Information and Software Technology 67 (2015) 1–12

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

Comparative case studies of open source software peer review practices

Jing Wang [a,*], Patrick C. Shih [b], Yu Wu [a], John M. Carroll [a]

[a] College of Information Sciences and Technology, The Pennsylvania State University, University Park, PA 16802, USA
[b] Department of Information and Library Science, Indiana University, Bloomington, IN, USA

ARTICLE INFO

Article history:
Received 24 November 2014
Received in revised form 5 May 2015
Accepted 10 June 2015
Available online 17 June 2015

Keywords:
Open source software
Virtual community
Software peer review
Design

ABSTRACT

Context: The power of open source software peer review lies in the involvement of virtual communities, especially users who typically do not have a formal role in the development process. As communities grow to a certain extent, how to organize and support the peer review process becomes increasingly challenging. A universal solution is likely to fail for communities with varying characteristics.
Objective: This paper investigates differences of peer review practices across different open source software communities, especially the ones engage distinct types of users, in order to offer contextualized guidance for developing open source software projects.
Method: Comparative case studies were conducted in two well-established large open source communities, Mozilla and Python, which engage extremely different types of users. Bug reports from their bug tracking systems were examined primarily, complemented by secondary sources such as meeting notes, blog posts, messages from mailing lists, and online documentations.
Results: The two communities differ in the key activities of peer review processes, including different characteristics with respect to bug reporting, design decision making, to patch development and review. Their variances also involve the designs of supporting technology. The results highlight the emerging role of triagers, who bridge the core and peripheral contributors and facilitate the peer review process. The two communities demonstrate alternative designs of open source software peer review and their trade-offs were discussed.
Conclusion: It is concluded that contextualized designs of social and technological solutions to open source software peer review practices are important. The two cases can serve as learning resources for open source software projects, or other types of large software projects in general, to cope with challenges of leveraging enormous contributions and coordinating core developers. It is also important to improve support for triagers, who have not received much research effort yet.

1. Introduction

A distinct and powerful characteristic of open source software (OSS) development is the involvement of communities that engage general users who do not belong to typical software development roles. This has made OSS development an appealing research area [2,24,42,43]. Recent advances of social computing infrastructure create opportunities for OSS projects to leverage an even larger crowd [12], as is demonstrated by GitHub's surpassing other open source forges in total number of commits in 2011 [15].

Among various forms of participation in a community, peer review is widely regarded as the one that significantly benefits from the community involvement. Raymond coined "Linus's law" ("given enough eyeballs, all bugs are shallow") to emphasize the advantage of extensive peer review in OSS development [31]. It is the practice in which community members evaluate and test software products, identify and analyze defects or deficiencies, and contribute and verify solutions (e.g., patches) that repair or improve them.

This community-based practice raises the question of how communities organize their peer review process, especially when they have reached a large scale. As OSS communities grow and mature, they encounter new challenges of engaging contributions, coordinating work, and ensuring software quality [27,38]. Without explicit coordination mechanisms and governance, their sustainability

* Corresponding author at: 316C IST Building, University Park, PA 16802 USA. Tel.: +1 814 863 8856.
E-mail addresses: jzw143@ist.psu.edu (J. Wang), patshih@indiana.edu (P.C. Shih), yuw132@ist.psu.edu (Y. Wu), jcarroll@ist.psu.edu (J.M. Carroll).

http://dx.doi.org/10.1016/j.infsof.2015.06.002
0950-5849/

and evolution will be challenged [17,26]. Therefore, reflecting on the practices of well-established large OSS communities could prepare other growing OSS projects of how to overcome such challenges.

Each project has its own uniqueness and no one-size-fits-all model can ensure success [4]. Studies on activities related to OSS peer review (e.g., [21,28,34]) have already showed initial evidence that the peer review practice is likely to vary across different OSS communities. However, those analyses did not spend much effort articulating the variances. They were largely focused on extracting commonalities among OSS projects or characterizing a single project, reporting the common constituting activities of OSS peer review including submission/bug reporting, analysis/design discussion, fix/patch development, test/patch review [10,45].

