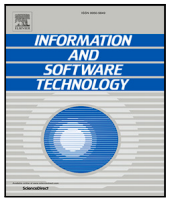
Information and Software Technology 177 (2025) 107568

Contents lists available at ScienceDirect

Information and Software Technology

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



Software solutions for newcomers' onboarding in software projects: A systematic literature review

Italo Santos [a,*], Katia Romero Felizardo [a,b], Igor Steinmacher [a], Marco A. Gerosa [a]

[a] Northern Arizona University, Flagstaff, AZ, USA
[b] Federal Technological University of Paraná, PR, Brazil

ARTICLE INFO

Dataset link: https://zenodo.org/records/10211339

Keywords:
Systematic literature review
Software projects
Open source software
Onboarding
Turnover
Tool
Newcomers
Novices

ABSTRACT

Context: Newcomers joining an unfamiliar software project face numerous barriers; therefore, effective onboarding is essential to help them engage with the team and develop the behaviors, attitudes, and skills needed to excel in their roles. However, onboarding can be a lengthy, costly, and error-prone process. Software solutions can help mitigate these barriers and streamline the process without overloading senior members.
Objective: This study aims to identify the state-of-the-art software solutions for onboarding newcomers.
Methods: We conducted a systematic literature review (SLR) to answer six research questions.
Results: We analyzed 32 studies about software solutions for onboarding newcomers and yielded several key findings: (1) a range of strategies exists, with recommendation systems being the most prevalent; (2) most solutions are web-based; (3) solutions target a variety of onboarding aspects, with a focus on process; (4) many onboarding barriers remain unaddressed by existing solutions; (5) laboratory experiments are the most commonly used method for evaluating these solutions; and (6) diversity and inclusion aspects primarily address experience level.
Conclusion: We shed light on current technological support and identify research opportunities to develop more inclusive software solutions for onboarding. These insights may also guide practitioners in refining existing platforms and onboarding programs to promote smoother integration of newcomers into software projects.

1. Introduction

Onboarding has become extremely relevant in a volatile labor and technological market [1–3]. In the software industry, onboarding is the process of integrating new developers into a software development team [3–6]. During onboarding, newcomers have to adapt to the new environment, understand the requirements of their role, and collaborate effectively with the team. Effective onboarding is essential for ensuring a smooth transition and productivity of the new members [7,8].

Newcomers have to consume new information in a short time, use new development processes, collaborate with new colleagues in a different work environment, and understand large and complex source code structures [9]. Thus, newcomers need significant time before being considered ready to work on a project to the best of their ability [10]. Companies strive to get the most out of their employees, and new team members look to prove themselves in the new setting [11]. Inadequately supported onboarding can lead to a substantial waste of company resources and talent and can be a source of frustration for all involved parties. According to Buchan et al. [12], poor onboarding can lead to anxiety in new team members due to their perceived lack of team contribution and trust. Additionally, it may result in a decline in the team's overall productivity [13,14]. Therefore, approaches supporting onboarding are desirable to help newcomers and mitigate these problems.

Although this is critical to any software development team, this is key to open source software (OSS) projects, which are expected to provide environments with low entry barriers to onboarding newcomers to maintain project sustainability [15,16]. Nevertheless, newcomers face challenging barriers in the OSS context. Social interaction, previous knowledge, finding a way to start, documentation, and technical hurdles are examples of barriers [17]. Consequently, these barriers posed during the onboarding may lead newcomers to give up on contributing [16]. Therefore, investigating newcomer onboarding in this context is crucial [16,18–23].

* Corresponding author.
E-mail addresses: italo_santos@nau.edu (I. Santos), katiascannavino@utfpr.edu.br (K.R. Felizardo), igor.steinmacher@nau.edu (I. Steinmacher), marco.gerosa@nau.edu (M.A. Gerosa).

https://doi.org/10.1016/j.infsof.2024.107568
Received 23 December 2023; Received in revised form 26 August 2024; Accepted 27 August 2024
Available online 2 September 2024

Onboarding strategies can include courses [9,24], bootcamps [3], and mentorship [3,25–27]. These strategies are known for being costly in terms of time and money and lack scalability [26]. For example, senior developers pointed out that working as mentors impacts their productivity [3]. Having new hires read relevant source code without assistance is also costly regarding time investment [28], leading to long adjustment periods.

Some solutions can be automated to facilitate the onboarding process for a large number of projects and newcomers. Such software solutions are still not largely used in practice but have been investigated in the scientific literature. However, this evidence is spread across different venues and disciplines.

This study aims to identify studies that propose software solutions that facilitate the onboarding of newcomers in software projects [7] using a Systematic Literature Review (SLR). Software solutions can actively support diverse aspects of onboarding. The literature is vast and covers many of these aspects, such as reducing onboarding time and cost for companies [1], supporting independent learning [6], supporting the need for training [29], helping newcomers to deal with the high amount of information [23,30], and supporting newcomers in understanding complex source code structures [26,27].

Our systematic literature review consolidates this information into a single resource, providing a clearer understanding of the existing software solutions that facilitate onboarding newcomers in software projects. This paper presents a comprehensive analysis of 32 primary studies published until 2023 to identify the state-of-the-art related software solutions for newcomers' onboarding and to identify potential gaps that can be addressed by developing new software solutions. The outcomes of this study inform practitioners and researchers working on smoothing onboarding for newcomers and provide a basis for further research in this area.

We have organized the remainder of this paper as follows. Section 2 details the SLR planning and its execution. Next, Section 3 presents the results and answers the study research questions. Section 4 outlines the paper discussion, Section 5, the implications. The threats to validity are discussed in Section 6. In Section 7, we introduce related work. Finally, Section 8 concludes the work concerning our main findings and suggests future work.

2. Research method

We conducted this study as a Systematic Literature Review (SLR) based on guidelines established for the Software Engineering domain [31]. We employed synthesis procedures similar to other SLRs (e.g., [5, 32]) to identify data patterns about solutions to facilitate newcomers' onboarding in software projects. In particular, we evaluated how far the proposed software solutions mitigate newcomers' barriers to joining software projects and how they address the diversity and inclusion of newcomers.

In this section, we detail the protocol used for the systematic literature review, specifying the research questions and defining the search strategy, selection process, selection criteria, and data collection and synthesis processes. We present the results for the research questions in Section 3.

2.1. Research questions

According to Park and Jensen [33], the continuous influx of newcomers and their active participation in development activities play a vital role in the success of software projects. In this context, this SLR aims to identify studies that propose software solutions that facilitate the onboarding processes for newcomers in software projects. We translated our research goal into the following research questions (RQs):

RQ1. What software solutions are proposed in the literature to facilitate newcomers' onboarding in software projects?

As the field of software development continually evolves, the challenges newcomers face during their onboarding process continue. By answering RQ1, we aim to provide a comprehensive understanding of the existing software solutions for supporting newcomers during the onboarding process. By leveraging existing knowledge and commonly used onboarding solutions, organizations can create a smooth onboarding process, promote productivity, and foster a positive team dynamic.

RQ2. How were the software solutions implemented?

While numerous software solutions have been proposed to enhance newcomers' integration into software projects, understanding the specific implementation details is essential to assess their feasibility, effectiveness, and real-world impact. Answering RQ2 enables the software development community to identify successful approaches and technological gaps.

RQ3. How do the proposed software solutions improve newcomers' onboarding?

Onboarding is a complex and multifaceted process. By answering RQ3, we aim to provide evidence and insights into which aspects of onboarding have been addressed by existing solutions. Understanding the goals of those proposed software solutions enables software projects to find solutions that better address their needs.

RQ4. How do the software solutions mitigate newcomers' barriers to joining software projects?

Newcomers face a variety of onboarding barriers [17]. By answering RQ4, we aim to gain insights into how existing software solutions address these barriers. This investigation aims to guide software projects in selecting appropriate software solutions and to identify potential gaps in the field.

RQ5. What research strategies were employed to evaluate the software solutions?

Software projects considering the adoption of a software solution may be particularly interested in how these solutions have been evaluated, especially in practical settings. Addressing RQ5 helps to understand the research strategies employed to evaluate the quality and applicability of these solutions, guiding transfer to practice and future research in the field.

RQ6. How do the software solutions address the diversity and inclusion of newcomers?

Literature shows [34–36] that the way information currently provided in software projects (e.g., documentation, issue description) benefits certain cognitive styles (e.g., those who learn by tinkering) over others (e.g., process-oriented learners). The prevalent approach in building software solutions is more beneficial to the majority, and the literature shows that not considering the minorities in the design increases barriers to their participation [37]. This is counter-intuitive to most designers because software is often built/designed by representatives of the majorities. Therefore, the information architecture of documentation and tools usually appeals to those who have high self-efficacy and are motivated by individual pursuits such as intellectual stimulation, competition, and learning technology for fun. These pursuits cater to characteristics associated with men, which can neglect women and other contributors who may have different motivations and personal characteristics [34]. RQ6 brings this awareness and contributes to the effort of making projects more welcoming for people who do not follow the cognitive and behavioral standards of the majority.

