Exploring Pull Requests in Code Samples

Matheus Melo  
matheus.albuquerque@ufms.br  
Faculty of Computing  
UFMS, Brazil

Gabriel Menezes  
gabriel.menezes@ufms.br  
Faculty of Computing  
UFMS, Brazil

Bruno Cafeo  
cafeo@facom.ufms.br  
Faculty of Computing  
UFMS, Brazil

ABSTRACT

The interaction between organizations and their clients (actors) around a common platform can be studied as Software Ecosystem (SECO). Code samples are software projects made available by organizations to help their clients in the learning process of the features from their platforms. In this context, actors’ interactions are important to keep a healthy SECO. Pull requests can be a tool to facilitate these interactions. In this study, we explore the contributions of code sample clients through pull requests. In addition, we assess the distribution of review activity between code sample maintainers. As a result, we found that majority of pull requests are reviewed and accepted. Also, accepted pull requests take less time to be reviewed. Finally, few maintainers are responsible to review them.

KEYWORDS

code samples, pull requests, software ecosystems

1 INTRODUCTION

Nowadays, software development is commonly supported by frameworks, libraries, and APIs (here called platforms). The platforms can provide feature reuse, improve productivity, and decrease costs [14, 24, 25]. These platforms support the development of different niches of tools such as mobile apps, web apps, responsive interfaces, cross-platform systems, cloud computing, distributed systems, and others. These platforms are widely used by developers and organizations. In the Java ecosystem, for example, there are more than 450,000 platforms available in the Maven repository [6]. In the Python ecosystem are more than 350,000 platforms [5] made available via Python Package Index.

In an environment with interactions between organizations that develop the platforms and developers (here called clients) that use the platform’s features to create value, this interaction results in contributions to the evolution of the platform, then this environment can be studied as a Software Ecosystem (SECO) [4, 15, 18]. In this context, clients need to learn how to use features provided by platforms. However, there are some barriers that developers may face learning these features [26, 31, 32, 35].

To help clients overcome these barriers, some organizations create code samples. Code samples are small software projects, with educational purposes and made available by organizations to support and accelerate clients’ learning of their platform features [3, 20, 21, 29]. Code samples are often provided by worldwide organizations, such as Android [9], Spring [28], Google Maps [10], Twitter [33], Microsoft [22], to name a few.

There are some characteristics of code samples and their SECO already explored in the literature. For example, there are works that evaluate source code characteristics, current and over time, frequency of changes, and the use of code samples by developers [20]. Also, other research explores the degree of experience of code sample clients [2]. In addition, a study assesses the main problems and needs raised by developers when using code samples [21]. Since the interaction and contribution between clients and organizations play an important key in SECO, it is necessary to understand how these interactions and contributions occur and how organizations deal with them. A way that clients can interact and contribute to code samples is GitHub pull requests.

To do so, in this paper, we explore how organizations deal with code sample contributions via pull requests on GitHub. To achieve this we developed the following research questions. (RQ1) Do organizations review and accept contributions from code sample clients? (RQ2) How long does it take organizations to review contributions from code samples clients? (RQ3) How do organizations distribute, among code sample maintainers, the review activity of clients contributions? To answer these questions, we conducted an exploratory study assessing around 12,000 pull requests of 2,179 code samples from Android, AWS, Azure, and Spring.

We found that the majority of contributions from code sample pull requests were already reviewed. Also, the majority of these contributions were accepted into the code sample repository. Regarding time to review, for accepted pull requests, in general, it takes less than 11 days. But for rejected pull requests, for most organizations, it takes more than 40 days, in general. Finally, we also found that pull requests review are performed by a few code sample maintainers.

The work of this paper is organized as follows. Section 2 presents the concepts necessary to understand this work. Section 3 presents the study design. Section 4 presents the results. Section 5 presents the implications of results. Section 6 presents the related works. Finally, Section 7 concludes the paper.

2 BACKGROUND

Software Ecosystems. SECO can be defined by interactions between actors, such as organizations, clients, and software systems around a common technological platform [1, 13, 18]. Organizations maintain the platform and are responsible for manage the SECO needs as clients demand. Clients use platform features to create value [13]. The interaction between these actors generate contributions that led to the evolution of the platform and SECO as well.

