Who, What, Why and How? Towards the Monetary Incentive in Crowd Collaboration: A Case Study of Github's Sponsor Mechanism

Xunhui Zhang, Tao Wang*, Yue Yu*, Qiubing Zeng, Zhixing Li, Huaimin Wang

{zhangxunhui,taowang2005,yuyue,lizhixing15}@nudt.edu.cn,qiubingzeng@gmail.com,whm_w@163.com

National University of Defense Technology

Changsha, Hunan, China

ABSTRACT

While many forms of financial support are currently available, there are still many complaints about inadequate financing from software maintainers. In May 2019, GitHub, the world's most active social coding platform, launched the SPONSOR mechanism as a step toward more deeply integrating open source development and financial support. This paper collects data on 8,028 *maintainers*, 13,555 *sponsors*, and 22,515 *sponsorships* and conducts a comprehensive analysis. We explore the relationship between the SPONSOR mechanism and developers along four dimensions using a combination of qualitative and quantitative analysis, examining **why** developers participate, **how** the mechanism affects developer activity, **who** obtains more sponsorships, and **what** mechanism flaws developers have encountered in the process of using it. We find a long-tail effect in the act of sponsorship, with most maintainers' expectations remaining unmet, and sponsorship has only a short-term, slightly positive impact on development activity but is not sustainable. While sponsors participate in this mechanism mainly as a means of thanking the developers of OSS that they use, in practice, the social status of developers is the primary influence on the number of sponsorships. We find that both the SPONSOR mechanism and open source donations have certain shortcomings and need further improvements to attract more participants.

CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

KEYWORDS

sponsor, donation, GitHub, open source, financial support

* indicates the corresponding authors.

1 INTRODUCTION

Open source development has brought prosperity to software ecosystems. Its characteristics of distributed coordination, free participation, and convenient sharing have led to the emergence of myriad open source projects, large-scale participation of developers, and continuous development of high-quality projects. However, the expansion of project scales has also brought challenges for software maintenance, such as continuously and rapidly increasing feature requests and bug fix reports [37] and an increasing pull request review workload [69]. Although there are many continuous integration (CI) tools and continuous deployment (CD) tools to help reduce the workload of project managers, the complicated and high-pressure maintenance work still subjects them to stress [66].

Past studies have shown that most current open source work is still spontaneously performed by volunteers [22]. They engage in open source work as a hobby, to improve their personal reputations or to learn new technologies. These intrinsic benefits motivate volunteers to make open source contributions [21]. However, many core managers and software maintainers would like to secure funding from others for their open source work because of the aforementioned challenges, thereby alleviating the related mental pressure and financial burdens [5, 57, 67].

At present, there are many ways in which the open source sphere obtains financial support, such as crowdfunding on Kickstarter, project donations on OpenCollective, and issue rewards on BountySource and IssueHunt [49]. However, these are mainly web portals serving open source contributors active in other social coding communities. The separation of development activities and financial support brings problems. First, it is difficult for sponsors to find active developers and open source projects in the open source community. Second, open source contributors need to spend considerable effort on maintaining the financial support platform. In May 2019, GitHub, the world's most popular software hosting platform, launched the SPONSOR mechanism, characterized by deep integration of financial support and the social coding platform. While the SPONSOR mechanism supports sponsorship of organizations and projects, it targets mainly individual contributors in the GitHub community. Therefore, unlike past related studies [52, 53], we can explore donation mechanism in the open source sphere from the perspective of individual developers. In this context, this paper aims to *explore donation in the open source sphere using the SPONSOR*

mechanism as an example. We conducted an empirical study based on mixed methods and answered the following research questions.

RQ1 ***Why* do individuals participate or not in the SPONSOR mechanism?**
From the feedback of GitHub developers, we summarized eight reasons for participation among sponsored developers, six reasons for participation among sponsors, and six reasons for not participating in the mechanism among other individuals. The main reason that participants used the SPONSOR mechanism was its relationship with open source software (OSS) usage. The main reason for not participating was that developers did not need sponsorship or that they were driven to participate in open source development because of its nonmonetary character. Our findings can help optimize the SPONSOR mechanism and attract more participants by satisfying the different motivations of contributors.

RQ2 ***How* effective is sponsorship in motivating developer OSS activity?**
We find through quantitative analysis that the sponsor mechanism has provided only a short-term, subtle boost to contributors' activities. According to the results of the qualitative analysis, most developers agree that sponsorship can provide them with motivation but are not satisfied with the available amounts. In contrast, most sponsors are satisfied with the current mechanism. Our findings shed light on the application of the SPONSOR mechanism in the open source sphere and the problems surrounding it. This work helps to rationalize the mechanism to promote greater participation in open source contributions among developers.

RQ3 ***Who* is likely to receive more sponsorship?**
The questionnaire results show that making useful OSS contributions and being active are the most critical factors for obtaining more sponsorship. However, according to the quantitative data analysis results, the factor that most affects sponsorship is the developer's social status in the community. Our findings can provide actionable suggestions for developers seeking more sponsorships, while the conflicting results also illuminate the problems with OSS donations.

RQ4 ***What* are the shortcomings of the SPONSOR mechanism?**
The research reveals that problems with the mechanism include usage deficiencies, object orientation with supported functions, and personalization. Many developers complain that the donations do not apply to open source ecosystems. A more relevant mechanism is needed to promote the healthy and sustainable development of the ecosystem.

The contributions of this paper are as follows:

- To the best of our knowledge, this is the first in-depth study that comprehensively analyzes the GitHub SPONSOR mechanism.
- We quantitatively and qualitatively analyze the SPONSOR mechanism along four dimensions, including developers' motivation to participate (why), the mechanism's effectiveness (how), the characteristics of developers who obtain more sponsorships (who), and the mechanism's shortcomings (what).

- We provide actionable suggestions to help developers participating in the SPONSOR mechanism obtain more sponsorship and feasible advice for improving the mechanism's effectiveness.

The remainder of this paper is organized as follows. Section 2 presents the related work, and Section 3 describes the background of the GitHub SPONSOR mechanism. Section 4 presents the study design of this paper. In Section 5, we describe the results for each research question. Then, we discuss the findings in Section 6, and describe the threats in Section 7. Finally, in Section 8, we conclude the paper and describe future work.

2 RELATED WORK

Open Innovation in Science (OIS) is a concept, which unifies the two domains of open and collaborative practices in science, *i.e.,* open science (OS) and open innovation (OI) [6]. For OS, the three pillars are accessibility, transparency, and inclusivity, among which the inclusivity (*e.g.,* citizen science) is directly related to the knowledge production process. For OI, various forms of collaborative practice exist, including crowdsourcing, OSS development, etc. Regarding these open initiatives, the motivation and incentives of participation has always been the focus of continuous research [4, 70]. Although there are different views on the relationship between citizen science, crowdsourcing, and OSS development, we follow the relationships described above and present the related work on participation motivation and monetary incentives of the three parts separately.

2.1 Citizen science

For traditional citizen science, the motivation of participants varies greatly depending on the age [2], gender [48], educational background [46], and level of involvement [63]. In many cases, both monetary and non-monetary incentives have a positive effect on participation [9]. However, Wiseman et al. found that non-monetary incentives alone were better for online HCI projects to promote high-quality data from participants [71]. Knowles [38] also confirmed that although monetary incentives enhanced participation, they undermined sustained participation in volunteering initiatives. While for some specific projects (*e.g.,* the conservation of species), monetary incentives even have the opposite effect [55].

Because participants act as sensors to collect data or volunteer their idling computer or brainpower to classify large data sets in the citizen science projects [71], their motivation to participate is primarily intrinsic [15, 43]. However, as motivation to participate varies for different projects, the imposition of monetary incentives can have different effects. Unlike traditional citizen science, OSS development is an open innovation activity requiring deep involvement and a great deal of experience, so the motivation and incentives for participation may vary considerably.

2.2 Crowdsourcing

Acting as a type of online activity, participants will receive the satisfaction of a given kind of need, be it economic, social recognition, self-esteem, or the development of individual skills [16]. Hossain [34] classified the motivators into extrinsic and intrinsic motivators, where extrinsic motivators include financial motivators

(*e.g., cash*), social motivators (*e.g., peer recognition*), and organizational motivators (*e.g., career development*). Intrinsic motivators are directly related to participants' satisfaction with the task (*e.g., enjoyment, fun*). Considering the related incentives, Liang et al. [45] highlighted that both intrinsic and extrinsic incentives could increase the effort of participation; however, extrinsic incentives weaken the impact of intrinsic motivation. By comparing paid and unpaid tasks, Mao et al. [47] concluded that monetary incentives make the task processing speed faster, but the quality is reduced. Based on this, Feyisetan et al. [18] improved the paid microtasks more engaging by including sociality features or other game elements. MTurk is a typical and popular crowdsourcing platform based on financial incentives and gamification, where participants are recruited, paid, and rated for their participation in microtasks, which ensure speed and quality at the same time [10]. Unlike MTurk, the contribution to Wikipedia is not incentivized by monetary rewards. Content contribution is more driven by reciprocity, self-development, while community participation relies on altruism, self-belonging, etc [73].

As can be seen from the related works above, there are many situations of crowdsourcing and different forms of motivation and incentive. However, unlike OSS development, traditional crowdsourcing tasks are mostly micro-tasks, which are relatively simple and require less time. Moreover, there is a clear distinction between the roles, *i.e.,* core developers and external contributors for OSS contributors. Contribution types include code contribution, code review, repository maintenance, management, etc.

2.3 Open source software development

Successful OSS initiatives can effectively change the method of software development [30, 39], improve software development efficiency [31, 60], and ensure software quality through effective management [1, 58]. Many projects have emerged along with the increasing number of users participating in the development of the OSS community [28]. In this context, many companies are involved in contributing to open source projects [32]. However, they have limited control and influence in day-to-day OSS work and decision processes [35], and OSS still relies on the voluntary participation of crowd labor [17].

Many studies have focused on analyzing individuals' motivations and the incentives for participating in OSS projects [14, 20, 33, 42, 59, 72]. Von Krogh et al. [68] classified contributors' motivations into three categories, namely, intrinsic motivation (*e.g.*, ideology and fun), internalized extrinsic motivation (*e.g.*, reputation and own use), and extrinsic motivation (*e.g.*, career and pay). Among developers who volunteer to contribute to open source projects, their motivation is mainly intrinsic or internalized extrinsic motivation [68]. They have full-time jobs and spend some spare time making open source contributions [21]. However, Hars et al. [3] found that being paid can promote continuous contribution from developers with all types of motivation.

Currently, there are many ways to obtain financial support for open source initiatives, *e.g.*, through donations or bounties. Many studies have focused on the characteristics, impact, or effectiveness of each form of financial support. For example, regarding bounties, Zhou et al. [77] studied the relation between issue resolution and bounty usage and found that adding bounties would increase the

likelihood of issue resolution. Acting as a way for recruiting developers, setting bounties attracts those developers who want to make money through open source contributions, which facilitate the completion of complex tasks. However, unlike bounty, the donation is a way of passively obtaining financial support. Regarding open source donation, Krishnamurthy et al. [40] studied the donation to the OSS platform and found the relation between donation level and platform association length and relational commitment. For the donation to OSS, Nakasai et al. [50, 51] analyzed the incentives of individual donors and found that the benefits for donors and software release could promote donations. In contrast, bugs in software will negatively affect the number of donations. However, they only focused on eclipse projects. Overney et al. [53] studied the impact of donations from a broader perspective of open source projects on GitHub, which corresponds to NPM packages and explicitly mentions the way of donation in the README.md files. They found that only a small fraction (mainly active projects) asked for donations, and the number of received donations was mainly associated with project age. Most donations are requested and eventually used for engineering activities. However, there was a slight influence of donation on project activities. Although Overney et al. did a thorough analysis of project-level donation, there lacks analysis of donation towards open source developers. Also, we think adding the qualitative analysis from the users' perspective can confirm the quantitative findings and help understand the pros and cons of system design and use.

3 BACKGROUND

3.1 Terminology

To help the reader understand the rest of the article, we introduce key terms related to the Sponsor mechanism.

Sponsor: **an entity who provides donations to others**.

Maintainer: **an entity who can be sponsored (developers who set up a Sponsor profile)**.

Nonmaintainer: **an entity who has not set up the *Sponsors***.

Sponsorship: **the donation relationship between a *sponsor* and a *maintainer***.

AccountSetUpTime: **the time when *maintainers* set up the Sponsor profile for their accounts**.

FirstSponsorTime: **the time when *maintainers* receive their first *sponsorship***.

