

## Guiding the way: A systematic literature review on mentoring practices in open source software projects

Zixuan Feng<sup>a,\*</sup>, Katie Kimura<sup>a</sup>, Bianca Trinkenreich<sup>a</sup>, Anita Sarma<sup>a</sup>, Igor Steinmacher<sup>b</sup>

<sup>a</sup> Oregon State University, School of Electrical Engineering and Computer Science, Corvallis, OR 97331, USA

<sup>b</sup> Northern Arizona University, School of Informatics, Computing, and Cyber Systems, Flagstaff, AZ 86011, USA

### ARTICLE INFO

**Keywords:**  
Mentoring  
Open source  
Literature review

### ABSTRACT

**Context:** Mentoring in Open Source Software (OSS) is important to its project's growth and sustainability. Mentoring allows contributors to improve their technical skills and learn about the protocols and cultural norms of the project. However, mentoring has its challenges: mentors sometimes feel unappreciated, and mentees may have mismatched interests or lack interpersonal skills. Existing research has investigated the different challenges of mentoring in different OSS contexts, but we lack a holistic understanding.

**Objective:** A comprehensive understanding of the current practices and challenges of mentoring in OSS is needed to implement appropriate strategies to facilitate mentoring.

**Method:** This study presents a systematic literature review investigating how literature has characterized mentoring practices in OSS, including their challenges and the strategies to mitigate them. We retrieved 232 studies from four digital libraries. Out of these, 21 were primary studies. Using this, we performed backward and author snowballing, adding another 27 studies. We conducted a completeness check by reviewing the references of the 4 most relevant primary studies, which resulted in us adding 1 additional study. We then conducted a full-text review and evaluated the studies using a set of criteria; as a result, 10 papers were excluded. We then employed an open-coding approach to analyze, aggregate, and synthesize the selected studies.

**Results:** We reviewed 39 studies to investigate the different facets of mentoring in OSS, encompassing motivations, goals, channels, and contributor dynamics. We then identified 13 challenges associated with mentoring in OSS, which fall into three categories: social, process, and technical. We also present a quick-reference strategy catalog to map these strategies to challenges for mitigation.

**Conclusions:** Our study serves as a guideline for researchers and practitioners about mentoring challenges and potential strategies to mitigate these challenges.

### 1. Introduction

Mentoring is one of the key ways for both current and novice contributors to enhance their technical skills and learn about project norms and cultural practices under the guidance of experienced contributors [1–7]. Mentoring can be formal or informal. In the former situation, mentors and mentees are paired through a structured program where regular sessions, goals, and evaluations are established to track progress and ensure a clear development path for the mentee [1, 4,8–10]. But, often, mentoring can be informal [2,11–14]. Feng et al. [2] defined a form of informal mentoring, implicit mentoring, where contributors provide guidance and feedback to other peers through everyday activities such as code reviews.

Research shows that mentoring in Open Source Software (OSS) projects is essential, given that these projects comprise a distributed

community where contributors from all over the world, across different time zones collaborate [5,15–17]. Together, these contributors create and maintain large and intricate software projects [15,16,18]. Newcomers in these projects must learn not only the technical expertise to contribute effectively but also gain an understanding of the cultural norms governing the creation of changes and the processes for their acceptance [10,19]. OSS communities are aware of the significance of mentoring, prompting large OSS foundations to invest time and efforts into formal mentoring initiatives, such as the Apache Mentorship Program and the Linux Mentorship Program [20,21].

However, despite these foundational programs, the time and effort required of mentors can still present significant challenges to their practicality. For example, mentors often do not receive the appropriate

\* Corresponding author.

E-mail addresses: [fengzi@oregonstate.edu](mailto:fengzi@oregonstate.edu) (Z. Feng), [kimuraka@oregonstate.edu](mailto:kimuraka@oregonstate.edu) (K. Kimura), [bianca.trinkenreich@oregonstate.edu](mailto:bianca.trinkenreich@oregonstate.edu) (B. Trinkenreich), [Anita.Sarma@oregonstate.edu](mailto:Anita.Sarma@oregonstate.edu) (A. Sarma), [Igor.Steinmacher@nau.edu](mailto:Igor.Steinmacher@nau.edu) (I. Steinmacher).

<https://doi.org/10.1016/j.infsof.2024.107470>

Received 6 November 2023; Received in revised form 3 April 2024; Accepted 6 April 2024

Available online 9 April 2024

0950-5849/© 2024 Elsevier B.V. All rights reserved.

recognition or acknowledgment for their work despite the amount of time and effort they invest [2,22]. Meanwhile, mentees may feel unfulfilled by their mentors due to mismatched interests and face an unwelcome environment because of biases in discussions [5–7,22–28]. Those biases may be conscious or unconscious and arise from destructive code reviews and cultural differences across mentees' gender, race/ethnicity, age, or seniority [29–32]. Mentoring challenges are pervasive. Some studies have investigated the challenges that newcomers face during onboarding [6,26], while others have investigated the challenges that mentors face with newcomers [5,22]. Several studies have proposed strategies that may mitigate these challenges [5,22,33]. Some studies have focused on formal OSS mentoring programs [4,9], while another study claimed that mentoring is more holistic and includes everyday development activities [2].

Our understanding of mentoring in OSS is currently fragmented, with insights dispersed across multiple studies. Without a comprehensive understanding of mentoring in OSS, it becomes challenging to pinpoint effective strategies for mentor-mentee relationships and determine best practices for mentoring. This knowledge gap can lead to redundancy in efforts and potentially missed opportunities to foster a nurturing and inclusive OSS community. Addressing this fragmented perspective is crucial for the success of mentoring initiatives as well as the long-term health and growth of OSS communities.

In this study, we review, summarize, and synthesize the current state of research on mentoring practices in OSS by conducting a systematic literature review (SLR) from 39 primary studies published between 2012 and 2023. Our SLR is guided by the **RQ: How have mentoring practices in OSS been characterized in the literature?**

Our contributions include (1) an overview of mentoring practices through five aspects; (2) the benefits of mentoring for OSS contributors and projects; (3) a “mentoring challenges” taxonomy serving as a quick reference for self-wellness checks by OSS communities; and (4) a quick-reference catalog of strategies mapped to challenges that can be used by OSS practitioners.

## 2. History of mentoring

Ericsson et al. [34] emphasized the value of seeking mentors during challenging tasks in software development, underscoring its role in fostering developer growth. In 2005, Google created a mentoring program for the OSS community through the Google Summer of Code (GSoC) mentorship program [4]. Similarly, Linux, a global OSS organization, has launched its mentorship program, committing over \$2.5 million to onboard new contributors [21]. Likewise, the Apache Software Foundation has launched a mentoring program to foster its community's development [20].

Studies in OSS have emphasized that mentoring plays a crucial role in onboarding newcomers and strengthening collaborations [2, 5,19,35,36]. For example, mentoring can provide newcomers with a more successful onboarding experience [19,35]. The “Code as Conversation” model (e.g., Pull Request comments on GitHub), where forum contributors ask and answer questions on a shared code snippet, is one example of a mentoring communication channel [37]. On the other hand, the absence of a mentor can be a significant obstacle to newcomer onboarding [35,38]. In 2022, Feng et al. [2] analyzed the daily activities (Pull Request comments) of OSS contributors to define a new form of informal mentoring, implicit mentoring, which is widespread in OSS, benefitting both mentors and mentees.

Currently, insights on mentoring are fragmented across studies. As a result, detailed research that can serve as guidance for mentoring in OSS or systematically introduce mentoring to OSS is lacking. This study provides a review of how mentoring has been characterized in the OSS community, serving as a foundational guide for mentoring in OSS.

## 3. Research method

This study was conducted as an SLR following the guidance proposed by Kitchenham [39] to review, aggregate, and categorize disparate knowledge about mentoring from studies in OSS.

### 3.1. Designing the SLR

The SLR was guided by the research question: **How have mentoring practices in OSS been characterized in the literature?** The authors held weekly meetings within one month to design the methodology of this study. Fig. 1 shows an overview of the methodology used in this study, including the research question, the definition of the search string, keyword piloting/searching, eligibility criteria, snowball sampling, quality assessment, data extraction, and the synthesis approach.

**Search String and Digital library.** We then determined a list of search keywords based on our research question. Proposed search keywords included, “mentorship”, “mentoring”, “mentee”, and “mentor”. To focus on the OSS community, we added “Open Source” or “OSS” to each string. This was important because the open-collaborative nature of OSS entails unique channels, activities, and communication forums that differ from those used in conventional software development [40]. We designed the search string to ensure the relevance of the captured studies, requiring that the search terms appear at least once in the title, abstract, or keywords. An example of a search query is as follows: (“Open Source” OR “OSS”) AND (“mentor” OR “mentee” OR “mentorship” OR “mentoring”).

Following the recommendations of Kitchenham [39] for secondary studies in computer science and software engineering, emphasizing their comprehensive coverage, we adopted a hybrid strategy to collect primary studies, which included a database search (ACM Digital Library, IEEE Xplore, SpringerLink, and ScienceDirect) and snowballing. The digital libraries used in this study have been frequently utilized in other SLR studies [41–43]. We decided not to use the year of publication as an additional filter, as mentoring is well-known but still relatively new to the OSS community and has only been studied in recent decades [2,22,35].

Before applying the search string to the targeted database (digital libraries), we conducted a pilot search on four libraries to refine and validate the search string and ensure the results were relevant to the study goal and could be used to answer our research question. We selected four well-known studies on mentoring in OSS as control studies from four digital libraries with which we were already familiar with [4–6,19]. By applying the defined search string and search approach, all four control studies were found.

**Eligibility criteria.** To narrow down the search results to only those relevant to mentoring in OSS communities, we designed inclusion criteria (IC) and exclusion criteria (EC)—as suggested by Kitchenham and Brereton [44]. Having criteria ensured that the selected studies were relevant to the study's goal and could adequately address the research questions.

- (+IC1.) The primary study investigates mentorship in OSS projects.
- (-EC1.) The study is a doctoral/master's thesis, research proposal, short communication, or technical report.
- (-EC2.) The study is not in the English language.
- (-EC3.) The study is a duplicate or a preliminary version of a previously reviewed article.
- (-EC4.) The study has not undergone peer review.
- (-EC5.) Unable to access the complete study.

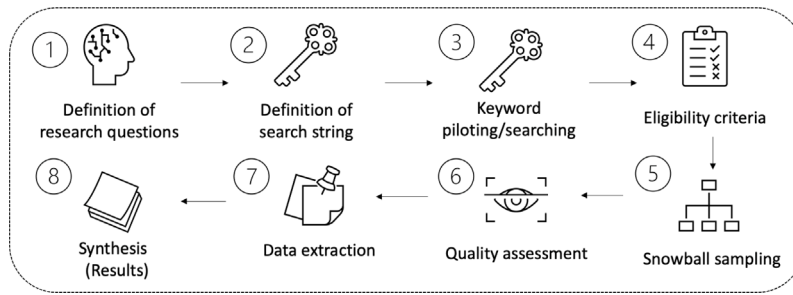


Fig. 1. The main SLR review process.

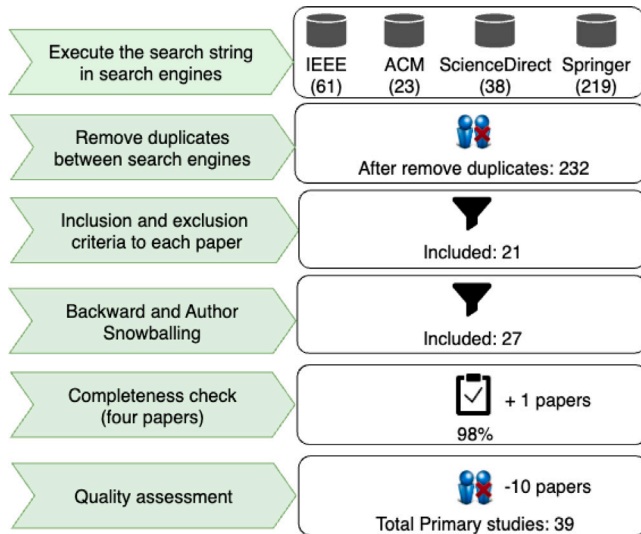


Fig. 2. Setups to filter and select the primary studies.

