How Important is Mentoring for New Contributors in an OSS Project?

A Mixed-Methods Study of the Rust Compiler Team

Vala Zeinali
School of Computer Science
Kent State University
Kent, OH, USA
vzeinali@kent.edu

Chris Bogart
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, USA
cbogart@andrew.cmu.edu

Daniel Klug
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, USA
dklug@cmu.edu

James Herbsleb
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA, USA
jdh@cs.cmu.edu

ABSTRACT

Our motivation comes from the problem of retention. In order for an open source software (OSS) ecosystem to survive, the ecosystem must attract and retain new contributors. Sustaining an ecosystem comes in many flavors, we attempt to identify how mentorship affects the habitants (contributors) of an OSS ecosystem and how those contributors evolve in the environment given Rust initiatives to retain them [8].

In this paper we use quantitative and qualitative practices to investigate how OSS teams matriculate and retain new contributors. We first mine nine years of Rust pull request/issue data from GitHub. We then clean and analyze first year contribution statistics to find significant factors that play a role in second year involvement. Through our quantitative approach, we find that contributors that emerged and submitted pull requests regarding issues that had a mentor to help, were more likely to stick around after their first year in the Rust compiler team ($p < .001$). In addition, we find, the amount of lines of code (LOC) a new contributor pushes is also statistically significant in retention ($p < .001$). After our quantitative approach, we randomly select new contributors and look at their comments with their mentors on the issue to rank the level of mentorship they had and how that impacted their pull request and ultimately their tenure in Rust.

We find that quantitatively and qualitatively, new contributors that submit pull requests with active mentorship are more likely to stay in the environment after their first year. We conclude that mentorship is paramount to retain new contributors in an OSS ecosystem project.

CHASE'20, MAY, 2020, Seoul, South Korea

https://doi.org/10.1145/1234567890

CCS CONCEPTS

• Empirical Software Engineering• Socio-Technical Complex Systems • Contributor Retention • Mentorship

KEYWORDS

Open source software, ecosystem, mentorship, mixed-method empirical study, data mining software repositories

ACM Reference format:

Vala Zeinali, Chris Bogart, Daniel Klug and James Herbsleb. 2020. How Important is Mentoring for New Contributors in an OSS Project? A mixed-Methods Study of the Rust Compiler Team. In *Proceedings of ACM CHASE conference workshop (CHASE'20). ACM, New York, NY, USA, 4 pages.* https://doi.org/10.1145/1234567890

1 Introduction

Contributor retention in open source software ecosystems has been an issue that affects the sustainability of the projects within the system [2, 5, 13, 16, 17]. There have been studies done to determine variables that result in contributor retention and disengagement. Miller and colleagues find that most disengagement comes from occupational reasons, i.e, got a new job that doesn't support OSS, changed role/project, left job where they contributed to OSS. There are many reasons a contributor may leave an OSS project. Another approach to studying disengagement and retention comes from a pure quantitative approach of studying the data produced from contributors, more specifically the amount of code they write for the project.

We perform similar analysis to Constantinou and Mens in *an empirical comparison of developer retention in the rubygems* and *npm software ecosystems*, however we present one more

dimension to the equation of solving retention. We present the aspect of contributing to issues that have a mentor present. We believe that a new contributor will remain in the OSS in their second if they have a mentor present their first year and/or contribute more effort in terms of lines of code (LOC). We explore the role of mentorship in the Rust compiler team ecosystem through both a quantitative and qualitative approach [6, 7, 22].

We raise the following research questions:

RQ1: *Are contributors that put in more effort in their first year more likely to put effort in their second year?*

RQ2: *Are contributors who propose more pull requests (PR) that reference issues that contain the label "E-Mentor" more likely to produce effort in their second year?*

RQ3: *Are contributors really having helpful mentorships on issues?*

2 Related Work

The study of employee retention and turnover has been widely studied [e.g. 4, 11, 12, 15]. There have been a few papers in the socio-technical systems track that break down possible reasons for contributor disengagement [2, 4, 5, 13, 16]. Miller identifies possible reasons for disengagement, such as occupational reasons like leaving a company that contributes to OSS, social reasons such as losing interest in the OSS project, and technical reasons like leaving GitHub for another repository system [3]. Likewise, Constantinou and Mens conducted *an empirical comparison of developer retention in the rubygems and npm software ecosystems* that focuses on the amount of communication a contributor makes and the frequency of commits the contributor makes in the ecosystem [5]. One of the best indicators in predicting the probability of a contributor survival is how often they commit to the project [5].