3.2 Introduction of the Sponsor mechanism

Currently, in GitHub, the workflow and key elements of *sponsorship* are shown in Figure 1, where the *sponsorship* is constructed on the *maintainer*'s sponsor page by clicking the "select" button of specific amount. The sponsor page is preset by the *maintainer* when setting up a Sponsor profile in the related GitHub account, which mainly consists of the following elements.

- Personal description: *maintainers* are free to add text and modify it at any time. The main content can cover basic personal information, project information, why they need to be sponsored, other ways of donation, etc.
- Preset goal: *maintainers* are allowed to set the number of *sponsors* or *sponsorships* that they want to get from the Sponsor mechanism and add related descriptions about the goal.
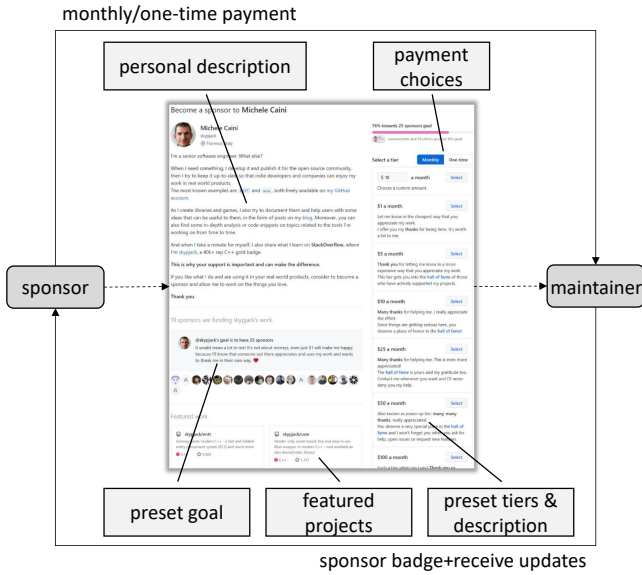
Figure 1: The workflow and key elements of *sponsorship*

- Featured projects: this part lists the related projects that the *maintainer* currently works on or with the most popularity.
- Preset tiers & description: this part contains the tiers set by the *maintainer*. *Sponsors* can choose which tier to pay according to the amount and the related description.
- Payment choices: *sponsors* can choose to monthly or one-time customized payment.

After choosing the way to construct the *sponsorship*, *sponsors* can get the sponsor badge and receive updates from the sponsored *maintainer* in the future.

3.3 Preliminary analysis

We conduct a statistical analysis of the use trends of the SPONSOR mechanism (Figure 2 shows the number of developers who set up the SPONSOR account and how the number of sponsorships changes over time). We can see that the number of developers who set up
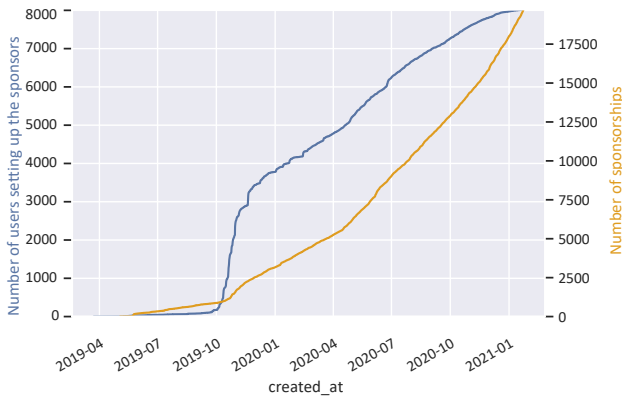

Figure 2: Cumulative participation over time

an account increased sharply around October 2019 (new things inspire people's interest). At other times, the growth rate shows a downward trend. Meanwhile, the absolute number of participants in this mechanism increased steadily, although the growth rate shows a slight upward trend. Compared to GitHub itself, which has shown a strong increase in its user base [74], the SPONSOR mechanism has not attracted as much attention. In this context, we formulate **RQ1: Why do individuals participate (or not) in the SPONSOR mechanism?**

According to our manual observation of GitHub developers' sponsorship pages, we find that developers can spend more time on their open source work if sponsored by others (with examples of this trend being Tim Condon [64] and Super Diana [61]). In short, we consider how the SPONSOR mechanism may affect developers' open source activities. In this context, we ask **RQ2: How effective is sponsorship in motivating developer OSS activity?**

There are some very successful cases of individuals receiving support under the GitHub SPONSOR mechanism (*e.g.,* Caleb Porzio, who was sponsored by 1,314 sponsors as of 7 August 2021 [1]). However, most SPONSOR participants have not been successful, and many have not received any sponsorships at all. According to Figure 3, only 14.1% of *maintainers* are sponsored at least once. Most people do not receive any sponsorships, despite setting up a SPONSOR account. Among *sponsors*, most (76.3%) sponsor others just one time. Based on the statistical analysis results, we consider which
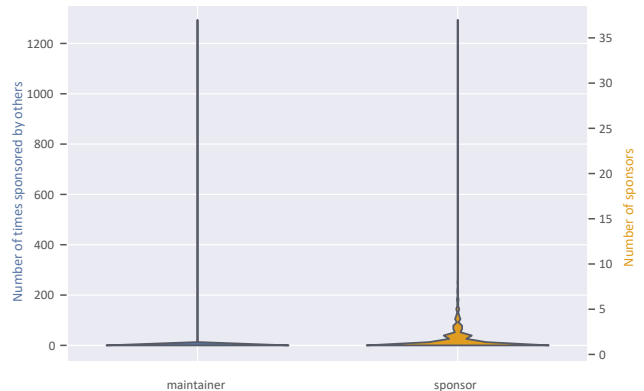
Figure 3: Differences between users in participation

developer characteristics lead to more sponsorships. In this vein, we ask **RQ3: Who is likely to receive more sponsorships?**

Currently, there are many ways to obtain financial support for open source initiatives, *e.g.,* through donations or bounties. The different types of financial support each have advantages and disadvantages [49]. It falls to participants (especially those who have participated in multiple financial support mechanisms) to judge the reasonableness and effectiveness of each. To better understand users' perceptions of the SPONSOR mechanism and thus enrich and improve it, we propose **RQ4: What are the shortcomings of the SPONSOR mechanism?**

---

[1] https://github.com/sponsors/calebporzio

4 STUDY OVERVIEW

4.1 Overall research methodology

The overall framework of this paper is shown in Figure 4, with the research methodology consisting of two main parts: data collection and research methods.

4.1.1 Data collection. The data is collected using GitHub API. The goal was to find different kinds of GitHub users (*maintainers*, *sponsors*, and *nonmaintainers*) and gather their related basic information and activities. Here, we focus on how to distinguish different kinds of users. The acquisition of relevant basic information and details on activities is described in the subsequent section (see Section 4.2) when we introduce each research method in detail. We acquired different types of users through the following steps.

(1) We used the RESTful API [27] to obtain all users. After that, we queried *maintainers* using the field hasSponsorsListing of the GraphQL API [26]. We obtained 60,732,250 users who had not deleted their accounts, among which 7,992 users were individual *maintainers*.

(2) We used the field sponsorshipsAsMaintainer of the GraphQL API [26] to look up all the sponsorships that *maintainers* had received and the corresponding *sponsors*.

(3) Using the list of *sponsors* queried in step (2), we used the field sponsorshipsAsSponsor of the GraphQL API [26] to query all the related *maintainers*. This step was to supplement the information on the *maintainers* who had set up the SPONSOR profiles identified during the query process in step (1).

(4) We repeated steps (2) and (3) until no new *maintainers* or *sponsors* appeared.

Through the above steps, we obtained 20,579 users, among which 8,028 are *maintainers*, 13,555 are *sponsors* (1,004 users are maintainers while sponsoring others at the same time). We also get 22,515 times of sponsorships. All users except maintainers were marked as *nonmaintainers*.

4.1.2 Research methods. To answer the research questions, we used a combination of quantitative and qualitative analysis. Regarding our **why (RQ1)** and **what (RQ4)** questions, since it was difficult to capture everyone's reasons for participation or nonparticipation and summarize the shortcomings of the mechanism based on just the platform information, we asked relevant people to complete a questionnaire. For the **how (RQ2)** and **who (RQ3)** questions, we collected maintainer-related data, quantitatively analyzed the impact of sponsorship behavior on *maintainer* open source activity, and explored the correlation between factors and the amount of sponsorship. On this basis, we again conducted a qualitative analysis using a questionnaire. This combination of quantitative and qualitative analysis led to our conclusions. Next, we describe each research method in detail.

4.2 Detailed introduction of research methods

4.2.1 Questionnaire. Since there are three types of interaction between the user and the SPONSOR mechanism, namely, interactions with a *sponsor*, a *maintainer*, or a *nonmaintainer* (see Section 3.1), we designed three different online surveys [75]. The surveys for both *sponsors* and *maintainers* relate to their expectations for and

satisfaction with the SPONSOR mechanism. The survey for *nonmaintainers* relates to their reason for not setting up the SPONSOR feature for their account. All the surveys start with an introduction to the research background and purpose. There are two types of questions in each survey.

- Demographic questions designed to obtain participants' information, including their role in and experience with OSS development (the predefined answers were inspired by prior research [44]).
- Main questions, designed to gather users' views on the SPONSOR mechanism.

Among the main questions, there are three kinds.

- Open-ended questions aimed at gathering answers.
- Rating scale questions soliciting users' satisfaction and agreement levels.
- Multiple-choice questions with "Other" text field options aimed at gathering large-scale user feedback while providing additional answers.

We provide a final, open-ended question to allow participants to talk freely about the SPONSOR mechanism. We discussed the questions with software engineering researchers to ensure that the items were well designed for our study and clear enough for participants to answer. Finally, we used SurveyMonkey [62] to deploy our online surveys.

There were two rounds of each survey: 1) the *pilot stage*, aimed at gathering answers to the open-ended questions from a limited number of participants, and 2) the *full-scale stage*, aimed at gathering the votes for each answer from a larger population. The statistics on the two stages can be seen in Table 1.

Table 1: Statistics on the two-stage survey

Stage	Statistic items	Maintainers	Sponsors	Nonmaintainers
Pilot	#selected participants	400	400	400
	#successful invitations	394	388	390
	#response (%)	45 (11.4%)	24 (6.2%)	9 (2.3%)
	Date for collection	June 8, 2021 - June 15, 2021		
Full-scale	#selected participants	6,104	6,359	7,500
	#successful invitations	5,951	6,224	7,343
	#response (%)	467 (7.8%)	396 (6.4%)	202 (2.8%)
	Date for collection	June 29, 2021 - July 13, 2021		

means the number, *e.g.*, #response implies the number of responses

Participant recruitment. To recruit participants for the two rounds of three different surveys, we took the following steps:

(1) For all three types of users (*maintainers*, *sponsors*, *nonmaintainers*), we filtered out those whose email or name information could not be openly accessed, as these users might not want to receive questionnaires.

(2) For all three types of users, we filtered out those who had not been active in the last month (since May 3, 2021), as they might not have focused on open source work on GitHub in recent days. In this step, we used the GitHub API to obtain users' recent activity, including the top repositories to which they had contributed in the last month and their last update time (field "updatedAt") on GitHub [26].
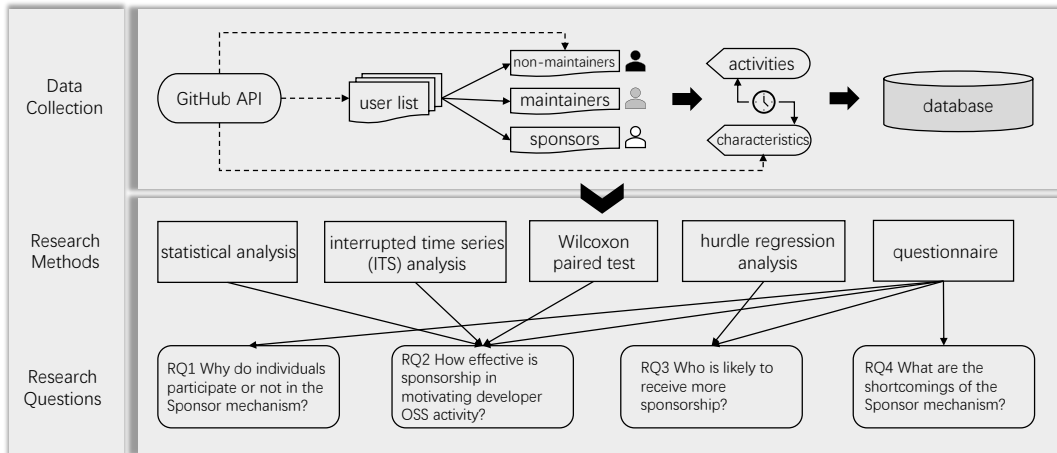
Figure 4: Framework of this paper

(3) For *nonmaintainers*, we selected only users who may be eligible to set up a Sponsor profile based on their location information and the list of countries or regions included under the GitHub Sponsor mechanism [25].