For clients generate value, they need to learn how to use platform features. However, there are some barriers to learning the platforms, for example, the possible lack of motivation in reading traditional documentation [32]. Difficulty in understanding and using platform-specific functionalities [31]. The need for quick learning [35]. And few complete examples [26]. As an alternative to help clients in the learning process of platform features, organizations are making available code samples.
**Code Samples.** As a relatively recent concept coming from the industry, organizations define code samples in different ways. For example, Oracle states that “code sample is provided for educational purposes or to assist your development or administration efforts [29].” Similarly, Spring reports that “code samples are designed to get you productive as quickly as possible [30].” And Mozilla says “code samples need to be simple enough to be understandable, but complex enough to do something interesting, and preferably useful [3].” Literature defines code sample as a complete software project with education purpose, made available by organizations, to assist their clients in understanding, using, and staying up to date with their product features [20, 21].

In the industry, there are guides for the construction and maintenance of code and in the literature, there are works that explore the characteristics found in code samples and their SECO as well: (1) Code samples should be simple and small to facilitate reuse and understanding [3, 20, 21]; (2) Code samples should provide a working environment as conventional projects [20, 21]; (3) Code samples should evolve and keep up to date, otherwise, they become outdated and less attractive to their clients [12, 16, 20].

**Actors Interactions and Pull Requests in SECO.** A key part of SECO is engaged actors and the interactions with each other [18]. Furthermore, these interactions between actors are just as important as interactions with the platform, for the survival of SECO [18]. One way for clients to interact with organizations is through pull requests. Pull requests allow developers (maintainers or clients) to make isolated changes over artifacts and then request their changes to be merged into the code sample repository [8, 27]. Figure 1 presents a pull request from gs-rest-service code sample, provided by Spring. In this example, the GitHub user veronx proposes to remove an unnecessary dependency from the code sample repository. The Buzzardo user, as a code sample maintainer, reviewed and commented about the change still needed. The veronx changes the pull requests to meet Buzzardo’s requirements. In the end, the maintainer accepted the pull request, and their code was merged to the code sample repository. The interaction between clients and the organization (through its maintainer) implicates the improvement of the code sample.

3 STUDY DESIGN

**Study Scenario:** It has already been noted that the most common problem faced by code sample clients is when they try to modify code samples [21]. It was also observed that the most recurring demand from clients is related to improving code samples, whether improving source code, documentation, or supporting tool [21]. This may indicate the clients’ desire to contribute to improving the code sample. To maintain a successful SECO, organizations need to meet the ecosystem needs and use business or motivation to encourage actors to contribute to the ecosystem evolution [18, 19]. Pull requests allow clients to interact and contribute with code samples, requesting changes to their repository [8, 27].

On the one hand, organizations can benefit from receiving contributions from their customers, this can improve their interactions and engagement, which is good for a healthy SECO. On the other hand, contribution acceptance, without proper review, may lead to a decay of code sample quality. This can be catastrophic due to code sample educational purposes. In this paper, we aim to explore how organizations deal with requests to change code samples repository through pull requests on GitHub. In addition, we aim to explore how are distributed the activities from pull requests management between the code sample maintainers.

**Pull Request Selection.** Our study focuses on contributions from code sample clients and evaluated by the organization, so we should not select pull requests created by code sample maintainers or code samples that have been closed by their creators. For that, in the first step, we select pull requests of the selected code samples. Second, we remove pull requests created by maintainers of the code sample (details about the maintainers are presented below). Finally, we removed pull requests that were closed by their own creators.

**Maintainers Selection.** We consider a code sample maintainer any GitHub user who accepted at least one pull request to a repository of studied code samples.

(RQ1) Do organizations review and accept contributions from code samples clients? To answer this question, we compute four metrics: number of reviewed pull requests, number of unreviewed pull requests, number of accepted pull requests, and number of rejected pull requests.

**Figure 1: GitHub page of a pull request from code sample gs-rest-service provided by Spring.**
pull requests, number of accepted pull requests and number of rejected pull requests. We consider a pull request to be reviewed when its state changes to closed and unreviewed when its state is open. We consider that a pull request has been accepted when its state changes to closed and it has been merged [23, 27]. And we consider that a pull request has been rejected when its state changes to closed but it has not been merged [23, 27]. Rationale: If there is a high proportion of unreviewed pull requests, it may indicate that organizations are not worried about reviewing them or have not been maintainers enough to attend to the needs of the community. This may discourage clients from contributing to the evolution of the code sample. In addition, if we find a low proportion of accepted pull requests, it may indicate that organizations are not open to receiving contributions, possibly from clients, in their code samples.

