Leveraging Software Bots to Enhance Developers' Collaboration in Online Programming Communities

Mairieli Wessel
University of São Paulo
São Paulo, Brazil
mairieli@ime.usp.br

ABSTRACT

Software bots are applications that are integrated into human communication channels, serving as an interface between users and other tools. Due to their focus on task automation, bots have become particularly relevant for Open Source Software (OSS) projects hosted on GitHub. While bots are adopted to save developers' costs, time, and effort, the interaction of these bots can be disruptive to the community. My research goal is two-fold: (i) identify problems caused by bots that interact in pull requests, and (ii) help bot designers to enhance existing bots, thereby improving the partnership with contributors and maintainers. Toward this end, we are interviewing developers to understand what are the problems on the human-bot interaction and how they affect human collaboration. Afterwards, we will employ Design Fiction to capture the developers' vision of bots' capabilities, in order to define guidelines for the design of bots on social coding platforms, and derive requirements for a meta-bot to deal with the problems. This work contributes more broadly to the design and use of software bots to enhance developers' collaboration and interaction.

CSCW '20 Companion, October 17–21, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-8059-1/20/10.
https://doi.org/10.1145/3406865.3418368

CCS CONCEPTS

• **Human-centered computing** → *Open source software*; • **Software and its engineering** → *Software creation and management*.

KEYWORDS

Software Bots; GitHub Bots; Open Source Software; Software Engineering

ACM Reference Format:
Mairieli Wessel. 2020. Leveraging Software Bots to Enhance Developers' Collaboration in Online Programming Communities. In *Companion Publication of the 2020 Conference on Computer Supported Cooperative Work and Social Computing (CSCW '20 Companion), October 17–21, 2020, Virtual Event, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3406865.3418368

Social coding platforms, such as GitHub, are widely used to host Open Source Software (OSS) projects. While providing features that aid collaboration and sharing, such as pull requests, these platforms contribute to increasing the workload of maintainers to communicate and review contributions. Recently, a new phenomenon has arisen to reduce this burden: software bots. As an interface that integrates humans and services, bots are playing an increasingly prominent role in online programming communities. These bots help reduce the intensive workload inherent to the pull request model by automating routine tasks and interacting with human developers. Such bots work on different tasks and are usually created as a GitHub user that can submit code contributions, interact through comments, and merge or close pull requests.

By executing tasks that were previously only performed by human developers, and interacting in the same communication channels as developers, bots have become new voices in the pull request conversation [7]. Nevertheless, the presence of these new "team members" can be disruptive, giving rise to problems that interfere in the community's interaction. According to Brown et al. [2], the human-bot interaction on pull requests can be inconvenient, leading developers to leave negative feedback or even abandon their contributions. This may be especially true for newcomers, who require special support during the onboarding process due to the barriers they face [9]. Newcomers can experience bots' complex answers as an additional and discouraging barrier, since bots can provide a long list of critical contribution feedback (e.g., style guidelines, failed tests), rather than supportive assistance.

Although some studies focus on bots' development [7], little has been done to investigate the potential problems introduced by bots at large. In fact, there are important aspects that must be considered by bot developers in order to increase the acceptance of bots by online programming communities. During a recent Dagstuhl seminar on bots in Software Engineering that I attended [10], some key themes of bot interaction that emerged are avoiding repetitive notifications, providing

RESEARCH QUESTIONS

To contribute to this problem, my research seeks to address the following research questions:

RQ1. What interaction problems do bots introduce when supporting pull requests?

RQ2. What features do maintainers, contributors, and bot developers envision for a bot to mitigate current problems?

RQ3. How might these envisioned features mitigate interaction problems?