(4) After completing the above three steps, we randomly selected 400 unique individuals of each type without overlap as participants in the *pilot stage*.

(5) For the *full-scale stage*, we selected all other *maintainers* (6,104) and *sponsors* (6,359) as participants. For *nonmaintainers*, due to the low response rate in the *pilot stage*, we filtered users according to the total number of stars of projects owned by developers (collected on 23 June 2021). We selected those with at least ten stars (we assumed that developers with popular projects are more likely to be interested in the Sponsor mechanism and use GitHub very often). After that, we randomly selected 7,500 participants.

Response and analysis. After selecting the participants, we published the questionnaire online and sent the web address to participants via email. The email invitation contained the basic information of the questionnaire publisher, the reason for the release, the number of questions, and the estimated time required to fill out the questionnaire.

Based on the participants' feedback of the *pilot* stage, we designed the questionnaires for the *full-scale stage*. We removed 1 question for *maintainers*, 1 question for *sponsors*, and 2 questions for *nonmaintainers* due to answers with repetitive content in relation to the answers to other questions. We extracted the essential information from all responses and turned some open questions into multiple-choice questions (3 for *maintainers*, 3 for *sponsors*, and 1 for *nonmaintainers*) through open coding of card sorting method [78] by the first, second and the fifth authors together. To avoid disturbing the participants, we extended the time to collect the responses in this stage relative to that in the *pilot* stage but did not send a second email reminder. At the same time, because different types of participants dedicate different amounts of attention to the Sponsor mechanism, the response rate varies greatly.

Nonmaintainers, who do not participate in the Sponsor mechanism, may not care about it and not want to reply to the email.

When analyzing the multiple-choice questions, we first calculated the voting rate for each preset option. After that, we manually included the textual response for the "Other" option into the preset taxonomy, if possible, via the closed coding method [78]. If a new topic emerged, we integrated it into the existing taxonomy. When analyzing the last open question ("Do you have anything else to tell us about the Sponsor mechanism?"), we extracted the essential information from the textual response for qualitative analysis. To facilitate analysis, we use *[MCx]*, *[SCx]*, and *[OCx]* to represent the textual response in the questionnaire for *maintainers*, *sponsors*, and *nonmaintainers*, respectively, where *x* indicates the serial number of the comment.

Through the first two questions of each questionnaire, we collected participants' demographic information, including their status and experience with open source development. For the *full-scale stage*, the results are shown in Table 2. More than 70% of participants in each category have more than three years of OSS development experience. More than 10% of *sponsors* have no OSS development experience, which indicates that many *sponsors* sponsor others solely to support OSS development or maintenance.

Table 2: Demographic information of participants in the *full-scale stage*

Questions	Answers	M (%)	S (%)	NM (%)
Q1: How would you best describe yourself?	Developer working in industry	62.3	80.0	65.5
	Full time independent developer	16.6	10.0	8.0
	Student	11.6	6.9	6.5
	Academic researcher	3.7	3.6	16.0
Q2: How many years of OSS development experience do you have?	Never	1.1	10.2	3.0
	<1 year	2.2	4.6	6.5
	1-3 years	10.1	14.5	12.6
	3-5 years	21.9	22.6	23.1
	5-10 years	33.6	26.9	27.1
	>10 years	31.2	21.3	27.6

M: *maintainer*; S: *sponsor*; NM: *nonmaintainer*

4.2.2   ITS analysis. The aim of this analysis was to determine when to treat sponsorship as an *intervention* and how it influences the potential trends in *maintainers'* activities (development and discussion activities) from a long-term perspective. Therefore, following the guidelines of previous studies [53, 65, 76], we used the ITS method. The settings of the ITS analysis are shown below.

Interventions: We set both *accountSetUpTime* and *firstSponsorTime* (see Section 3.1) as separate interventions. We assumed that *maintainers* may increase their activity after *accountSetUpTime* to attract others' attention for future sponsorship or be motivated to increase their open source contributions after *firstSponsorTime*.

Responses: We set the *number of commits* (development activity) and the *number of discussions* (discussion activity) as responses, as they indicate different kinds of activities on GitHub.

Unstable period: Similar to previous studies [53, 65, 76], we set 15 days before and after interventions as the unstable period.

Before & after intervention periods: To retain enough analyzable data, we selected *maintainers* with at least six months of activity before and after interventions in addition to the unstable period. Therefore, each *maintainer* has at least $15 * 2 + 6 * 2 * 30 = 390$ days of activity on GitHub.

Time window: Each month in *before & after intervention periods* is a time window, and the unstable period is also a time window. Therefore, there are $6 * 12 + 1 = 13$ time windows in all.

The independent variables are as follows.

Basic items.

- *intervention*: Binary variable indicating an intervention
- *time*: Continuous variable indicating the time by month from the start of an observation to each time window, with a value range of [0, 12]
- *time after intervention*: Continuous variable indicating how many months have passed after an intervention (if there is no intervention, *time after intervention=0*; otherwise, *time after intervention=time-6*).

Developer characteristics.

- *number of stars before*: Continuous variable, measured as the total number of stars of *maintainer*-owned repositories before the start of each time window
- *in company*: Binary variable indicating whether company information exists at data collection time
- *has goal*: Binary variable indicating whether a *maintainer* sets a goal for sponsorship at data collection time
- *has another way*: Binary variable indicating whether a *maintainer* sets other methods for receiving donations at data collection time
- *is hireable*: Binary variable indicating whether a *maintainer* declares a hireable status at data collection time

Developer activities.

- *number of commits before*: Continuous variable measured as the number of commits before the start of each time window
- *number of discussions before*: Continuous variable measured as the number of discussions before the start of each time window

We built a mixed effect linear regression model for ITS analysis with a *maintainer* identifier as the random effect and all the measured factors as fixed effects. A major advantage of the mixed effect model is that it can eliminate the correlated observations within a subject [19]. Here, the time windows for the same *maintainer* tend to have a similar trend. We used the *lmer* function of the *lmerTest* package in R [41] to fit models for the *maintainer's* commit and discussion activities. For better model performance, we transformed the continuous variables to make them approximately normal and on a comparable scale through log-transformation (plus 0.5) and standardization (mean 0, standard deviation 1) [56]. To reduce the multicollinearity problem, we excluded factors with variance inflation factor (VIF) values ≥ 5 using the *vif* function of the *car* package in R [11]. We report the coefficients and the related *p* values obtained in this way. We also report the explained variance of the factor, which can be interpreted as the effect size relative to the total variance explained by all the factors. For the fitness of models, we report both marginal (R_m^2) and conditional (R_c^2) R-squared values using the *r.squaredGLMM* function of the *MuMIn* package in R [7].

Together with ITS analysis, we visually present how *responses* change over time to show the activity change more intuitively (statistical analysis). Since there is an *unstable period* in the ITS analysis, we analyze this period separately using the *Wilcoxon paired test* method, which is presented in the following section.

4.2.3   Wilcoxon paired test. For the ITS analysis, the unstable period is ignored. However, the SPONSOR mechanism involves a small amount of money, which may influence *maintainer* behavior in the short term only. We assume that *maintainers* may have great fluctuations in OSS activity during the unstable period. We used a paired, nonparametric test method called the Wilcoxon paired test [8]. Through two-sided tests (both *alternative=greater* and *alternative=less*) [12], we can see whether the intervention increases or decreases a *maintainer's* activity. We considered three kinds of interventions, including *accountSetUpTime*, *firstSponsorTime*, and before and after each sponsorship. We used Cliff's delta (δ) to measure the effect size [29], with $|\delta| < 0.147$ indicating a negligible effect size, $0.147 \leq |\delta| < 0.33$ indicating a small effect size, $0.33 \leq |\delta| < 0.474$ indicating a medium effect size, and $|\delta| \geq 0.474$ indicating a large effect size.

4.2.4   Hurdle regression analysis. The critical idea of hurdle regression is to create a dataset with *maintainer* characteristics and the amount of sponsorship established. Therefore, we collected different characteristics of each *maintainer* heuristically, including *basic information*, *social characteristics*, SPONSOR *mechanism characteristics*, *developer activities*, and *project characteristics*. For the amount of sponsorship, we used the number of times that a *maintainer* is sponsored. Next, we present detailed descriptions of the collected variables.

Developer basic information.

- *user age*: Continuous variable measured as the time interval by month since the creation of the user account in the GitHub community until the data collection time
- *in company*: Binary variable indicating whether a *maintainer* introduces the personal work situation in detail

- *has email*: Binary variable indicating whether a *maintainer* publicly provides the contact information
- *has location*: Binary variable indicating whether the *maintainer* discloses the geographical location information
- *is hireable*: Binary variable indicating whether a *maintainer* indicates availability for hire

Social characteristics.

- *followers*: Continuous variable measured as the number of followers
- *followings*: Continuous variable indicating how many users the *maintainer* follows

Sᴘᴏɴsᴏʀ *mechanism characteristics.*

- *min tier*: Continuous variable measured as the minimum number of dollars set by the *maintainer* for donations
- *max tier*: Continuous variable indicating the maximum donation
- *has goal*: Binary variable indicating whether a *maintainer* sets a goal for sponsorship
- *has another way*: Binary variable indicating whether a *maintainer* introduces other modes for receiving donations. Here, we identified other donation modes by finding links to other funding platforms in the description on the sponsorship page. Other platforms are shown in Table 9, which was compiled according to the collection by Overney et al. [53] and the supported external links of GitHub [24]
- *introduction richness*: Continuous variable measured as the length of the introduction on the personal sponsorship page
- *user age after sponsor account*: Continuous variable indicating the time interval by month (to see how time influences the amount of sponsorship)

Activity characteristics.

- *number of commits*: Continuous variable measured as the total number of commits in GitHub from *accountSetUpTime* until the data collection time
- *number of discussions*: Continuous variable measured as the number of comments, including issue comments, pull request comments, and commit comments from *accountSetUpTime* until the data collection time

Project characteristics.

- *sum star number*: Continuous variable measured as the total number of stars of repositories created by a *maintainer*
- *sum fork number*: Continuous variable indicating the number of forks
- *sum watch number*: Continuous variable indicating the number of watchers
- *sum top repository star number*: Continuous variable measured as the total number of stars of top repositories that a *maintainer* contributed in the four months before data collection [23]
- *number of dependents*: Continuous variable measured as the number of repositories that rely on the project with the most watchers among all projects owned by the *maintainer*

When building the hurdle regression models, we removed *maintainers* with less than 3 months of activity after *accountSetUpTime*

to reduce the impact of time on sponsorship. We reasoned that *sponsors* need time to find *maintainers* to donate to. To reduce the zero-inflation in the response variance, we used hurdle regression [36] by splitting the sample into two parts:

(1) *maintainers* who have not received any donations from others, to examine which factors influence whether a *maintainer* receives donations
(2) *maintainers* with at least 1 sponsorship, to examine how the amount of received donations is influenced by the aforementioned characteristics

For the reduction of the multicollinearity problem and the report of results, we use the same methods (see Section 4.2.2).

5 RESULTS

5.1 RQ1: *Why* do individuals participate or not in the Sᴘᴏɴsᴏʀ mechanism?

For this research question, the questionnaire had a dedicated item for each of the three types of participants, *i.e.,* **Q3** for *maintainers*, *sponsors*, and *nonmaintainers*. Table A shows the motivations or reasons elaborated by different types of developers in the *full-scale stage* and the percentage of votes for each option.

5.1.1 Related motivations. From the results, we find that some of the motivations of *maintainers* and *sponsors* are related.

Project use relationship. For $\mathcal{RM}1$ and $\mathcal{RS}1$, they all indicate that the usage of related projects leads to sponsorship. Some 64.9% of *maintainers* and 85.8% of *sponsors* cite this factor as one motivation for participating in the Sᴘᴏɴsᴏʀ mechanism; this consensus puts it in first place on both groups' motivation lists. People think that users should give back to contributors in various ways, among which the Sᴘᴏɴsᴏʀ mechanism serves as a *"nice way to say thanks"*[MC23] and *"allow people to easily fund their projects."* [MC20]. From the perspective of *sponsors*, developers are grateful for the OSS that they use and hope to express their gratitude and, *e.g., "show support for OSS, which I heavily rely on in my daily work. Without OSS, I could not have built a career in data science"* [SC3].