**Table 1**  
Digital Libraries and initial number of articles.

Search String	ACM	IEEE	SpringerLink	ScienceDirect
mentorship AND (OSS OR Open Source)	1	7	26	5
mentoring AND (OSS OR Open Source)	11	26	73	16
mentor AND (OSS OR Open Source)	11	24	110	16
mentee AND (OSS OR Open Source)	0	4	10	1

For ACM, IEEE, ScienceDirect, search query were applied for Title OR Abstract OR Keyword  
For Springer, search query were applied for exact same phase AND title

3.2. Conducting the SLR

**Step 1 & 2: Searching with a search string and removing duplicates.** Table 1 shows the digital libraries and the number of search results using search strings (search date: February 3rd, 2023). We noted that the search results returned duplicate entries in the libraries, as studies are often cross-listed in multiple libraries. After removing 109 duplicates, this step yielded 232 studies.

**Step 3. Selecting Primary Studies.** The first two authors held weekly meetings to conduct a negotiated agreement process for evaluating papers based on the predefined inclusion and exclusion criteria. These meetings between the first and second authors were done synchronously (in-person or remotely), mitigating any biases to ensure an agreement was reached by the end of the meeting. As previously mentioned, we conducted a pilot search using four well-known studies

with which the first and second authors were already familiar with [4–6,19]. These papers served as a foundational knowledge base, guiding the selection process of primary studies.

For each paper, we read its abstract, introduction, and conclusion to understand the paper’s topic and primary findings. We also read the methodology section to understand the study’s approach, and we reviewed the results section to discern the paper’s contribution(s). After applying the inclusion and exclusion criteria, we retrieved 21 primary studies, with 211 papers removed.

**Step 4. Snowball sampling.** As suggested by Kitchenham and Brereton [44], we then applied single-step backward snowball sampling from the references of the papers for any additional literature [45]. After reviewing the publication records, we also conducted “author snowballing” on the top three most prolific authors (Igor Steinmacher, Marco Gerosa, and Anita Sarma). This involved investigating the authors of the chosen papers to see if they had written any other relevant papers. We searched these researchers’ websites and Google Scholar profiles until we could no longer find research on mentoring. The snowball sampling helped identify 27 more relevant studies in OSS (as shown in Fig. 2).

**Step 5. Completeness check.** To guarantee the comprehensiveness of our research, we performed a completeness check on the top four most cited studies from our list: [5,22,27,46]. The first two authors independently reviewed every reference in each of the four papers, specifically seeking those relevant to mentoring. The two authors then reviewed each other’s lists and reached a negotiated agreement. These four papers included 12 unique studies on mentoring, 2 of which were missing from our list [47,48]. Since [48] is only comprised of four pages, we only added [47] to our primary list. Our completeness ratio is 98% (1 additional paper added to the 48 papers in our original list).

**Step 6. Quality assessment and data extraction.** The first two authors conducted a full-text review of each paper. We evaluated each paper using a set of criteria, defined and illustrated in Table 2. These criteria were established to ensure the papers’ relevance to answer our research question. After a thorough full-text reading, ten papers were excluded because they did not primarily focus on mentoring-related collaboration. As a result, 39 primary studies were ultimately included in our analysis (see supplementary spreadsheet [49] for details on selected papers).

3.3. Extracting and analyzing data

The 39 primary studies published between 2012 and 2023 have an average of 58 references ( $sd = 33$ ) and 31 citations ( $sd = 38$ ). Based on the titles, keywords, abstracts, introductions, and conclusions, the topics of the 39 primary studies were classified into three categories. As shown in Table 3, the most prevalent research topic among the studies is OSS contributor dynamics and collaboration strategies (14 studies). The topic includes characteristics and motivations of OSS contributors, retaining/attracting contributors in OSS, strategies for improving collaboration systems in OSS, and barriers for general contributors in OSS. 20 studies investigated OSS newcomer onboarding

**Table 2**

Assessment criteria definition.

ID	Assessment criteria	Criteria definitions
AC1	Mentoring in OSS	The paper explicitly investigates mentoring relationships in OSS.
AC2	Collaborative system in OSS	The paper investigates contributors' characteristics, motivations, and barriers to analyze/improve collaborative and social computing systems in OSS.
AC3	Data collection/analysis	The paper explains the methodology in detail, including data collection and analysis.
AC4	Usability/Repeatability	The paper discusses the validity and threats in detail, and results are reflective for the majority of contributors/projects in OSS community.
AC5	Empirical validation	The paper reports case studies, controlled experiments, and other empirical support.

**Table 3**

Categories of data detailing OSS mentoring and their corresponding primary studies.

Categories of data	Primary studies
Contributor dynamics and collaboration strategies in OSS	[2,11,19,22,27,28,36,46,50–53] [54,55]
Newcomer onboarding and engagement in OSS	[5–7,9,23–26,33,35,56–58] [59–65]
Diversity and inclusion in OSS	[31,66]
Challenges of mentoring in OSS	[2,5,6,19,23–25,27,31,33,35,46] [28,50,51,53,56–58,61–64] [66–68]
Mentoring Strategies in OSS	[2,4–6,9,11,19,22,24,31,33,36,46] [55–59,61,64]

and engagement; they focus on onboarding guidance, recommending first issues to newcomers, and retaining and attracting newcomers. Two studies were conducted on diversity and inclusion in OSS, focusing on factors such as seniority, gender, and location [31,66]. Among our primary studies, 26 investigated the challenges mentors or contributors encounter within the OSS community. Meanwhile, 20 studies discussed strategies to address these mentoring challenges in OSS.

We analyzed the 39 selected papers qualitatively, following the open coding protocol after a full-text review [69]. During the analysis, each emerging code was compared to the existing codes to identify whether it was a separate category or a subset of an existing code. The first three authors applied the procedure via continuous comparison throughout coding sessions [70,71].

We first investigated how mentoring is characterized in existing studies, consolidating our understanding into five perspectives (Section 4): characteristics, motivations and goals, channels and activities, benefits, and forms. We then investigated mentoring challenges in OSS and created a taxonomy table for challenges based on the categories developed in prior work [5] (Section 5). We have also synthesized strategies for addressing mentoring challenges and created quick-reference tables to improve the effectiveness of mentoring in OSS communities (Section 6).

#### 4. Overview of mentoring in OSS

We started our research by investigating how the literature characterizes mentoring practices in OSS. Specifically, we looked into the nature of mentoring in OSS, such as why OSS needs mentoring, who are mentors and mentees, what is considered mentoring activities, and where mentoring takes place (channels). We characterize mentoring in OSS through five distinct aspects (see Fig. 3): characteristics of mentoring, motivations and goals, channels and activities, benefits of mentoring, and forms of mentoring (see Table 6 for quick references in Appendix)

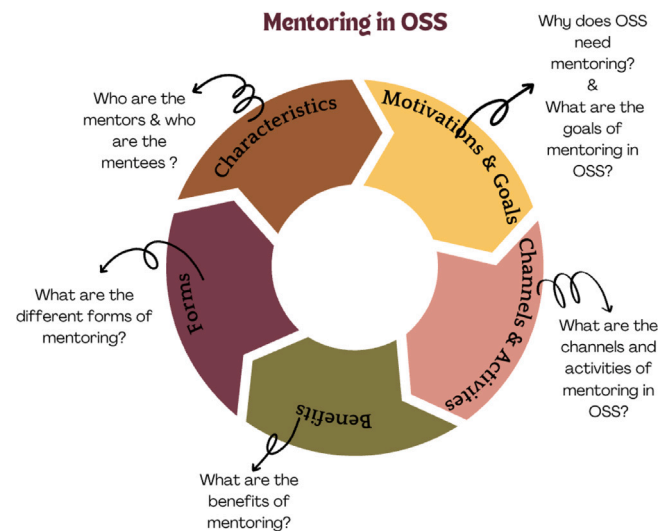


Fig. 3. Mentoring in OSS.

##### 4.1. In OSS communities, who are mentees, and who are mentors?

There are no restrictions on who can act as a mentee or mentor, yet certain patterns frequently emerge. During the onboarding process, mentees are newcomers — individuals new to the OSS community or contributors joining new OSS projects [63]. These mentees are often paired with experienced contributors (mentors) from the project, who may either be assigned or volunteer for the role [35]. These mentors provide guidance and address specific challenges [38]. Such mentors hold positions of seniority and have considerable impacts on projects [65].

In addition, contributors act as mentors when they support one another by sharing insights, knowledge, and interests, regardless of skill



proficiency or seniority. A survey with 231 OSS contributors in Apache Projects revealed that nearly 90% of contributors had adopted such roles as mentors or mentees, irrespective of their seniority [2]. Therefore, mentoring can address issues that may arise due to OSS communities lacking formal training. Furthermore, mentoring can help connect contributors facing similar challenges in learning certain concepts or software [11].

**Characteristics:** In OSS, mentors and mentees are not strictly defined as senior contributors and newcomers to a project. For existing contributors, the roles of mentors and mentees are fluid, and contributors often assist each other based on shared challenges. This assistance may be provided voluntarily or due to encouragement from the community.

#### 4.2. Why does OSS need mentoring?

**OSS projects do not assist newcomers:** When newcomers contribute to OSS projects, they often begin as “outsiders”, selecting open issues from the issue tracker system for their initial contributions [23]. However, these environments are usually complex and unfamiliar, and unfortunately, many OSS projects leave newcomers to fend for themselves [33,35].

**Newcomers face significant barriers:** Further complicating this onboarding process are various barriers. Steinmacher et al. [7] introduced a conceptual model outlining 58 obstacles newcomers face when joining OSS projects. In addition to these obstacles, newcomers may also face inclusion issues related to gender, race/ethnicity, or seniority [5,28,31]. In a field study involving five professional software development teams, Mendez et al. [28] found that most OSS tool and infrastructure barriers are tied with literature-established newcomer barriers, with 80% of them showing bias against women.

**Not only mentees face challenges and barriers:** While the absence of onboarding mentoring may decrease the likelihood of newcomers becoming long-term contributors [54,68], having an onboarding mentoring program is not without its challenges. By interviewing contributors who have previously served as mentors in OSS projects, Balali et al. [5] pinpointed 44 challenges that may occur during the onboarding mentoring process. Of these, 19 challenges particularly impact mentors, while 34 primarily affect mentees. Further insights from surveys and interviews conducted by Steinmacher et al. [22] highlighted 25 additional mentoring challenges faced by mentors.

**Contributors want to build better communities:** Tan et al. [57] interviewed contributors who volunteered to be GSoC mentors and found that most of the mentors’ decisions to volunteer were intrinsically motivated; they sought feelings of self-accomplishment and wanted to assist others. Feng et al. [2] also found that informal mentors derived satisfaction from voluntarily mentoring. One mentor in their study stated they “never get tired of seeing the lights come on when the mentee gets it”.

Mentees from GSoC are often extrinsically motivated. In other words, they aim to gain work experience and the prestige of associating with notable organizations on their resumes [4,57]. Furthermore, mentees see the GSoC as a chance to interact with mentors and community members equipped with OSS experiences, which can expand their professional network and help them get familiar with software engineering practices [4].

#### 4.3. What are the goals of mentoring in OSS?

The goals of mentoring are related to helping mentees acquire the technical, social, and organizational knowledge and skills necessary to succeed in OSS [5,65].

**Help newcomers join OSS projects:** One of the primary goals of mentoring is to help newcomers overcome initial barriers and challenges during their early participation in OSS [5,9,19,22,33,35]. The

outcomes are more favorable when an experienced mentor directly supports a new contributor in comparison to when a new contributor navigates things alone [35]. Fagerholm et al. [19] found that mentoring effectively maintains a contributor’s motivation and improves collaboration within OSS communities. Typically, a specialized onboarding mentoring program is designed to provide mentees with any necessary upfront training, as well as continuous access to mentoring support throughout the project [72]. Mentoring helps newcomers expand their knowledge and skill set, facilitates newcomers’ assimilation into the culture of OSS communities, and boosts newcomers’ work productivity [6,9,72].

**Help contributors internalize the community culture in OSS:** Prana et al. [31] suggested a “proximity-based” mentoring where mentors and mentees are paired up based on their geographical (or even cultural) closeness to one another, potentially assisting contributors in overcoming shared inclusion biases. When computer science students enter the industry and engage in peer code review, they inadvertently learn to promote a community-oriented learning environment [36]. A microclimate created by a focused and effective peer group, who help each other adopt the project culture and foster a collaborative environment, enhances the likelihood of an individual to become a long-term contributor [54].

**Help OSS communities improve their sustainability:** Silva et al. [4] mentioned that fostering and encouraging student participation in mentoring programs can increase the size of the OSS workforce, especially given that the OSS community is interest-based and volunteer-led [2]. As Zhou and Mockus [54] found, future long-term contributors are more likely than other OSS candidates to play an active role and show a community-oriented attitude. Creating a collaborative, newcomer-friendly environment to attract and retain newcomers is crucial to project survival and success [54,66]. For instance, mentors in the GSoC support the future development of their projects as their mentees bring fresh insights [57]. Moreover, 18% of mentees give back to the community by becoming future mentors to assist the next generation [9].