(RQ2) How long does it take organizations to review contributions from code samples clients? Is there a difference in time between accepted and rejected? To answer this question, we selected three metrics: time to review, time to accept and time to reject. Time to review is computed by the difference between the date pull requests became closed and its creation date. Time to review considers both accepted and rejected pull requests. Time to accept is calculated by the difference between the date pull requests became closed and its creation date, but only considers accepted pull requests. Similarly, time to reject is calculated by the difference between the date pull requests became closed and its creation date, but considers only rejected pull requests. Rationale: If the time to close is too high, this may deter clients to contribute, as they don’t see their contributions being inserted or even reviewed by code sample maintainers. By another view, if the time to close and the time to accept shows a high value, it could be an indication that maintainers took more time to review the pull requests may be due to a detailed review in order to reduce the insert of low-quality code or even errors in the code sample, and this is extremely important given the educational purpose of code samples.

(RQ3) How do organizations distribute, among code sample maintainers, the review activity of clients contributions? Given the set of code sample maintainers, we compute the number of pull requests reviewed performed for each maintainer. Next, we compute the Gini coefficient [7] along with the Lorenz curve [17]. Rationale: By calculating the Gini coefficient and Lorenz curve, we can understand how your pull request reviews are distributed in code samples. If revisions are the responsibility of a few maintainers, this can slow down the time it takes to review pull requests, and could be a bottleneck.

4 RESULTS

(RQ1) Do organizations review and accept contributions from code samples clients? Table 1 presents the number of reviewed and unreviewed pull requests from code samples. We found 483 pull requests for code samples from Android, 123 (25.5%) unreviewed and 360 (74.5%) reviewed. For AWS samples, we found 6,194 pull requests, 1,166 (18.9%) unreviewed and 5,028 (81.1%) reviewed. For Azure samples, we found 4,933 pull requests, 1,579 (32.1%) unreviewed and 3,354 (67.9%) reviewed. For Spring samples, we found 1,150 pull requests, 150 (13.1%) unreviewed and 1,000 (86.9%) reviewed. These results seem to show that most pull requests have already been reviewed by organizations. In any case, there is a non-negligible percentage of pull requests unreviewed, and organizations must show efforts to review them.

Table 1: Unreviewed vs Reviewed Pull Requests

<table>
<thead>
<tr>
<th>Project</th>
<th>Unreviewed</th>
<th>Reviewed</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>123 (25.5%)</td>
<td>360 (74.5%)</td>
<td>483</td>
</tr>
<tr>
<td>AWS</td>
<td>1,166 (18.9%)</td>
<td>5,028 (81.1%)</td>
<td>6,194</td>
</tr>
<tr>
<td>Azure</td>
<td>1,579 (32.1%)</td>
<td>3,354 (67.9%)</td>
<td>4,933</td>
</tr>
<tr>
<td>Spring</td>
<td>150 (13.1%)</td>
<td>1,000 (86.9%)</td>
<td>1,150</td>
</tr>
</tbody>
</table>

Table 2: Accepted vs Rejected Pull Requests

<table>
<thead>
<tr>
<th>Project</th>
<th>Accepted</th>
<th>Rejected</th>
<th>Reviewed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>292 (81.1%)</td>
<td>68 (18.9%)</td>
<td>360</td>
</tr>
<tr>
<td>AWS</td>
<td>4,422 (87.9%)</td>
<td>606 (12.1%)</td>
<td>5,028</td>
</tr>
<tr>
<td>Azure</td>
<td>2,901 (86.4%)</td>
<td>453 (13.6%)</td>
<td>3,354</td>
</tr>
<tr>
<td>Spring</td>
<td>705 (70.5%)</td>
<td>295 (29.5%)</td>
<td>1,000</td>
</tr>
</tbody>
</table>

(RQ2) How long does it take organizations to review contributions from code samples clients? Is there a difference in time between accepted and rejected? Figure 2 presents the time taken to pull requests be reviewed. We can see that, in the median, Android samples take 7.5 days to be reviewed. While pull requests from AWS samples take 3.6 days, in the median. Pull requests from Azure samples take 12.73 days and Spring samples take 32.58 days on the median. This seems to indicate that there is no standard across organizations, while pull requests for code samples from AWS typically take approximately 3 days, Spring pull requests take more than 30 days to be reviewed. However, on all platforms, there are pull requests that take longer than others, and there are even cases that take more than 500 days to review. This may be due to the complexity of the modifications, given that the number of lines added or modified and the number of commits is an attributes to determine the time to review [34].

Figure 3 (left) presents the time taken to review pull requests accepted into code sample repositories. We noted that in Android
samples, in the median, pull requests took 6.61 days to be accepted. In AWS, was 2.67 days, on the median. While in Azure samples were 7.75 days and Spring samples were 7.54 days, in the median. Figure 3 (right) shows the time taken to review pull requests rejected. For Android samples, we noted that, in the median, was spent 24 days reviewing rejected pull requests. In AWS samples, pull requests took 29 days, in the median. While Azure and Spring present 75.5 and 108.38 days respectively, in the median. We found that, in general, accepted pull requests take less time to review than rejected pull requests. This behavior is also found in conventional projects [11].