Promotion of continuous OSS contributions. $\mathcal{RM}2$ and $\mathcal{RS}2$ reflect participants' uniform motivation to engage in further OSS contributions. Some 63.1% and 78.4% of *maintainers* and *sponsors*, respectively, cite this factor as a motivation; this factor thus ranks 2nd among all the enumerated reasons for participation. For open source developers, if they want to devote themselves to open source projects, they need to solve the problem of daily costs and open source maintenance costs (*e.g., "I believe in the open source and good-for-humanity idea. I need to get paid only to live a decent life"* [MC37]). Therefore, the emergence of the Sᴘᴏɴsᴏʀ mechanism may help them solve the above problems to a certain extent and then invest more time in open source projects (*e.g., "I was really hoping to get sponsorship so I could spend more time focusing on developing open source projects"* [MC11]). For *sponsors*, they also hope to inspire contributors to continue to make outstanding contributions (*e.g., "motivate them to do the awesome work"* [SC5]).

Recognition of OSS work. For $\mathcal{RM}4$ and $\mathcal{RS}3$, they all indicate *sponsors'* recognition of *maintainers*. A total of 39.9% of *maintainers* and 49% of *sponsors* cite this factor as a motivation for participation;

Table 3: Reasons for participating or not participating in the Sponsor mechanism

Reason_maintainers	Votes (%)	Reason_sponsors	Votes (%)	Reason_non-maintainers	Votes (%)
$\mathcal{RM}$1 It allows users of my projects to express thanks/appreciation	64.9	$\mathcal{RS}$1 Because I benefit from the developer's projects	85.8	$\mathcal{RO}$1 No need to be sponsored	39.3
$\mathcal{RM}$2 Sponsorship can motivate my future OSS contribution	63.1	$\mathcal{RS}$2 To encourage the developer to continue the contribution	78.4	$\mathcal{RO}$2 I contribute to OSS not for money	38.3
$\mathcal{RM}$3 Side income for OSS contribution	60.6	$\mathcal{RS}$3 To show my recognition of the developer's work	69.5	$\mathcal{RO}$3 My work is not worth being sponsored	28.4
$\mathcal{RM}$4 It can reflect community recognition for my work	39.9	$\mathcal{RS}$4 Because I'm interested in the developer's projects	49.0	$\mathcal{RO}$4 Never heard of it	26.4
$\mathcal{RM}$5 Just for fun	28.9	$\mathcal{RS}$5 To motivate the developer to work harder on a specific feature	9.4	$\mathcal{RO}$5 It's cumbersome	8.5
$\mathcal{RM}$6 I deserve to be rewarded for my past OSS contribution	21.8	$\mathcal{RS}$6 Because I know the developer	8.9	$\mathcal{RO}$6 Not available in my region	2.0
$\mathcal{RM}$7 I am able to prioritize the requirements of sponsors (e.g., fixing bugs)	18.8	Other	1.0	Other	10.4
$\mathcal{RM}$8 It's a way for me to make a living	13.1				
Other	1.9				

this motivation ranks 4th and 3rd for these two groups, respectively. For some people, sponsorship is a manifestation of greater recognition by *sponsors* than income.

Support for specific features. For $\mathcal{RM}$7 and $\mathcal{RS}$5, 18.8% of *maintainers* and 9.4% of *sponsors* hope that the Sponsor mechanism can help set the agenda for issue resolution priorities, although many people think that OSS should not be related to money (*e.g., "If there was money given by others involved, I would feel pressed to implement whatever they want (like in industry projects). I want FLOSS to be completely independent from corporate requests"* [OC5]).

5.1.2 Motivation across different user types. In addition to the motivations mentioned above related to the *sponsor* and *maintainer* relationship, there are other motivations or reasons related to the kinds of users.

Maintainers: More than 60% of these participants chose $\mathcal{RM}$3, but only 13% chose $\mathcal{RM}$8. In the **Other** option, 4 participants mentioned that they hope sponsorship can cover some of their infrastructure costs. Moreover, 28.9% of participants even chose $\mathcal{RM}$5 **Just for fun**. This indicates that different people have different expectations for the income. The Sponsor mechanism, as a donation mode, can bring maintainers a small amount of side income, but it is hardly a means for making a living (*e.g., "Sponsor is not an earning medium but a support mechanism"* [MC91]).

Sponsors: Some 49% of these respondents chose $\mathcal{RS}$4. They may not have used the *sponsors'* project but may be interested only in contributions or projects. We also find that only 8.9% of these participants chose $\mathcal{RS}$6, which indicates that the *sponsor* may not know the *maintainer* before the sponsorship. Therefore, social relationships are not particularly relevant to donations.

Nonmaintainers: Among developers who have not set up the Sponsor feature in their user account, we find that 38.3% chose $\mathcal{RO}$2. This is related to the results on users setting up their Sponsor profile. There have always been two perspectives in the open source spheres. Some people think that money is needed to maintain projects even if they are open source; others feel that open source projects should be free. In Table 2, we can see that the proportion of full-time independent developers is lowest among *nonmaintainers*. At the same time, we examined why full-time independent developers do not set up the Sponsor feature in their accounts. Only 3 out

of these 16 developers chose $\mathcal{RO}$2 (however, they all had successful products). This indicates that full-time independent developers recognize and hope to gain benefits through open source contributions, and the sponsorship mechanism can be a way to meet their needs. There are other reasons for not participating in the Sponsor mechanism. Some 39.3% of participants do not currently need to be sponsored ($\mathcal{RO}$1), 28.4% play down the value of their own work ($\mathcal{RO}$3), and 26.4% report that they have never heard of it ($\mathcal{RO}$4). Others complain about problems with the mechanism, including its cumbersomeness (8.5% – $\mathcal{RO}$5), unavailability (2% – $\mathcal{RO}$6), and tax problems (4% – one reason cited under the "Other" option).

The main reason cited for participation is to obtain or express appreciation for the use of open source projects or to recognize the *maintainer*'s OSS contribution. In turn, such support may promote better contributions. *Maintainers* seeking to make money tend to obtain extra income rather than a full livelihood through sponsorship. For *nonmaintainers*, in addition to personal reasons, the mixing of open source projects and money is another critical consideration preventing them from participating.

5.2 RQ2: *How* effective is sponsorship in motivating developer OSS activity?

We used the following methods for this research question: statistical analysis (visualization), ITS analysis, unstable period analysis based on the Wilcoxon paired test method, and qualitative analysis based on a questionnaire survey. We also explored the two kinds of interventions, namely, *accountSetUpTime* and *firstSponsorTime*.

5.2.1 Visualization. Figures 5-8 present the change in activities over time. We can see from the figures that both commit and discussion activities remain stable before and after the intervention. However, during the unstable period, developers tend to be more active than usual. In response to this phenomenon, we analyzed the persistent and transient effects of the interventions using the ITS method and Wilcoxon paired test method, respectively.

5.2.2 ITS analysis. Table 4 shows the results of the ITS analysis. The results show that the factor with the strongest correlation to OSS activity is the associated historical activity (*i.e., number of*
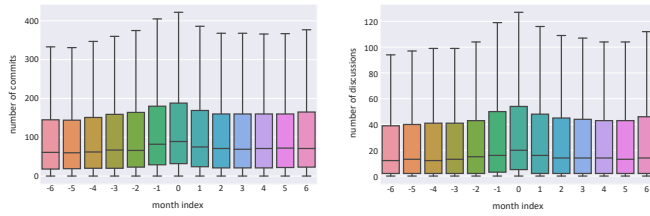
Figure 5: *Number of commits before and after accountSetUpTime*
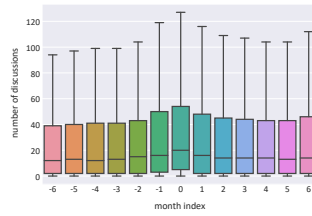
Figure 6: *Number of discussions before and after accountSetUpTime*

Figure 7: *Number of commits before and after firstSponsorTime*

Figure 8: *Number of discussions before and after firstSponsorTime*

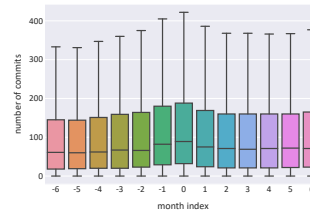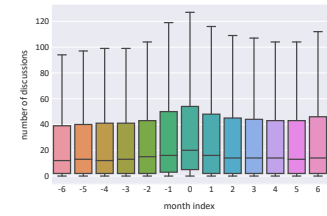*commits before* for **Commit Model**, *number of discussions before* for **Discussion Model**). For all four models, the associated historical activity explains more than 80% of the total variance. For the impact of other funding sources, we find that the variance explained by this factor does not exceed 1.1% in all four models. Therefore, it is somewhat clear that the existence of funding sources other than the SPONSOR mechanism does not influence our exploration of the association of this mechanism with open source activity.

For the *number of commits*, we find that for both *accountSetUpTime* and *firstSponsorTime*, there is a slight growth trend before the intervention. After the intervention, both show a negative growth trend ($\beta(time) + \beta(time\ after\ intervention) < 0$). Additionally, we find that the intervention itself is negatively correlated with the *number of commits* ($\beta(intervention) < 0$).

For the *number of discussions*, we find results similar to those for the commit activity. The intervention of the SPONSOR mechanism changes the original slowly increasing dynamics and reduces the discussion activity. Specifically, the intervention has no effect at *accountSetUpTime* but a slightly negative effect at *firstSponsorTime*.

In regard to the above results, it is surprising that the setup of the SPONSOR mechanism or the first sponsorship does not contribute to the *maintainer*'s commit activity or discussion activity growth. In contrast, there is a slight inhibitory effect. To illuminate this situation, we followed up with a questionnaire to explore the *maintainers'* subjective satisfaction with the SPONSOR mechanism and its motivating effect (see Section 5.2.4).

5.2.3 Wilcoxon paired test analysis. Table 5 shows the results of the Wilcoxon paired test and Cliff's delta.

For the *number of commits*, when the *maintainer* sets up the SPONSOR account, is sponsored for the first time, or receives a new sponsorship, the *number of commits* after the intervention is significantly higher. For the *number of discussions*, we find no significant changes around the three kinds of interventions.

This result indicates that sponsor behavior leads to a short-term increase in commit activity. For discussion, however, the sponsorship does not lead to short-term changes. In contrast to the ITS analysis, the Wilcoxon paired test analyzes changes in activity during the unstable period, further demonstrating that the SPONSOR mechanism can give a short-term boost to development activity.

5.2.4 Questionnaire survey. To further explore the effectiveness of the SPONSOR mechanism, we conducted independent research with *maintainers* and *sponsors* to uncover their subjective judgments about the efficacy of the mechanism. In response to this goal, we
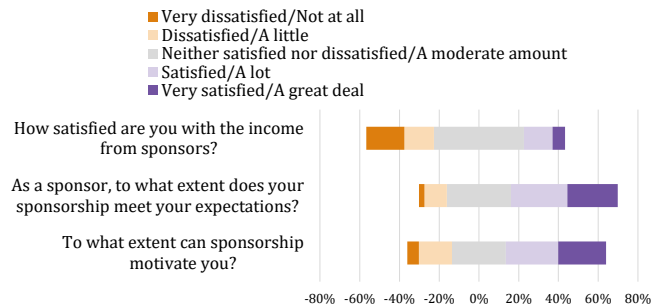
Figure 9: Results of 5-point Likert scale questions

asked maintainers (**Q4 "How satisfied are you with the income from sponsors?"**) and sponsors (**Q4 "As a sponsor, to what extent does your sponsorship meet your expectations?"**). Meanwhile, we asked the maintainers directly about their internal perceptions of the effectiveness of sponsorship incentives (**Q5 "To what extent can sponsorship motivate you?"**). The results are shown in Figure 9.

For *sponsors*, we find that 53.7% think that sponsorship meets their expectations fully or a great deal and only 14.1% report that their expectations are hardly met or not met at all. For *maintainers*, we find that 50.4% consider that sponsorship motivates them fully or a great deal but 22.5% think that it does not bring any motivating effect. However, in terms of the amount of sponsorship, we find that only 20.7% of *maintainers* are either satisfied or very satisfied with their income from sponsorship and 30.1% are dissatisfied or very dissatisfied with the amount.