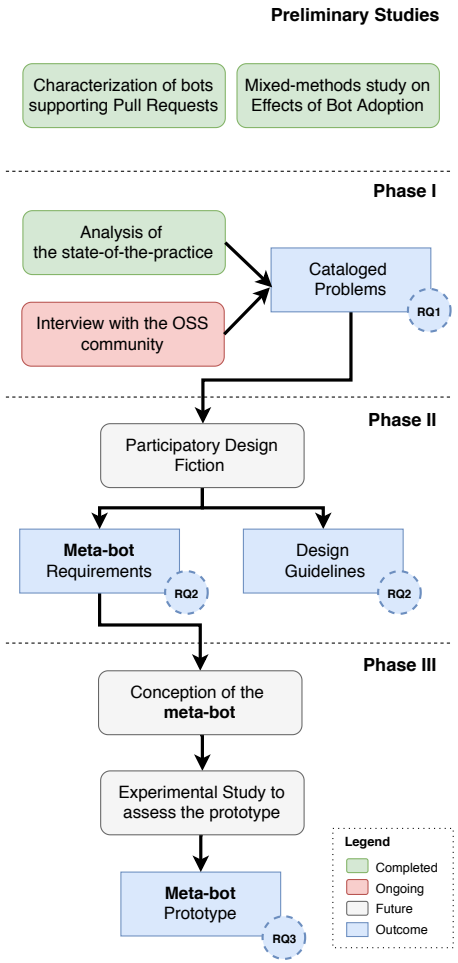

Figure 1: An Overview of the Research Design.

consistency in the tasks being done, and making bots adaptive. Thus, it is critical to understand that software bots are socio-technical rather than technical applications, and must be designed to consider human interaction, developers' collaboration, and ethical concerns.

Considering this context, the goal of my research is to *design and evaluate strategies to mitigate problems related to the interaction with software bots on social coding platforms, thereby assisting developers to communicate and accomplish their tasks.* Thus, the overall question addressed by this research is: **How can bots better support OSS developers' interaction in online programming communities?** To answer this question, this research employs mixed-methods analysis of surveys, interviews and participatory design fiction, and quasi-experimental research studies.

BACKGROUND

Software bots are extensively proposed and analyzed in the literature of different domains including social media [14], online learning [4], and Wikipedia [15]. On Wikipedia, for example, bots change the overall ecosystem by interacting with their operators, managers, and human editors, as well as other bots. Zheng et al. [15] describe that while editors acknowledge bots for streamlining knowledge production, they complain that bots not only solve problems but creates them as well.

In software engineering, bots have been proposed to support technical and social aspects of software development activities [5], such as communication and decision-making [11]. However, understanding how bots' interaction affects human developers is a major challenge. Storey and Zagalsky [11] highlight that bots' potential negative impact is still neglected. Mirhosseini and Parnin [6], for example, reported that maintainers are overwhelmed by bots notification, which interrupts their workflow. According to Brown and Parnin [2], bots still need to enhance interaction with humans, in order to overcome problems such as notification workload. Considering this gap, we go a step further, investigating problems on the human-bot interaction on pull requests, and how to mitigate them.

RESEARCH OVERVIEW

The research design comprises three phases, as presented in Figure 1.

Preliminary Studies — The Role Bots Play and How They Affect the Community

In 2018, we conducted a preliminary study to characterize the bots that support pull requests on GitHub. Our results indicate that bots perform several tasks, including ensuring license agreement signing, reporting continuous integration failures, reviewing code and pull requests, triaging issues, and refactoring the source code [12]. We also openly asked contributors and maintainers about the "challenges of using bots" on pull requests. Several contributors complained about the way the bots interact, saying that the *bots provide non-comprehensive or poor feedback*, while others mentioned that *bots introduce communication noise* and that there is a *lack of information on how to interact with the*

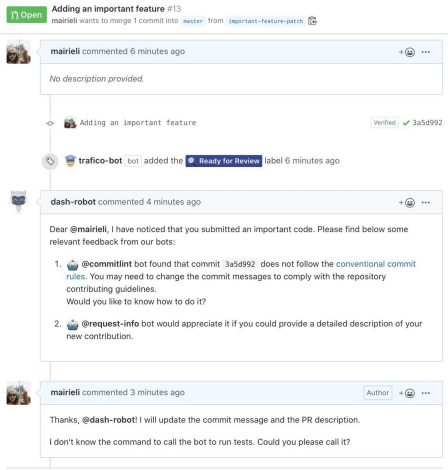

Figure 2: Interface mockup of the interaction between a contributor/maintainer and the meta-bot on a pull request.

bot. Our survey respondents deem the current bots as *not smart enough* and provided insights into the bots' potential new features, such as *improving notification and awareness*, *enhancing user interaction*, *improving communicability*, and *answering specific questions*. Our results suggest that *GitHub bots* serve as a useful way to access services and to automate tasks; however, in terms of supporting developers' interaction, they are not as evolved as in other domains (e.g., education, customer service).