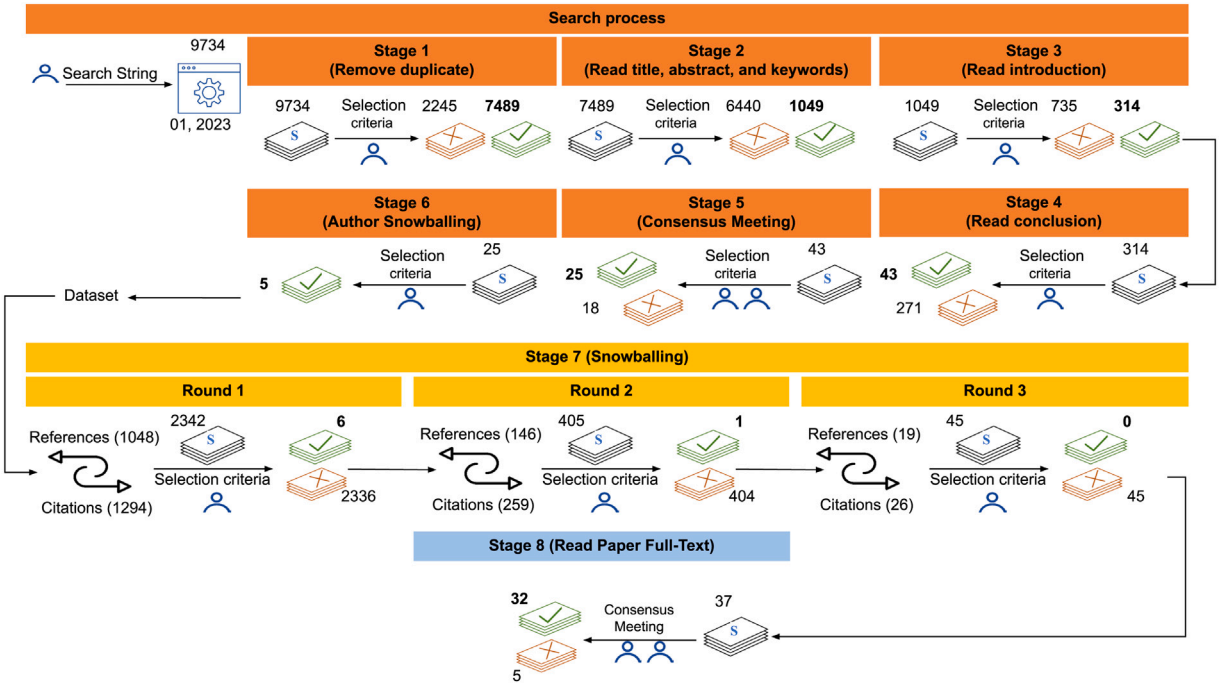


Fig. 1. Search and selection process describing the number of studies selected in each stage: Ⓢ: studies | √: included | ×: excluded.

2.2. Selection criteria

For the selection criteria, we established one Inclusion Criteria (IC) and five Exclusion Criteria (EC), detailed below:

IC1—The primary study proposes software solutions for newcomers' onboarding in software projects;
EC1—The study does not have an abstract;
EC2—The study is just published as an abstract;
EC3—The study is not written in English;
EC4—The study is an older version of another study already considered;
EC5—The study is not a scientific paper—such as editorials, summaries of keynotes, workshop proposals/reports, and tutorials.

In our review, we focused on papers that propose software solutions — such as tools, applications, or platforms — designed to facilitate the onboarding of newcomers to software projects. These software solutions support various aspects of onboarding, such as reducing time and cost and aiding newcomers learning. We excluded papers that only investigated the onboarding process without proposing software solutions, such as studies that examined the code of conduct, as they do not align with our focus on automated and software-driven solutions. To clarify, we consider that software solutions are alternatives to mitigate onboarding barriers and offer (semi-) automated support, helping newcomers adapt to new environments, understand complex systems, and access the necessary information without requiring constant human guidance.

2.3. Search strategy and selection process

We systematically searched for relevant studies, as illustrated in Fig. 1. The search process included eight stages, applied sequentially, as follows.

Stage 1. For our search string formulation, we defined our population as 'software projects' and the intervention as 'onboarding newcomers' derived from our research questions. Upon careful analysis of terms associated with the population and intervention components, we formulated a set of keywords and their synonyms to construct our search string. The selection of these synonyms was carried out with the assistance of domain experts, and we also drew upon relevant SLR [5,38] to enrich our collection of synonyms further. Subsequently, we performed a pilot search on Google Scholar to fine-tune the search string, and we created a control group containing a set of five (5) studies previously known by the authors for search string validation [1,21,39–41]. The first author (named R1 — Researcher 1 — from this point on) used the keywords and their respective synonyms, presented in Table 1, to build the search string, as detailed in Table 2. The final search string was derived after numerous trials and iterations, considering the studies established as the control group. R1 applied the search string in the most commonly used publication databases in Computer Science [31,42,43], including IEEE Xplore,[1] ACM digital library,[2] Scopus,[3] Springer link,[4] and Web of science.[5] We did not include Google Scholar in our search because it can produce inaccurate results and has considerable overlap with other databases we used in our search. For example, Valente et al. [44] found that Scopus alone returns 93% of relevant papers in a computer science literature review, and although Google Scholar's recall is high, its precision is low due to the inclusion of non-peer-reviewed documents like arXiv, PhD theses, and technical reports. Similarly, Harzing and Alakangas [45] concluded that while Google Scholar provides broader coverage for most disciplines, Web of Science and Scopus yield fairly similar results. This is consistent with the concerns of other researchers [46–48] about Google Scholar's effectiveness in retrieving primary studies. For instance, Kitchenham et al. [47] suggest that Google Scholar is more suitable for searching grey literature, which was not the focus of our review.
Our search across the five selected digital libraries yielded 9734 candidate studies, was conducted in January 2023, and no period restrictions were applied. Subsequently, we removed 2245 duplicate

[1] http://ieeexplore.ieee.org
[2] http://portal.acm.org
[3] http://www.scopus.com
[4] https://link.springer.com/
[5] https://www.webofscience.com/wos/woscc/basic-search

Table 1
Keyword and synonyms used to build search string terms.

Keyword	Synonyms
Software project	"software project", "software engineering", "software development" OSS, "open source", "open-source", "free software", FOSS, FLOSS, "OSS projects", "open source software"
Onboarding newcomers	Onboarding, onboard, joining, engagement, newcomer, contributors, novice, newbie, "new developer", "early career", "new member", "new contributor", "new people", beginner, "potential participant", joiner, "new committer"

Table 2
Final search string.

("software project" **OR** "software engineering" **OR** "software development" **OR** "open source" **OR** "open-source" **OR** "free software" **OR** FOSS **OR** FLOSS **OR** OSS **OR** "OSS projects" **OR** "open source software") AND ("joining process" **OR** onboarding **OR** onboard **OR** joining **OR** engagement **OR** newcomer **OR** novice **OR** newbie **OR** "new developer" **OR** "early career" **OR** "new member" **OR** "new contributor" **OR** beginner **OR** "potential participant" **OR** joiner **OR** entrance)

candidate studies, resulting in an initial set of 7489 unique candidate studies to commence the selection process. To mitigate biases related to the search string, we included the forward and backward snowballing approaches to find other relevant studies that could not be returned in the initial search.

Stage 2. Our study selection was a multistage process [31]. Initially, R1 reviewed the candidate studies' titles and abstracts to assess their adherence to the inclusion and exclusion criteria described in Section 2.2, and 1049 studies were included. We applied the selection criteria, and unless a study could be excluded only based on the title and abstract, we obtained its full text to have additional information [31].

Stage 3. Since many SE abstracts are too poor to rely on when selecting studies [49], we decided to exclude a study after reading other sections (such as the introduction and, if necessary, the conclusions). R1 re-evaluated and added the reading of the introduction section of the studies selected in the previous stage. 314 candidate studies were included since they matched the inclusion criteria. For example, in some cases, we identified that a software solution was proposed reading the abstract. However, whether it could support onboarding newcomers needs to be clarified, as required in our inclusion criteria (IC1—The primary study proposes software solutions for newcomers' onboarding in software projects). Therefore, we read the introduction section to clarify the context. In cases of doubt, we also read the conclusion to understand better how the software solutions proposed support newcomers to ensure that the IC1 was met.

Stage 4. Aiming to obtain a new layer of information, in addition to the sections already read, the conclusions of the studies included in the previous stage were then analyzed, and R1 reapplied the selection criteria, resulting in 43 studies included.

Stage 5. At this stage, another researcher (R2) applied the selection criteria (Section 2.2) in the previously selected candidate studies, independently. They reached 88% of agreement. R1 and R2 conducted a consensus decision-making meeting for the cases of disagreement. When a consensus was not possible (only 1 case), we included the study to avoid premature exclusion. As a result, we selected 25 primary studies and excluded 18.

Stage 6. R1 applied author snowballing on the 25 studies selected by the search in the digital libraries. R1 searched other papers published by the 68 authors of these 25 studies by checking the authors' Google Scholar profiles. In cases where R1 could not find the author's profile page, R1 scrutinized other sources, such as ACM Digital Library, IEEE Xplore, and DBLP. R1 found 5436 other candidate papers, which were analyzed using the same process used for papers found in digital libraries: title, abstract, and keyword analysis (Stage 2), resulting in 5 more studies included.

Stage 7. We also conducted citation backward and forward snowballing to mitigate the risk of missing studies. R1 conducted full snowballing, which identifies new studies based on the starting set, followed by backward and forward snowballing, according to the guidelines for snowballing proposed in [74]. R1 performed three rounds of full snowballing, applying the same selection process for papers found in digital libraries: title, abstract, and keyword analysis (Stage 2).

1. Round 1. Thirty studies formed the starting set. The backward snowballing resulted in 1048 papers and the forward in 1294 papers. Six (6) studies met the inclusion criteria and were included.
2. Round 2. R1 analyzed the six (6) studies selected in Round 1 and applied backward snowballing, finding 146 other candidate studies. The forward snowballing identified 259 studies. In this round, we included only one (1) study.
3. Round 3. R1 analyzed the study selected in Round 2 and applied the backward and forward snowballing, finding 19 and 26 studies, respectively. We did not include any new studies in this round.