We think that the main reason for this difference is that sponsors' main motivation to participate is to display their gratitude, inspire others, etc., by giving funds. Therefore, most *sponsors* are satisfied with their own behavior. For *maintainers*, although more than half think that sponsorship can be stimulating, we find that only approximately 20% are satisfied with the amount of sponsorship received. This shows that open source sponsorship has a positive effect on some developers, but in fact, the amount of monetary rewards that can be received through sponsorship is relatively small and unlikely to meet the expectations of *maintainers*.

In terms of short-term effects, the SPONSOR mechanism makes a slightly positive contribution to the development activity but has no significant impact on discussion activity. However, this impact is not sustained. One possible reason

Table 4: Results of ITS analysis

| | Commit Model Dependent variable: scale(log(number of commits + 0.5)) | | | | Discussion Model Dependent variable: scale(log(number of discussions + 0.5)) | | | |
| | *accountSetUpTime* | | *firstSponsorTime* | | *accountSetUpTime* | | *firstSponsorTime* | |
	Coeffs (Err.)	Chisq	Coeffs (Err.)	Chisq	Coeffs (Err.)	Chisq	Coeffs (Err.)	Chisq
(Intercept)	-0.10^{***} (0.01)		-0.13^{***} (0.03)		0.01 (0.01)		-0.02 (0.04)	
scale(log(*number of commits before* + 0.5))	0.59^{***} (0.01)	5190.72^{***}	0.58^{***} (0.02)	1185.38^{***}	0.03^{***} (0.01)	16.78^{***}	-0.02 (0.02)	1.14
scale(log(*number of discussions before* + 0.5))	$-0.02^{\cdot}$ (0.01)	3.45^{*}	-0.03 (0.02)	2.29	0.57^{***} (0.01)	4570.67^{***}	0.60^{***} (0.02)	1333.36^{***}
scale(log(*number of stars before* + 0.5))	-0.06^{***} (0.01)	55.23^{***}	-0.07^{***} (0.01)	22.71^{***}	-0.02^{**} (0.01)	6.41^{*}	-0.05^{***} (0.01)	11.11^{***}
has goal (TRUE)	0.06^{***} (0.01)	17.43^{***}	0.07^{*} (0.03)	5.97^{*}	0.01 (0.01)	1.02	0.03 (0.03)	1.21
has other way (TRUE)	0.16^{**} (0.05)	8.22^{**}	0.14 (0.09)	2.36	0.28^{***} (0.05)	26.17^{***}	0.14 (0.09)	2.33
in company (TRUE)	0.09^{***} (0.01)	38.56^{***}	0.11^{***} (0.03)	15.60^{***}	0.01 (0.01)	0.31	0.02 (0.03)	0.60
is hireable (TRUE)	0.00 (0.01)	0.02	0.01 (0.03)	0.22	-0.08^{***} (0.01)	30.34^{***}	-0.06^{*} (0.03)	4.41^{*}
time	0.02^{***} (0.00)	96.11^{***}	0.03^{***} (0.00)	61.22^{***}	0.01^{***} (0.00)	21.42^{***}	0.02^{***} (0.00)	21.80^{***}
intervention (TRUE)	-0.02^{*} (0.01)	5.66^{*}	-0.09^{***} (0.02)	25.54^{***}	0.01 (0.01)	0.30	-0.05^{**} (0.02)	6.71^{**}
time after intervention	-0.04^{***} (0.00)	245.92^{***}	-0.05^{***} (0.00)	97.38^{***}	-0.03^{***} (0.00)	111.52^{***}	-0.04^{***} (0.00)	63.73^{***}
Number of Observations	75,516		20,148		75,516		20,148	
R_m^2	0.34		0.32		0.37		0.35	
R_c^2	0.64		0.64		0.66		0.65	

$^{***}p < 0.001, ^{**}p < 0.01, ^{*}p < 0.05, ^{\cdot} < 0.1$

Table 5: Results of Wilcoxon paired test

| | | Commit | | | Discussion | | |
	Num obs	Wilcoxon(*greater*) p value	Wilcoxon(*less*) p value	Cliff's delta	Wilcoxon(*greater*) p value	Wilcoxon(*less*) p value	Cliff's delta
accountSetUpTime	7,969	$8e-05^{***}$	0.9999	0.022	0.6801	0.3199	-0.005
firstSponsorTime	2,796	0.0015^{**}	0.9985	0.025	0.7433	0.2567	-0.009
all sponsored time	21,153	$3e-12^{***}$	1.0000	0.021	0.9809	0.0191	-0.003

$^{***}p < 0.001, ^{**}p < 0.01, ^{*}p < 0.05, ^{\cdot} < 0.1$

is that the actual amount of support does not meet *maintainers'* expectations, which makes it difficult for *maintainers* to rely on sponsorship income to keep investing in open source contributions.

5.3 RQ3: *Who* is likely to receive more sponsorships?

For this research question, we tried to identify the important factors influencing the amount of sponsorship and provide further advice on *maintainers*. We again analyzed and verified the results through a combination of quantitative and qualitative analysis. For the qualitative analysis, we analyzed both *maintainers* and *sponsors* and explored the consistency of their perceptions of sponsorship.

5.3.1 Hurdle regression. From an overall perspective (see Table 6), the hurdle regression models fit well, with $R^2 = 34\%$ and $R^2 = 39\%$, respectively. Even though 7,465 *maintainers* have more than 3 months of activity after setting up their SPONSOR profile, only 2,750 (36.8%) of them receive at least one sponsorship. Moreover, only 6% receive sponsorships more than 10 times, and only 25 *maintainers* receive more than 100 sponsorships. Therefore, although many people want to obtain sponsorship, only a small number of people succeed.

When we consider whether the *maintainer* receives any sponsorships (columns 2 and 3 of Table 6), the *followers* factor, representing social status, has the most substantial positive effect, explaining 45.8% of the total variance. However, the factor *followings* is negatively correlated with the likelihood of receiving sponsorship (effect size: 3.1%). It is likely that compared to *followings*, *followers* better represents the centrality of *maintainers* in the community, while *maintainers* with large *followings* tend to learn more from others in the community. Discussion activity is positively correlated with the

likelihood of sponsorship (*number of discussions*, effect size: 22.7%), while relatively speaking, commit activity explains only 0.3% of the variance. A possible explanation is that sponsored developers tend to focus more on issues or pull requests submitted by sponsors to give back or attract the attention of others. Commit activity is common among GitHub developers, where many developers may just focus on their own issues. For sponsor tiers, the *min tier* is negatively correlated with the likelihood of sponsorship acquisition (effect size: 12.3%). However, *max tier* is positively correlated and explains 5% of the variance. Both of the tiers have sizable effects but opposite directions of influence. It is likely that many sponsors tend to donate only a little money and that setting a high *min tier* may cause them to abstain from sponsorship. However, if *maintainers* want to obtain sponsorships, they cannot undervalue themselves. Trying to increase the *max tier* can increase the possibility of being sponsored. Another thing for *maintainers* to note is the importance of the introduction text when setting up their SPONSOR account. If *maintainers* introduce themselves at greater length, they are more likely to become sponsored (effect size: 5.1%). Other factors have negligible effects, with explained variances of less than 5%.

When we consider the amount of sponsorship received by *maintainers* (columns 4 and 5 of Table 6), the social status of *maintainers* is also positively correlated with the response (*followers*, effect size: 65.3%). At the same time, *followings* oppositely correlates with the response (effect size: 10.7%). The factor *number of discussions* explains 9.6% of the total variance. The *min tier* variable becomes nonsignificant, unlike in the *receive sponsorship* model. A possible explanation for this result is that the setting of the *min tier* is not a long-term solution for securing more sponsorship. Developers need to be more focused on their status and daily activities in the community. Other factors have negligible effects.

Table 6: Result for factors influencing sponsorship

	Dependent variable: receive sponsorship		Dependent variable: the amount of sponsorship	
	Coeffs (Err.)	Chisq	Coeffs (Err.)	Chisq
(Intercept)	-0.53^{***} (0.09)		1.80^{***} (0.07)	
scale(log(*user age* + 0.5))	-0.10^{**} (0.03)	8.62^{**}	-0.02 (0.02)	0.41
in company (TRUE)	-0.26^{***} (0.06)	18.08^{***}	-0.12^{**} (0.04)	7.48^{**}
has email (TRUE)	-0.03 (0.06)	0.31	0.12^{**} (0.04)	7.53^{**}
has location (TRUE)	-0.11 (0.09)	1.41	-0.19^{**} (0.06)	8.36^{**}
is hireable (TRUE)	-0.19^{**} (0.06)	9.70^{**}	-0.07 (0.04)	2.28
scale(log(*followers* + 0.5))	0.96^{***} (0.04)	545.36^{***}	0.68^{***} (0.03)	722.89^{***}
scale(log(*followings* + 0.5))	-0.19^{***} (0.03)	37.39^{***}	-0.24^{***} (0.02)	118.53^{***}
scale(log(*min tier* + 0.5))	-0.42^{***} (0.04)	146.89^{***}	$-0.04^{\cdot}$ (0.02)	$3.39^{\cdot}$
scale(log(*max tier* + 0.5))	0.23^{***} (0.03)	59.82^{***}	0.14^{***} (0.02)	38.46^{***}
has goal (TRUE)	0.18^{**} (0.06)	8.32^{**}	0.01 (0.04)	0.10
has other way (TRUE)	0.28 (0.22)	1.54	0.44^{***} (0.13)	13.51^{***}
scale(log(*user age after sponsor account* + 0.5))	0.02 (0.03)	0.40	-0.02 (0.02)	0.81
scale(log(*number of commits* + 0.5))	$0.08^{\cdot}$ (0.04)	$3.42^{\cdot}$	0.10^{***} (0.03)	10.68^{**}
scale(log(*number of discussions* + 0.5))	0.73^{***} (0.05)	270.29^{***}	0.31^{***} (0.03)	106.73^{***}
scale(log(*sum star number* + 0.5))	-0.10^{**} (0.04)	7.48^{**}	-0.07^{**} (0.03)	6.87^{**}
scale(log(*sum top repository star number* + 0.5))	-0.13^{**} (0.04)	9.55^{**}	-0.15^{***} (0.03)	31.11^{***}
scale(log(*introduction richness* + 0.5))	0.23^{***} (0.03)	60.84^{***}	0.11^{***} (0.02)	25.74^{***}
scale(log(*number of dependents* + 0.5))	-0.02 (0.03)	0.62	$-0.03^{\cdot}$ (0.02)	2.69
Number of Observations		7,465		2,750
delta R^2		0.34		0.39

$^{***}p < 0.001, ^{**}p < 0.01, ^{*}p < 0.05, ^{\cdot} < 0.1$

5.3.2 Questionnaire. We asked questions related to *maintainers* (**Q6 "In which way do you think you can obtain more sponsorships?"**) and *sponsors* (**Q5 "What kind of developer do you prefer to sponsor?"**) separately. Table 7 presents the results.

For maintainers. The results reveal that from the *maintainers'* perspective, producing useful projects and tools ($\mathcal{WM}1, \mathcal{WM}4$) is seen as more likely to draw sponsorships than just participating in projects ($\mathcal{WM}5, \mathcal{WM}6, \mathcal{WM}7, \mathcal{WM}8, \mathcal{WM}9$). One possible reason for this is that the Sponsor mechanism is to credit funds to individual accounts, and the sponsorship button on the project homepage also needs to be configured by the owner. Some *sponsors* who want to donate to a project through the Sponsor mechanism (*e.g., those reporting that "I prefer to sponsor projects, not a specific developer" [SC167]*) may end up sponsoring only the project's owner.

Some 54.5% of *maintainers* think that by working hard, they can obtain more sponsorships ($\mathcal{WM}2$). However, some *maintainers* said sponsorship is simply a matter of popularity (*e.g., "Purely popularity basically... OSS Creators from YouTube earn a ton of money" [MC292]; "I think it is mostly a function of being a celebrity so it operates on the same rules" [MC262]*). This is probably why 54.1% of the *maintainers* chose $\mathcal{WM}3$.

More than 1 option was chosen by 85.6% of the sponsored participants. Moreover, 20.5% chose at least 5 options, which shows that in fact, the options that we offered are feasible for promoting sponsorships among *maintainers*. Some relevant participants indicated that *"Donations just don't work" [MC284] or "It doesn't matter; people take when it's free" [MC281]*. These responses suggest that the reasons that prevent most people from obtaining more sponsorships that would meet their expectations are not limited to individual participation characteristics and platform mechanism design; rather, the act of sponsorship itself may not be suitable for the open source sphere. Indeed, 10 participants who selected $\mathcal{WM}11$ indicated that there was no way to obtain more sponsorship.