**Motivations & Goals:** Both OSS newcomers and ongoing contributors face challenges, from project complexities to inclusive issues. Mentoring aims to provide contributors with the necessary skills to contribute, help them mitigate initial challenges, and ensure they integrate well into the community.

#### 4.4. What are the channels and activities of mentoring in OSS?

**Mentoring in OSS is all-pervasive:** Mentoring can take place in a variety of places and ways. The channels mentors and mentees may use to interact with one another are emails, pull-request (PR) comments, code reviews, in-person meetings, online meetings, mailing lists, discussion forums, blogs, internet relay chat (IRC), and other online communication tools [2,35,65,72]. A survey of OSS newcomers from Rehman et al. [62] showed that over half of the respondents’ first contributions were related to development. Mentors are encouraged to continue assisting mentees locate and understand tasks by communicating through online discussion forums, mailing lists, issue tracking systems, and online meetings [19,22,35,63]. An issue tracking system (e.g., pull requests, issue lists) is one of the most popular platforms where mentees and mentors engage in discussions and code reviews [7,35]. To track mentees’ progress, an administrator of GSoC suggested that OSS mentors should encourage mentees to submit their code to the issue tracking system [4,9].

**Mentoring exists within routine OSS activities:** When contributors join OSS projects, they must familiarize themselves with the projects’ code base, tools, and procedures [19]. Support can come in various ways, such as recommending tasks or explaining the software architecture [19]. Kilamo et al. [73] highlighted that code reviews

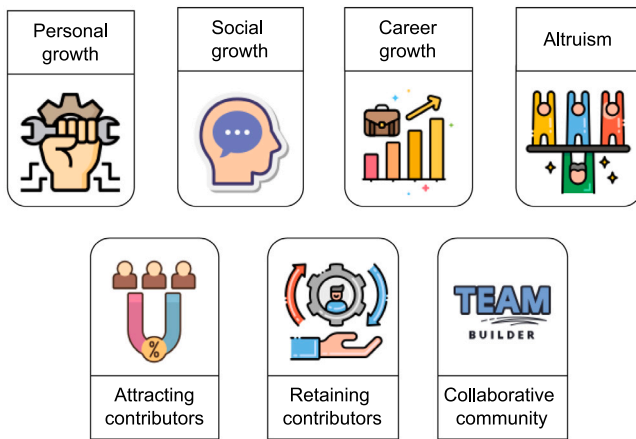


Fig. 4. Benefits of Mentoring in OSS.

are among the most integral mentoring activities; they found that more than half of the QT project [74,75] contributors conducted code reviews to assist their peers. Informal mentoring is common in daily OSS tasks. This type of mentoring occurs during routine OSS activities and is facilitated through various channels, including code reviews, emails, video conferences, and more [2].

**Channels & Activities:** The channels and activities of mentoring in OSS are all-pervasive and exist within routine OSS activities. The channels mentors and mentees may use to interact with one another are emails, pull-request (PR) comments, code reviews, in-person meetings, online meetings, mailing lists, and other online communication tools.

#### 4.5. What are the benefits of mentoring in OSS?

The growth and success of any organization is rooted in learning [57,76]. Research on education, management, and software engineering indicates that mentoring is valuable to both mentees and mentors [2,36,57,77,78].

**Benefits for mentees and mentors:** As shown in Fig. 4, we found that mentoring in OSS presents many benefits for both mentees and mentors. These benefits were broadly categorized into four primary areas: **Personal Growth, Career Growth, Social Skill Development, and Altruism** [2].

Through mentoring, *mentors* refine their technical expertise and nurture non-technical skills, such as leadership, problem-solving, and interpersonal skills. Mentors mention that mentoring often aids their career growth through promotions or potential salary increments [2]. The altruistic benefits of mentoring, such as observing mentees' growth, receiving appreciation, and developing friendships, further strengthen its value [2,5,55,57,60,79]. Gerosa et al. [52] found that senior contributors frequently remain active in OSS projects due to altruistic motivations.

The primary benefit for *mentees* in contributing to OSS is **learning** [11,22,52,57,60]. During onboarding, mentors help mentees familiarize themselves with the project's culture norms and established contribution process [11,19]. As mentees understand the structures and procedures of an OSS project, they gain confidence in their contributions and are less likely to fear judgment [11]. Mentors primarily assist in **improving mentees' technical abilities** [22,26,38], by explaining software architecture, assigning tasks, and performing code reviews [19,22]. In addition, mentoring in OSS enhances mentees' **social skills and networks** and **strengthens their career progression** [2,52,55,57,60]. For example, mentees from GSoC were motivated

by peer recognition, networking expansion, resume building [9], and career progression [2]. Moreover, employers often assess a candidate's contributions to OSS during the hiring process [52].

**Benefits for projects:** Mentoring plays a significant role in **attracting and retaining contributors** in OSS communities. Established formal mentoring programs, like GSoC, leverage onboarding mentoring to attract and retain contributors [4,9]. Mentoring improves mentee engagement and improves the chances of their sustained, long-term contribution to the project [80–82]. In GSoC, students who complete all program phases are compensated with 3000 to 6600 USD; 18% ( $n = 22$ ) of these students later become mentors in GSoC [83]. In addition, a supportive work environment, nurtured by effective mentoring relationships, can improve the overall *health of a collaborative environment* [2,54].

**Benefits:** Mentoring in OSS provides mutual benefits to mentors and mentees in the forms of personal growth, career growth, social skill development, and altruism while enhancing community sustainability and fostering a healthy collaborative environment.

#### 4.6. What are the different forms of mentoring?

While reviewing the literature on mentoring in OSS, we found two forms of mentoring: **formal mentoring** and **informal mentoring**. Feng et al. [2] defines the former as when “*mentors and mentees are formally connected through scholarships or mentorship programs*”, and informal mentoring as when “*a mentor or mentee reaches out to others to seek/give guidance in a particular area*”.

GSoC is a typical example of formal mentoring as it provides scholarships to students interested in contributing to OSS projects and assigns them a mentor from the project they select [27,33]. In contrast, informal mentoring can occur anytime and anywhere in OSS, such as in code reviews, emails, and video conferences.

**Forms:** Mentoring in OSS has different forms. Formal onboarding caters to newcomers, whereas informal mentoring is utilized by all contributors.

### 5. Challenges of mentoring in OSS














This section consolidates mentoring challenges in OSS projects (see Table 4); these challenges were grouped into three categories: **social-related, process-related, and technical-related**.




- **Social-related** challenges are classified as personal or interpersonal, such as personality conflicts and interpersonal challenges between mentees and mentors.
- **Process-related** challenges relate to the workflows of projects/communities, such as bureaucratic procedures.
- **Technical-related** challenges include difficulty setting up a software development environment and similar technical-related issues.

#### 5.1. Social-related challenges

Mentors may struggle to help mentees with **low self-confidence** and anxiety, who may be too shy to speak up due to fear of judgment [6,7,24]. For instance, contributors may fear receiving judgment for poor variable names, misused technical terminology, or errors in their code [11]. In addition, contributors may believe that their abilities and talents are predetermined. This mindset may influence mentees' attitudes towards coding errors, perceiving them as reflections of their skills rather than as valuable learning opportunities [11]. Low self-confidence may arise from receiving unfriendly feedback, which can ultimately cause mentees to give up [5]. Additionally, contributors

**Table 4**  
Challenges.

Challenge	Definition	Literature
<b>Social-related Challenges</b>		
 Low self-efficacy	Mentees lack confidence in completing tasks or fear judgment from harsh feedback, making them hesitate to submit code.	[5–7,22–25] [26,27,46,57,58]
 Unacknowledged/uncompensated efforts	Mentors do not receive financial compensation or acknowledgment for their mentoring efforts.	[2,7,22,46,50]
 Task overload/time constraints	Mentors and mentees are overwhelmed with tasks and responsibilities. Additionally, they both struggle with time allocation and transitioning between accounts, such as their GitHub accounts.	[2,5,22,35,53,57] [64]
 Engagement challenges	Mentees do not receive recognition for their contributions, while mentors struggle to understand mentees' backgrounds and support mentees who lack motivation or are resistant to guidance.	[5–7,22,25–27] [31,46,57,63,64]
 Gaps in experience/technical skills	Mentees struggle with comprehension due to differences in experience and often encounter tasks beyond their skill level; the project lacks guidance and resources.	[5–7,22–25] [26–28,31,46,53] [56–58,62–64] [67,68,73]
 Unsupportive community	Mentees face difficulties with a hostile project environment, slow response rates, and securing assistance.	[5,7,22,24–26] [28,31,35,46,56,57] [61,67,68]
 Communication challenges	Mentees or mentors struggle with communication, connection, and relationship-building. Additionally, they face unprofessional and biased treatment from project members.	[2,5,7,22,25,26,31] [28,46,53,57,61,66] [67,68]
 Language skills	Mentees or mentors have difficulty communicating due to limited English proficiency.	[5,7,22,25,26,46] [57,58]
 Working in different time zones	Mentees or mentors experience communication challenges due to time zone and geographical differences.	[5,7,19,22,25,26] [46,57]
<b>Process-related Challenge</b>		
 Bureaucratic process	Projects have internal processes that slowdown or prevent mentees from contributing.	[5,7,24–27] [31,46,53,57,63,67] [68]
 Ambiguous documentation/governance	Projects lack clear documentation, guidelines, resources, and established governance processes.	[5,7,22–25] [26,27,31,33,46,51] [28,57,58,63,64,67] [68,73]
 Difficulties finding tasks to contribute to	Mentees struggle to find feasible tasks, while mentors find it challenging to identify mentee-appropriate tasks.	[5–7,24–27] [28,33,53,56–58] [62,67,68]
<b>Technical-related Challenge</b>		
 Infrastructure hurdles	Projects require complex setup or have poorly designed code and architecture. Mentees' limited technical skills hinder their understanding of technologies. Mentees and mentors use incompatible devices/operating systems.	[5–7,22–25] [26–28,31,46,51] [53,58,62–64,67] [68]

Key: Mentor Mentee Mentor and Mentee

with low self-efficacy may express what Singh [61] defined as a “*low initiative to learn*”. In the study, contributors with this characteristic seemed to lack the desire to learn how to solve problems independently and, instead, showed a preference for having someone else fix them [61]. Notably, these contributors are not the only group struggling with low self-efficacy; ambitious first-time contributors are in a similar situation. Believing that their initial contribution needs to bring significant change, these contributors tend to favor more complex tasks, which they often find themselves unable to complete [5,22].

Non-code contributions are often less recognized than code contributions [50]. Considering the activity of mentoring is a non-code contribution, continuing to mentor (rather than committing code) can be challenging when it is an **unacknowledged effort** [2,22]. Mentors also need financial compensation; without it, their motivation and

availability to mentor other contributors may diminish [22]. This is very prevalent issue in informal mentoring, as mentors often experience burnout due to their efforts being rarely acknowledged and awarded [2].

When contributing, mentors and mentees in OSS projects often struggle with a **heavy workload and time constraints**, in which both parties are overwhelmed with tasks and responsibilities [5,22,53]. Mentors from the Apache projects mentioned they mentor multiple mentees in addition to their other duties [22]. Mentors may also struggle to coordinate their schedules with mentees' schedules, making it challenging to find a suitable time for communication [22]. For example, Apache Software Foundation and Linux kernel have both paid and unpaid contributors; one contributor mentioned that “*the difference in availability of time, makes a huge difference in how people can contribute*” [22].

Mentees' personalities and characteristics can hinder mentors' ability to guide them, which ultimately becomes an **engagement challenge**. One mentor in Balali et al. [5] admitted experiencing difficulties in guiding mentees who showed little interest in actively learning how to do things. Additionally, mentors may encounter contributors who are easy to anger, resistant to compromise, or dismissive of new ideas, making it difficult for mentors to foster a welcoming environment for their mentees [7,46,53].

Engagement becomes further complicated when larger projects or communities overlook contributions from mentees, which often require additional care [5,22]. Moreover, cultural differences between mentees and other members can hinder the former's engagement, especially in OSS projects where communication can be careless, causing misunderstandings and disengagement [7].

Differences in work experience and project seniority between mentees and mentors can create **gaps in experience and technical skills**. Mentees may struggle to understand concepts when assigned tasks above their skill level. Balali et al. [5] found that individuals with extensive experience often struggle to see concepts through the eyes of someone new to a project. Additionally, some mentors may have high technical expertise but may not use foundational skills when mentoring. This can discourage mentees when they fail to understand concepts or processes [22]. Similarly, when mentees cannot complete an assigned task, they may experience a decrease in motivation or even decide to leave the project [5].