It seems that the overall pattern in the papers are, you put in more work you get out more work and more work equals higher probability of surviving in the ecosystem. It is known that organizations want their employees to be engaged. There are indicators that engaged employees are more productive [10] and there is a link between employee engagement and discretionary effort, innovation, customer loyalty, quality, and productivity [3]. Such studies have led to increased interest in what drives employee engagement and this is where 'feeling valued' is important. Robinson et al (2004) in their research on the NHS found that "The strongest driver of all (drivers) for engagement is a sense of feeling valued and involved" [19]. Instinctively, it is clear that employees want to feel valued at work or have a 'sense of value' and they report that this is what makes them feel engaged.

Humans yearn for connection with other humans. Connection comes in many forms, however, in our paper we will view human connection as mentorship within the Rust compiler team. Mentoring involves the use of an experienced individual to teach and train someone with less knowledge in a given area. Through individualized attention, the mentor/teacher transfers needed information, feedback, and encouragement to the protégé/learner [18].

3 Experimental Procedure

3.1 Dataset

Our dataset contains every pull request and issue from every contributor (number of contributors = 3008) in the Rust compiler team from July 2010 to July 2019 (number of pull requests referencing issues = 84004, *figure 2).* In addition, we have every issue ever created during that duration. We specifically look at how many lines of code a contributor adds and deletes per pull request. In addition, we look at how many pull requests reference issues that have been labeled with the "E-Mentor" label. Labels are a signal to contributors, these labels signal to a contributor what type of issue is present in the system. In the Rust ecosystem, there are currently 264 types of signaling labels. Labels can range from denoting what type of feature an issue works on, the area of expertise, the type of problem the issue references, and the type of engagement required. Rust uses the "E-Mentor" label and a way to matriculate new contributors [1]. In this paper we focus on the type of engagement required for the issue. Specifically, in this paper we will focus on the "E-Mentor" label. We match every pull request pushed that references an issue that is labeled with the "E-Mentor" label to the corresponding contributor. We keep a count of the number of pull requests pushed that reference an issue labeled with "E-Mentor" for each contributor for their first year in the environment. Overall, our table consists of the 7 variables: effort_y1, additions_y1, deletions_y1, effort_y2, additions_y2, deletions_y2, num_mentor_done_y1.

3.2 Quantitative Methodology

First, we mine nine years of GitHub repository data on the Rust compiler team [23]. Then we explore the overall effort put in by all levels of contributors over time and within their first two years in the environment (effort is noted by additions of LOC + deletions of LOC). Next, we classify all contributors in the repository as "new" or "old" *(figure 1)* contributors for every year and graph the shift in moving average contributor effort (*figure 2).* We then explore if issues labeled "E-Mentor" help new contributors stick around and put effort in their second year.

$$if(PrProposedYear != AuthorsFirstActionYear)\ THEN$$
$$Status \leftarrow OLD$$
$$ELSE$$
$$Status \leftarrow NEW$$

Figure 1 shown above.

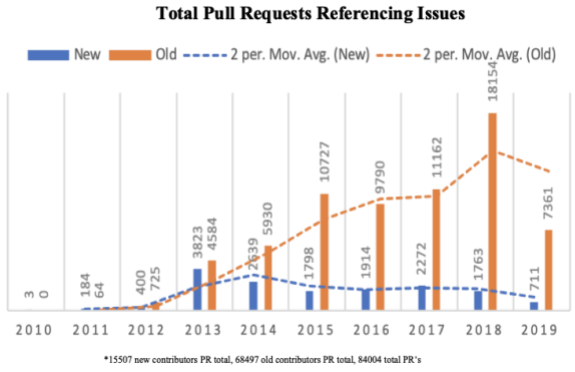


Figure 2 show above

We label the contributors as "present" or "absent" based on if they had effort in their second year (*figure 3*). Lastly, for our quantitative approach, we run a multi-linear regression model to predict second year effort based first year effort and number of pushed pull requests referencing issues that contain the label "E-Mentor" [14].