For sponsors. The vast majority (85.1%) chose $\mathcal{WS}1$, which suggests that most *sponsors* support developers involved in the open source projects that sponsors use. This corresponds to the top-ranked way of obtaining sponsorship ($\mathcal{WM}1$) selected by the *maintainers*, suggesting that the best way to obtain more sponsorship, in the opinion of both *maintainers* and *sponsors*, is to create projects that more people use. Similarly, more than half of the participants wanted to sponsor projects of personal interest ($\mathcal{WS}2$) and developers who had made significant contributions ($\mathcal{WS}3$). We find that 31.1% of the *sponsors* chose to sponsor independent developers ($\mathcal{WS}5$). However, some *sponsors* said that just being an independent developer is not enough and that the development and maintenance of good open source projects or tools are needed (*e.g., "Independent developers with nice tools" [SC30]*).

Most *sponsors* do not consider the act of sponsorship as a form of charity—few people reported doing so simply because the person being rewarded was in hardship ($\mathcal{WS}7$) or had not received many rewards ($\mathcal{WS}6$). Likewise, *sponsors* do not want to reward another developer simply because they know one another (only 15.4% chose $\mathcal{WS}8$; *e.g., "It is usually a library I am using in my own project and I know the developer in person" [SC168]*).

Most maintainers and sponsors think that sponsorship builds on relationships forged through using OSS. Active and meaningful participation in open source contributions can also help maintainers gain more attention. However, the quantitative analysis reveals that the social popularity of the *maintainer* in the community is the decisive factor in obtaining more sponsorships.

5.4 RQ4: *What* are the shortcomings of the Sponsor mechanism?

For this research question, we investigated the mechanism shortcomings found by participants while using the Sponsor mechanism. We asked the question "**What are the shortcomings of the

Table 7: Ways of obtaining more sponsorship

Way_maintainers	Votes (%)		Who_sponsors	Votes (%)	
$\mathcal{WM}$1 Producing useful projects	62.6		$\mathcal{WS}$1 Developers whose projects I benefit from	85.1	
$\mathcal{WM}$2 Staying active and contributing more in the community	54.5		$\mathcal{WS}$2 Developers whose projects I'm interested in	60.3	
$\mathcal{WM}$3 Advertising myself or my work to the community	54.1		$\mathcal{WS}$3 Developers who make important contributions	50.9	
$\mathcal{WM}$4 Producing valuable code	38.5		$\mathcal{WS}$4 Developers who are active in community	42.0	
$\mathcal{WM}$5 Getting involved in popular projects	29.1		$\mathcal{WS}$5 Independent developers	31.1	
$\mathcal{WM}$6 Getting involved in projects adopted by companies	25.5		$\mathcal{WS}$6 Developers who haven't received much sponsorship	24.1	
$\mathcal{WM}$7 Getting involved in long-term projects	21.6		$\mathcal{WS}$7 Developers who are in hardship	18.7	
$\mathcal{WM}$8 Getting involved in less maintained yet important projects	19.1		$\mathcal{WS}$8 Developers who I know	15.4	
$\mathcal{WM}$9 Getting involved in projects led by companies	8.8		$\mathcal{WS}$9 Other	1.0	
$\mathcal{WM}$10 Providing localized content	7.4				
$\mathcal{WM}$11 Other	3.6				

Sponsor mechanism?" of both maintainers (**Q7**) and sponsors (**Q6**) separately. Table 8 presents the results.

Among maintainers, 13.1% thought that the Sponsor mechanism was perfect ($\mathcal{SM}$6) and could meet their personal needs well, while among sponsors, 33.1% thought that the mechanism was perfect ($\mathcal{SS}$2). This indicates that the satisfaction of different types of mechanism participants, especially maintainers, varies greatly. The current Sponsor mechanism does not meet maintainers' needs well. The shortcomings include the following main aspects (some of these were resolved by GitHub during the research process).

Discoverability of maintainers. The results reveal that 51.3% of maintainers found it difficult to be discovered by sponsors ($\mathcal{SM}$1); however, based on feedback from sponsors, only 19.6% found it difficult to determine whom they should sponsor ($\mathcal{SS}$3). A larger share (40.1%) found it difficult to assess who urgently needed sponsorship ($\mathcal{SS}$1).

Interactivity of participants. From the results, we find that among maintainers, 29.4% thought that the current Sponsor mechanism cannot support good direct communication with sponsors ($\mathcal{SM}$2)), while among sponsors, 11.8% wanted communication support ($\mathcal{SS}$5). Some thought that they should not burden developers by interrupting their normal development process (*"I don't want to burden the developers [by asking them] to communicate with sponsors. The sponsor should be string-free" [SC195]*).

Payments. Many people, including maintainers and sponsors, highlighted existing payment problems with the Sponsor mechanism, including limited payment options (25.1% of maintainers – $\mathcal{SM}$3)), limited sponsorship tiers, inconvenient tax payments (19.3% of the maintainers – $\mathcal{SM}$5)), and limited payment providers. Some of these shortcomings, *e.g.,* the limited payment options, may have been resolved by GitHub during the research process.

User distinction. A total of 20.7% ($\mathcal{SM}$4) of maintainers and 10.5% ($\mathcal{SS}$6) of sponsors mentioned the distinction between sponsors and others in project development activities.

Geographical restrictions. From $\mathcal{SM}$7 and $\mathcal{SS}$4, we see that 11% of maintainers and 13.2% of sponsors thought that support for regions limits the popularity of participation. As of 27 July 2021, only 37 regions were supported, leaving many people unable to participate in the mechanism ($\mathcal{RO}$6) and *sponsors* unable to sponsor as many people as they want (*e.g., "Not all organizations I want to support joined GitHub sponsors" [SC192]*).

Lack of contribution indicators. Five participants noted that there was a lack of valid OSS contribution indicators. OSS contributions are not limited to commits and pull requests. If not involved in the current project, the *sponsor* hardly knows who has played a significant role in the project development (*e.g., "It is not easy to measure my OSS contribution. Sometimes it is just filing issues; other times, it is documentation PRs" [MC350]*). Moreover, contributions of small patches to large projects are difficult for others to find and thus are unlikely to gain sponsorships (*e.g., "In my case, you will be hard-pressed to get anything for your work when you are making just a little addition to a massive piece of software" [MC379]*). Among sponsors, some want to sponsor a project, not individual maintainers (*e.g., "I prefer to sponsor projects, not a specific developer" [SC167]*).

OSS donations. The Sponsor mechanism itself is an act of donation. On GitHub, sponsorship is primarily for users or organizations that have created a GitHub account. We find from the results that 16 participants thought that the donation mechanism itself was not suitable for the current open source sphere. Many reasons were cited for this evaluation: People take open source projects for granted, and no one wants to pay for them (*e.g., "People still do not like to pay for software" [MC355]*). Companies that use open source initiatives to gain revenue do not want to give back to the open source project (*e.g., "Most companies don't fund any of their open source dependencies" [MC354]*). Donations are passive income, and without a regular income, developers have little motivation to work full-time on open source projects (*e.g., "Donation makes far less revenue than charging for things" [OC78]*).

To solve the problems mentioned above, we offer the following actionable suggestions after taking into account the participant feedback.

Discoverability of maintainers.

- Add "Sponsor" buttons for the relevant project or people on the release webpage (*"Recognition of sponsors in release of the repository would be something I can think of" [SC217]*).
- Add support for integrated development environments (IDEs), allowing developers to discover package dependencies and quickly jump to sponsor pages while developing with IDEs (*"Better discoverability and integration with other developer tooling" [SC65]*).
- Provide a more straightforward way to show personal OSS contributions (*e.g., "Promote efforts like a dashboard" [MC126]*).

Table 8: Shortcomings of the Sponsor mechanism

Shortcoming_maintainers	Votes (%)	Shortcoming_sponsors	Votes (%)
SM1 It's hard for others to discover me for sponsorship	51.3	SS1 I cannot assess how urgently a developer needs to be sponsored	40.1
SM2 I can't interact with my sponsors on GitHub (e.g., for expressing appreciation)	29.4	SS2 None. It's perfect	33.1
SM3 Lack of a wide range of payment options (e.g., one-time/yearly/quarterly payment)	25.1	SS3 It's hard for me to find the developer I should sponsor	19.6
SM4 GitHub does not distinctly mark my sponsors (e.g., I cannot easily tell whether an issue submitter is my sponsor)	20.7	SS4 It is not supported in many regions	13.2
SM5 I have to pay taxes	19.3	SS5 I can't interact with the developer I sponsored on GitHub	11.8
SM6 None. It's perfect to me	13.1	SS6 I'm not distinctly marked in the projects whose maintainers have been sponsored by me (e.g., when I submit an issue)	10.5
SM7 It is not supported in many regions	11.0	SS7 Other	8.1
SM8 I can't declare how I dealt with the received money	10.1		
SM9 Other	9.4		

During the research process, GitHub fixed some shortcomings, *e.g.,* the one-time payment method.

Interactivity among participants.

- Allow maintainers to configure themselves whether they wish to communicate directly with sponsors. The interaction can be set up in different groups for different sponsors, similar to Patreon's integration solution with Discord [54] (*e.g., "Lack of integration with the payment tiers like the Discord integration with Patreon" [MC337]*).
- Allow maintainers to configure their own thank-you emails that can be sent automatically when they receive a sponsorship (*e.g., "Some kind of thank-you setup where I can send notes, etc." [MC109]*).
- Allow sponsors to upload statements to disclose expenses related to sponsorship proceeds (*"Distribution of the money, especially in FOSS [free and open source software] projects" [MC88]*).

Payments.

- Provide clear income and expense statements to the *sponsor* and *maintainer* automatically.
- Integrate as many payment providers as possible on the basis of meeting tax requirements.

User distinctions.

- Let maintainers decide, through in a configurable form in their personal settings, whether they want to treat sponsors differently from nonsponsors.
- In addition to an option to show distinctions, add configuration options such as what development activities to show and whether to distinguish between sponsors with different sponsorship amounts (*e.g., "Developers should be allowed to set permission levels based on sponsorship. E.g., you can only comment or make requests if you're a sponsor (or if the developer directly opts you in, or if you've made contributions to the project, things like that). This would really positively change the culture of GitHub collaboration" [SC212]*).

Geographical restrictions. Provide support for more regions.

Lack of contribution indicators. Set up a multidimensional indicator of contributions, and ensure rational allocation of project sponsorship funds.

OSS donations. Future research should synthesize feedback from all types of open source participants and reconsider how to improve the sponsorship mechanism or design a more appropriate form of open source financial support.

The shortcomings of the Sponsor mechanism relate to three main aspects. *Usage deficiencies*: **difficulty of participants in finding each other, lack of good interaction support, lack of promotion, lack of adequate payment and billing support, etc.** *Object orientation with supported functions*: **despite support for organizations and projects, main targeting of individuals. For** *sponsors,* **a need for better support for corporate sponsorship; for** *maintainers,* **a need for better support for multicontributor projects.** *Personalization*: **a need for configurability of the Sponsor mechanism to reflect variation in participant types and motivations.**

6 DISCUSSION

Through this study of the integrated sponsorship mechanism on the world's most popular open source platform (GitHub), we found that participation in the mechanism has not shown the same rapid growth as participation in open source projects. Meanwhile, there is a long-tail effect regarding the number of sponsorships obtained by maintainers; *i.e.,* most maintainers do not obtain many sponsorships or even any at all. Compared to the work of Overney et al. [53], this research brings us one step closer to understanding the incentive effect of sponsorship on individual developers by collecting feedback from participants in open source donations, taking the GitHub Sponsor as an example.

Although this article considers only the Sponsor mechanism, it lacks overall consideration and comparative analysis of all open source sponsorship platforms. However, we think that the article still provides some guidance in helping improve the mechanism itself and exploring the essence of open source donation.

This paper explored four aspects of the Sponsor mechanism: its who, what, why, and how. The main findings and insights are as follows.

Why do individuals participate or not in the Sponsor mechanism? Not all open source contributors endorse open source donation. There were more nonparticipants than participants. Like the motivations for participation in traditional citizen science [15, 43] and information-sharing crowdsourcing systems like Wikipedia [73],

developers are primarily intrinsically motivated to participate in open source contributions [21]. However, because open source development activities are more complex and require significant maintenance, many contributors are looking for financial support [5, 57, 67]. Among the groups that support and use it, it is generally relationships built through the use of specific software that serve as the backbone of the sponsorship behavior. In fact, many users want to reflect the difference between sponsors and nonsponsors in development activities and, in this way, change the method of open source collaboration and participation in open source donation. Such a change might not be very pleasant and could lead to the open source sphere becoming money driven. We think that making the format personalized and configurable may meet the needs of more people without changing the nature of the open source sphere.