An **unsupportive community** relates to a project's toxic environment, slow response times, and limited availability of assistance, which can create challenges for mentees. Mentees may experience delays in receiving an answer from project members; some may not even receive an answer [5,25,84]. In addition, feedback can be harsh and sometimes includes bias. For instance, a participant in Prana et al. [31]'s study stated that they stopped contributing to OSS projects due to racism, gender bias, or unprofessional conduct by maintainers. Bosu and Sultana [66] found female contributors have lower code acceptance and waited longer for initial feedback on their code submissions than male contributors. Additionally, Guizani et al. [46] found that contributors from non-western countries are underrepresented in OSS project communities.

Effective communication is crucial in OSS projects, yet mentees and mentors frequently encounter **communication challenges**. Paul et al. [85] found that the proportion of negative sentiments in code review comments is greater than positive ones. However, it is widely acknowledged that code reviews are not always helpful. Since all conversations in OSS are public, working with unfamiliar contributors is common but can be scary for newcomers [46,53].

Mentoring may also be impacted by **language skills** [5]. An example from language papers showed a typical misunderstanding case between English and Spanish, “¡Viva México, cabrones!” (“Long live Mexico, you bastards!”), when translated into English, switches from a pleasant wish to a hostile and insulting statement [86].

In addition, mentoring often transcends geographical boundaries within OSS communities, presenting unique challenges associated with **working in different time zones**. Such disparities can lead to communication delays, making coordinating real-time interactions challenging [5].

### 5.2. Process-related challenges

As shown in Table 4, there are three mentoring challenges we have synthesized that are imposed by the organization, internal procedures, or practices in OSS communities.

The effectiveness of mentoring may be reduced by **bureaucratic processes** [54]. Contributors may feel discouraged by projects' long and complex processes, as adopting such processes may require more effort than simply contributing to OSS projects [53].

**Ambiguous documentation or governance** poses significant challenges in OSS projects, often hindering both mentees and mentors. Mentees need a clear understanding of a project before they can contribute [31], which can be difficult when the project lacks documentation [26]. For instance, an OSS contributor in Prana et al. [31]'s study felt a project was poorly documented due to its lack of precise code discipline, involvement rules, and README files. In addition, mentors are negatively affected when a project lacks a formal procedure for introducing mentees to the community [22]. Instead of following an established procedure, mentors must devote extra time to creating an introduction from scratch or rely on a recollection of what may have been done before [22].

Selecting the right task is a pivotal step when contributing to OSS, yet both mentees and mentors often have **difficulties finding tasks to contribute to**. This is especially true for newcomers, who usually give up on contributing because they could not find a feasible task [24]. While mentors can assist mentees by identifying tasks for them, this process can be challenging because mentors must consider mentees' backgrounds, goals, and expertise, which are often unclear or not provided [5–7,22,54]. Moreover, when determining the complexity of a task, mentors often rely on their experience and consequently fail to perceive concepts from a mentee's perspectives [6,22]. Tasks that mentors identify as low complexity may still take mentees a significant amount of time and effort to complete [6,22].

### 5.3. Technical-related challenges

Technical challenges regarding **infrastructure hurdles** are related to or caused by technology, such as frameworks, programming languages, and/or tooling used in the project. A mentee's level of knowledge before joining an OSS project may cause them to experience difficulties setting up their development environment before completing tasks. Mentees may feel demotivated when setting up a development environment, as the process can take time [25] or have complex procedures due to outdated infrastructure [46]. While mentors should be able to assist mentees with these technical challenges, providing assistance becomes more difficult when mentors and mentees are not using compatible devices or operating systems [5,22].

## 6. Strategies for enhancing mentoring in OSS

Enhancing the effectiveness of mentoring is crucial for fostering a collaborative environment in OSS. We synthesized 19 strategies from primary studies to address previously identified challenges. We have further categorized these strategies into six groups (as shown in Table 5): knowing the mentees (Fig. 5), establishing governance (Fig. 6), preparing mentors (Fig. 7), tasks recommendation (Fig. 8), supporting mentees (Fig. 9), and rewarding mentors and mentees (Fig. 10).

### 6.1. Knowing the mentees

A practical method to enhance interactions during mentoring is to ensure mentors are familiar with their mentees. Strategies such as **making mentees identifiable** and **asking mentees' interests and backgrounds** enable mentors to provide targeted support. For example, knowing mentees' interests and backgrounds allows mentors to identify and support mentees, fostering a welcoming, responsive, and collaborative environment [5]. In addition, assigning tasks in alignment with a mentee's experience can enhance mentees' performance and motivation [6,22]. As shown in Fig. 5, these strategies help mitigate social and process challenges. In Balali et al. [5]'s study, a mentor suggested that a “*newcomer tag*” can prompt mentors and other project members to be more patient and welcoming. By directly asking mentees about their interests and past experiences, mentors can use specific strategies to monitor mentee engagement and craft practical mentoring approaches [6,22,59].



**Table 5**  
Strategies for mitigating mentoring challenges.

Strategy	Definition	Literature
<b>Knowing the mentees</b>		
#	Making mentees identifiable	Identifying mentees so others can recognize them and be more patient, welcoming, and responsive. [5,6,22,59]
☺	Asking mentees' interests and background	Communicating with mentees to better understand their expertise and interests. [6,9,22,57] [61,64]
<b>Establishing governance</b>		
📄	Keeping documentation concise and updated	Presenting structured documentation with clean and organized information. [5,6,9,22,24] [33,46,51,55] [58,61,64]
📜	Adopting a code of conduct	Adopting a code of conduct to establish norms and expectations for behavior and create a more inclusive and diverse community. [2,4,9,19,22] [31,33,36,46] [57,61]
🗨️	Establishing multiple methods of communication	Agreeing on effective ways to facilitate communication. [5,19,55,57]
<b>Preparing mentors</b>		
👉	Encouraging mentors to be supportive	Proposing that mentors should consistently encourage mentees and hold orientation sessions for mentees. [5,9,11,33] [56,57]
🏠	Having orientation sessions for mentors/mentees	Providing a comprehensive guide for mentors. [5,6,9,19,22] [24,33,46,55] [57,61,64]
<b>Task recommendation</b>		
👤	Allowing mentees to choose their own tasks	Allowing mentees to choose tasks based on their expertise and interests. [6,9,22,33] [55]
☰	Suggesting repetitive tasks	Having mentees gradually learn specific skills and gain confidence before they move to more complex tasks. [6,22,57]
☰	Suggesting small tasks	Assigning mentees small, manageable tasks. [5,6,22,33] [55,58]
📌	Tagging tasks based on their complexity	Having issues tagged based on their complexity allows mentees to choose a task that matches their background. [5,6,22,24] [33,55,57] [58,59]
☰	Suggesting tasks that match mentees' interests and skills	Assigning mentees tasks that are feasible to complete helps determine their skill level and match their interests. [5,6,22,33] [55,58]
<b>Supporting mentees</b>		
📖	Recommending resources to aid mentees in task completion	Recommending a variety of resources such as peers, local groups, technical resources (tutorials and textbooks), and online communities. [11,24,55,57] [64]
👥	Encouraging mentees to share their work	Recommending mentees present their work, publish their code, and present technical jargon. Also consider publicly recognizing their contributions. [5,9,11,55] [57,61]
☰	Providing communication guidelines for mentees	Using a positive tone; acknowledging mentors' help; writing direct, to-the-point questions; and using technical language/forum jargon. [33,61]
👥	Fostering peer discussion groups among demographically similar mentees	Encouraging mentees from shared demographics to create a community or group where they can connect and collaborate. [5,11,46]
👉	Encouraging mentees to seek informal peer mentors throughout the project	Recommending that mentees collaborate and support one another to foster peer mentorships. [2,4,9,11,33] [36,55,57,61] [64]
<b>Rewarding all</b>		
☆	Rewarding/acknowledging mentors' efforts	Recognizing and appreciating the time, effort, and guidance that mentors provide to their mentees in projects. [2,4,19,22] [61]
📅	Rewarding mentees' progress and task completion	Acknowledging the achievements and successful completion of goals or tasks by mentees. [4,5,9,64]

## 6.2. Establishing governance

Effective governance in OSS projects is essential to creating an inclusive and productive environment. Table 5 outlines three strategies for establishing governance: **keeping documentation concise and updated**, **adopting a code of conduct**, and **establishing multiple**

**methods of communication**. The mapping between these strategies and the identified challenges is shown in Fig. 5.

**Keeping documentation concise and updated** supports contributors' decision to continue contributing to projects [36] and directly addresses issues like ambiguous documentation or governance [6,22, 24,33]. Engagement challenges, where mentees may lack motivation

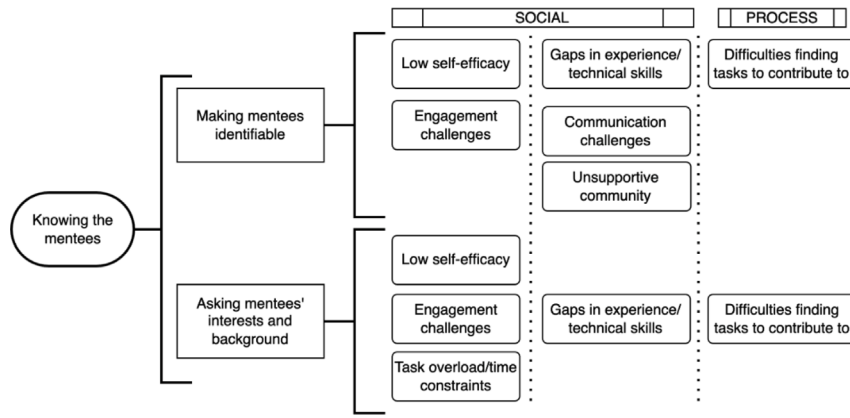


Fig. 5. Strategies mapped to challenges in mentoring (Knowing the mentees).

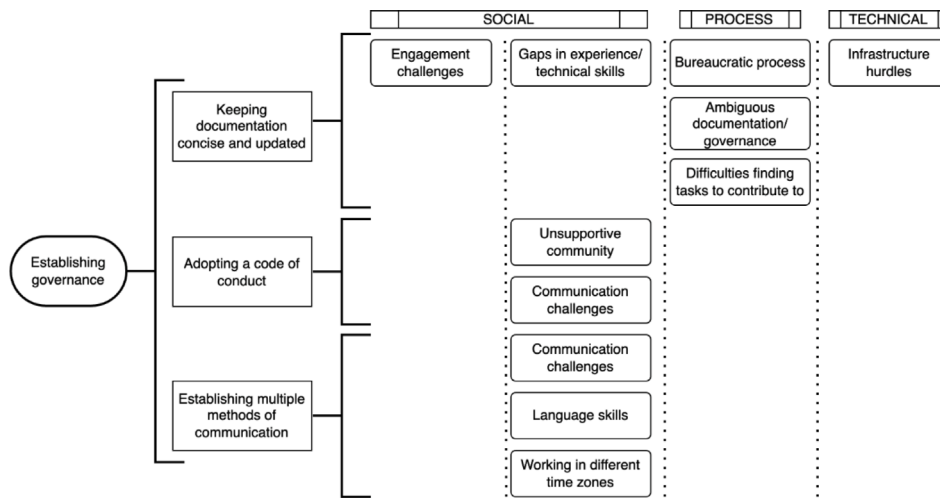


Fig. 6. Strategies mapped to challenges in mentoring (Establishing governance).

or resist guidance, can be mitigated with structured documentation that provides recognition of contributions and clear paths for involvement [9,61]. Up-to-date documentation helps mentors and mentees bridge any experience or technical gaps [55,57,58]. Furthermore, concise and updated documentation aids mentees in selecting feasible tasks and mentors in assigning mentee-friendly tasks [57,61]. Lastly, infrastructure hurdles such as complex project setups, poor code structures, or device incompatibilities, can be mitigated with detailed setup instructions and troubleshooting tips [5,6,24,46].

**Adopting a code of conduct** in OSS projects fosters an inclusive environment, directly addressing social challenges. A clear code of conduct outlines expected behavior, making mentees feel more welcome in the community [2,4,9,19,22,33,61]. With clear guidelines on appropriate interactions, response times, and the assistance process, mentees can navigate the community with greater confidence [4,55,57]. Moreover, a code of conduct establishes culture norms that promote open discussions, fostering relationship-building between mentees, mentors, and other community members [19,36]. Addressing unprofessional and biased interactions makes everyone treat each other kindly, ensuring that both mentors and mentees can work collaboratively in a positive environment [9,22,31,46].

**Establishing multiple methods of communication** directly addresses challenges caused by diverse linguistic backgrounds and different time zones [19,55,57]. For example, varied communication tools, like scheduled calls or collaborative documentation, help ensure continuous mentoring regardless of geographical disparities [22,31,53].