Stage 8. R1 and R2 independently read the full text of the 37 candidate studies at this stage and jointly conducted a consensus decision-making meeting with 100% agreement, including 32 out of the 37 studies (Table 3). Supplementary material related to this paper can be found online[6] and includes files detailing aspects of the study selection and analysis process.

2.4. Data collection and analysis

We extracted two types of data from the primary studies: (i) general bibliometric information (i.e., author affiliations, countries, publication type, title, year, keywords) and (ii) specific data related to the research questions identified during the full-text analysis (i.e., type of software solution, implementation methods, focus of the solution, research strategies used to assess them, barriers the solutions mitigate, and diversity and inclusion aspects), as illustrated in Table 4.

We compiled the quantitative and qualitative data extracted from each study included in our SLR. The quantitative data allowed us to examine the trends reported in the literature. We also analyzed qualitative data, applying an inductive approach inspired by open coding and axial coding from Grounded Theory (GT) [75] to establish data categories and systematically organize the insights provided by the literature regarding software solutions for onboarding. Although the purpose of the GT method is the construction of substantive theories, according to Corbin and Strauss [75], the researcher may use only some of its procedures to meet one's research goals. While addressing each research question, specific data properties were defined and consistently extracted from all relevant publications.

2.5. Data synthesis

Most primary studies were published in conferences, accounting for 30 primary studies (94%), while we identified only two (2) studies published in journals (6%). Table 5 provides a comprehensive list of the conferences and journals where these primary studies were published. This information can be valuable for practitioners and researchers interested in this topic, as it helps identify relevant conferences for future publication opportunities. Notably, ICSE, the flagship software engineering conference, had the highest number of published primary studies, followed by the FSE conference. The journals that featured publications were IEEE Access and Science China Information Sciences.

[6] https://zenodo.org/records/10211339

Table 3
List of included studies.

ID	Reference	Title	Public. year
[PS01]	Azanza et al. [1]	Onboarding in Software Product Lines: Concept Maps as Welcome Guides	2021
[PS02]	Canfora et al. [29]	Who is Going to Mentor Newcomers in Open Source Projects?	2012
[PS03]	Cubranic and Murphy [30]	Hipikat: Recommending Pertinent Software Development Artifacts	2003
[PS04]	Diniz et al. [50]	Using Gamification to Orient and Motivate Students to Contribute to OSS projects	2017
[PS05]	Dominic et al. [51]	Conversational Bot for Newcomers Onboarding to Open Source Projects	2020
[PS06]	Fu et al. [52]	Expert Recommendation in OSS Projects Based on Knowledge Embedding	2017
[PS07]	Guizani et al. [53]	Attracting and Retaining OSS Contributors with a Maintainer Dashboard	2022
[PS08]	He et al. [54]	GFI-Bot: Automated Good First Issue Recommendation on GitHub	2022
[PS09]	Kagdi et al. [55]	Who Can Help Me with this Source Code Change?	2008
[PS10]	Medeiros and Díaz [56]	Assisting Mentors in Selecting Newcomers' Next Task in Software Product Lines: A Recommender System Approach	2022
[PS11]	Nagel et al. [57]	Ontology-Based Software Graphs for Supporting Code Comprehension During Onboarding	2021
[PS12]	Sarma et al. [58]	Training the Future Workforce through Task Curation in an OSS Ecosystem	2016
[PS13]	Serrano Alves et al. [59]	How to Find My Task? Chatbot to Assist Newcomers in Choosing Tasks in OSS Projects	2022
[PS14]	Stanik et al. [40]	A Simple NLP-Based Approach to Support Onboarding and Retention in Open Source Communities	2018
[PS15]	Steinmacher et al. [21]	Overcoming Open Source Project Entry Barriers with a Portal for Newcomers	2016
[PS16]	Steinmacher et al. [60]	Recommending Mentors to Software Project Newcomers	2012
[PS17]	Toscani et al. [61]	A Gamification Proposal to Support the Onboarding of Newcomers in the FLOSScoach Portal	2015
[PS18]	Wang and Sarma [41]	Which Bug Should I Fix: Helping New Developers Onboard a New Project	2011
[PS19]	Xiao et al. [62]	Recommending Good First Issues in GitHub OSS Projects	2022
[PS20]	Yin et al. [63]	Automatic Learning Path Recommendation for Open Source Projects Using Deep Learning on Knowledge Graphs	2021
[PS21]	Ford et al. [64]	ReBOC: Recommending Bespoke Open Source Software Projects to Contributors	2022
[PS22]	Liu et al. [65]	Recommending GitHub Projects for Developer Onboarding	2018
[PS23]	Santos et al. [37]	Designing for Cognitive Diversity: Improving the GitHub Experience for Newcomers	2023
[PS24]	Santos et al. [66]	Can I Solve It? Identifying APIs Required to Complete OSS Tasks	2021
[PS25]	Minto and Murphy [67]	Recommending Emergent Teams	2007
[PS26]	Heimburger et al. [39]	Gamifying Onboarding: How to Increase Both Engagement and Integration of New Employees	2020
[PS27]	Malheiros et al. [68]	A Source Code Recommender System to Support Newcomers	2012
[PS28]	Yang et al. [69]	RepoLike: Personal Repositories Recommendation in Social Coding Communities	2016
[PS29]	Zhou et al. [70]	GHTRec: A Personalized Service to Recommend GitHub Trending Repositories for Developers	2021
[PS30]	Venigalla et al. [71]	GitQ- Towards Using Badges as Visual Cues for GitHub Projects	2022
[PS31]	Sun et al. [72]	Personalized Project Recommendation on GitHub	2018
[PS32]	Sarma et al. [73]	Tesseract: Interactive Visual Exploration of Socio-Technical Relationships in Software Development	2009

Table 4
Form containing items extracted from selected studies.

General extracted data
Author affiliations and countries
Publication type (journal, conference, or workshop)
Study metadata (title, authors, year)
Keywords
Research questions
RQ1—Type of software solution
RQ2—Software solution implementation
RQ3—Outcomes of software solutions for onboarding
RQ4—Research strategies to assess the software solution
RQ5—Newcomers' barriers mitigated by the software solution
RQ6—Software solutions focus on newcomers aspects of diversity and inclusion

Table 5
Selected studies classified by published venue.

Venue	# of studies	ID	%
ICSE	7	[PS01], [PS03], [PS07], [PS15], [PS19], [PS23], [PS32]	22%
FSE	3	[PS02], [PS08], [PS12]	9%
CHASE	2	[PS04], [PS18]	6%
ICSME	2	[PS09], [PS14]	6%
COMPSAC	2	[PS20], [PS27]	6%
MSR	2	[PS24], [PS25]	6%
BotSE	1	[PS05]	3%
IWCSN	1	[PS06]	3%
CAiSE	1	[PS10]	3%
SEAA	1	[PS11]	3%
CONVERSATIONS	1	[PS13]	3%
RSSE	1	[PS16]	3%
IHC	1	[PS17]	3%
VL/HCC	1	[PS21]	3%
AHFE	1	[PS26]	3%
ICWS	1	[PS29]	3%
ICPC	1	[PS30]	3%
IEEE Access	1	[PS22]	3%
Internetware	1	[PS28]	3%
Science China Information Sciences	1	[PS31]	3%

Table 6 presents the geographic distribution of the selected primary studies, which originate from five continents and nine different countries. Many primary studies involved collaboration among authors from multiple countries. The majority of publications were from the USA (34%), followed by Brazil (28%) and China (25%). The remaining countries contributed with approximately 1 to 3 publications each.

Fig. 2 illustrates the yearly distribution of primary studies published over time. The analysis reveals that researchers published the earliest study in the dataset in 2003. From 2003 to 2011, there was a consistent trend of one study published per year. Furthermore, starting in 2012, there was a notable increase in published primary studies, with the count rising from 3 to 7. This growth suggests a heightened interest and research activity in the field during the subsequent years.

Table 6
Selected studies per country.

Continent	Country	# of studies	%
Asia	China	8	25%
	India	1	3%
Europe	Germany	3	9%
	Italy	1	3%
	Spain	2	6%
North America	Canada	2	6%
	USA	11	34%
South America	Brazil	9	28%
Oceania	Australia	1	3%

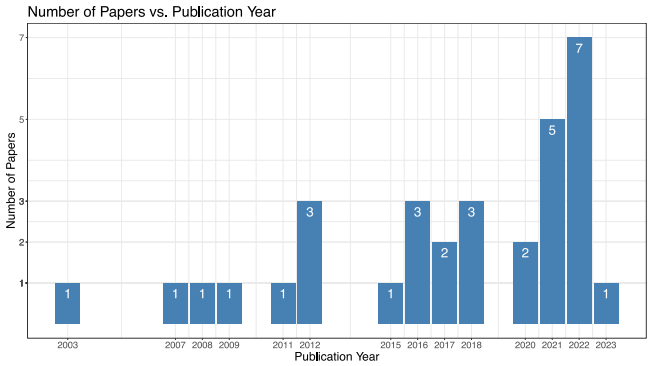


Fig. 2. Publication years and relevant paper counts.

We created a word cloud by aggregating the keywords extracted from the 32 chosen primary studies, as illustrated in Fig. 3. Word clouds gained popularity as a simple yet visually captivating method for representing textual information. They are widely employed in various domains to provide an overview by highlighting the most frequently occurring words, serving as a concise textual summary [76].

3. SLR results

The following subsections discuss the findings for each research question.

3.1. RQ1. What software solutions are proposed in the literature to facilitate newcomers' onboarding in software projects?