It is necessary for system designers to consider regional support and then make the Sponsor mechanism accessible and better for more people who want to participate by improving the user experience (*e.g.,* better access to bill for tax).

How effective is sponsorship in motivating developer OSS activity? In a study of donations to projects, Overney et al. [53] found that donation did not improve engineering activity. And in our study, we also found that sponsorship has only a short-term positive stimulating effect on *maintainers'* development activity. However, the impact does not last, and there is even a slight negative effect in the long term. A possible reason for this result is that most *maintainers* do not receive sufficient sponsorship through the Sponsor mechanism to be motivated to contribute continuously. This may reflect the characteristics of open source donations. The *maintainer* passively receives sponsorship from the sponsor, and there is no compulsion for the act of sponsorship to occur. Thus, situations may arise that are similar to that of one of our questionnaire participants, who created heavily used tools but received no sponsorships. When compared horizontally with the results of other maintainers, such an outcome may have the negative effect of dealing a blow to maintainers and reducing their enthusiasm for making open source contributions.

For system designers, it is important to consider how to design conjunctive mechanisms, such as adding a ranking list according to the number of received or given sponsorships in the annual report or other locations. Therefore, the sponsorship mechanism can become a more continuous driving force, enhancing the impact of the sponsorship on developer activities.

Who is more likely to receive sponsorships? Participants' subjective perceptions conflict with the actual phenomenon. Participants believe that creating useful open source projects should lead to more sponsorships. However, we find that the most significant factor influencing the amount of sponsorship is social status. This inconsistent finding illustrates that participants want to express their gratitude or receive appreciation from others through the software usage relationship. However, it is not the case that those who develop sufficiently useful tools receive substantive sponsorship. Given the feedback from participants in our questionnaire, this situation is likely to cause maintainers to complain about a lack of publicity for themselves or about the fact that their work leads to no more sponsorships. At the same time, developers who make minor

contributions to popular projects or outstanding contributions to niche projects may be ignored under this mechanism. Comparing to project-oriented donation, *e.g.,* open collective, patreon [53]. Although the Sponsor mechanism is targeted at developers, which allows external contributors who do not own but are actively involved in popular projects to get donations. However, it is found through the results that *sponsors* prefer project-oriented donation, *i.e.,* the core developers or owners of popular or used projects are more likely to receive sponsorship. Since some of the money donated to projects is spent on travel/food [53], we think it is needed to consider the percentage of contributors' contributions to achieve greater equity.

As for now, we think that for open source developers who want to get more sponsorship, it is essential to increase one's community visibility through advertising and help oneself get more attention by building open source projects that more people use.

What are the shortcomings of the Sponsor mechanism? The defects of the Sponsor mechanism are manifested in three main aspects: usage defects, object-oriented and support mechanisms, and personalization setting problems. At the same time, many developers believe that sponsorship behavior is not suitable for the open source ecosystem. The free nature of OSS leads to an unwillingness to pay. This finding shows that in addition to the problems with the mechanism itself, donations are not perfectly adapted to the open source ecosystem. The passivity, uncertainty, and instability inherent to donations make it difficult for *maintainers* to rely on them and continue to make open source contributions for a long time. At the same time, the lack of reasonable evaluations of contributions and funding allocation makes it difficult for *sponsors* to determine whom to sponsor and by how much. So the bounty approach of "getting paid to do more" is recognized by some people than the donation approach, through which they can get paid immediately for the work and have more precise goals [77]. But how to balance the advantages of bounty and avoid regarding money as the guide of open source development may be the goal of future monetary incentive system design. For more specific system design recommendations, see Section 5.4.

Overall, the Sponsor mechanism is a good attempt and an essential step toward achieving reasonable and effective open source financial support. As of now, the mechanism still needs further improvement to meet the needs of more developers.

7 THREATS TO VALIDITY

For the questionnaire, we did not do the detection of carelessly invalid responses [13]. First of all, the number of questions is small, the time required to answer is short, and there is no overlap between questions, so it is not feasible to judge the validity of the responses simply by the results. Secondly, we did not set attention check items to shorten user participation time. However, since users need to click on our questionnaire and jump to the SurveyMonkey site to respond after receiving the email, we think this has ensured the validity of the responses we received to some extent. When conducting the second round of the questionnaire survey, to avoid disturbing participants excessively, we sent it only once. We did not send second or third reminder emails. At the same time, people

who have not set up a SPONSOR account may not care about the mechanism. As a result, the response rate was low.

For ITS analysis, data should be collected for the different factors for each time window. However, due to the lack of availability of timestamps in the GitHub API, some factors were measured only at their values at the time of data collection (*e.g.*, *in company*), as they do not change frequently.

For hurdle regression, the factors included in the models were several aspects related to the sponsorship of developers. However, other factors may influence whether a developer can obtain sponsorship or how much funding is received. Moreover, the number of sponsorships does not accurately indicate the amount of money that a developer receives from donations, as there exist different tiers and sponsors can withdraw their monthly sponsorship at any time. However, we do not have access to data on the actual donations received by each developer. Developers may obtain donations from other platforms to maintain related projects. We did not consider all this funding in total or the activities of developers on other platforms.

This paper explored only the effectiveness of the SPONSOR mechanism for individual users, but the SPONSOR mechanism itself can also be used for organizational accounts. To avoid our analysis being confounded by the impact of such users, we processed our data accordingly. Therefore, the results do not apply to GitHub's organizational accounts. According to statistics, 92% of users who set up sponsors are individual users.

8 CONCLUSION AND FUTURE WORK

This paper took GitHub's SPONSOR mechanism as a case study and used a mixed qualitative and quantitative analysis method to investigate four dimensions of the mechanism. Regarding why developers participate in the SPONSOR mechanism, we found that it is mainly related to the use of OSS. Regarding the mechanism's effectiveness, we found that the SPONSOR system has only a short-term effect on development activities but that in the long term, there is a slight decrease. We studied who obtains more sponsorships and found that the social status of the *maintainer* in the community correlates most strongly with this outcome (the more followers, the more sponsorships a developer acquires). Regarding the drawbacks of the mechanism, we found that in addition to the shortcomings in its use, participants felt that the SPONSOR mechanism should better attract and support corporate sponsors. Some people thought that the open source donation method needed to be improved to attract more developers to participate. Overall, we have explored the correlation between donation behavior and developers in open source communities using the GitHub SPONSOR mechanism. In future work, we will further explore the following aspects: 1) the advantages and disadvantages of different open source donation platforms and the effectiveness of incentives for open source activities and 2) different types of open source financial support and the reasonableness and effectiveness of each mode.

ACKNOWLEDGMENTS

REFERENCES

[1] Mark Aberdour. 2007. Achieving quality in open-source software. *IEEE software* 24, 1 (2007), 58–64.
[2] Bethany Alender. 2016. Understanding volunteer motivations to participate in citizen science projects: a deeper look at water quality monitoring. *Journal of Science Communication* 15, 3 (2016), A04.
[3] Shaosong Ou Alexander Hars. 2002. Working for free? Motivations for participating in open-source projects. *International journal of electronic commerce* 6, 3 (2002), 25–39.
[4] Maria J Antikainen and Heli K Vaataja. 2010. Rewarding in open innovation communities–how to motivate members. *International Journal of Entrepreneurship and Innovation Management* 11, 4 (2010), 440–456.
[5] Dryden Ashe. 2013. The ethics of unpaid labor and the oss community. https://www.ashedryden.com/blog/the-ethics-of-unpaid-labor-and-the-oss-community. [Online; accessed June 8, 2021].
[6] Susanne Beck, Carsten Bergenholtz, Marcel Bogers, Tiare-Maria Brasseur, Marie Louise Conradsen, Diletta Di Marco, Andreas P Distel, Leonhard Dobusch, Daniel Dörler, Agnes Effert, et al. 2020. The Open Innovation in Science research field: a collaborative conceptualisation approach. *Industry and Innovation* (2020), 1–50.
[7] Kenneth P. Burnham and David R. Anderson. 2002. *Model Selection and Multimodel Inference: a Practical Information-Theoretic Approach* (2nd ed.). Springer.
[8] G. Canfora, L. Cerulo, M. Cimitile, and MD Penta. 2014. How changes affect software entropy: an empirical study. *Empirical Software Engineering* 19, 1 (2014), 1–38.
[9] Francesco Cappa, Jeffrey Laut, Maurizio Porfiri, and Luca Giustiniano. 2018. Bring them aboard: rewarding participation in technology-mediated citizen science projects. *Computers in Human Behavior* 89 (2018), 246–257.
[10] Krista Casler, Lydia Bickel, and Elizabeth Hackett. 2013. Separate but equal? A comparison of participants and data gathered via Amazon's MTurk, social media, and face-to-face behavioral testing. *Computers in human behavior* 29, 6 (2013), 2156–2160.
[11] Jacob Cohen, Patricia Cohen, Stephen G West, and Leona S Aiken. 2013. *Applied multiple regression/correlation analysis for the behavioral sciences.* Routledge.
[12] The SciPy community. 2008. API Reference of scipy.stats.wilcoxon. https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.wilcoxon.html. [Online; accessed July 31, 2021].
[13] Paul G Curran. 2016. Methods for the detection of carelessly invalid responses in survey data. *Journal of Experimental Social Psychology* 66 (2016), 4–19.
[14] Paul A David and Joseph S Shapiro. 2008. Community-based production of open-source software: What do we know about the developers who participate? *Information Economics and Policy* 20, 4 (2008), 364–398.
[15] Margret C Domroese and Elizabeth A Johnson. 2017. Why watch bees? Motivations of citizen science volunteers in the Great Pollinator Project. *Biological Conservation* 208 (2017), 40–47.
[16] Enrique Estellés-Arolas and Fernando González-Ladrón-de Guevara. 2012. Towards an integrated crowdsourcing definition. *Journal of Information science* 38, 2 (2012), 189–200.
[17] Yulin Fang and Derrick Neufeld. 2009. Understanding sustained participation in open source software projects. *Journal of Management Information Systems* 25, 4 (2009), 9–50.
[18] Oluwaseyi Feyisetan, Elena Simperl, Max Van Kleek, and Nigel Shadbolt. 2015. Improving paid microtasks through gamification and adaptive furtherance incentives. In *Proceedings of the 24th international conference on world wide web.* 333–343.
[19] Andrzej Gałecki and Tomasz Burzykowski. 2013. Linear mixed-effects model. In *Linear Mixed-Effects Models Using R.* Springer, 245–273.
[20] Rishab Aiyer Ghosh. 2005. Understanding free software developers: Findings from the FLOSS study. *Perspectives on free and open source software* 28 (2005), 23–47.
[21] GitHub. 2016. Getting Paid for Open Source Work. https://opensource.guide/getting-paid/. [Online; accessed June 8, 2021].
[22] GitHub. 2017. Open Source Survey. https://opensourcesurvey.org/2017/. [Online; accessed June 8, 2021].
[23] GitHub. 2021. About your personal dashboard. https://docs.github.com/en/github/setting-up-and-managing-your-github-user-account/managing-user-account-settings/about-your-personal-dashboard#finding-your-top-repositories-and-teams. [Online; accessed May 24, 2021].
[24] GitHub. 2021. Displaying a sponsor button in your repository. https://docs.github.com/en/github/administering-a-repository/managing-repository-settings/displaying-a-sponsor-button-in-your-repository. [Online; accessed May 22, 2021].
[25] GitHub. 2021. Invest in the software that powers your world. https://github.com/sponsors. [Online; accessed July 30, 2021].
[26] GitHub. 2021. Reference of GraphQL User API. https://docs.github.com/en/graphql/reference/objects#user. [Online; accessed July 30, 2021].

[27] GitHub. 2021. Reference of RESTful List users API. https://docs.github.com/en/rest/reference/users#list-users. [Online; accessed August 1, 2021].

[28] Github. 2021. The 2020 State of the OCTOVERSE. https://octoverse.github.com. [Online; accessed February 4, 2021].

[29] R. J. Grissom and J. J. Kim. 2007. *Effect Sizes for Research: A Broad Practical Approach*. Effect sizes for research : a broad practical approach.

[30] Carl Gutwin, Reagan Penner, and Kevin Schneider. 2004. Group awareness in distributed software development. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work*. ACM, Chicago Illinois, USA, 72–81.