Figure 3 shown below

$$if\,(SecondYearEffort > 0)\ THEN$$
$$Status \leftarrow PRESENT$$
$$ELSE$$
$$Status \leftarrow ABSENT$$

3.3 Qualitative Methodology

After running the multi-linear regression, we randomly select 5 new contributors that have pushed PR's referencing issues labeled with "E-Mentor" from Rust and run a quick observational study on the footprints within the issue. We look at two things: within the issue thread, was there an assigned mentor to the issue (Y/N) and if so, was the mentor apathetic (did they respond to mentees questions). We then track our 5 individuals from the date of an issue and see how they evolved since the marked issue. We will omit any selections from 2019, for there is not enough data to conclude they stayed in the ecosystem.

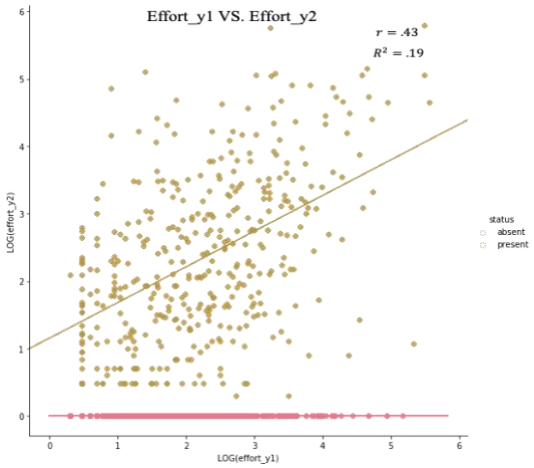
4 Experimental Results

4.1 Quantitative Results

After running our multi-linear regression model, we find that effort_y1 ($p < .001$), additions_y1 ($p < .001$), and num_mentor_done_y1 ($p < .001$) are all statistically significant when effort_y2 is our dependent variable. When running the multi-linear regression, we found that, in our multiple linear regression model above, the model predicts that effort_y2 will change by approximately 2.18 units as effort_y1 increases by 1 (units) on average. In addition, we have statistically significant evidence ($p < .001$) to believe that effort in year one increases your chance of putting effort in year two.

```
Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       481.8266   209.5231   2.300   0.0216 *
effort_y1           2.1841     0.1188  18.378  < 2e-16 ***
additions_y1       -3.5891     0.2266 -15.838  < 2e-16 ***
num_mentor_done_y1 391.7194    96.1625   4.074  4.8e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From our multi-linear regression model, we found that the model predicts that effort_y2 will change by approximately 391.7194 units as num_mentor_done_y1 increases by 1 (units) on average. Since our p-value < .001, we have reason to reject the null hypothesis and have reason to believe our alternative hypothesis that on average, pushing more pull requests that reference issues labeled with "E-Mentor" increases the average amount of effort put in year two.



In our sample 19.0% (R-squared*100) of the variability in effort_y2 can be explained by the linear association between effort_y2 and effort_y1. Overall, since our p-value < .001, we have reason to reject the null hypothesis and have reason to believe our alternative hypothesis that on average, putting in more effort in year one increases the average amount of effort put in year two.

After running our quantitative empirical study on the Rust compiler team ecosystem, we answer our first two research questions (***RQ1*** and ***RQ2***). We then move on to inspecting the mentorship relationship with contributors in the Rust compiler team through a qualitative observational study.

4.2 Qualitative Results

Individual	Mentor Present	Mentor Apathetic	Days in Eco. Before Leaving Rust
1	YES	NO	2130 1/24/14 – 11/24/19

2	YES	NO	9 7/26/16 – 8/4/16
3	YES	YES	5 6/25/14 – 6/30/14
4	YES	NO	Present (2.75 years+)
5	YES	YES	Present (2.5 years+)

Note: An individual is considered no longer in the ecosystem if they don't contribute for longer than 50 days

From our random selection of 5 individuals, we create the table shown above. We anonymize the names of the individuals to protect their identity. From our very small random selection and observational study, we found two cases of apathetic mentors. However, of those two mentors, only one mentee left immediately after the experience of an apathetic mentor. It's hard to say our result above answers our ***RQ3*** since we only selected a simple random sample of 5 individuals. However, we observe in our sample that half the contributors that experienced an apathetic mentor on their first attempt to solve an issue were likely to leave Rust within 10 days. In addition, we observe that a third of contributors that had non apathetic mentors were likely to leave within 10 days.

