Social and Technical Evolution of Software Ecosystems

A Case Study of Rails

Eleni Constantinou
COMPLEXYS Research Institute
University of Mons, Belgium
eleni.constantinou@umons.ac.be

Tom Mens
COMPLEXYS Research Institute
University of Mons, Belgium
tom.mens@umons.ac.be

ABSTRACT

Software ecosystems evolve through an active community of developers who contribute to projects within the ecosystem. However, development teams change over time, suggesting a potential impact on the evolution of the technical parts of the ecosystem. The impact of such modifications has been studied by previous works, but only temporary changes have been investigated, while the long-term effect of permanent changes has yet to be explored. In this paper, we investigate the evolution of the ecosystem of Ruby on Rails in GitHub in terms of such temporary and permanent changes of the development team. We use three viewpoints of the Rails ecosystem evolution to discuss our preliminary findings: (1) the base project; (2) the forks; and (3) the entire ecosystem containing both base project and forks.

CCS Concepts

•**Software and its engineering → Software evolution; Open source model; Programming teams;**

Keywords

Software ecosystems; Social evolution; Technical evolution

1. INTRODUCTION

Open source software ecosystems and their evolution have been studied by the research community in order to gain a better understanding of their dynamics [9]. This paper considers Lungu's definition that defines a software ecosystem as a collection of software projects that are developed and evolve together in the same environment [6]. Examples of software ecosystems include programming archive networks, mobile app stores, distributions of the Linux operating system [7] and project ecosystems in GitHub, which are created through forking, sharing of developers and links via dependencies [2].

The environment of each software ecosystem consists of the development team and the source code artefacts, corresponding to the social and technical aspects of the ecosystem, respectively. The evolution of both the social and technical aspects of software ecosystems has been widely studied, but current studies focus on temporary changes of the ecosystem [12, 11]. On the contrary, this paper focuses on permanent changes in the ecosystem environment and measures the effect of these changes in the ecosystem evolution. For example, an analysis measuring temporary changes will consider a contributor who becomes inactive for only a quarter but contributes in future quarters, while our analysis only considers contributors who become and stay inactive throughout the ecosystem's evolution.

The goal of this paper is to gain an understanding of the long-term effect of changes in the development team to both the overall ecosystem's evolution and the file activity. We are interested in ecosystems that follow a pull-based development process [3], with most development activity going on in forks and pull requests merging these changes into the base project. We study these questions on the long-lived ecosystem of Ruby on Rails in GitHub[1].

2. EXPERIMENTAL SETUP

Ruby on Rails is a web-application framework for creating database-backed web applications according to the Model-View-Controller (MVC) pattern. We used the 2016-09-05 dump of the GHTorrent dataset [2] to obtain the historical evolution of the base project and the forks of Rails, and we queried GitHub to obtain more information about the files modified in each commit. We applied several filters to the initial dataset to ensure the validity of our results: (1) *Forks*: we only considered the forks whose pull requests were merged back to the base project in order to eliminate the effect of simple copies which do not affect the evolution of the base project; (2) *Files*: We only considered the source code files we found in commits to eliminate the effect of irrelevant files, e.g., temporary files; and (3) *Contributors*: We eliminated contributors whose average activity in terms of quarters was less than 2 to reduce potential noise from one-time or occasional contributors.

Our empirical study focuses on three research questions:
$\boldsymbol{RQ_1}$ How does the commit activity of the ecosystem (in base and forks) evolve over time?
$\boldsymbol{RQ_2}$ How do changes in the development team and in file activity change over time?
$\boldsymbol{RQ_3}$ How do changes in the development team affect the file activity of the ecosystem?