### 6.3. Preparing mentors

Table 5 shows strategies to prepare mentors, including **encouraging mentors to be supportive** and **having orientation sessions for mentors and mentees**. Fig. 7 shows strategies mapping to challenges across social, process, and technical domains.

**Encouraging mentors to be supportive** is fundamental to effective mentoring. For instance, Balali et al. [5] highlighted the importance of mentors working closely with mentees on bugs or issues. Such proactive engagement makes problem-solving more straightforward and builds trust. Effective mentoring engagement also includes monitoring mentees' progress through mentees' blogs, meetings, or other communication channels [9]. Tan et al. [56] mentioned the significance of being patient when newcomers are struggling; mentors should also recognize mentees' achievements by providing encouragement. Moreover, fostering diversity and nurturing an inclusive team culture can encourage mentees by mitigating their fear of judgment [66].

**Having orientation sessions for mentors and mentees** helps ease mentees' entry into the OSS community, especially when onboarding newcomers [5,46]. The onboarding process can be streamlined using step-by-step tutorials or targeted mentoring activities [24,55,57]. By equipping hands-on demonstrations with foundational concepts, mentors can instill confidence and a deeper understanding among their mentees [46,64].

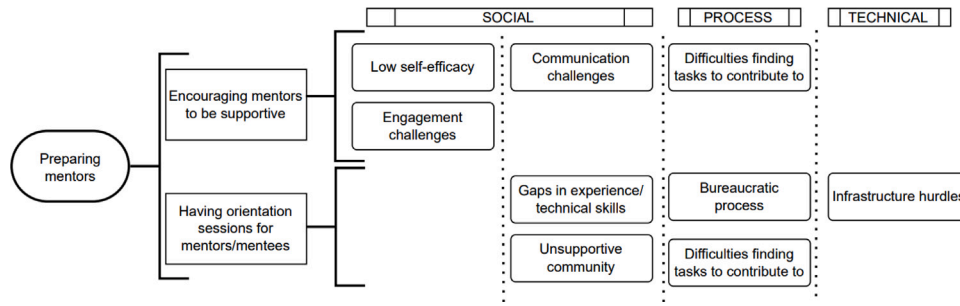


Fig. 7. Strategies mapped to challenges in mentoring (Preparing mentors).

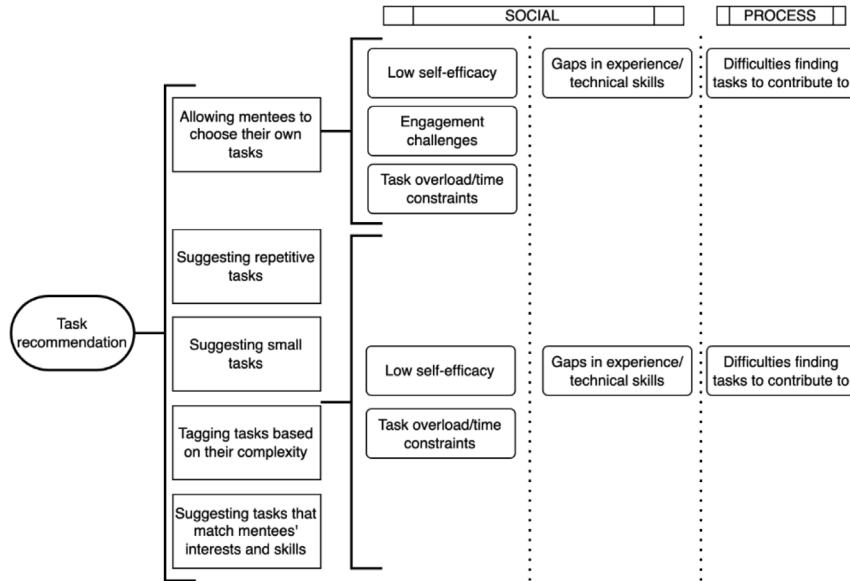


Fig. 8. Strategies mapped to challenges in mentoring (Task recommendation).

6.4. Task recommendation

Table 5 shows strategies for recommending tasks to mentees to mitigate challenges, including offering repetitive tasks and helping mentees familiarize themselves with the projects [6,22]. Fig. 8 presents the mapping of these strategies to social and process challenges.

Steinmacher et al. [22] mentioned that **allowing mentees to choose their own tasks** can address challenges arising from the diverse backgrounds of mentees. Balali et al. [6] suggested mentors should understand mentees' past contributions and suggest **repetitive and small tasks** as a contribution starting point. Beginning with manageable tasks can address challenges like low self-efficacy and task overload. A similar effect can be seen from the strategy of **tagging tasks by complexity**. Steinmacher et al. [33] proposed a tagging system that shows task difficulty, affected modules, required skills, and the contact information of project members. Including these details would assist mentees and mentors in developing a deeper understanding of the task [6,22]. Explicitly marking task complexity also helps mentees **select tasks aligned with their capabilities and interests** [5,22,61].

6.5. Supporting mentees

Table 5 lists five strategies for supporting mentees that may mitigate challenges across social, process, and technical categories. These strategies are mapped to challenges in Fig. 9.

**Recommending resources to aid mentees in task completion**

range from peer connections to in-depth technical materials like tutorials, textbooks, and online communities [11]. Providing code samples, highlighting mentored bugs, and suggesting related artifacts or comparable challenges can narrow the mentees' knowledge gap [24].

**Encouraging mentees to share their work** is essential to mentees' development. Mentees often fear the potential judgment of their work. However, presenting their work to others enables mentees to confront their fears [5]. Over 90% of participants from [57] agreed with having mentees submit regular reports, such as blog posts, with one participant stating, "I asked students to write blog posts during the three months so that I can see the progress". Encouraging public sharing and providing the proper support can boost mentees' confidence in contributing to OSS projects [6].

In addition, **providing communication guidelines for mentees** can ensure mentees seek assistance effectively, present their challenges clearly, and avoid unintentional biases or misunderstandings. As highlighted during the evaluation of FLOSScoach, communication templates were greatly valued [33]. Such templates not only benefit the mentees but also aid community members in quickly addressing issues. Singh [61] found that messages with specific attributes, such as an informative subject line, direct questions, a positive tone, and mention of prior efforts, increased the likelihood of receiving a response. Moreover, introducing this communication guidance resulted in mentees receiving more polite, appreciative, and humble interpersonal interactions from community members [33,61]. An example message template was provided by Steinmacher et al. [33]: "Hello, My name is [your name] and I am a newcomer trying to place my first code contribution to Amarak.

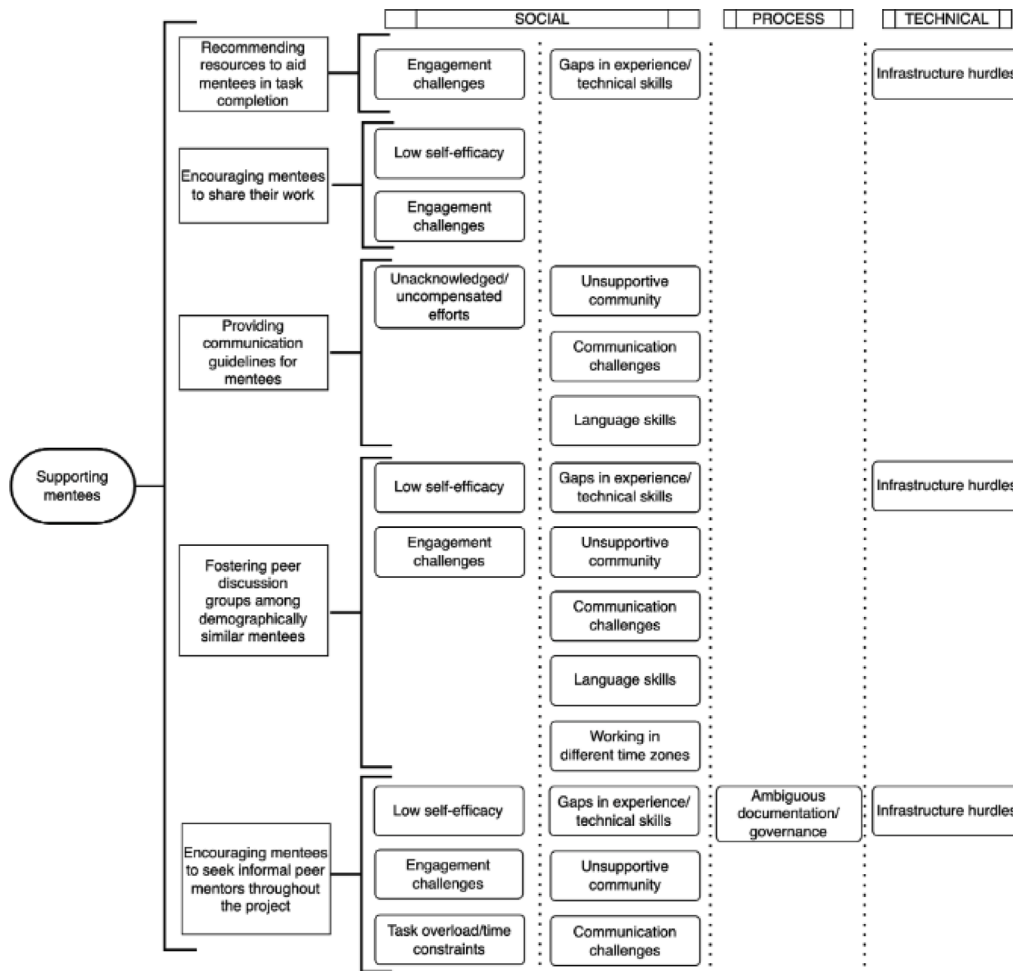


Fig. 9. Strategies mapped to challenges in mentoring (Supporting mentees).

*I am facing problems [during my first steps/finding a task/setting up my workspace]. Can someone help me [clarifying some questions/ mentoring me]? I have already [mention the things you have done already to try to solve your problem] [If you are getting an error, include it in the message]. [Mention the OS you are working on and the tools you are using]*".

**Fostering peer discussion groups among demographically similar mentees** offers more than just technical guidance, they also provide a platform where contributors can bond over shared experiences and cultural differences [5]. The shared language, culture, and aligned career trajectories foster a collaborative environment where contributors can share their problem-solving strategies and more effectively support one another [5,11,46].

Additionally, having cohorts of similar demographics can help create environments where individuals from underrepresented groups feel safe and supported [5,87]. However, as our strategies here are synthesized mainly from literature about onboarding newcomers, it is important to recognize that projects should not always limit themselves to cohesive groups only. One of the most enriching opportunities in OSS communities lies in interacting with contributors from different backgrounds and gaining diverse insights and perspectives. Projects are encouraged to foster diversity and nurture an inclusive team [31,32,87]. By balancing onboarding newcomers with continuously promoting diversity, projects can dynamically create an inclusive collaboration environment.

**Encouraging informal mentoring** in OSS communities is crucial as it diversifies the mentoring landscape. This approach allows mentees to engage with multiple community members, each of whom does not need to dedicate extensive time to any single mentee. Challenges

such as low self-efficacy, task overload, time constraints, technical gaps, an unsupportive community, language barriers, ambiguous documentation, and infrastructure hurdles can all be potentially addressed by informal mentoring. For instance, contributors who believe things like, “programming is not for me”, or, “I am not good at coding”, can be convinced otherwise by receiving guidance from peers who have faced the same challenges [36]. Penoyer et al. [88] highlighted the importance of intrinsic motivation in contributions to platforms like StackOverflow, emphasizing the alignment of such motivations with the ethos of informal mentoring. Through collaborative strategies and prioritizing informal mentoring, OSS communities can bolster their support for all contributors [55,64].

### 6.6. Rewarding all

**Acknowledging and rewarding the contributions of both mentors and mentees** is essential for addressing social challenges, as illustrated in the mapping in Fig. 10.