We identified three primary categories of software solution strategies to support newcomers onboarding in software projects as observed in Table 7.

Recommendation system. According to Robillard et al. [77], recommendation systems help users find information and make decisions where they lack experience or cannot consider all the data at hand. These systems proactively tailor suggestions that meet users' information needs and preferences. Recommendation systems play a crucial role by offering tailored recommendations to assist in various aspects of project engagement. These recommendations can span a wide range of areas. For example, multiple studies suggest projects that align with newcomers' skills, preferences, and interests [PS05,PS21,PS22,PS28,PS29,PS31].

Some software solutions recommend initial issues for newcomers to contribute [PS07,PS08,PS14,PS19,PS24]. The goal of those solutions is to aid in task labeling, mainly those suitable for newcomers, by enhancing those tasks' visibility, and it contributes to easing newcomers' integration into OSS projects.

Other software solutions focused on facilitating connections between newcomers and experienced mentors [PS02,PS06,PS09,PS16,PS25]. These studies aim to ease newcomers' integration, enhance learning, and foster collaboration within software development environments. The studies introduce software solutions to connect newcomers with proficient developers, mentors, or experts who can offer guidance, support, and collaboration. The studies used historical records [PS09,PS02,PS16], emergent team information [PS25], or knowledge embedding [PS06] to recommend suitable mentors or experts to newcomers. Each of these studies emphasized the importance of personalization in matching newcomers with mentors or experts who possess the relevant skills and knowledge for the tasks at hand. Furthermore, these studies underscore the significance of mentorship and expert guidance in easing newcomers' integration and enhancing their skills.

Fig. 3. Keywords word cloud.

Some other papers propose software solutions recommending artifacts [PS03,PS10,PS20,PS27]. They focus on designing software solutions to expedite newcomers' productivity and engagement in OSS projects. The studies offered tailored recommendations (e.g., relevant artifacts [PS03], feature selection [PS10], learning paths [PS20], and change requests [PS27]) to guide newcomers during the contribution process. Other software solutions guided newcomers toward relevant tasks/bugs to tackle [PS12,PS13,PS18], providing mechanisms for newcomers to discover and select tasks suited to their skills and interests. These studies aimed to streamline the task selection process, making it easier for newcomers to engage in the OSS process. [PS12] and [PS18] indirectly aid newcomers in selecting tasks by providing curated tasks and bug-related resources.

Presentation of project information. According to Moody [78], visual representations are effective because they leverage the capabilities of the robust and highly parallel human optical system. Humans prefer receiving information in a graphic format to process it efficiently. Some solutions focused on information visualization tools [PS01,PS11,PS18,PS20,PS32] that provide dynamic and visual representations of project resources, documentation, and contributions. In addition, some solutions use tools and techniques to capture, organize, and present data within a project environment, enhancing the accessibility and comprehensibility of project-related information, including metrics [PS07,PS30] and structured documentation [PS15].

Information visualization tools can enhance user engagement and retention by making content more interactive. [PS01,PS11,PS18,PS20], and [PS32] propose dependency visualization tools for organizing and visually presenting information related to the project. For example, Azanza et al. [PS01] introduced SPL Cmaps to aid newcomers in grasping the complexity of SPL by visually representing concepts and connections, and Nagel et al. [PS11] developed node-link diagrams to visually represent source code by presenting code relationships. The other three studies [PS18,PS20,PS32] explored the different aspects of relationships between OSS projects: socio-technical networks including developers, code, and software bugs [PS18] and [PS32]; and the

Table 7
Software solution strategies for newcomers onboarding.

Category	Software solution	Description	Study references
Recommendation system (23 studies)	Project recommendation (6 studies)	Assign suitable projects for newcomers based on their skills and interests.	[PS05,PS21, PS22,PS28,PS29, PS31]
	Issue label recommendation (5 studies)	Label tasks or issues for newcomers to engage with OSS projects.	[PS07,PS08, PS14,PS19,PS24]
	Mentor/expert recommendation (5 studies)	Recommend experienced mentors in OSS projects.	[PS02,PS06, PS09,PS16,PS25]
	Artifact recommendation (4 studies)	Recommend artifacts (related tasks, software product line features, learning paths, and source files) for newcomers to explore and integrate into projects.	[PS03,PS10, PS20,PS27]
	Task/bug recommendation (3 studies)	Recommend relevant tasks or bugs for new-comers to tackle in software projects.	[PS12,PS13, PS18]
Presentation of project information (8 studies)	Information visualization (5 studies)	Use visualization tools for organizing and representing knowledge.	[PS01,PS11, PS18,PS20,PS32]
	Metrics (2 studies)	Create metrics to support community managers to track and acknowledge new-comers' contributions.	[PS07,PS30]
	Structured documentation (1 study)	Guide newcomers by structuring existing project documents.	[PS15]
Environment redesign (4 studies)	Gamification (3 studies)	Apply gamification to enhance newcomer engagement and motivation in software projects.	[PS04,PS17, PS26]
	Platform usability enhancement (1 study)	Modify newcomers' interaction with the OSS project environment.	[PS23]

Note: A single study may fit into multiple categories.

relationship between program structure and project versions to explore the software evolution [PS20].

In the metrics subcategory, Guizani et al. [PS07] propose a dashboard solution to support community managers in monitoring and acknowledging newcomers' contributions. In addition, Venigalla et al. [PS30] presents GitQ to automatically augment GitHub repositories with badges representing source code and project maintenance information.

Concerning structured documentation, Steinmacher et al. [PS15] proposed a web portal that guides newcomers in their first contribution. These solutions encompass pertinent and complementary concepts and provide valuable information for software projects, aiding the onboarding of new contributors.

Environment redesign. Some software solutions were designed to foster an environment facilitating active newcomer engagement. The studies often include the implementation of gamification [PS04, PS17,PS26], which introduces game-like elements to enhance newcomers' motivation, participation, and learning within the project context. Among the studies, two [PS04] and [PS17] delved into the integration of game design elements such as Rankings, Quests, Points, and Levels [PS04] and Gameboard, Unlocking, Tips, Badges, Forum, Voting, Profile, and Leaderboard [PS17]. The authors applied those game elements in distinct contexts, specifically in GitLab [PS04] and the FLOSScoach portal [PS17]. Heimburger et al. [PS26] was the only study that explored gamification by developing a mobile onboarding application tailored explicitly for youth generations. The gamification solutions used game elements to orient, engage, and motivate users [PS04, PS17,PS26]. These findings emphasize increased newcomers' motivation when using these solutions, even though they took place in specific contexts, like OSS platforms [PS04] and [PS17] and private companies [PS26].

Additionally, other software solutions encompass changes to the project interface [PS23], such as platform usability enhancements, to create a more user-friendly and welcoming atmosphere for newcomers. PS23 aims to optimize GitHub's effectiveness by addressing distinct aspects. Santos et al. [PS23] included in the GitHub interface visual elements such as tooltips, progress bars, and feedback messages. Environment redesign solutions focus on enhancing the platform's usability for newcomers during the contribution process [PS23]. Santos et al. [PS23] highlight that the current environment does not adequately support newcomers' onboarding. However, with changes in the interface, the platform can become more inclusive [PS23] and enhance users' performance when onboarding.

Research Question 1

Answer: The software solution strategies proposed in the literature incorporate systems that recommend projects, artifacts, tasks, labels, labeling, and mentors. Other solutions focus on gamification for engagement and enhancements, providing information via dashboards, web portals, and graphical aids.

3.2. RQ2. How were the software solutions implemented?

The software solutions for onboarding were organized in a taxonomy by implementation type, presented in Table 8. The lines represent categories on how the software solutions are implemented, such as web environment, machine learning model, and IDE plugin. The columns are the software solutions types previously mentioned in RQ1, including project and issue label recommendations. It is important to note that a study may fit into multiple categories.

Web environment. In a web environment, end users can configure or program applications using domain-specific or even application-specific languages [79]. Throughout our research, we identified studies that proposed modifications to the environment to facilitate the success of newcomers during the onboarding process and implemented in a web environment setting, with a focus on *gamification* [PS04,PS17], *platform usability enhancement* [PS23], *metrics* [PS07,PS30], *structured documentation* [PS15], *information visualization* [PS18,PS32], *issue label* [PS07, PS08], *mentor/expert* [PS02], *project* [PS21], *artifact* [PS03,PS10,PS27] and *task/bug* [PS12,PS13,PS18].

Table 8
Taxonomy overview of software solutions for newcomers' onboarding by implementation types.

	Project recommendation	Issue label recommendation	Mentor/Expert recommendation	Artifact recommendation	Task/Bug recommendation	Information visualization	Metrics	Structured documentation	Gamification	Platform usability enhancement
Web environment	PS21	PS07*, PS08*	PS02	PS03, PS10, PS27	PS12, PS13* PS18*	PS18*, PS32	PS07*, PS30	PS15	PS04, PS17	PS23
Machine learning model	PS22, PS28, PS29	PS08*, PS14, PS19, PS24	PS06							
IDE plugin	PS31		PS09, PS16, PS25							
Interactive graph				PS20*		PS01, PS11, PS20*				
Chatbot	PS05	PS08*			PS13*					
Mobile application									PS26	

Concerning the gamification solutions, two studies [PS04] and [PS17] demonstrate the potential of integrating gamification elements into web environments to enhance engagement and motivation among newcomers in OSS projects.

Diniz et al. [PS04] integrated gamification elements on GitLab for undergraduate students, and Toscani et al. [PS17] demonstrate that gamification can be effective in engaging a diverse range of newcomers. This opportunity implies that gamification can be customized to cater to various demographic groups, ensuring inclusivity and widespread participation.