ECSAW '16, November 28-December 02, 2016, Copenhagen, Denmark
© 2016 ACM. ISBN 978-1-4503-4781-5/16/11...$15.00
DOI: http://dx.doi.org/10.1145/2993412.3003384

[1]https://github.com/rails/rails

Table 1: Dataset descriptive statistics

	Dataset			Filtered dataset		
	Base	Forks	Ecosystem	Base	Forks	Ecosystem
Count	1	1,896	1,897	1	692	693
Developers	1,827	2,154	3,121	430	681	765
Commits	43,195	25,938	69,133	40,660	22,923	63,583

Table 2: Definitions of team and file metrics

$Leavers(t)$	$\{c \mid \forall\ i \geq t, isContr(c, t-1) \wedge \neg isContr(c, i)\ \}$
$Joiners(t)$	$\{c \mid \forall\ i < t, isContr(c, t) \wedge \neg isContr(c, i)\ \}$
$Stayers(t)$	$\{c \mid\ isContr(c, t) \wedge isContr(c, t-1)\ \}$
$TeamTurnover(t)$	$\lvert Joiners(t) \rvert\ /\ \lvert \{c \mid\ isContr(c, t)\} \rvert$
$TeamAbandonment(t)$	$\lvert Leavers(t) \rvert\ /\ \lvert \{c \mid\ isContr(c, t-1)\} \rvert$
$Obsolete(t)$	$\{f \mid \forall\ i \geq t, isTouched(f, t-1) \wedge \neg isTouched(f, i)\ \}$
$New(t)$	$\{f \mid \forall\ i < t, isTouched(f, t) \wedge \neg isTouched(f, i)\ \}$
$Maintained(t)$	$\{f \mid isTouched(f, t) \wedge isTouched(f, t-1)\}$
$FileTurnover(t)$	$\lvert New(t) \rvert\ /\ \lvert \{f \mid isTouched(f, t)\} \rvert$
$FileAbandonment(t)$	$\lvert Obsolete(t) \rvert\ /\ \lvert \{f \mid isTouched(f, t-1)\} \rvert$

Table 1 presents descriptive statistics of our initial and filtered dataset in terms of the studied projects, developers and commits. The contributor elimination threshold resulted in the exclusion of contributors with five or less commits for the base and forks, and seven or less for the entire ecosystem. While this approach eliminated a large fraction of contributors, it only has a limited effect on the number of commits, confirming that the bulk of commits is done by frequent contributors.

From this point forward, our results will refer to the filtered dataset. It consists of the Rails base project and 692 forks, while 430 developers only contribute to the base project and 681 developers contribute to its forks. It should be noted that the ecosystem's number of developers equals 765, meaning that developers of the forks contribute to the base project as well. Finally, the majority of commits is observed for the base project with 40,660 commits, while the forks' commits equals 22,923.

GHTorrent reports that Rails was created in April 2008. From that date forward, we divided the dataset in 34 quarters (i.e., three-month intervals). We excluded from our analysis the first and last quarter since we do not have past and future data. For each quarter, we registered the following information: (1) *Contributors:* We record the set of active contributors, new developers who joined the development team and contributors who permanently abandoned the development team; (2) *Commits:* We measure the number of commits in each quarter to assess the overall development effort; (3) *Modified files:* We register the files that are actively developed/maintained, the new files that are developed in the system and the files that become obsolete in each quarter; (4) *Orphaned files:* We also register the orphaned files that were maintained by developers who abandoned the development team.

Table 2 presents our metrics of team and file changes. In our study we consider the permanent actions of contributors, i.e., users who left the project and never contributed again, and users who joined the project but had never contributed in a previous quarter. We only consider contributors and files that remain inactive until the end of the observed duration of the ecosystem. Therefore, these measurements take into account only permanent changes in the ecosystem with regard to the observed time interval in our dataset.

Given a contributor c and a quarter t, where $t-1$ is the previous quarter, and a predicate $isContr(c,t)$ that is true if and only if c made a source code commit in t, we formally define **Leavers(t)**, **Joiners(t)** and **Stayers(t)**. Given a file f and a predicate $isTouched(f,t)$ that is true if and only if f was touched through commits in t, we define the file modifications **Obsolete(t)**, **New(t)** and **Maintained(t)**. We define the ratios **TeamTurnover(t)** and **FileTurnover(t)** as the fraction of the team and files in a given quarter that is different with respect to the previous quarter. **TeamTurnover(t)** takes into account the new members who joined the development team in quarter t and **FileTurnover(t)** considers the files that were never touched in previous commits. Accordingly, we define the ratios **TeamAbandonment(t)** and **FileAbandonment(t)** as the fraction of the team and files in a given quarter that became inactive with respect to the previous quarter, e.g., contributors who never commit again and files that never appear in commits.