Even if direct financial compensation is not feasible, alternative methods exist to acknowledge and encourage mentoring in OSS. Steinhilber et al. [22] suggested that OSS projects can create mechanisms that encourage mentoring through non-monetary motivators. Fagerholm et al. [19] mentioned that mentors should receive some form of benefit or compensation for their efforts. Feng et al. [2] proposed a simple tagging system: “@mentor” that allows contributors to publicly acknowledge mentoring efforts. Moreover, simple messages such as saying “thanks” or expressing gratitude in any form can serve as valuable compensation. Additionally, responding with phrases like, “I



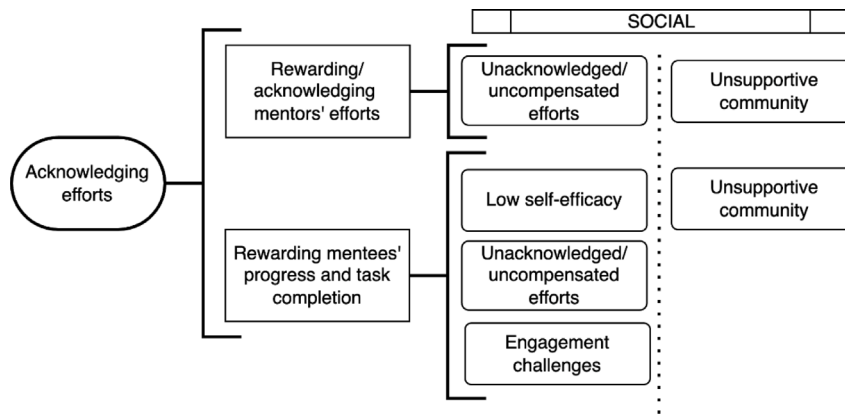


Fig. 10. Strategies mapped to challenges in mentoring (Rewarding all).

tried this”, upon receiving advice is another form of acknowledgment that shows mentees’ responsiveness [61].

Given the considerable time and effort that mentees invest during the mentoring process, they may quickly feel disheartened. As Balali et al. [5] noted, rewards can help motivate mentees to contribute. Strategies that publicly recognize mentees’ contributions, especially among peers, can be beneficial. Other practical approaches include publicly displaying mentees’ names on team pages to boost their exposure and offering them tangible rewards, such as certificates of contribution, which can assist their professional career growth [9]. In addition, Foundjem et al. [64] described a competitive approach where mentors could use reward strategies to motivate mentees to work collaboratively on challenging tasks.

## 7. Discussion

This study conducted an SLR to investigate how mentoring practices in OSS have been characterized in existing literature. The findings have been compiled into a guidance book for practicing mentoring in OSS communities. This section discusses the implications of this study’s findings for future research and practice.

### 7.1. The dynamics and diversity of mentoring in OSS

#### 7.1.1. Dynamic roles of mentoring

Traditionally, formal mentoring involves structured programs where mentors and mentees are systematically paired. For example, mentees in the GSoC are paired with mentors from their chosen OSS projects [4]. In these settings, expectations and boundaries are often explicitly outlined [4,8]. Mentees from formal mentoring programs often return to OSS, not just as contributors but also as mentors [52].

However, mentoring in OSS does not adhere to a singular, one-size-fits-all mold. Informal mentoring is more casual and natural than community interactions, such as when a contributor guides mentees through OSS daily activities [2]. This type of mentoring exists through various channels, such as emails, video conferences, and even informal meetings. Such informal mentoring is the key to boosting a healthy collaboration environment in OSS [2,11]. It is worth noting that the roles of mentors and mentees are not rigidly compartmentalized by characteristics or tenure. Even senior and experienced contributors can find themselves in the mentee position when they face unfamiliar challenges. This highlights that informal mentoring is more suitable for the nature of the OSS. This role dynamic fosters an environment of mutual learning and knowledge transfer [2,46,52]. The statistics show that nearly 90% of Apache Project contributors have acted as either informal mentors or mentees, irrespective of their tenure [2].

#### 7.1.2. Dynamic motivation of mentoring

The motivations behind contributions to OSS are complex and influenced by varying personal, social, and demographic factors [52]. Delving deeper into mentoring in OSS reveals that motivations exhibit additional layers of complexity and are dynamic.

Mentors often find fulfillment in guiding and witnessing the growth of their mentees, deeply rooted in their intrinsic desire to pass on knowledge and build a better community [2,56]. In the meantime, mentors and mentees are joining large organizations’ mentoring programs to build more social connections [2]. In addition, motivations can also shift towards quick promotions, improved salaries, and other extrinsic factors [2,4]. Similarly, mentees operate under a blend of intrinsic and extrinsic motivations. Some are eager to pave the path to joining large organizations like Google, Apache, or Linux, while others are drawn to the global collaboration opportunities that OSS communities offer. This multifaceted, dynamic, and interactive motivation landscape offers valuable insights to OSS projects, researchers, and tool developers.

#### 7.1.3. Dynamic benefits of mentoring

Mentoring goes beyond assisting contributors in overcoming technical and systemic challenges; it also serves as an essential instrument in fostering a sustainable, diverse, and inclusive collaboration environment. Mentors help with technical challenges by guiding mentees and fostering a culture of knowledge-sharing and collaboration. Ultimately, today’s mentees evolve into tomorrow’s mentors [2,9]. Shifting to diversity, mentoring can be strategically designed to pair individuals based on geographical or cultural demographics. This approach directly counteracts biases and fosters a welcoming, inclusive OSS community. This can simultaneously help community growth and sustainability, attracting and retaining contributors for OSS projects’ long-term success.

### 7.2. Strength of strategies for mitigating the challenges in OSS

As discussed in Section 6, we have mapped the proposed evidence-based strategies to mitigate mentoring challenges within OSS to enhance their efficacy and efficiency. These strategies were synthesized from contributors’ feedback with mentoring experiences but have not been formally evaluated or researched. To guide communities in choosing and prioritizing these strategies, we evaluated the strengths of the sources by using a methodology called GRADE (Grading Recommendations Assessment, Development, and Evaluation) [89]. This categorizes evidence strength into high, moderate, low, or very low tiers, a method often employed in software engineering secondary research [90–96]. GRADE recommends evaluating the strength of a study’s evidence through four aspects: design, quality, consistency, and directness.

**Design:** The primary studies we reviewed were all case studies detailing strategies suggested by participants, with no controlled experiments for intervention evaluation. According to GRADE, this observational nature of evidence is “low”. The **quality** of the primary studies was trustworthy, particularly in terms of method description and data collection. Additionally, potential validity threats were acknowledged, and measures were taken to mitigate them. Thus, considering the strengths and shortcomings, we rated the quality “moderate”. **Consistency:** This aspect assesses how similar effect estimates are across different studies. The primary studies lacked consistency, making quantitative result synthesis potentially unreliable. However, each strategy is synthesized from at least three studies (refer to Table 5), indicating a positive consistency trend. **Directness:** We observed that most strategies were derived from interviews, open-ended survey questions, or discussion sections based on results. Thus, there is significant uncertainty about the directness. Therefore, as part of future studies, evaluating the effectiveness of strategies from the contributors’ perspectives is necessary.

### 7.3. Implications

#### 7.3.1. For researchers

In our primary studies, only one study investigated informal mentoring in OSS. Such informal mentoring is still unrecognized and unacknowledged in OSS. However, by investigating pull-request comments, Feng et al. [2] found that over 27% of the pull requests included implicit mentoring comments. Additionally, in their survey, around 75% of the respondents confirmed they had been implicit mentors, with the majority of them serving as implicit mentors for more than two years; however, their efforts were often not acknowledged. This suggests that implicit mentoring in OSS is not limited to review comments; other channels have yet to be investigated. By investigating mentoring in OSS in this study, including definition, channels and activities, motivations and goals, characteristics, forms, and benefits, as discussed in Section 4, future studies on informal mentoring could investigate its feasibility across different types of projects and working environments in OSS. Furthermore, it is worthwhile to investigate if other types of mentoring are exhibited in OSS but invisible.

On the other hand, real-world feedback from active OSS contributors can give vital insights into mentoring approaches. In our study, although most reported mentoring benefits were summarized from empirical studies, it is necessary to investigate and validate these benefits directly from contributors’ perspectives. However, developing a flawless mentoring program might not be feasible for all OSS contributors and projects. We identified challenges and mitigation strategies, as discussed in Sections 5 and 6. Further research is needed to assess the significance and prevalence of each challenge OSS contributors face and investigate the effectiveness of the proposed strategies. This can involve case studies or field studies that delve deeper into the challenges and strategies. Additionally, controlled studies can validate these strategies as suggested by [97]. Such studies would offer OSS communities valuable insights into addressing challenges, strengthening mentoring processes, and improving the community’s sustainability.

#### 7.3.2. For OSS communities

In this study, we systematically investigated how mentoring has been characterized in existing studies. The goals, characteristics, channels, and activities related to mentoring definitions can be used by project managers, core contributors, and PMC (Project Management Committee) [98] chairs to implement and recognize mentoring activities in OSS. In addition, OSS project maintainers who either actively engage in mentoring or are interested in practicing mentoring in OSS can use the table of mentoring challenges (Table 4) as a diagnostic reference to evaluate the health of their micro-project environment.

Table 5 in Section 6 may serve as a “prescription” for enhancing the effectiveness of mentoring. This study identified multiple strategies

for each mentoring challenge, providing flexible options for various projects in OSS. For example, the prevalence of negative sentiments in code review comments [85] could be potentially mitigated by strategies such as preparing mentors, establishing governance, and supporting mentees, which could implicitly improve the proportion of useful code reviews.

However, only giving feedback on what needs improvement can sound negative. For example, Bosu et al. [99] developed a classifier for identifying “useful” code reviews by investigating the characteristics of effective code reviews, including the sentiment of comments, and found half of the comments were “*Extremely Negative*”, and only about half were useful. So, it is possible that comments that were about fixing things could be seen as unfavorable, while on the other hand, comments that did not talk about improvements could be seen as unuseful. Given the extra effort and time that reviewers need to invest, they may also want to consider the best way to frame their review comments to initially provide encouragement before delving into the improvements.

Furthermore, as most contributors to OSS are volunteers, not understanding the benefits of participating in mentoring for both mentees and mentors can make it challenging for OSS communities to attract and retain mentors. OSS communities could use this study as a guide to recognize and reward mentoring activities and raise awareness about mentoring.

For example, *mentees* can benefit from the taxonomy of challenges (see Table 4) and the quick-reference catalog of strategies uncovered in Section 6. By self-checking when encountering similar issues, mentees can proactively communicate with mentors and choose practical strategies to overcome their challenges. In addition, contributors who do not have assigned mentors can refer to strategies for actions to undertake independently when facing barriers, such as seeking informal peer mentors within the project; showing appreciation for other contributors’ time, effort, and guidance; and starting with and continuing to work on small, repetitive tasks before taking on more ambitious ones.

In addition, *mentors* can also benefit from the taxonomy of challenges (see Section 5) by gaining awareness of the potential challenges they may encounter while working with mentees or other contributors in their projects. Additionally, they can use the quick-reference catalog of strategies in Section 6 as a handy tool to support other contributors effectively.

In addition, it would be highly influential if OSS communities included this study in their documentation, like an annual report or code of conduct. That way, this study can serve as a public service advertisement to encourage OSS communities to engage implicitly or explicitly in mentoring activities.

## 8. Threats to validity

As with all research, there are risks associated with this study. We have taken every feasible step to limit the impact of these potential threats, which are discussed in this section.

A SLR study from Zhou et al. [100] identified the four most common threats to validity as non-comprehensive venues (e.g., choice of libraries), bias in selected studies and data extraction, and incorrect search terms in automatic search engines. To avoid these biases, this study followed the SLR process suggested by Kitchenham [39] and Petersen et al. [101]. We first defined the search string by reviewing well-known foundational studies on mentoring in OSS. To validate the defined search strings, we conducted a pilot search. Keyword searching was performed on four well-known digital libraries, as they were used in prior studies [41–43].

The first two authors held weekly meetings during the selection and filtering of primary studies. They conducted a negotiated agreement to evaluate the studies based on the inclusion and exclusion criteria designed by the research team, as discussed in Section 3.

To mitigate potential bias from our choice of libraries and ensure no studies were overlooked, we employed one-step backward and

**Table 6**  
Mentoring in OSS.

Aspects	Explanations	Literature
Characteristics	The roles of mentors and mentees are fluid and not strictly defined.	[2,11,35,38] [63,65]
Motivations	OSS projects do not assist newcomers. Newcomers face significant barriers. All contributors face challenges and barriers. Contributors want to build better communities.	[23,33,35] [5,7,28,31] [5,22,54,68] [2,4,57]
Goals	Assist newcomers in joining OSS projects.  Assist contributors in internalizing community culture in OSS. Assist OSS communities in improving their sustainability.	[5,6,9,19,22] [33,72] [31,36,54]  [2,4,9,54,57] [66]
Channels and Activities	Mentoring in OSS is all-pervasive and exists within routine OSS activities.	[2,4,7,9,19] [22,35,62,63] [65,72,73]
Benefits	Mentors/mentees: Personal growth, career growth, social skills, and altruism.  Projects: Contributor attraction and retention.	[2,5,9,11,19] [22,26,38,52] [55,57,60,79] [2,4,9,54,82]
Forms	Formal mentoring: mentors and mentees are formally connected through mentorship programs. Informal mentoring: mentors/mentees reach out to others to seek/give guidance anytime and anywhere in OSS.	[27,33] [2]

author snowball sampling. After that, we observed that the results from snowballing were more than those from the search string. This discrepancy happened because mentoring in studies can take various forms, such as structured programs like the Summer of Code [9] or studies focusing on onboarding newcomers [58]. However, these diverse forms of mentoring are not always explicitly mentioned in abstracts, titles, or keywords, making them difficult to capture through search strings. Our snowballing sampling method enabled a comprehensive identification of relevant papers. To further reinforce our primary study list, our research team conducted a completeness check, and the completeness ratio was 98%.