Platform usability enhancement solutions, such as the OSS environment redesign [PS23], facilitated newcomers' understanding of repositories and aided their decision-making process. Santos et al. [PS23] tackled inclusivity bugs on the GitHub interface by implementing fixes via a JavaScript plugin, contributing to a more inclusive experience.

Concerning project information visualization, [PS30] presented visual cues conveying project information to developers on GitHub repositories, and [PS07] introduced dashboard prototypes. In addition, [PS15] developed a web portal to provide targeted information and recommendations. Other studies [PS03,PS10,PS27,PS15] emphasize the need to facilitate newcomers' access to relevant information. Some studies proposed software solutions that assist newcomers with issue labels [PS07,PS08]. Moreover, other studies presented software solutions to engage newcomers with tasks matching their skills and interests [PS12,PS13] and enabling newcomers to explore project bug descriptions [PS18].

Machine learning. According to Lo et al. [80], machine learning is adopted broadly in many areas, and data plays a critical role in machine learning systems due to its impact on model performance. Machine learning is an artificial intelligence technique that makes decisions or predictions based on data [81]. We identified eight (8) studies that harnessed the power of machine learning techniques. These studies predominantly center on offering recommendations to newcomers, honing in on crucial aspects such as *issue label* [PS08,PS14,PS19, PS24], *mentor/expert* [PS06] and *projects* [PS22,PS28,PS29]. Across these studies, Fu et al. [52] [PS06] used machine learning techniques to provide expert recommendations by using the random forest method to suggest suitable experts for developers based on domain-specific file embedding. Meanwhile, He et al. [PS08] showcases the integration of machine learning into newcomer onboarding by automating task selection and enhancing newcomers' participation. Software projects can optimize collaboration and knowledge sharing using domain-specific file embedding and behavioral patterns, as demonstrated by Fu et al. [PS06], by connecting newcomers with experienced individuals who can guide them.

The utilization of historical data and machine learning techniques [PS14,PS19,PS22,PS24] highlights the importance of automating the categorization of issues based on their characteristics and historical context. Projects can improve efficiency by automatically assigning relevant labels and tags, analyzing resolved issues, extracting pertinent details from titles and descriptions, and simplifying the issue management process.

Two studies [PS28] and [PS29] introduced ML-driven solutions recommending repositories to developers. Both works leverage historical development activities, technical features, and social connections to predict developers' interests and preferences.

IDE plugin. Integrated Development Environment (IDE) plugins are software extensions or add-ons that enhance the functionality and features of software. Four software solutions [PS09,PS16,PS25,PS31] developed a plugin they applied as an external software component in an IDE, which users can add to enhance and extend its functionality. Those software solutions are related to *mentor/expert* [PS09,PS16, PS25] and *project* recommendation [PS31].

Each study offers unique perspectives on how the solutions can guide and engage developers. A significant subset of studies [PS09, PS16,PS25] focuses on enhancing collaboration among newcomers, developers, and the project community through various means, such as suggesting mentors [PS16] and identifying experts in real-time [PS25]. Some studies [PS09,PS16,PS25,PS31] leverage historical project data, such as source code history, email threads, development activities, and social connections, to inform their recommendations and tailor their software solutions to individual newcomers.

Interactive graph. When developers aim to commit a contribution to an existing project, their initial step involves reading and comprehending the project's code in alignment with their contribution objectives [63]. In our results, we came across three studies [PS01, PS11,PS20] incorporating visualizations to aid newcomers in understanding complex aspects of software projects. These visualizations range from domain-specific visualizations in SPL [PS01], visualizations for unfamiliar codebases [PS11], and visualizations for knowledge graphs [PS20]. These studies support newcomers' comprehension of complex concepts, navigate project environments, and facilitate their learning paths within software projects.

Chatbot. According to Nagarhalli et al. [82], chatbots can perform many tasks at lower costs across a wide range of fields, such as customer service, healthcare, pedagogy, and personal assistance, many companies have invested heavily in this technology. Three primary studies proposed chatbots to aid onboarding. They proposed chatbots that focus on different types of interactions with users by recommending *issue label* [PS08], *project* [PS05], and *task/bug* [PS13]. These chatbots utilize machine learning techniques [PS08], natural language processing (NLP) methods [PS05], and conversational interfaces [PS13] to interact with newcomers and provide tailored recommendations. These studies emphasize the potential of chatbots as software solutions to enhance the onboarding journey for newcomers in OSS projects. Using chatbots to implement those solutions enhanced the engagement and productivity of newcomers in software projects.

Mobile application. Mobile devices and their applications offer substantial benefits to users, including portability, location awareness, and accessibility [83]. One study [PS26] proposed a mobile application

Table 9
Onboarding aspects focused by the software solutions.

Category impacted	Onboarding aspect	Study references
Process (19 studies)	Project discovery (6 studies)	[PS21,PS22, PS28,PS29,PS30, PS31]
	Choosing tasks (5 studies)	[PS08,PS10, PS13,PS14,PS24]
	Information overload (4 studies)	[PS01,PS11, PS20,PS32]
	Issue labeling (3 studies)	[PS07,PS19, PS24]
	Mitigation of barriers related to the orientation and contribution process (1 study)	[PS15]
Personal (4 studies)	Engagement and motivation (3 studies)	[PS04,PS17, PS26]
	Self-efficacy (1 study)	[PS23]
	Onboarding of newcomers with different cognitive styles (1 study)	[PS23]
Interpersonal (5 studies)	Mentor/expert recommendation (4 studies)	[PS02,PS06, PS09,PS25]
	Social integration and team building (1 study)	[PS26]
Technical (3 studies)	Artifact selection (2 studies)	[PS03,PS27]
	Code comprehension (1 study)	[PS11]

Note: A single study may fit into multiple categories.

solution to guide and assist newcomers during onboarding. Heimburger et al. [PS26] incorporated gamification elements, such as QR-Hunting, Company-Quiz, Team Bingo, Company-Whisper, and the Onboarding Tree, showcasing how gamification solutions tap into the intrinsic motivation of newcomers. PS26 underscores the app's potential to revolutionize onboarding for tech-savvy professionals.

Research Question 2

Answer: The studies implemented software solutions utilizing web environment enhancements, machine learning, IDE plugins, interactive graphs, chatbots, and mobile applications. A trend is the prevalence of web-based implementations over the years.

3.3. RQ3. How do the proposed software solutions improve newcomers' onboarding?

To address RQ3, we categorized the goal of each primary study into four categories, as presented in Table 9. The categories draw parallels with the categorization outlined by Balali et al. [25], although we tailored them to the context of software solutions for onboarding. Software solutions focusing on *process* revolve around refining onboarding procedures and workflows within a software project. Regarding the *personal* aspects, we found solutions geared toward enhancing individual newcomers' needs and experiences during the onboarding process. Software solutions that focus on *interpersonal* aspects encompass those that enhance relationships among team members, including both newcomers and existing contributors. Furthermore, software solutions focusing on *technical* aspects aimed to provide newcomers with the necessary tools, resources, and technical skills required for their roles within the software project. It is important to note that some studies appeared in multiple categories.

Process. [PS08,PS10,PS13,PS14,PS24] proposed solutions that improved how newcomers select a task to start contributing by streamlining the assignment process based on newcomers' skills and interests. Additionally, [PS07,PS19,PS24] improved how issues could be better labeled to support maintainers. Four studies [PS01,PS11,PS20,PS32] changed the artifact representation and enabled interactive exploration of the relationships among different project elements to reduce information overload. Furthermore, some primary studies [PS21,PS22,PS28, PS29,PS30,PS31] enhanced project discovery, helping newcomers find projects aligned with their interests and skills.

Personal. In our analysis, we identified four studies [PS04,PS23, PS17,PS26] that enhanced individual newcomers' needs and experiences. Such solutions increased engagement and motivated newcomers to accomplish tasks [PS04,PS17,PS26]. These software solutions primarily utilized gamification techniques with newcomers, fostering their engagement and boosting motivation. Additionally, the solution proposed by Santos et al. [PS23] improved the newcomers' self-efficacy by providing a software solution that enhances newcomers' belief in their ability to perform tasks within the project context. Further, their solution improved the onboarding experience of newcomers with different cognitive styles.

Interpersonal. We identified five studies [PS02,PS06,PS09,PS25, PS26] that propose solutions that foster community building among newcomers in OSS projects. One of these solutions [PS26] enhanced social integration and team building by introducing an application designed to support the onboarding process within a software company, particularly targeting users from generations Y and Z. Four solutions [PS02,PS06,PS09,PS25] facilitate mentorship for newcomers by enhancing mentor and expert recommendations.

Technical. Two studies [PS03,PS27] improved artifact recommendation based on user requirements. [PS03] and [PS27] aimed to refine how OSS projects suggest and deliver artifacts to newcomers, aligning with their needs and preferences. Additionally, one study [PS11] enhanced newcomers' code comprehension by providing visual representations of OSS projects.

Research Question 3

Answer: Our research emphasizes the significant impact of software solutions on newcomers' onboarding in OSS projects, categorizing onboarding into *personal* aspects (focusing on boosting motivation and self-efficacy); *interpersonal* (focusing on community building and mentorship); *process* (addressing task selection and information overload); and *technical* (emphasizing skill development and artifact recommendations).

3.4. RQ4. How do the software solutions mitigate newcomers' barriers to joining software projects?

Steinmacher et al. [17] conducted a qualitative analysis of relevant literature and collected data from practitioners to identify the barriers that hinder newcomers' initial contributions to OSS projects. As a result of their comprehensive investigation, the authors developed a model comprising 58 distinct barriers. Based on the previously published studies, our study analyzes the existing software solutions for onboarding and how they could mitigate these identified barriers. It is important to note that only 18 out of the 58 barriers were covered by the existing software solutions, as illustrated in Table 10.

