid forms in open source software development. Management science 52(7), 1000–1014 (2006)
[15] Sinclair, A.: License profile: Apache license, version 2.0. IFOSS L. Rev. 2, 107 (2010)
[16] Sirkkala, P., Aaltonen, T., Hammouda, I.: Opening industrial software: planting an onion. In: IFIP International Conference on Open Source Systems. pp. 57–69. Springer (2009)
[17] Valetto, G., Helander, M., Ehrlich, K., Chulani, S., Wegman, M., Williams, C.: Using software repositories to investigate socio-technical congruence in develop- ment projects. In: Fourth International Workshop on Mining Software Repositories (MSR'07: ICSE Workshops 2007). pp. 25–25. IEEE (2007)
[18] West, J., O'mahony, S.: The role of participation architecture in growing sponsored open source communities. Industry and innovation 15(2), 145–168 (2008)
[19] Wolfram,S.:Mathematica website (2020), https://www.wolfram.com/mathematica/, accessed on 5 January 2020