3. RESULTS

***RQ*$_1$ How does the commit activity of the ecosystem (in base and forks) evolve over time?** The evolution of Rails in terms of commits is presented in Figure 1. The main development activity takes place in the base project during the first 12 quarters. From quarter 13 onwards, forks start to have increasing commit activity, while an important decrease in the base project commits arises at quarter 18. From quarter 18 onwards, the development effort appears to be equally spread between the base project and the forks. These results reveal that the Rails project started to form a fork-based ecosystem after quarter 12, and the evolution of the base project started to heavily depend on forks.

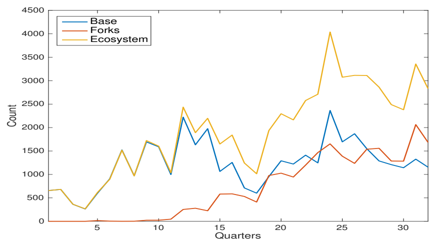
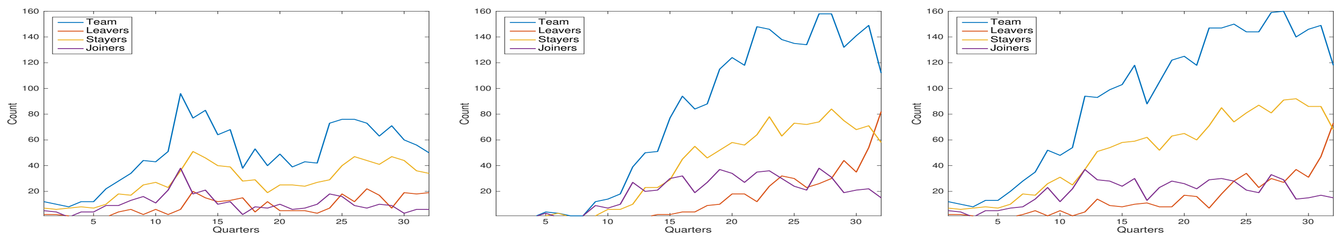


Figure 1: Quarterly evolution of #commits

***RQ*$_2$ How do changes in the development team and in file activity change over time?** For each quarter t, we measured $Stayers(t)$, $Joiners(t)$ and $Leavers(t)$ as well as the total team size. Figure 2 shows their evolution for (a) the base project, (b) the forks and (c) the entire ecosystem. The evolution of the team size shows that the base project was rapidly growing until quarter 12, and started to shrink again until quarter 17. After quarter 11 the development of forks started to increase, explaining the entire ecosystem's stability with respect to the team size. Concerning permanent Joiners and Leavers, Figure 2 shows that the increase in the size of the development team of Rails can mainly be attributed to an increase in the number of Stayers, implying that core contributors seldomly leave the development team. The number of Leavers and Joiners shows a stable behavior with a small number of contributors



(a) Base Project (b) Forks (c) Ecosystem

Figure 2: Quarterly evolution of team changes

becoming core members and few contributors permanently leaving the core development team.

Figure 3 shows the number of file changes ($Obsolete(t)$, $Maintained(t)$ and $New(t)$) per quarter t in (a) the base project, (b) the forks and (c) the ecosystem. The base project shows increased activity from the first quarters, indicating activity prior to moving Rails to GitHub, and a decrease is observed at quarter 10. An increased number of files changes in forks is observed from quarter 12 onwards. The fact that after quarter 18 the number of file changes in forks is larger than in the base project in most quarters, indicates that development effort has shifted from the base project to the forks. The ecosystem's increasing trend with respect to the number of files shows that the pull-based development model benefits Rail's evolution. We also observe an evident trend that the more files are actively developed, the more files are maintained in consecutive quarters. With a few exceptions, the number of obsolete and new files stays relatively low.

***RQ_3* How do changes in the development team affect the file activity of the ecosystem?** Our results show that during the evolution of Rails, both the development team and the file activity are subject to constant and permanent modifications. In order to investigate the effect of these changes, we measured the team and file turnover and abandonment, as defined in Section 2. Table 3 provides the average and standard deviation of the quarterly ratio of turnover and abandonment (denoted as average ± standard deviation). The average team turnover varies between 23 and 29% and the average team abandonment varies between 11 and 16% for the three ecosystem viewpoints. Our results show that 14% of the team permanently leaves the project on average and that more than 25% of the members of the development team are newcomers, indicating moderate renewal of the core development team. The team changes are accompanied by file changes and according to Table 3, on average 15% of the files are newly created in each quarter, while 10% of the ecosystem's files are completely abandoned in each quarter.