Newcomers' orientation. Newcomers' orientation is a critical phase in facilitating newcomers' successful integration and contribution to various settings, and several barriers hinder this process. Among the primary studies, 13 could address the challenge of *finding a task for newcomers*. [PS12,PS13,PS18,PS24] offer insights into task selection, providing clear guidelines [PS04,PS14,PS15,PS17], utilizing task recommendation system [PS08,PS19], and leveraging task complexity levels to match newcomers' skills and interests [PS05,PS07,PS31]. Concerning the barrier of *finding a mentor*, some studies shed light on solutions to streamline finding a mentor from different perspectives, such as mentorship programs [PS12,PS15], mentor–mentee and matching systems [PS02,PS06,PS09,PS16,PS25], and establishing efficient communication channels between newcomers and mentors [PS05, PS13].

Table 10
Software solutions for onboarding to overcome barriers identified by Steinmacher et al. [17], only 18 out of the 58 barriers are addressed by existing software solutions.

BARRIERS/SOFTWARE SOLUTIONS	PS 01	PS 02	PS 03	PS 04	PS 05	PS 06	PS 07	PS 08	PS 09	PS 10	PS 11	PS 12	PS 13	PS 14	PS 15	PS 16	PS 17	PS 18	PS 19	PS 20	PS 21	PS 22	PS 23	PS 24	PS 25	PS 26	PS 27	PS 28	PS 29	PS 30	PS 31	PS 32
Newcomers' orientation																																
Finding a task to start with	–	–	–	X	X	–	X	X	–	–	–	X	X	X	X	–	X	X	X	–	–	–	–	X	–	–	–	–	–	–	X	–
Finding a mentor	–	X	–	–	X	X	–	–	X	–	–	X	X	–	X	X	–	–	–	–	–	–	–	–	X	–	–	–	–	–	–	–
Finding the correct artifacts to fix an issue	–	–	X	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Poor "How to Contribute" available	–	–	–	–	–	–	–	–	–	–	–	–	–	–	X	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Newcomers don't know what is the contribution flow	–	–	–	–	–	–	–	–	–	–	–	–	–	–	X	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Newcomers' characteristics																																
Lack of domain expertise	X	–	X	–	X	–	X	X	–	X	X	X	–	–	X	–	–	–	–	X	–	–	X	–	–	–	X	–	X	–	X	X
Lack of knowledge in project process and practices	X	X	X	X	X	X	X	X	–	X	X	–	–	–	X	X	X	–	–	X	X	X	X	X	–	X	X	X	X	X	X	–
Lack of technical background	X	–	X	–	X	–	–	–	X	X	X	–	–	–	X	–	–	X	–	X	–	–	X	–	X	–	X	–	–	–	X	–
Communication																																
Not receiving an answer	–	–	–	–	–	–	–	–	–	–	–	–	–	–	X	X	–	–	–	–	–	–	–	–	X	–	–	–	–	–	–	–
Send a message that is considered impolite	–	–	–	–	–	–	–	–	–	–	–	–	–	–	X	X	–	–	–	–	–	–	–	–	X	–	–	–	–	–	–	–
Need to contact a real person	–	X	–	–	X	X	–	–	X	–	–	X	–	–	X	X	–	–	–	–	–	–	–	–	X	X	–	–	–	–	–	–
Delayed answers	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	X	–	–	–	–	–	–	–	–	X	–	–	–	–	–	–	–
Documentation problems																																
Information overload	X	–	X	–	–	–	–	–	–	–	X	–	–	–	X	–	–	–	–	X	–	–	X	–	–	–	–	–	–	–	–	X
Lack of documentation	–	–	–	–	–	–	–	–	–	–	–	–	–	–	X	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Spread documentation	–	–	–	–	–	–	–	–	–	–	–	–	–	–	X	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Technical hurdles																																
Local environment setup hurdles	X	–	X	X	–	–	–	–	–	X	–	–	–	–	X	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Code/architecture hurdles	–	–	X	–	–	–	–	–	–	–	X	–	–	–	X	–	–	–	–	X	–	–	–	–	–	–	X	–	–	–	–	–
Understanding flow of information	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	X	–	–	–	–	–	–	–	–	–

The literature lacks methods to assist newcomers in *finding the correct artifacts to fix an issue*. Cubranic and Murphy [PS03] is the only study that presents a solution to recommended artifacts from the archives that are relevant to a task that a newcomer is trying to perform—and it was published 20 years ago. Concerning the barrier of *poor "How to Contribute" availability*, it is crucial to emphasize the need for improving the availability and accessibility of comprehensive, user-friendly resources that can guide newcomers through the contribution process. To overcome this barrier, [PS15] delivers well-structured documentation, tutorials, and interactive guides. Only the solution presented by Steinmacher et al. [PS15] offers clear and concise guidance to address the barrier of *newcomers' lack of awareness of the contribution flow*, ensuring that newcomers comprehend the necessary steps and expectations for their contributions.

Newcomers' characteristics. Newcomers are expected to possess a minimum requirement of previous technical background to perform a development task [17]. Fifteen solutions can address the barrier of *lack of domain experience*, bridging the knowledge gap and gradually empowering newcomers to acquire domain expertise, enabling them to contribute to their expertise domain. These solutions include broadening newcomers' domain knowledge and reducing information overload [PS01,PS11,PS20,PS32], forming an implicit group memory from the information stored in a project's archives [PS03,PS31], and providing newcomers' support not only during their first contribution [PS23,PS27,PS29] but by acting as an agent to engage them in the project [PS05,PS10,PS12,PS15] and promoting collaboration between newcomers and domain experts [PS07,PS08].

To mitigate the barrier of *lack of knowledge in project process and practices*, 24 solutions can enhance newcomers' technical skills, fill the gaps in their knowledge, and build their confidence to contribute to technical projects actively. These include providing comprehensive documentation [PS03,PS15] and resources that explain project workflows [PS01,PS04,PS07,PS11,PS17,PS20,PS23,PS26], provides project recommendation [PS05,PS21,PS22,PS28,PS29,PS31] mentoring [PS02, PS06,PS10,PS16], coding standards [PS08,PS24,PS27,PS30], and communication channels [PS05]. Additionally, to overcome the barrier of *lack of technical background*, 13 solutions can help by offering guidance during the contribution process [PS01,PS11,PS15,PS20,PS23, PS27], recommendation of project documentation [PS03,PS05,PS15, PS18,PS31], and pairing newcomers with experienced developers as mentors [PS09,PS10,PS25].

Communication. According to Steinmacher et al. [17], newcomers are sometimes unaware of community communication protocol. Three solutions [PS15,PS16,PS25] can tackle the barriers related to *not receiving an answer* and *sending impolite messages*. To alleviate the first barrier, [PS15] focuses on creating designated communication channels to better visibility of newcomers' questions and increase the chances of receiving timely answers from the community members. [PS16] and [PS25] recommend appointing experienced members as mentors, ensuring newcomers receive timely responses. For the second barrier, [PS15] offers newcomers guidance on effective communication with other project members, while [PS16] and [PS25] recommend avoiding unintentional rudeness or misunderstandings.

Nine studies can address the barrier of *need to contact a "real" person*. These include mentoring initiatives such as pairing newcomers with experienced community members by recommending mentors to newcomers [PS02,PS06,PS09,PS16,PS25] and providing clear guidelines [PS05,PS12,PS15,PS26]. Concerning the barrier of receiving *delayed answers*, two solutions [PS16,PS25] recommended mentors who can expedite responses and collaborate with newcomers to assist them in their initial contributions.

Documentation problems. We identified six (6) software solutions that can mitigate barriers related to documentation problems. The solutions can tackle the barrier of *information overload* include creating clear and concise documentation [PS15], breaking down complex concepts into manageable sections [PS03], providing a straightforward visual representation of the project [PS01,PS11,PS20,PS32], and offering contextual guidance to help newcomers find the most relevant information based on their specific needs [PS23]. Moreover, to mitigate the barrier of *lack of documentation*, only one solution [PS15] focuses on actively creating and improving documentation resources, including dedicating resources and efforts to document essential aspects of the project. To tackle the barrier of *spread documentation*, one solution [PS15] delved into methods of consolidating and centralizing documentation resources. Steinmacher et al. [PS15] offered newcomers a dedicated "Documentation" section, housing project documentation organized into subsections for easy access and navigation.

Technical hurdles. We found 9 (nine) software solutions targeting technical challenges newcomers encounter when trying to understand

Table 11
Evaluation strategies of software solutions.

Study type		Study references
Laboratory experiment (23 studies)	Programmed actors (10 studies)	[PS06,PS08,PS09, PS10,PS19,PS22, PS25,PS27,PS28, PS29]
	Both (human and programmed actors) (7 studies)	[PS02,PS14,PS15, PS20,PS21,PS24, PS31]
	Human participants (6 studies)	[PS04,PS11,PS13, PS23,PS26,PS30]
Judgment study (16 studies)	External experts (7 studies)	[PS02,PS13,PS14, PS17,PS20,PS21, PS32]
	Newcomers (7 studies)	[PS01,PS13,PS15, PS17,PS26,PS30, PS32]
	Maintainers (2 studies)	[PS07,PS24]
Experimental simulation (2 studies)		[PS01,PS03]

Note: A single study may fit into multiple categories.

and navigate the technical aspects of a project. Concerning the barrier of *local environment setup hurdles*, three solutions [PS04,PS10,PS15] can provide orientation on how to set up the development environment. [PS01] and [PS03] suggest pre-configured development environments to ensure a smooth onboarding experience. Five (5) software solutions can mitigate the barrier of *code/architecture hurdles*. These solutions encompass various initiatives to assist newcomers in their codebase navigation and comprehension of the project's architecture, such as furnishing architectural diagrams [PS15] and presenting high-level project structure overviews [PS03,PS10,PS20,PS27]. We want to highlight that Santos et al. [PS23] was the only work that could mitigate newcomers' cognitive barriers during the contribution process.