[31] Stefan Haefliger, Georg Von Krogh, and Sebastian Spaeth. 2008. Code reuse in open source software. *Management science* 54, 1 (2008), 180–193.

[32] Cynthia Harvey. 2017. 35 Top Open Source Companies. https://www.datamation.com/open-source/35-top-open-source-companies. [Online; accessed February 5, 2021].

[33] Andrea Hemetsberger. 2002. Fostering cooperation on the Internet: Social exchange processes in innovative virtual consumer communities. *ACR North American Advances* 29 (2002), 354–356.

[34] Mokter Hossain. 2012. Users' motivation to participate in online crowdsourcing platforms. In *2012 International Conference on Innovation Management and Technology Research*. IEEE, 310–315.

[35] Javier Luis Cánovas Izquierdo and Jordi Cabot. 2018. The role of foundations in open source projects. In *Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Society*. ACM, Gothenburg, Sweden, 3–12.

[36] S. Jackman, C. Kleiber, and A. Zeileis. 2008. Regression Models for Count Data in R. *Journal of Statistical Software* 27, 8 (2008), 1–25.

[37] Jaweria Kanwal and Onaiza Maqbool. 2012. Bug Prioritization to Facilitate Bug Report Triage. *Journal of Computer Science and Technology* 27 (2012), 397–412.

[38] Bran Knowles. 2013. *Cyber-sustainability: towards a sustainable digital future*. Lancaster University (United Kingdom).

[39] Bruce Kogut and Anca Metiu. 2001. Open-source software development and distributed innovation. *Oxford review of economic policy* 17, 2 (2001), 248–264.

[40] Sandeep Krishnamurthy and Arvind K Tripathi. 2009. Monetary donations to an open source software platform. *Research Policy* 38, 2 (2009), 404–414.

[41] Alexandra Kuznetsova, Per B. Brockhoff, and Rune H. B. Christensen. 2017. lmerTest Package: Tests in Linear Mixed Effects Models. *Journal of Statistical Software* 82, 13 (2017), 1–26. https://doi.org/10.18637/jss.v082.i13

[42] Karim Lakhani and Robert Wolf. 2005. *Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects*. MIT Press, Cambridge.

[43] Lincoln R Larson, Caren B Cooper, Sara Futch, Devyani Singh, Nathan J Shipley, Kathy Dale, Geoffrey S LeBaron, and John Y Takekawa. 2020. The diverse motivations of citizen scientists: Does conservation emphasis grow as volunteer participation progresses? *Biological Conservation* 242 (2020), 108428.

[44] Zhixing Li, Yue Yu, Tao Wang, Gang Yin, Shanshan Li, and Huaimin Wang. 2021. Are You Still Working on This An Empirical Study on Pull Request Abandonment. *IEEE Transactions on Software Engineering* (2021), 1–1. https://doi.org/10.1109/TSE.2021.3053403

[45] Huigang Liang, Meng-Meng Wang, Jian-Jun Wang, and Yajiong Xue. 2018. How intrinsic motivation and extrinsic incentives affect task effort in crowdsourcing contests: A mediated moderation model. *Computers in Human behavior* 81 (2018), 168–176.

[46] Victoria J MacPhail and Sheila R Colla. 2020. Power of the people: A review of citizen science programs for conservation. *Biological Conservation* 249 (2020), 108739.

[47] Andrew Mao, Ece Kamar, Yiling Chen, Eric Horvitz, Megan E Schwamb, Chris J Lintott, and Arfon M Smith. 2013. Volunteering versus work for pay: Incentives and tradeoffs in crowdsourcing. In *First AAAI conference on human computation and crowdsourcing*.

[48] Debra J Mesch, Patrick M Rooney, Kathryn S Steinberg, and Brian Denton. 2006. The effects of race, gender, and marital status on giving and volunteering in Indiana. *Nonprofit and Voluntary Sector Quarterly* 35, 4 (2006), 565–587.

[49] Nadia. 2015. A handy guide to financial support for open source. https://github.com/nayafia/lemonade-stand/blob/master/README.md. [Online; accessed June 8, 2021].

[50] Keitaro Nakasai, Hideaki Hata, and Kenichi Matsumoto. 2018. Are donation badges appealing?: A case study of developer responses to eclipse bug reports. *IEEE Software* 36, 3 (2018), 22–27.

[51] Keitaro Nakasai, Hideaki Hata, Saya Onoue, and Kenichi Matsumoto. 2017. Analysis of donations in the eclipse project. In *8th International Workshop on Empirical Software Engineering in Practice (IWESEP)*. IEEE, Tokyo, Japan, 18–22.

[52] Cassandra Overney. 2020. Hanging by the Thread: An Empirical Study of Donations in Open Source. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Companion Proceedings* (Seoul, South Korea) *(ICSE '20)*. Association for Computing Machinery, New York, NY, USA, 131–133. https://doi.org/10.1145/3377812.3382170

[53] Cassandra Overney, Jens Meinicke, Christian Kästner, and Bogdan Vasilescu. 2020. How to Not Get Rich: An Empirical Study of Donations in Open Source. In

[54] Patreon. 2021. Discord (Text chat for creators that's free, secure, and works on both your desktop and phone). https://www.patreon.com/apps/discord. [Online; accessed August 8, 2021].

[55] Anett Richter, Orr Comay, Cecilie S. Svenningsen, Jonas Colling Larsen, Susanne Hecker, Anders P. Tøttrup, Guy Pe'er, Robert R. Dunn, Aletta Bonn, and Melissa Marselle. 2021. Motivation and support services in citizen science insect monitoring: A cross-country study. *Biological Conservation* 263 (2021), 109325. https://doi.org/10.1016/j.biocon.2021.109325

[56] Jackson Samantha. 2019. Feature Transformation (Understanding When to Scale and Standardize Data for Machine Learning). https://medium.com/@sjacks/feature-transformation-21282d1a3215. [Online; accessed August 8, 2021].

[57] Isaac Schlueter. 2013. Money and Open Source. https://medium.com/open-source-life/money-and-open-source-d44a1953749c. [Online; accessed 8-June-2021].

[58] Douglas C Schmidt and Adam Porter. 2001. Leveraging open-source communities to improve the quality & performance of open-source software. In *Proceedings of the 1st Workshop on Open Source Software Engineering*, Vol. 1. Citeseer, Toronto, Canada.

[59] Andrew Schofield and Grahame S. Cooper. 2006. Participation in Free and Open Source Communities: An Empirical Study of Community Members' Perceptions. In *Open Source Systems*, Ernesto Damiani, Brian Fitzgerald, Walt Scacchi, Marco Scotto, and Giancarlo Succi (Eds.). Springer US, Boston, MA, 221–231.

[60] Manuel Sojer and Joachim Henkel. 2010. Code reuse in open source software development: Quantitative evidence, drivers, and impediments. *Journal of the Association for Information Systems* 11, 12 (2010), 2.

[61] Diana Super. 2020. Become a sponsor to Super Diana. https://github.com/sponsors/alphacentauri82. [Online; accessed May 26, 2021].

[62] SurveyMonkey. 1999. https://www.surveymonkey.com/. [Online; accessed May 26, 2021].

[63] Patrícia Tiago, Maria João Gouveia, César Capinha, Margarida Santos-Reis, and Henrique M Pereira. 2017. The influence of motivational factors on the frequency of participation in citizen science activities. *Nature Conservation* 18 (2017), 61.

[64] Condon Tim. 2020. Become a sponsor to Tim Condon. https://github.com/sponsors/0xTim. [Online; accessed May 26, 2021].

[65] Asher Trockman, Shurui Zhou, Christian Kästner, and Bogdan Vasilescu. 2018. Adding Sparkle to Social Coding: An Empirical Study of Repository Badges in the Npm Ecosystem. In *Proceedings of the 40th International Conference on Software Engineering* (Gothenburg, Sweden) *(ICSE '18)*. Association for Computing Machinery, New York, NY, USA, 511–522. https://doi.org/10.1145/3180155.3180209

[66] Liam Tung. 2020. Redis database creator Sanfilippo: Why I'm stepping down from the open-source project. https://www.zdnet.com/article/redis-database-creator-sanfilippo-why-im-stepping-down-from-the-open-source-project/. [Online; accessed June 8, 2021].

[67] Steven J. Vaughan-Nichols. 2021. Hard work and poor pay stresses out open-source maintainers. https://www.zdnet.com/article/hard-work-and-poor-pay-stresses-out-open-source-maintainers/. [Online; accessed Jun 8, 2021].

[68] Georg Von Krogh, Stefan Haefliger, Sebastian Spaeth, and Martin W. Wallin. 2012. Carrots and Rainbows: Motivation and Social Practice in Open Source Software Development. *MIS Q.* 36, 2 (jun 2012), 649–676.

[69] Jing Wang, Patrick C. Shih, and John M. Carroll. 2015. Revisiting Linus's law: Benefits and challenges of open source software peer review. *International Journal of Human-Computer Studies* 77 (2015), 52–65. https://doi.org/10.1016/j.ijhcs.2015.01.005

[70] John Willinsky. 2005. The unacknowledged convergence of open source, open access, and open science. *First Monday* 10, 8 (Aug. 2005). https://doi.org/10.5210/fm.v10i8.1265

[71] Sarah Wiseman, Anna L Cox, Sandy JJ Gould, and Duncan P Brumby. 2017. Exploring the effects of non-monetary reimbursement for participants in HCI research. *Human Computation* (2017).

[72] Bo Xu, Donald R. Jones, and Bingjia Shao. 2009. Volunteers' involvement in online community based software development. *Information & Management* 46, 3 (2009), 151–158. https://doi.org/10.1016/j.im.2008.12.005

[73] Bo Xu and Dahui Li. 2015. An empirical study of the motivations for content contribution and community participation in Wikipedia. *Information & management* 52, 3 (2015), 275–286.

[74] Yue Yu, Gang Yin, Huaimin Wang, and Tao Wang. 2014. Exploring the Patterns of Social Behavior in GitHub. In *Proceedings of the 1st International Workshop on Crowd-Based Software Development Methods and Technologies* (Hong Kong, China) *(CrowdSoft 2014)*. Association for Computing Machinery, New York, NY, USA, 31–36. https://doi.org/10.1145/2666539.2666571

[75] Xunhui Zhang, Tao Wang, Yue Yu, Qiubing Zeng, Zhixing Li, and Huaimin Wang. 2022. Questionnaire design for GitHub Sponsor mechanism. (2022). https://doi.org/10.5281/ZENODO.5715824

[76] Yangyang Zhao, Alexander Serebrenik, Yuming Zhou, Vladimir Filkov, and Bogdan Vasilescu. 2017. The impact of continuous integration on other software development practices: A large-scale empirical study. In *2017 32nd IEEE/ACM*

Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering (Seoul, South Korea) *(ICSE '20)*. Association for Computing Machinery, New York, NY, USA, 1209–1221. https://doi.org/10.1145/3377811.3380410

International Conference on Automated Software Engineering (ASE). 60–71. https://doi.org/10.1109/ASE.2017.8115619

[77] Jiayuan Zhou, Shaowei Wang, Cor-Paul Bezemer, Ying Zou, and Ahmed E. Hassan. 2020. Studying the Association between Bountysource Bounties and the Issue-addressing Likelihood of GitHub Issue Reports. *IEEE Transactions on Software Engineering* (2020), 1–1. https://doi.org/10.1109/TSE.2020.2974469

[78] T. Zimmermann. 2016. Card-sorting: From text to themes. In *Perspectives on Data Science for Software Engineering*, Tim Menzies, Laurie Williams, and Thomas Zimmermann (Eds.). Morgan Kaufmann, Boston, 137–141. https://doi.org/10.1016/B978-0-12-804206-9.00027-1

A OTHER PLATFORMS BESIDES THE SPONSOR MECHANISM

Table 9: Other platforms for obtaining OSS financial support

Name	URL
Bountysource	https://www.bountysource.com
Flattr	https://flattr.com
IssueHunt	https://issuehunt.io
Kickstarter	https://www.kickstarter.com
Liberapay	https://liberapay.com
Gittip	https://gratipay.com
Gratipay	https://gratipay.com
OpenCollective	https://opencollective.com
Otechie	https://otechie.com
Patreon	https://www.patreon.com
PayPal	https://www.paypal.com
Tidelift	https://tidelift.com
Tip4Commit	https://tip4commit.com
LFX Mentorship (formerly CommunityBridge)	https://lfx.linuxfoundation.org/tools/mentorship
Ko-fi	https://ko-fi.com