4 Experimental Limitations

One major threat to validity is the notion of confounding variables. It may very well be the case that there is a better indicator to solving retention issues. Second, just because mentorship worked in one ecosystem, does not guarantee that mentorship will work in another ecosystem. Mentorship may work for a certain culture but be offensive or an insult to another culture. We simply analyze one ecosystem for signs of beneficiary factors to retain new contributors in an OSS ecosystem. Next, we only analyze what is available to the eye on GitHub, there may have been private conversations on other platforms that may tip the differential. Lastly, our quick observational study only randomly selected 5 individuals within the Rust compiler team. We are aware of our small "n" and thusly accept our observation with a grain of salt. [20]

5 Conclusion / Future Work

We conclude that effort put in during your first year is indeed significant in predicting your survival in the ecosystem for your second year. Since our (p-value < .001), we have reason to reject the null hypothesis and have reason to believe our alternative hypothesis, that on average, putting in more effort in year one increases the average amount of effort put in year two. In addition, from our multi-linear regression model, we found that the model predicts that effort_y2 will change by approximately 391.7194 units as num_mentor_done_y1 increases by 1 (units) on average. Likewise to our first test, since our (p-value < .001), we have reason to reject the null hypothesis and have reason to believe our alternative hypothesis, that on average, pushing more pull requests that reference issues labeled with "E-Mentor" increases the average amount of effort put in year two and thus being retained in the Rust OSS ecosystem. Our qualitative observational study showed that there might be some sort of association between mentorship and retention of the contributor. We observed that that half the contributors that experienced an apathetic mentor on their first attempt to solve an issue were likely to leave Rust within 10 days. Since our "n" in our survey was small, we can't statistically prove there is significance between mentorship and retention in a **qualitative approach**.

However, we plan to incorporate survey data in our future work on this mentorship topic [9]. We will acquire survey data by a stratified random sample based on years ranging from 2010-2018 [24]. We will select 6 random contributors from each year (total n = 48) and then ask them the following questions: during your first year in Rust, how many prior years of OSS experience did you possess, on a scale from 1-10, do or did you feel wanted in the Rust ecosystem during your first year, did you have a mentor during your first year contributing to the Rust compiler team, and on a scale from 1-10, how would you rate your mentor? Our goal is to cross validate to really see if contributors are getting mentorship if they want it and if the mentorship systems set in OSS ecosystems are actually doing their job in keeping contributors engaged with projects.

6 Acknowledgements

This research was possible by the REU - SE program at Carnegie Mellon University. I am very grateful for my experience in the program. Specifically, I want to thank Chris Bogart, Jim Herbsleb, and Daniel Klug for their time and knowledge. Thank you to my significant others for their support as always.

6 References