Research Question 4

Answer: Most software solutions for onboarding presented in the literature focus on mitigating the barriers related to newcomers' characteristics. The software solutions assist newcomers in finding suitable tasks and mentors, bridging gaps in domain knowledge, project processes, and technical background, improving communication, maintaining user-friendly documentation, simplifying technical aspects, and enhancing their onboarding experience. Our results also reveal a need for solutions that target communication barriers, documentation issues, technical challenges, and newcomers' orientation.

3.5. RQ5. What research strategies were employed to evaluate the software solutions?

This question investigates the research strategies used to evaluate the proposed software solutions for newcomers' onboarding. Table 11 presents the study types identified in the selected primary studies.

We categorized the evaluation methods employed in the primary studies according to the ABC Framework, as initially defined by Stol and Fitzgerald [84]. The ABC Framework underscores the essence of knowledge-seeking research, emphasizing the involvement of actors (A) engaging in behavior (B) within a specific context (C). Within this framework, we identified three predominant research strategies to assess primary studies concerning software solutions for onboarding.

The predominant research strategy employed by the primary studies (23 studies, 71%) was laboratory experimentation, involving meticulous manipulation of variables to precise measurements of actors' behavior [84]. These experiments encompassed diverse studies involving human participants and programmed actors—such as algorithms or prototype tools. Furthermore, it is noteworthy that a subset of primary studies [PS04,PS11,PS13,PS23,PS26,PS30] utilized laboratory experiments involving human participants. These studies typically featured treatment and control groups, allowing precise measurements to detect potential differences. Conversely, other studies [PS06,PS08,PS09,PS10, PS19,PS22,PS25,PS27,PS28,PS29] employed programmed actors, such as algorithms or prototype tools, in their laboratory experiments. It is worth mentioning that particular studies [PS02,PS14,PS15,PS20,PS21, PS24,PS31] conducted laboratory experiments involving humans and algorithms, encompassing a more comprehensive evaluation.

Our results show that among the primary studies, 13 (40%) applied judgment studies. According to Stol and Fitzgerald [85], a judgment study involves collecting empirical data from a group of participants who assess or rate behaviors in response to stimuli presented by a researcher. In these instances, researchers introduced specific stimuli to observe participants' responses. The goal was to gather input or "judgment" from stakeholders, requiring intensive stimulus–response communication, as discussed by Stol and Fitzgerald [84].

Two studies [PS01] and [PS03] employed experimental simulation to evaluate participants' behavior in tasks that mimic real-world scenarios. As defined by Stol and Fitzgerald [85], experimental simulation studies assess the behavior of participants or systems in a controlled setting that resembles the real world. The studies conducted these simulations in SPL settings [PS01] and the software development environment [PS03].

Four studies [PS05,PS12,PS16,PS18] did not evaluate their proposed software solutions, as they were still in the early stages of their development process at publication. In terms of analysis type, more than half of the studies, 16 out of 32 (50%), employed qualitative analysis to gain insights into software solutions for onboarding by interpreting data to understand subjective experiences associated with the onboarding process from the perspectives of newcomers [PS01,PS02,PS03,PS04,PS07,PS10,PS13,PS14,PS15,PS17,PS20, PS21,PS23,PS26,PS31,PS32]. Additionally, 13 studies (40%) made use of quantitative analysis to evaluate their proposed solutions, collecting and analyzing measurable data related to onboarding, such as success rates, completion times, user satisfaction ratings, or performance metrics [PS01,PS02,PS06,PS11,PS13,PS14,PS15,PS19,PS22,PS23,PS24, PS29,PS30]. Out of these studies, six (19%) employed mixed methods [PS01,PS02,PS13,PS14,PS15,PS23].

Research Question 5

Answer: The primary studies employed three research strategies to evaluate software solutions for onboarding: experimental simulation, laboratory experimentation, and judgment studies. Laboratory experiments were the most frequently used research strategy (mostly comparing algorithms, with no human in the loop).

3.6. RQ6. How do the software solutions address diversity and inclusion of newcomers?

According to Jehn et al. [86], team diversity encompasses individual differences among team members, manifesting in dimensions like value diversity (e.g., beliefs, goals, values), information diversity (e.g., experience, knowledge, background), and social diversity (e.g., gender, age, race). Our study examined the software solutions for onboarding proposed in the literature to assess their potential for facilitating diversity and inclusion among newcomers in OSS projects. Table 12 illustrates each software solution's specific target populations.

Many companies are aware of the lack of diversity in their organizations, prompting a surge in initiatives to enhance employee diversity across global technology companies [87]. Past research has revealed challenges related to perceived diversity within software engineering teams in industrial and OSS settings [87,88].

The importance of social diversity in OSS projects has been well-established, with numerous studies showing its positive impact on

Table 12
Software solutions target population.

Target population	Diversity dimension	Study reference
Newcomers with different educational background (10 studies)	Information (Background)	[PS01,PS03,PS04, PS11,PS13,PS15, PS17,PS23,PS26, PS30,PS32]
Newcomers with different professional experience (9 study)	Information (Experience)	[PS02,PS07,PS13, PS14,PS17,PS20, PS21,PS24,PS32]
Newcomers with different cognitive styles (1 study)	Social (Gender)	[PS23]
Newcomers from generations Y and Z (1 study)	Social (Age)	[PS26]

productivity, teamwork, and the quality of contributions [89,90]. Conversely, the lack of diversity has significant drawbacks: (i) OSS projects miss out on the benefits of a broader range of contributors and the diverse perspectives they bring; (ii) underrepresented groups miss out on valuable learning and experience opportunities offered by these projects; and (iii) individuals from minority backgrounds may face limited job opportunities when hiring decisions use OSS contributions [37,91,92]. Despite the long-standing recognition of the diversity gap in OSS, progress in addressing this issue has been limited [93–95].

Our analysis of the selected studies showed that 15 out of 32 (47%) proposed software solutions for onboarding targeting a general newcomer population without considering or evaluating their effectiveness for integrating different types of users into OSS projects. Eleven studies [PS01,PS03,PS04,PS11,PS13,PS15,PS17,PS23,PS26,PS30,PS32] proposed solutions addressing the diversity aspect of educational backgrounds, specifically aiding students during the onboarding process. This is particularly pertinent given previous research indicating that variations in educational backgrounds can lead to heightened task-related discussions within work teams [86]. Additionally, nine studies [PS02,PS07,PS13,PS14,PS17,PS20,PS21,PS24,PS32] presented software solutions targeting newcomers with more development experience—developers transitioning to new software projects seeking solutions to comprehend project characteristics and source code structures.

Only two studies [PS23] and [PS26] focused on providing support tailored to newcomers with specific cognitive styles [PS23] and concerning newcomers' age [PS26]. Santos et al. [PS23], focused on mitigating cognitive barriers faced by newcomers due to inclusivity bugs. The study revealed that platforms like GitHub, which newcomers use to contribute to OSS, create barriers for users with different characteristics, disproportionately impacting underrepresented groups. Heimburger et al. [PS26] developed a mobile app for generations Y and Z entering the workforce. This solution acknowledges these generations' unique characteristics and communication styles, allowing organizations to create onboarding experiences that resonate with their target audience. Our results highlight the need for more research in the software engineering field that specifically targets increasing diversity and inclusion in software communities to improve and facilitate more inclusive software solutions for onboarding.

Research Question 6

Answer: Among the 32 analyzed studies, the predominant focus on diversity and inclusion dimensions pertained to information diversity (i.e., background and experience). Only two studies specifically addressed the unique needs of newcomers from minority groups, focusing on gender and age.

4. Discussion

This section delves into our research findings, exploring insights and potential areas for further investigation.

Momentum of recommendation systems and machine learning. There is a rise in recommendation systems designed to aid newcomers in diverse activities. These systems assist developers in finding relevant information and evaluating alternative decisions, thereby covering a broad spectrum of software engineering tasks [77,96]. Machine learning and software engineering intersection has become increasingly prominent [97,98]. By harnessing machine learning techniques, it can tackle software engineering problems that are challenging to model purely through algorithms or lack satisfactory solutions [99]. This integration allows for innovative solutions and advancements in the field. Among the primary studies, machine learning techniques were employed to improve recommendation systems, enabling personalized and automated suggestions.

Web environment offers a versatile platform for creating software applications that are universally accessible and can be executed through web browsers. Furthermore, the openness and flexibility of the web simplify the process of writing and deploying code, contributing to the proliferation of a rich and diverse array of applications globally [100]. In the software solutions highlighted in this study, the predominant implementation types observed were based on web environments. These solutions significantly contribute to fostering a more welcoming and supportive onboarding experience for newcomers by leveraging the advantages offered by web environments.

Increasing newcomers' engagement and motivation. The OSS movement has attracted a globally distributed community of volunteers, and the increasing demand for professionals with OSS knowledge has prompted students to contribute to OSS projects [101]. Students gain real-world skills and experiences by engaging in OSS projects, making them more competitive in the job market [102,103]. Additionally, exposing students to OSS projects benefits the communities by increasing the number of potential contributors and fostering collaboration.