In addition, to ensure clarity in understanding mentoring, we initiated a series of preliminary mentoring assessment sessions, as discussed in Section 3. To counteract potential biases from any single researcher, three researchers initially read the full text independently and then reached agreements during the coding sessions. Since classification is a human process relying on subjective criteria, this process may impact this study's results. To mitigate this threat, the first three authors completed three rounds of card sorting and compared their categorizations until they reached an agreement. The findings from this analysis were presented to the entire research team for validation. Given our rigorous verification procedures and analysis methodology, we believe the reliability of our data aligns well with our research goals.

Our research findings are built on case studies of large-scale OSS projects such as Apache, Linux, and Firefox [2,4,5,22]. Thus, our findings may not apply to all OSS projects. For instance, large OSS foundations often have structured mentoring programs with a large pool of mentors and mentees, whereas in small-scale OSS projects, mentoring occurs informally and relies on social connections. Therefore, the challenges and strategies may differ between small and large OSS projects, and additional studies are needed to validate our findings in small-scale OSS projects.

Lastly, given the number of studies included in this SLR, delving into individual results in detail was not feasible. Instead, our primary goal was to synthesize existing studies, providing a systematic overview of fragmented knowledge within mentoring in OSS.

## 9. Conclusion and future work

This study investigated how mentoring practices in OSS have been characterized in existing studies. The empirical evidence shows that

mentoring in OSS benefits mentors and mentees by improving their skills, careers, and social networks. Moreover, mentoring can bolster the success and growth of OSS projects and communities by improving sustainability. However, multiple challenges are associated with mentoring in OSS. This study offers a taxonomy detailing the challenges of mentoring in OSS based on empirical evidence. In addition, this study also provides a quick-reference strategy catalog to mitigate challenges, aiming to enhance the effectiveness of mentoring.

As a next step, we plan to conduct an in-depth analysis of mentoring challenges and identify how they develop and their prevalence. Additionally, it will be essential to evaluate the applicability and effectiveness of the proposed strategies from the viewpoint of OSS contributors.

## CRediT authorship contribution statement

**Zixuan Feng:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Katie Kimura:** Writing – review & editing, Writing – original draft, Visualization, Validation, Resources, Methodology, Investigation, Data curation, Conceptualization. **Bianca Trinkenreich:** Writing – review & editing, Writing – original draft, Visualization, Validation, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Anita Sarma:** Writing – review & editing, Writing – original draft, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Igor Steinhilber:** Writing – review & editing, Writing – original draft, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization.

## Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Anita Sarma reports financial support was provided by National Science Foundation. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

Data will be made available on request.

## Acknowledgments

This work was supported by the National Science Foundation, United States under Grant No. 1901031 and 2303043.

## Appendix

See Table 6.

## References

- [1] K.E. Kram, *Mentoring at Work: Developmental Relationships in Organizational Life*, University Press of America, 1988.
- [2] Z. Feng, A. Chatterjee, A. Sarma, I. Ahmed, A case study of implicit mentoring, its prevalence, and impact in apache, in: *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2022, pp. 797–809.
- [3] A. Sarma, M.A. Gerosa, I. Steinmacher, R. Leano, Training the future workforce through task curation in an oss ecosystem, in: *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, 2016, pp. 932–935.
- [4] J.O. Silva, I. Wiese, D.M. German, C. Treude, M.A. Gerosa, I. Steinmacher, Google summer of code: Student motivations and contributions, *J. Syst. Softw.* 162 (2020) 110487.
- [5] S. Balali, I. Steinmacher, U. Annamalai, A. Sarma, M.A. Gerosa, Newcomers' barriers... is that all? an analysis of mentors' and newcomers' barriers in OSS projects, *Comput. Support. Coop. Work (CSCW)* 27 (3–6) (2018) 679–714.
- [6] S. Balali, U. Annamalai, H.S. Padala, B. Trinkenreich, M.A. Gerosa, I. Steinmacher, A. Sarma, Recommending tasks to newcomers in oss projects: How do mentors handle it? in: *Proceedings of the 16th International Symposium on Open Collaboration*, 2020, pp. 1–14.
- [7] I. Steinmacher, T. Conte, M.A. Gerosa, D. Redmiles, Social barriers faced by newcomers placing their first contribution in open source software projects, in: *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, 2015, pp. 1379–1392.
- [8] Google, Google summer of code, 2022, [Online; accessed 16 March 2022].
- [9] J. Silva, I. Wiese, D.M. German, C. Treude, M.A. Gerosa, I. Steinmacher, A theory of the engagement in open source projects via summer of code programs, in: *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2020, pp. 421–431.
- [10] A. Schilling, S. Laumer, T. Weitzel, Train and retain: the impact of mentoring on the retention of floss developers, in: *Proceedings of the 50th Annual Conference on Computers and People Research*, 2012, pp. 79–84.
- [11] O. Cereceda, D.E. Quinn, A graduate student perspective on overcoming barriers to interacting with open-source software, *FACETS* 5 (1) (2020) 289–303.
- [12] B.R. Ragins, J.L. Cotton, Mentor functions and outcomes: a comparison of men and women in formal and informal mentoring relationships, *J. Appl. Psychol.* 84 (4) (1999) 529.
- [13] L.D. Inzer, C.B. Crawford, A review of formal and informal mentoring: Processes, problems, and design, *J. Leadersh. Educ.* 4 (1) (2005) 31–50.
- [14] Y.P. Bynum, The power of informal mentoring, *Education* 136 (1) (2015) 69–73.
- [15] C. Costa, J. Figueirêdo, J.F. Pimentel, A. Sarma, L. Murta, Recommending participants for collaborative merge sessions, *IEEE Trans. Softw. Eng.* 47 (6) (2021) 1198–1210, <http://dx.doi.org/10.1109/TSE.2019.2917191>.
- [16] M.S. Andreasen, H.V. Nielsen, S.O. Schröder, J. Stage, Usability in open source software development: opinions and practice, *Inf. Technol. Control* 35 (3) (2006).
- [17] P.N. Courant, R.J. Griffiths, *Software and Collaboration in Higher Education: A Study of Open Source Software*, Organization for Open Source Software Study, 2006.
- [18] N. Ducheneaut, Socialization in an open source software community: A socio-technical analysis, *Comput. Support. Coop. Work (CSCW)* 14 (4) (2005) 323–368.
- [19] F. Fagerholm, A.S. Guinea, J. Münch, J. Borenstein, The role of mentoring and project characteristics for onboarding in open source software projects, in: *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, 2014, pp. 1–10.
- [20] The Apache Software Foundation, *Mentoring Programme*, 2020, [Online; accessed 16 March 2022].
- [21] Linux, *Diversity Inclusivity* - Linux Foundation, 2022, [Online; accessed 16 March 2022].
- [22] I. Steinmacher, S. Balali, B. Trinkenreich, M. Guizani, D. Izquierdo-Cortazar, G.G. Cuevas Zambrano, M.A. Gerosa, A. Sarma, Being a mentor in open source projects, *J. Internet Serv. Appl.* 12 (1) (2021) 1–33.
- [23] C. Stanik, L. Montgomery, D. Martens, D. Pucci, W. Maalej, A simple nlp-based approach to support onboarding and retention in open source communities, in: *2018 IEEE International Conference on Software Maintenance and Evolution, ICSME, IEEE*, 2018, pp. 172–182.
- [24] I. Steinmacher, T.U. Conte, M.A. Gerosa, Understanding and supporting the choice of an appropriate task to start with in open source software communities, in: *2015 48th Hawaii International Conference on System Sciences, IEEE*, 2015, pp. 5299–5308.
- [25] I. Steinmacher, A.P. Chaves, T.U. Conte, M.A. Gerosa, Preliminary empirical identification of barriers faced by newcomers to open source software projects, in: *2014 Brazilian Symposium on Software Engineering, IEEE*, 2014, pp. 51–60.
- [26] I. Steinmacher, T.U. Conte, C. Treude, M.A. Gerosa, Overcoming open source project entry barriers with a portal for newcomers, in: *Proceedings of the 38th International Conference on Software Engineering*, 2016, pp. 273–284.
- [27] I. Steinmacher, M. Gerosa, T.U. Conte, D.F. Redmiles, Overcoming social barriers when contributing to open source software projects, *Comput. Support. Coop. Work (CSCW)* 28 (2019) 247–290.
- [28] C. Mendez, H.S. Padala, Z. Steine-Hanson, C. Hilderbrand, A. Horvath, C. Hill, L. Simpson, N. Patil, A. Sarma, M. Burnett, Open source barriers to entry, revisited: A sociotechnical perspective, in: *Proceedings of the 40th International Conference on Software Engineering*, 2018, pp. 1004–1015.
- [29] S.D. Gunawardena, P. Devine, I. Beaumont, L. Garden, E.R. Murphy-Hill, K. Blincoe, Destructive criticism in software code review impacts inclusion, 2022.
- [30] J. Sarker, A.K. Turzo, M. Dong, A. Bosu, Automated identification of toxic code reviews using toxic, *ACM Trans. Softw. Eng. Methodol.* (2023).
- [31] G.A.A. Prana, D. Ford, A. Rastogi, D. Lo, R. Purandare, N. Nagappan, Including everyone, everywhere: Understanding opportunities and challenges of geographic gender-inclusion in oss, *IEEE Trans. Softw. Eng.* (2021).
- [32] H.S. Qiu, A. Nolte, A. Brown, A. Serebrenik, B. Vasilescu, Going farther together: The impact of social capital on sustained participation in open source, in: *2019 IEEE/ACM 41st International Conference on Software Engineering, ICSE, IEEE*, 2019, pp. 688–699.
- [33] I. Steinmacher, C. Treude, M.A. Gerosa, Let me in: Guidelines for the successful onboarding of newcomers to open source projects, *IEEE Softw.* 36 (4) (2018) 41–49.
- [34] K.A. Ericsson, R.T. Krampe, C. Tesch-Römer, The role of deliberate practice in the acquisition of expert performance, *Psychol. Rev.* 100 (3) (1993) 363.
- [35] F. Fagerholm, A.S. Guinea, J. Borenstein, J. Münch, Onboarding in open source projects, *IEEE Software* 31 (6) (2014) 54–61.
- [36] H. Pon-Barry, B.W.-L. Packard, A. St. John, Expanding capacity and promoting inclusion in introductory computer science: a focus on near-peer mentor preparation and code review, *Comput. Sci. Educ.* 27 (1) (2017) 54–77.
- [37] S. Kumar, C. Wallace, Communication strategies for mentoring in software development projects, in: *Proceedings of the 7th International Workshop on Cooperative and Human Aspects of Software Engineering*, 2014, pp. 111–114.
- [38] I. Steinmacher, I.S. Wiese, M.A. Gerosa, Recommending mentors to software project newcomers, in: *2012 Third International Workshop on Recommendation Systems for Software Engineering, RSSE, IEEE*, 2012, pp. 63–67.
- [39] B. Kitchenham, *Procedures for Performing Systematic Reviews*, vol. 33, no. 2004, Keele University, Keele, UK, 2004, pp. 1–26.
- [40] J.W. Castro Llanos, S.T. Acuña Castillo, Differences between traditional and open source development activities, in: *International Conference on Product Focused Software Process Improvement*, Springer, 2012, pp. 131–144.
- [41] T. Ambreen, N. Ikram, M. Usman, M. Niazi, Empirical research in requirements engineering: trends and opportunities, *Requir. Eng.* 23 (1) (2018) 63–95.
- [42] B. Musa Shuaibu, N. Md Norwawi, M.H. Selamat, A. Al-Alwani, Systematic review of web application security development model, *Artif. Intell. Rev.* 43 (2) (2015) 259–276.
- [43] A.W. Khan, M.U. Khan, J.A. Khan, A. Ahmad, K. Khan, M. Zamir, W. Kim, M.F. Ijaz, Analyzing and evaluating critical challenges and practices for software vendor organizations to secure big data on cloud computing: An AHP-based systematic approach, *IEEE Access* 9 (2021) 107309–107332.
- [44] B. Kitchenham, P. Brereton, A systematic review of systematic review process research in software engineering, *Inf. Softw. Technol.* 55 (12) (2013) 2049–2075.
- [45] C. Wohlin, Guidelines for snowballing in systematic literature studies and a replication in software engineering, in: *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering*, 2014, pp. 1–10.
- [46] M. Guizani, A. Chatterjee, B. Trinkenreich, M.E. May, G.J. Noa-Guevara, L.J. Russell, G.G. Cuevas Zambrano, D. Izquierdo-Cortazar, I. Steinmacher, M.A. Gerosa, et al., The long road ahead: Ongoing challenges in contributing to large OSS organizations and what to do, *Proc. ACM Hum.-Comput. Interact.* 5 (CSCW2) (2021) 1–30.