Towards understanding the effect bot adoption, we conducted an exploratory empirical investigation of the effects of adopting bots to support the code review process on pull requests. First, we statistically analyzed data from 1,466 open source projects hosted on GitHub. Analyzing the statistical models, on the one hand, we found that more pull requests are merged into the codebase after the bot adoption, and there is less communication between contributors and maintainers. On the other hand, merging pull requests takes more time. Considering non-merged pull requests, after bot adoption, projects have less monthly non-merged pull requests, and faster pull requests rejections. To understand their perspective on the effects of bot adoption, we surveyed 127 open-source project maintainers. Respondents reported 13 changes in the maintenance process. Three responses, although less recurrent, called our attention to negative effects caused by bot adoption. Bots are *impersonating human developers*, *introducing noise* and *intimidating newcomers*.

Phase I — Understanding Human-bot Interaction Through the Lens of the OSS Community

Understanding how developers and bots interact on GitHub is the first step towards designing strategies to improve the human-bot interaction. To further this investigation, we gathered some anecdotal evidence of problems in the human-bot interaction from the state-of-the-practice. We manually analyzed pull requests, looking for (i) human users mentioning bots, and (ii) bots' interactions—such as opening, merging, or commenting on pull requests. We noticed that the bots used in pull requests indeed (i) overwhelm developers' communication with notifications and feedback, (ii) perform wrong actions, and (iii) are misused [13].

Currently, we are interviewing contributors, maintainers, and bot developers to understand their perspectives about the problems encountered in the state-of-the-practice, and to identify new problems. This study currently consists of 21 semi-structured interviews with participants that were recruited directly or using a snowballing sampling. Participants were expected to have experience contributing or maintaining projects which use bots to support any pull request activity. Emergent problems include technical, social, and cultural issues. This analysis provides the foundation to our catalog of programmer bot interaction challenges.

Phase II — Designing Strategies to Support Developers' Interactions

After identifying the human-bot interaction problems, we will design strategies for better supporting interactions on pull requests. By capturing the expectations of developers who interact with bots,

we will: (i) define guidelines for the design of future bots, and (ii) elicit the features to a meta-bot. This meta-bot was inspired by Sadeddin et al. [8] work. In order to deal with several responses from different bots, Sadeddin et al. [8] showed that a meta-bot would obtain product information from several shopping bots, and then summarize and present it to the user. The concept of meta-bot is also present in the literature of software agents [3]. Therefore, we will use Design Fiction [1] as a participatory method to explore ways to overcome a subset of the most relevant problems evidenced. The speculative nature of this technique amplifies critical views of current social and technological developments, creating a fictional context narrated through designed artifacts. Using semi-structured interviews, we will present to participants a fictional history of a meta-bot capable of better supporting developers' interactions on pull requests, operating as a middleman between developers and the existing bots. Participants will act as storytellers, answering questions to ground the end of the fictional history, in order to raise social and ethical concerns around the use of bots, as well as the requirements of the meta-bot.

Phase III — Transforming Design Strategies to Bot Prototypes

Grounded on the findings of the participatory design fiction, we will design and implement a preliminary prototype of the meta-bot to be further evaluated. Essentially, we envision the meta-bot as a promising approach to assist developers to perform their tasks more efficiently and help OSS projects attract, engage and retain contributors. Our envisioned meta-bot (see Fig. 2) will provide more flexibility to developers, enabling them to configure the dynamics of the interaction. In short, the meta-bot solves interaction problems by mediating the action of other bots used on pull requests. Compared to other GitHub bots, the meta-bot will provide additional value to the interaction of already existing bots through these key features: (i) summarizing other bots' outcomes to avoid information overload; (ii) supporting developers' questions and requests; (iii) providing customizable feedback; and (iv) helping developers deal with bots' exceptions.

To evaluate the bot prototype, we will conduct a preliminary user study with OSS project specialists. Since software bots are typically evaluated using factors that are related to both accuracy and usability [2], we will conduct an evaluation of whether the implemented features mitigate interaction problems. The results will enable us to improve the bots' requirements and the prototype according to the feedback received. In addition to the study involving specialists, the developed meta-bot will be integrated to a real OSS project. The goal of this integration is to receive feedback from contributors and maintainers and understand to what extent the adopted strategies support the developers' collaboration. In this case, we plan to conduct debrief sessions with the developers that interacted with the meta-bot and quantitatively analyze this data. In order to receive large scale feedback, the meta-bot will be available to be integrated to projects hosted on GitHub in the long term.

EXPECTED CONTRIBUTIONS

In summary, my proposal contains three main novel contributions: (i) an empirical catalog of human-bot interaction problems in social coding platforms, with focus on the pull request submission process; (ii) guidelines for designing *bots* to support software development tasks; and (iii) a meta-bot to support developers' interactions. In the long-term, this research will provide a more nuanced view of the human-bot interaction in social coding platforms. With a more in-depth understanding of this interaction, researchers and practitioners can invest their efforts in designing or improving bots, ultimately supporting developers on submitting and reviewing pull requests.