Next, we measured the number and ratio of abandoned files, i.e., files that became obsolete when their contributors abandoned the ecosystem. Figure 4 shows the orphaned files per quarter for each ecosystem viewpoint. On average, each quarter 86.94 files become obsolete in the ecosystem, corresponding to 10% of the developed files in each quarter, and the number of orphaned files equals 21.65. This means that on average approximately 25% of the obsolete files were maintained by contributors that have abandoned the ecosystem. An open question is whether these files are not subject to further development or maintenance, or their

Table 3: Average and standard deviation values of turnover and abandonment ratios

	Base	Forks	Ecosystem
$TeamTurnover$	0.23 ± 0.12	0.29 ± 0.25	0.25 ± 0.12
$TeamAbandonment$	0.16 ± 0.09	0.11 ± 0.13	0.14 ± 0.10
$FileTurnover$	0.15 ± 0.11	0.34 ± 0.34	0.15 ± 0.11
$FileAbandonment$	0.10 ± 0.08	0.13 ± 0.16	0.10 ± 0.07

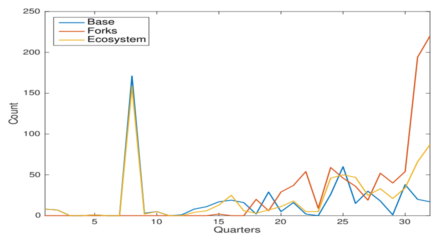


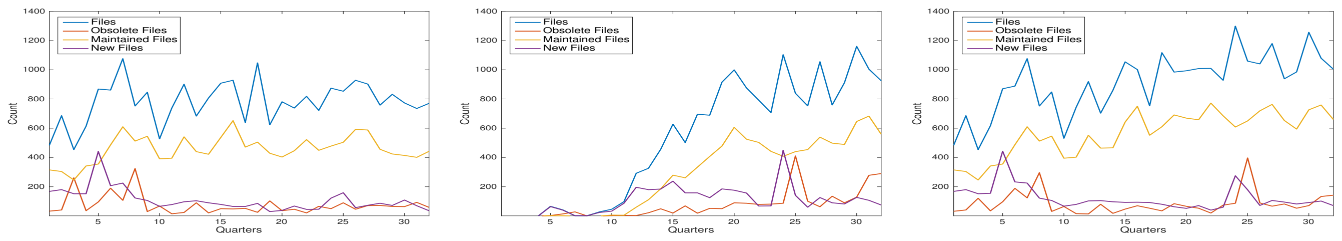
Figure 4: Orphaned files per quarter

functionality might contain unresolved bugs.

Our study on the core development team of Rails showed that the ecosystem started to intensively use the fork and push mechanisms of GitHub after quarter 12, with both the development team and files showing a roughly linearly increasing trend. Therefore, fork and push mechanisms seem to facilitate the effort of the core development team of Rails. Also, our filtering of occasional contributors showed that the core team carries out the bulk of the development effort. From a research perspective, it is important to filter evolution datasets from such contributors to draw valid conclusions [5]. From the ecosystem perspective, further research is required to identify if joiners, stayers and leavers of the core development team are engaged in other ecosystems, and to investigate which practices can eliminate the effect of occasional contributions and orphaned files in the ecosystem.

4. THREATS TO VALIDITY

While contributors tend to use multiple accounts across different repositories (e.g., version control system, bug tracker and mailing list), it is much less common to use multiple accounts for committing source code within the same GitHub repository. Thus, the effect of multiple identities is minimal in our study and it does not pose a substantial threat to validity. Since our study focuses on only one ecosystem, larger studies must be performed to understand the effect of permanent changes in the evolution of different ecosystems. Finally, our measurement of contributors' effort in terms of commits poses a threat to the validity of our work due to the use of commit squashing [5]. Ruby developers are rec-



(a) Base Project (b) Forks (c) Ecosystem

Figure 3: File modifications

ommended to squash their branch before any pull requests. To mitigate this threat, our future work will also take into account the size and frequency of the commits.