- [47] J.D.O. Silva, I.S. Wiese, D.M. German, I.F. Steinmacher, M.A. Gerosa, How long and how much: What to expect from summer of code participants? in: 2017 IEEE International Conference on Software Maintenance and Evolution, ICSME, IEEE, 2017, pp. 69–79.
- [48] C. Hannebauer, M. Book, V. Gruhn, An exploratory study of contribution barriers experienced by newcomers to open source software projects, in: Proceedings of the 1st International Workshop on CrowdSourcing in Software Engineering, 2014, pp. 11–14.
- [49] Supplementary Spreadsheet, 2023, Online spreadsheet. URL: <https://shorturl.at/bqty2>.
- [50] B. Trinkenreich, M. Guizani, I. Wiese, A. Sarma, I. Steinmacher, Hidden figures: Roles and pathways of successful oss contributors, Proc. ACM Hum.-Comput. Interact. 4 (CSCW2) (2020) 1–22.
- [51] W. Ding, P. Liang, A. Tang, H. Van Vliet, M. Shahin, How do open source communities document software architecture: An exploratory survey, in: 2014 19th International Conference on Engineering of Complex Computer Systems, IEEE, 2014, pp. 136–145.
- [52] M. Gerosa, I. Wiese, B. Trinkenreich, G. Link, G. Robles, C. Treude, I. Steinmacher, A. Sarma, The shifting sands of motivation: Revisiting what drives contributors in open source, in: 2021 IEEE/ACM 43rd International Conference on Software Engineering, ICSE, IEEE, 2021, pp. 1046–1058.
- [53] A. Lee, J.C. Carver, A. Bosu, Understanding the impressions, motivations, and barriers of one time code contributors to FLOSS projects: a survey, in: 2017 IEEE/ACM 39th International Conference on Software Engineering, ICSE, IEEE, 2017, pp. 187–197.
- [54] M. Zhou, A. Mockus, What make long term contributors: Willingness and opportunity in oss community, in: 2012 34th International Conference on Software Engineering, ICSE, IEEE, 2012, pp. 518–528.
- [55] K. Carillo, S. Huff, B. Chawner, What makes a good contributor? Understanding contributor behavior within large Free/Open Source Software projects—A socialization perspective, J. Strateg. Inf. Syst. 26 (4) (2017) 322–359.
- [56] X. Tan, M. Zhou, Z. Sun, A first look at good first issues on GitHub, in: Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2020, pp. 398–409.
- [57] X. Tan, M. Zhou, L. Zhang, Understanding mentors' engagement in OSS communities via google summer of code, IEEE Trans. Softw. Eng. (2023).
- [58] I. Santos, I. Wiese, I. Steinmacher, A. Sarma, M.A. Gerosa, Hits and misses: Newcomers' ability to identify skills needed for oss tasks, in: 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering, SANER, IEEE, 2022, pp. 174–183.
- [59] I. Steinmacher, I. Wiese, T.U. Conte, M.A. Gerosa, Increasing the self-efficacy of newcomers to open source software projects, in: 2015 29th Brazilian Symposium on Software Engineering, IEEE, 2015, pp. 160–169.
- [60] C. Hannebauer, V. Gruhn, Motivation of newcomers to FLOSS projects, in: Proceedings of the 12th International Symposium on Open Collaboration, 2016, pp. 1–10.
- [61] V. Singh, Newcomer integration and learning in technical support communities for open source software, in: Proceedings of the 17th ACM International Conference on Supporting Group Work, 2012, pp. 65–74.
- [62] I. Rehman, D. Wang, R.G. Kula, T. Ishio, K. Matsumoto, Newcomer OSS-candidates: Characterizing contributions of novice developers to GitHub, Empir. Softw. Eng. 27 (5) (2022) 1–20.
- [63] C. Hannebauer, V. Gruhn, On the relationship between newcomer motivations and contribution barriers in open source projects, in: Proceedings of the 13th International Symposium on Open Collaboration, 2017, pp. 1–10.
- [64] A. Foundjem, E. Eghan, B. Adams, Onboarding vs. Diversity, productivity and quality—Empirical study of the OpenStack ecosystem, in: 2021 IEEE/ACM 43rd International Conference on Software Engineering, ICSE, IEEE, 2021, pp. 1033–1045.
- [65] G. Canfora, M. Di Penta, R. Oliveto, S. Panichella, Who is going to mentor newcomers in open source projects? in: Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering, ACM, 2012, p. 44.
- [66] A. Bosu, K.Z. Sultana, Diversity and inclusion in open source software (OSS) projects: Where do we stand? in: ESEM, IEEE, 2019, pp. 1–11.
- [67] I. Steinmacher, M.A.G. Silva, M.A. Gerosa, D.F. Redmiles, A systematic literature review on the barriers faced by newcomers to open source software projects, Inf. Softw. Technol. 59 (2015) 67–85.
- [68] I. Steinmacher, M.A.G. Silva, M.A. Gerosa, Barriers faced by newcomers to open source projects: a systematic review, in: IFIP International Conference on Open Source Systems, Springer, 2014, pp. 153–163.
- [69] B.G. Glaser, Open coding descriptions, Grounded Theory Rev. 15 (2) (2016) 108–110.
- [70] D.R. Garrison, M. Cleveland-Innes, M. Koole, J. Kappelman, Revisiting methodological issues in transcript analysis: Negotiated coding and reliability, Internet High. Educ. 9 (1) (2006) 1–8.
- [71] J.H. Forman, L.J. Damschroder, Qualitative content analysis, 2007.
- [72] F. Fagerholm, P. Johnson, A.S. Guinea, J. Borenstein, J. Münch, Onboarding in open source software projects: A preliminary analysis, in: 2013 IEEE 8th International Conference on Global Software Engineering Workshops, IEEE, 2013, pp. 5–10.
- [73] T. Kilamo, M. Nurminen, T. Männistö, Supporting management of hybrid OSS communities—A stakeholder analysis approach, in: Product-Focused Software Process Improvement: 17th International Conference, PROFES 2016, Trondheim, Norway, November 22–24, 2016, Proceedings, Vol. 10027, Springer, 2016, p. 102.
- [74] T. Hirao, S. McIntosh, A. Ihara, K. Matsumoto, Code reviews with divergent review scores: An empirical study of the openstack and qt communities, IEEE Trans. Softw. Eng. 48 (1) (2020) 69–81.
- [75] E. Eng, Qt GUI toolkit: Porting graphics to multiple platforms using a gui toolkit, Linux J. 1996 (31es) (1996) 2–es.
- [76] G. Cole, The value of mentoring: A mutually beneficial experience for mentor and mentee, Dev. Learn. Organ.: Int. J. (2015).
- [77] S.P. Kissau, E.T. King, Peer mentoring second language teachers: A mutually beneficial experience? Foreign Lang. Ann. 48 (1) (2015) 143–160.
- [78] P. Garcia, M. Perez, D. Farrell, S. Bork, B. Ericson, J.-L. Mondisa, Supporting mutually beneficial near-peer mentoring relationships within computing education programs, in: 2021 Conference on Research in Equitable and Sustained Participation in Engineering, Computing, and Technology, RESPECT, IEEE, 2021, pp. 1–9.
- [79] R. Spencer, Naturally occurring mentoring relationships involving youth, Blackwell Handb. Mentor.: Multiple Perspect. Approach (2007) 97–117.
- [80] T.G. Brashear, D.N. Bellenger, J.S. Boles, H.C. Barksdale Jr., An exploratory study of the relative effectiveness of different types of sales force mentors, J. Pers. Sell. Sales Manag. 26 (1) (2006) 7–18.
- [81] G.L. Ciampaglia, D. Taraborelli, MoodBar: Increasing new user retention in wikipedia through lightweight socialization, in: Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing, 2015, pp. 734–742.
- [82] A. Schilling, S. Laumer, T. Weitzel, Who will remain? an evaluation of actual person-job and person-team fit to predict developer retention in floss projects, in: 2012 45th Hawaii International Conference on System Sciences, IEEE, 2012, pp. 3446–3455.
- [83] E.H. Trainer, C. Chaihirunkarn, A. Kalyanasundaram, J.D. Herbsleb, Community code engagements: summer of code & hackathons for community building in scientific software, in: Proceedings of the 18th International Conference on Supporting Group Work, 2014, pp. 111–121.
- [84] H.S. Shim, G.L. Roth, Sharing tacit knowledge among expert teaching professors and mentees: Considerations for career and technical education teacher educators, J. STEM Teach. Educ. 44 (4) (2007) 4.
- [85] R. Paul, A. Bosu, K.Z. Sultana, Expressions of sentiments during code reviews: Male vs. female, in: 2019 IEEE 26th International Conference on Software Analysis, Evolution and Reengineering, SANER, IEEE, New York, NY, USA, 2019, pp. 26–37.
- [86] S.B.K. de Marlangeon, L.A. Juez, A typology of verbal impoliteness behaviour for the english and spanish cultures, Rev. Esp. Lingüíst. Apl. (25) (2012) 69–92.
- [87] A. Ramakrishnan, D. Sambuco, R. Jagsi, Women's participation in the medical profession: insights from experiences in Japan, Scandinavia, Russia, and Eastern Europe, J. Women's Health 23 (11) (2014) 927–934.
- [88] S. Penoyer, B. Reynolds, B. Marshall, P.W. Cardon, Impact of users' motivation on gamified crowdsourcing systems: A case of stackoverflow, Issues Inf. Syst. 19 (2) (2018).
- [89] T. Dybå, T. Dingsøyr, Strength of evidence in systematic reviews in software engineering, in: Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, 2008, pp. 178–187.
- [90] T. Dybå, T. Dingsøyr, Empirical studies of agile software development: A systematic review, Inf. Softw. Technol. 50 (9–10) (2008) 833–859.
- [91] V. Alves, N. Niu, C. Alves, G. Valença, Requirements engineering for software product lines: A systematic literature review, Inf. Softw. Technol. 52 (8) (2010) 806–820.
- [92] J.d.O. Alexandre, P. Kruchten, M.L. do E Pedrosa, H.R. de Almeida Neto, H.P. de Moura, State of the art of agile governance: A systematic review, Int. J. Comput. Sci. Inf. Technol. 6 (5) (2014) 121.
- [93] F.S. Silva, F.S.F. Soares, A.L. Peres, I.M. de Azevedo, A.P.L. Vasconcelos, F.K. Kamei, S.R. de Lemos Meira, Using CMMI together with agile software development: A systematic review, Inf. Softw. Technol. 58 (2015) 20–43.
- [94] A.S. Guinea, G. Nain, Y. Le Traon, A systematic review on the engineering of software for ubiquitous systems, J. Syst. Softw. 118 (2016) 251–276.
- [95] H.G. Gurbuz, B. Tekinerdogan, Model-based testing for software safety: a systematic mapping study, Softw. Qual. J. 26 (4) (2018) 1327–1372.
- [96] B. Trinkenreich, I. Wiese, A. Sarma, M. Gerosa, I. Steinmacher, Women's participation in open source software: A survey of the literature, ACM Trans. Softw. Eng. Methodol. (TOSEM) 31 (4) (2022) 1–37.
- [97] J.J. Schlesselman, Case-Control Studies: Design, Conduct, Analysis, vol. 2, Oxford University Press, 1982.

- [98] D. Riehle, How open source is changing the software developer's career, *Computer* 48 (5) (2015) 51–57.
- [99] A. Bosu, M. Greiler, C. Bird, Characteristics of useful code reviews: An empirical study at microsoft, in: 2015 IEEE/ACM 12th Working Conference on Mining Software Repositories, IEEE, 2015, pp. 146–156.
- [100] X. Zhou, Y. Jin, H. Zhang, S. Li, X. Huang, A map of threats to validity of systematic literature reviews in software engineering, in: 2016 23rd Asia-Pacific Software Engineering Conference, APSEC, IEEE, 2016, pp. 153–160.
- [101] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, Systematic mapping studies in software engineering, in: 12th International Conference on Evaluation and Assessment in Software Engineering (EASE) 12, 2008, pp. 1–10.