ACKNOWLEDGMENTS

I would like to thank my advisor Prof. Marco A. Gerosa and my co-advisor Prof. Igor Steinmacher. This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001 and CNPq (# 141222/2018-2).

REFERENCES

[1] Mark Blythe. 2014. Research through design fiction: narrative in real and imaginary abstracts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 703–712.

[2] Chris Brown and Chris Parnin. 2019. Sorry to Bother You: Designing Bots for Effective Recommendations. In *Proceedings of the 1st International Workshop on Bots in Software Engineering (BotSE)*.

[3] Meric Dagli. 2019. *Designing for Trust*. Ph.D. Dissertation.

[4] Supratip Ghose and Jagat Joyti Barua. 2013. Toward the implementation of a topic specific dialogue based natural language chatbot as an undergraduate advisor. In *Informatics, Electronics & Vision (ICIEV), 2013 International Conference on*. IEEE, Washington, DC,USA, 1–5.

[5] Bin Lin, Alexey Zagalsky, Margaret-Anne Storey, and Alexander Serebrenik. 2016. Why developers are slacking off: Understanding how software teams use slack. In *Proceedings of the 19th ACM Conference on Computer Supported Cooperative Work and Social Computing Companion*. ACM, 333–336.

[6] Samim Mirhosseini and Chris Parnin. 2017. Can Automated Pull Requests Encourage Software Developers to Upgrade Out-of-date Dependencies?. In *Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE 2017)*. IEEE Press, Piscataway, NJ, USA, 84–94. http://dl.acm.org/citation.cfm?id=3155562.3155577

[7] Martin Monperrus. 2019. Explainable Software Bot Contributions: Case Study of Automated Bug Fixes. In *Proceedings of the 1st International Workshop on Bots in Software Engineering (BotSE '19)*. IEEE Press, Piscataway, NJ, USA, 12–15. https://doi.org/10.1109/BotSE.2019.00010

[8] Khaled W Sadeddin, Alexander Serenko, and James Hayes. 2007. Online shopping bots for electronic commerce: the comparison of functionality and performance. *International Journal of Electronic Business* 5, 6 (2007), 576.

[9] Igor Steinmacher, Tayana Uchoa Conte, Christoph Treude, and Marco Aurelio Gerosa. 2016. Overcoming Open Source Project Entry Barriers with a Portal for Newcomers. In *Proceedings of the 38th International Conference on Software Engineering (ICSE)*.

[10] Margaret-Anne Storey, Alexander Serebrenik, Carolyn Penstein Rosé, Thomas Zimmermann, and James D. Herbsleb. 2020. BOTse: Bots in Software Engineering (Dagstuhl Seminar 19471). *Dagstuhl Reports* 9, 11 (2020), 84–96.

[11] Margaret-Anne Storey and Alexey Zagalsky. 2016. Disrupting Developer Productivity One Bot at a Time. In *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE 2016)*. ACM, New York, NY, USA, 928–931. https://doi.org/10.1145/2950290.2983989

[12] Mairieli Wessel, Bruno Mendes de Souza, Igor Steinmacher, Igor S. Wiese, Ivanilton Polato, Ana Paula Chaves, and Marco A. Gerosa. 2018. The Power of Bots: Characterizing and Understanding Bots in OSS Projects. *Proceedings of the ACM Conference on Computer Supported Cooperative Work Social Computing* 2, CSCW, Article 182 (Nov. 2018), 19 pages. https://doi.org/10.1145/3274451

[13] Mairieli Wessel and Igor Steinmacher. 2020. The Inconvenient Side of Software Bots on Pull Requests. In *Proceedings of the 2nd International Workshop on Bots in Software Engineering (BotSE)*. https://doi.org/10.1145/3387940.3391504

[14] Bin Xu, Tina Chien-Wen Yuan, Susan R. Fussell, and Dan Cosley. 2014. SoBot: Facilitating Conversation Using Social Media Data and a Social Agent. In *Proceedings of the Companion Publication of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW Companion '14)*. ACM, New York, NY, USA, 41–44. https://doi.org/10.1145/2556420.2556789

[15] L Zheng, Christopher M Albano, and Jeffrey V Nickerson. 2018. Steps toward Understanding the Design and Evaluation Spaces of Bot and Human Knowledge Production Systems. In *Wiki Workshop'19*.