[1] Bauer, T. N. Erdogan, B. "Organizational socialization: The effective onboarding of new employees" in S. Zedeck (Ed.), APA Handbook of industrial and organizational psychology, Vol. 3, pp. 51-64. Washington, DC, USA, 2011, American Psychological Association.

[2] Benjamin Gidron. Predictors of retention and turnover among service volunteer workers. Journal of Social Service Research, 8(1):1–16, 1985.

[3] Blessing White, (2008), The State of Employee Engagement 2008: Highlights for U.K. and Ireland, 3. 12.

[4] Jailton Coelho and Marco Tulio Valente. Why modern open source projects fail. In Proc. Int'l Symposium Foundations of Software Engineering (FSE),pages 186–196. ACM, 2017.

[5] Eleni Constantinou and Tom Mens. An empirical comparison of developer retention in the rubygems and npm software ecosystems. Innovations in Systems and Software Engineering, 13(2-3):101–115, 2017.

[6] Creswell, J. W. (2003). Research design: Qualitative, quantitative, and mixed methods approaches (2nd ed.). Thousand Oaks, CA: Sage.

[7] John W Creswell and Vicki L Plano Clark. Designing and conducting mixed methods research. Wiley Online Library, 2007.

[8] Fagerholm, F., Sanchez Guinea A., Münch, J., Borenstein, J. "The Role of Mentoring and Project Characteristics for Onboarding in Open Source Software Projects". Empirical Software Engineering and Measurement (ESEM), 2014.

[9] Fuller, W.A. (1984). Least Squares and Related Analyses for Complex Survey Designs. Survey Methodology 10, 97-118.

[10] Gruman J.A. and Saks A.M., (2011), Manage Employee Engagement to Manage Performance, Industrial and Organizational Psychology, Vol. 4, Iss. 2, pp. 204–207, Wiley

[11] Peter W Hom, Thomas W Lee, Jason D Shaw, and John P Hausknecht.
One hundred years of employee turnover theory and research. Journal of Applied Psychology, 102(3):530, 2017.

[12] Hyosu Kim and Dennis Kao. A meta-analysis of turnover intention predictors among us child welfare workers. Children and Youth Services Review,
47:214–223, 2014.

[13] Irma Browne Jamison. Turnover and retention among volunteers in humanservice agencies. Review of Public Personnel Administration, 23(2):114–132,
2003.

[14] Kutner MH, Nachtsheim CJ, Neter J, et al. Applied Linear Statistical Model. 5th ed. New York, NY: McGraw-Hill/Irwin; 2005:664 – 665, 1173–1183.

[15] Lynn E Miller, Gary N Powell, and Joseph Seltzer. Determinants of turnover among volunteers. Human Relations, 43(9):901–917, 1990

[16] Courtney Miller, David Widder, Christian Kastner, and Bogdan Vasilescu.
Why do People Give Up FLOSSing? A Study of Contributor Disengagement in Open Source
10.1007/978-3-030-20883-7_11, 2019.

[17] Minghui Zhou, Audris Mockus, Xiujuan Ma, Lu Zhang, and Hong Mei. Inflow and retention in OSS communities with commercial involvement: A case study of three hybrid projects. ACM Trans. Softw. Eng. Methodol. (TOSEM), 25(2):13, 2016.

[18] Newby, T. J. and Heide, A. (1992), The Value of Mentoring. Performance Improvement Quarterly, 5: 2-15. doi:10.1111/j.1937-8327.1992.tb00562.x

[19] Robinson, D., Perryman, S. and Hayday, S. (2004), The Drivers of Employee Engagement, Report 408, Institute for Employment Studies, Brighton

[20] Sandelowski M. (1995) Focus on qualitative methods: sample size in qualitative research. Research in Nursing and Health 18, the information needs of the study. The ultimate aim is to 179–183.

[21] Steinmacher, I., Wiese, I., Chaves, A.P., Gerosa, M.A., "Why do newcomers abandon open source software projects?," 6th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE), pp. 25-32, 2013.

[22] Tashakkori, A., & Teddlie, C. (1998). Mixed methodology: Combining qualitative and quantitative approaches. Thousand Oaks, CA: Sage.

[23] T. Xie, S. Thummalapenta, D. Lo and C. Liu, Data mining for Software Engineering, IEEE Computer, 2009

[24] Trost, J.E. (1986). Statistically nonrepresentative stratified sampling: A sampling technique for qualitative studies. Qualitative Sociology, 9, 54-57.