5. RELATED WORK

Software ecosystem evolution has been widely studied by the research community. The Ruby ecosystem has been studied by Kabbedijk and Jansen [4], who identified different types of contributors: networkers, lone wolfs, and one day flies. Syed et al. [10] used social network analysis combined with a survey to identify clusters of contributors, and revealed that Ruby's ecosystem consists mostly of independent developers. Vasilescu et al. [12] present a large dataset of social diversity attributes of GitHub contributors contributing to 23,493 GitHub projects. In [11], the authors studied gender and tenure diversity and how they relate to team productivity and turnover, and their findings show that increased gender and tenure diversity are associated with greater productivity. Also, they found that turnover is positively associated with tenure diversity. Unlike these studies, we focus on permanent turnover and abandonment of both users and files to investigate the long-term effects on the ecosystem evolution, in terms of base and forks. German et al. [1] studied the R ecosystem evolution and found that a healthy community and well-maintained packages are essential to the successful evolution of the ecosystem. Rigby et al. [8] quantified the extent of abandoned source files and used methods from financial risk analysis to assess the susceptibility of the project to developer turnover. They found that when tight relationships between the author and the source code exist, then it is more difficult to replace the authors by newcomers since they are less productive and more prone to making errors. However, their study investigates isolated projects, while our work focuses on software ecosystems.

6. CONCLUSIONS

This article presented a case study of the evolution of the Ruby on Rails ecosystem in GitHub. We studied the effect of permanent changes in the development team and the source code files of the ecosystem. We used the viewpoints of the base project, forks and the entire ecosystem to present our empirical findings. Our results show that the ecosystem's evolution is accompanied by modifications in both the development team and the maintained files, while the impact of contributors abandoning the ecosystem is rather limited. Future work will focus on investigating more and larger ecosystems to confirm our observations for the Rails ecosystem and generalize our results to other ecosystems.

Acknowledgments. This research was carried out in the context of ARC research project AUWB-12/17-UMONS-3.

7. REFERENCES

[1] D. M. German, B. Adams, and A. E. Hassan. The evolution of the R software ecosystem. In *European Conf. Software Maintenance and Reengineering*, pages 243–252, 2013.

[2] G. Gousios. The GHTorrent dataset and tool suite. In *Working Conf. Mining Software Repositories*, pages 233–236, 2013.

[3] G. Gousios, M.-A. Storey, and A. Bacchelli. Work practices and challenges in pull-based development: The contributor's perspective. In *Int'l Conf. Software Engineering*, pages 285–296, 2016.

[4] J. Kabbedijk and S. Jansen. Steering insight: An exploration of the Ruby software ecosystem. In *Int'l Conf. Software Business*, pages 44–55, 2011.

[5] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D. M. German, and D. Damian. An in-depth study of the promises and perils of mining GitHub. *Empirical Software Engineering*, 21(5):2035–2071, 2016.

[6] M. Lungu. Towards reverse engineering software ecosystems. In *Int'l Conf. Software Maintenance*, pages 428–431, 2008.

[7] T. Mens and P. Grosjean. The ecology of software ecosystems. *Computer*, 48(10):85–87, 2015.

[8] P. C. Rigby, Y. C. Zhu, S. M. Donadelli, and A. Mockus. Quantifying and mitigating turnover-induced knowledge loss: Case studies of Chrome and a project at Avaya. In *Int'l Conf. Software Engineering*, pages 1006–1016, 2016.

[9] A. Serebrenik and T. Mens. Challenges in software ecosystems research. In *European Conf. Software Architecture Workshops*, pages 40:1–40:6, 2015.

[10] S. Syed and S. Jansen. On clusters in open source ecosystems. In *Int'l Workshop on Software Ecosystems*, volume 987, pages 19–32, 2013.

[11] B. Vasilescu, D. Posnett, B. Ray, M. G. van den Brand, A. Serebrenik, P. Devanbu, and V. Filkov. Gender and tenure diversity in GitHub teams. In *ACM Conf. Human Factors in Computing Systems*, pages 3789–3798, 2015.

[12] B. Vasilescu, A. Serebrenik, and V. Filkov. A data set for social diversity studies of GitHub teams. In *Working Conf. Mining Software Repositories*, pages 514–517, 2015.