Our investigation is aimed at extending prior research with a dedicated codification of different OSS peer review practices, which can provide contextualized implications for other developing OSS projects. We use the term "peer review" in a broad sense to include all types of efforts in detecting software defects or deficiencies rather than confine it to code reading. As a distinctive and pivotal characteristic of OSS peer review, user involvement undoubtedly contributes to the differences among the practices. Users are also a substantial participating group of OSS communities, from which peer review practices cannot be detached. Considering the unique quality and the context complexity, we contrast OSS peer review practices through case studies on two well-established large communities that produce software targeting remarkably different types of users—*software developers* and *end-users*. Two communities can by no means cover all the variations of OSS peer review practices, but can still provide important insights [4]. Moreover, comparing the two ends of a spectrum of user technical expertise highlights the differences of significance as well as enables an appropriate and flexible combination in-between.

Our study contributes to alternative designs of social mechanism and technology for OSS peer review. We identify differences in *bug reporting*, *design decision*, and *patch development and review*, the key activities constituting peer review practices, as well as in the tool affordance and use between two well-established large OSS communities. Our findings highlight the importance of *triagers*, an emerging group of contributors who mediate between the core and periphery and facilitate the peer review process. We also characterize how core developers collaborate differently in response to the different types and sizes of peripheral participants. This extends prior research that primarily focused on comparing between core and periphery.

2. Related work

2.1. Community-based open source software development

The openness of OSS to users and the engagement of virtual communities have been attracting considerable research efforts. Findings and discussions have centered on the different contributions from core and peripheral members. Quantitative examinations repeatedly detected skewed contribution distribution: a small group of developers in the core contributed majority of the code, while the rest in the community mainly made occasional contributions by reporting bugs [18,25,28]. Subsequent work elaborated the roles of core and peripheral members in OSS development. Dahlander and Frederiksen [13] studied how peripheral participants affect OSS innovation. They found that one's position in the core/periphery structure is more consequential for innovating than his/her expertise. Rulliani and Haefliger [36] described the role of core developers and those in the peripheries, and how the propagation of such standards is communicated through non-material artifacts such as code and virtual discussions as a social practice. Another related theme focused on how peripheral participants advanced to core developers through either legitimate peripheral participation [29] or socialization [14,44].

Some OSS studies described the roles in OSS development at a finer level than the dichotomy, but most of them still classified through the lens of users versus developers. The best-known examples from this group of work probably include the "onion model" [8] and the layered community structure [52]. They share similar ideas: developers other than the core contribute their patches or review or revise others' code, either regularly (i.e., active developer/co-developer) or sporadically (i.e., peripheral developer); users consist of active ones who report bugs and passive ones who only use the software. Ko and Chilana [24] also identified a group of users, *power users*, who submit quality bug reports. A very small portion of research discussed the roles between users and developers. Barcellini et al. [3] characterized the emerging roles in the Python community. They studied 2 mailing lists (i.e., one user-oriented and one developer-oriented) and found that several key participants act as boundary spanners across the communities for driving feature sets. They proposed a "role emerging design" to support the coordination process in OSS.

Our study extends the understanding of roles in OSS communities through identifying and elaborating the emerging role of bug triagers. While the OSS peer review process is not the same as feature discussions reported in the aforementioned OSS literature, we found that bug triagers serve a similar boundary-spanning role for resolving issues surrounding bugs. This role entails different types of tasks depending on the characteristics of users and developers in the community.

2.2. Software peer review in open source

Peer review is the evaluation of the quality of one's work products by others. In software development, the primary objective of peer review is to discover defects (or bugs) as early as possible during development processes, suggest improvements and even help developers create better products [48]. Code is not the only object that peer review assesses but any artifacts created during the software development process, such as requirements specifications, use cases, project plans, user interface design and prototypes, and documentation [22].

To this end, we use the term OSS peer review inclusively to refer to the evaluation of the entire software application instead of confining it to reading code or reviewing patches (i.e., change sets to software products). We consider that such inclusiveness best suits the original description of "extensive peer review" in OSS literature [31]. Previous studies have identified common activities in the peer review process [10,45]. It often begins with one submitting a bug report (i.e., submission/bug reporting). Others diagnose the defect causes and request additional information to determine whether the bug should be fixed (i.e., analysis/problem identification/design decision). Once a solution is generated (i.e., fix/solution generation/patch development), they evaluate then commit the solution to the current software product (i.e., test/patch review and commit).

Previous OSS studies have touched upon each individual activity involved in the OSS peer review process, but were largely focused on the common characteristics across projects. With respect to bug reporting, researchers found a mismatch between the information users reported and the information developers needed [6,45]. Ko and Chilana [24] examined the massive bug reports in Mozilla, suggesting that reports that led to changes were reported by a comparably small group of experienced frequent reporters, who were the only valuable users to recruit in open