<table>
<thead>
<tr>
<th>Time to Accept</th>
<th>Time to Reject</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.61 AWS</td>
<td>Android</td>
</tr>
<tr>
<td>2.67 AWS</td>
<td>AWS</td>
</tr>
<tr>
<td>7.75 Azure</td>
<td>Azure</td>
</tr>
<tr>
<td>7.54 Spring</td>
<td>Spring</td>
</tr>
</tbody>
</table>

(RQ3) How do organizations distribute, among code sample maintainers, the review activity of clients contributions?

Figure 4 presents the Lorenz curve for the relation between code sample maintainers and the number of pull requests reviewed. In addition to the Lorenz curve, we also compute the Gini coefficient. The values were 0.8 for Android, 0.91 for AWS, 0.91 for Azure, and 0.82 for Spring. These results show that the pull request review activity is mostly performed by a small group of maintainers. This behavior can be explained because the pull request review task requires experienced maintainers and organizations are concerned about selecting only competent maintainers for this task. However, as we saw in the results of RQ2, there is a certain delay in some cases of review. This can be caused by a bottleneck in the distribution of review tasks and organizations should pay attention to this.

The distribution of review activity from code sample pull requests focuses on a small group of maintainers. This may be due to the need for experienced maintainers for such a task, but organizations should note that this can be a bottleneck.

5 IMPLICATIONS

The community matters. Our results showed that, in general, organizations care about interacting with their clients’ community, by reviewing their contributions from pull requests. This goes further, as we realize that organizations are receptive to clients’ contributions from pull requests, given the acceptance rate. These interactions between actors are just as important as interactions with the platform and play an important role in SECO’s survival [18]. Therefore, it is important that organizations use efforts to foster and increase these interactions and contributions.

To each organization, its rules. We observed that there are some differences between how organizations handle contributions from pull requests in their code samples. Some are more receptive than others to these contributions. Also, some take less time to review these contributions than others. Also, some take less time to review these contributions than others.

Expand to conquer. We’ve seen that the task of reviewing contributions from pull requests is focused on a small group of maintainers. And that could be causing a delay in the review. Therefore, we believe that it would be important for organizations to better distribute this activity between their maintainers. That way, pull requests would possibly take less time to review, and it can be an incentive for more clients to contribute and contribute more often.

6 RELATED WORKS

Code Samples in SECO. Menezes et al. et al. assessed the characteristics of source code, evolution, and the use of code samples [20].
Menezes et al. observed the main problems faced and the main demands demanded by code sample clients [21]. Braga et al. analyzed the target audience of code samples, and how experienced their clients are. [2]. Unlike these studies, our study assesses how organizations handle clients’ contributions to code samples.

**Pull Requests.** Soares et al. explored around 97,000 pull requests from 30 GitHub projects to find characteristics that have affected pull requests lifetime [23]. Soares et al. conducted a study with 22,523 pull requests to understand the factors that influence the assignment of reviewers to pull requests [27]. Gousios et al. assessed around 166,000 pull requests to understand the characteristics of lifetime, acceptance, and rejection of pull requests in 291 projects [11]. Similarly, with these studies, we also assess time, review, acceptance, and rejection. But code samples are different from conventional projects due to their educational purpose [21]. This makes necessary an exclusive study of their pull requests.

### 7 CONCLUSION

In this paper, we conduct an empirical study to understand how organizations handle contributions from pull requests of their code samples. In addition, we also explore how organizations distribute the review activity of these pull requests. We assess around 12,000 pull requests from 2,179 code samples, investigating the percentage of reviewed, unreviewed, accepted and rejected pull requests. We also assess the time it takes for accepted and rejected pull requests to be reviewed, and how the review task is distributed among code sample maintainers. We found that the majority of pull requests were already reviewed and accepted to the code sample repository. We noted that the accepted pull requests take less time to be reviewed than rejected ones. Finally, we found that the review activity is performed by a small group of maintainers.

In future work, we aim to compare our results with pull requests from projects that are not code samples, to understand their differences. Also, we want to explore, from the accepted and rejected pull requests, which one comes from code samples clients and from organization maintainers themselves. This can better clarify if organizations are accepting contributions from external developers or only from internal ones. In addition, the source of pull requests (client or maintainer) can impact the time to be reviewed, so it’s interesting also extract time for its source.

### ACKNOWLEDGEMENTS

This research is supported by CAPES and CNPq. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

### REFERENCES