Gamification has gained attention to enhance student engagement and motivation in software projects. Gamification applies game elements in non-gaming contexts to motivate and engage participants [104]. In the context of OSS, gamification techniques are vital in promoting healthy competition and instilling a sense of achievement [105,106]. Our findings show a growing interest in utilizing gamification and modifying the OSS environment to enhance newcomer engagement and motivation. By incorporating gaming elements, students remain engaged, persist in their contributions, and derive satisfaction from their involvement. Furthermore, gamification offers learning and skill development opportunities as students acquire new technical skills, learn collaboration, and gain insights into project management practices [107–109].

Impact of software solutions for onboarding. Newcomers need proper orientation to navigate the project and correctly make contributions [21]. Motivating, engaging, and retaining new developers in a project is essential to sustain a healthy OSS community [110]. Our findings demonstrate that software solutions significantly impact newcomers' onboarding experiences in OSS projects, with onboarding aspects categorized into four key areas (i.e., personal, interpersonal, process, and technical). Collectively, these software solutions shape and enhance newcomers' onboarding journeys, facilitating their integration into OSS projects. Begel and Simon [111] discuss the importance, advantages, and challenges of mentoring novices in the software industry. Mentoring is crucial in pairing experienced contributors with newcomers to provide guidance, support, and knowledge transfer. By establishing constructive learning relationships between mentors and mentees, these solutions fostered the growth and integration of newcomers in the OSS project.

Our findings highlight the diverse impact of software solutions on newcomers' onboarding in OSS projects. Focusing on solutions such as engagement and motivation, mentoring, labeling and task selection, project recommendation, and reducing information overload contribute to facilitating the integration of newcomers in software development communities.

Investigating newcomers' barriers. A better understanding of the barriers enables communities and researchers to design and produce tools and conceive software solutions to support newcomers [25]. We identified research gaps in addressing barriers newcomers face during onboarding. Only 18 out of the 58 barriers were covered by the existing software solutions. In particular, software solutions are lacking to tackle barriers related to communication, documentation issues, technical challenges, and newcomers' orientation. Additionally, there is room for exploring tools and techniques to assist newcomers in finding the correct artifacts to understand the contribution process workflow. Existing software solutions for onboarding addressed communication barriers to some extent. However, research opportunities remain for further improvements to support newcomers in better communicating with members of the OSS communities. Furthermore, new studies can explore documentation barriers by removing the overload of information newcomers face when onboarding and making it simple to share documentation. Additionally, future studies can investigate another interesting gap in supporting newcomers in understanding code and architecture hurdles, focusing on the cognitive processes required to comprehend the code information flow.

Beyond the laboratory to explore new horizons. Software engineering is a dynamic and interdisciplinary domain encompassing various social and technological aspects. It is crucial to deeply understand human activities to explore how individual software engineers engage in software development and how teams and organizations coordinate their efforts to achieve success. By studying these aspects, researchers can gain a holistic understanding of software engineering practices and enhance the ability to support software development processes [112]. The analysis of the selected primary studies revealed several types of evaluations. Overall, our findings highlight the different research strategies employed to evaluate the software solutions for onboarding, with the predominant strategy being laboratory experiments. However, future research endeavors could benefit from transitioning beyond the laboratory and conducting field experiments in real-world settings to offer a more comprehensive evaluation of software solutions for onboarding over an extended period, ensuring their long-term success.

Diversity and inclusion in software solutions. Newcomers encounter various challenges, which affect underrepresented populations differently and can result in a steeper learning curve, a lack of community support, and difficulties in initiating contributions, all contributing to the existing diversity imbalance in OSS [16,95,113]. Numerous studies emphasized the positive impact of social diversity on productivity, teamwork, and the quality of contributions. The literature has highlighted concerns regarding the low diversity in OSS, considering factors such as gender, language, and location [95,114–116]. Previous research has demonstrated that diverse teams are more productive, reinforcing the significance of addressing diversity-related issues in OSS [90]. Our analysis revealed that most of the proposed software solutions for onboarding targeted a general newcomer population without considering or evaluating different user types in OSS projects.

Developing inclusive software solutions for onboarding is required to foster diversity and inclusion in software communities. Our study underscores the scarcity of software solutions for onboarding addressing diversity and inclusion. By addressing the specific needs and barriers underrepresented groups face, it is possible to create more inclusive onboarding processes and foster greater diversity within OSS projects. Our study serves as a call to action for the software engineering community to actively work towards creating inclusive environments that welcome individuals from diverse backgrounds and leverage their unique perspectives to benefit the community.

5. Implications for practitioners

In this section, we outline the implications of our study for practitioners.

Implications for project maintainers. Project maintainers have many responsibilities, including attracting and retaining new contributors to promote the project's growth and sustainability. They can leverage the insights gained from our study to create welcoming, inclusive, and supportive environments to onboard and retain newcomers. For example, they can facilitate the integration of newcomers into their projects by recognizing the value of mentorship recommendations solutions and focusing on developing structured documentation and resources to lessen newcomers' cognitive overload when onboarding a new software project.

Implications for tool developers. Tool developers can use our results to understand how to alleviate newcomers' onboarding barriers and use this knowledge to implement new tools. These tools could represent project information through dashboards, web portals, and visualization techniques to support newcomers with the necessary resources for successful navigation and performing better at tasks. Moreover, developers could focus on designing tools that consider the needs of minority groups, such as women or generations Y and Z.

6. Limitations

Although we have adopted the SLR guidelines proposed by Kitchenham et al. [117], this study has some limitations. This section presents the study's limitations and discusses how we mitigate them.

Search strategy. It is possible that the search process might miss relevant primary studies [118]. We defined and followed the search strategy described in Section 2.3 to mitigate this threat. One author extracted the search terms based on our research questions, and the search string was iteratively developed. The search string terms (detailed in Table 2) are broad, aiming to retrieve as many relevant studies as possible. Moreover, we incorporated author and citation analysis, which allowed us to identify other studies beyond our initial search.

Studies selection. A significant threat in secondary studies is recognized to be the validity of study selection [119]. We predefined inclusion and exclusion criteria (see Section 2.2) in the protocol and used them to filter relevant studies. Additionally, two researchers applied the selection criteria in different stages of the study's selection process and jointly conducted a consensus decision-making meeting.

Data extraction. Inconsistency extraction is a fundamental threat in SLR studies Khan et al. [120]. We mitigate this threat by defining a data extraction form, detailed in Section 2.4, to extract relevant data to answer our RQs consistently. One author initially extracted the data, and the other authors participated in the discussion meetings to solve doubt and double-check data, as suggested by Wohlin et al. [121].

Data analysis. The risk of inaccurate data classification and mapping can cause subjective interpretation bias. We lessened this threat following an inductive approach inspired by open coding and axial coding procedures from GT by Corbin and Strauss [75] for analyzing qualitative data.

Generalizability. We do not assert the complete generalizability of this study. Nevertheless, we have tried to enhance its applicability by providing a comprehensive overview of software solutions for onboarding and by logically structuring the study's collected data, results, analysis, and conclusions. To promote the potential for generalizability in our findings, we thoroughly examined a wide array of studies across various subfields of software engineering. As an outcome, we described the implications of our results to social coding platforms, software development organizations, maintainers of OSS projects, software projects, tool developers, and researchers.

7. Related work

This section overviews the relevant work concerning newcomers' onboarding in software projects and literature reviews focusing on onboarding practices. By exploring these areas, we aim to understand the challenges and software solutions associated with integrating newcomers into software projects.

Newcomer's onboarding. Onboarding is a crucial process that facilitates the transition of new employees and enables them to acquire the necessary attitudes, knowledge, skills, and behaviors for effective work [122–124]. According to Bauer and Erdogan [7], onboarding is a crucial process encompassing the activities and initiatives designed to equip new hires with the knowledge, skills, and behaviors necessary to succeed in the new work environment. Newcomers in the software development environment face challenges in becoming fully integrated and productive team members, which includes acquiring organizational knowledge, project knowledge, product and domain knowledge, and knowledge of the technical environment [125]. Fagerholm et al. [27] executed a case study to evaluate the influence of mentoring support on developers. Their findings revealed that mentoring played a crucial role in the onboarding process for newcomers, empowering them to become more engaged and active participants. Gregory et al. [125] examined onboarding practices in a co-located agile project team within a large IT department that regularly welcomed inexperienced newcomers, exploring the activities and adjustments made by individuals and the workplace. As a result, they developed an agile onboarding model encompassing various onboarding activities, individual adjustments made by newcomers, and workplace adjustments to facilitate their integration into the team.

A multitude of empirical studies dedicated their focus to examining the process of newcomers joining community-based OSS projects [20,29,33,41,126,127]. These studies offer insights into the factors influencing newcomers' onboarding experiences within OSS communities. Fronchetti et al. [128] investigated the factors influencing the onboarding of new contributors in OSS projects. The authors analyzed 450 repositories and identified project popularity, review time for pull requests, project age, and programming languages as the main factors explaining newcomers' growth patterns. Understanding these factors helps project maintainers optimize software solutions for onboarding. Furthermore, a separate body of research has focused on understanding newcomers' barriers during their onboarding journey [129,130].

Our study stands out from existing literature due to its unique focus on providing knowledge on software solutions for newcomers' onboarding within software projects. To the best of our knowledge, our research is the first to investigate software solutions for onboarding. We offered a literature review detailing software solutions and their practical implementation, impact on the onboarding process, research methodologies employed, and potential to reduce barriers for newcomers. We also investigated whether these solutions prioritize aspects of diversity and inclusion for newcomers into software projects.

Literature reviews. The systematic mapping study conducted by Kaur et al. [38] examined community participation and engagement in OSS projects. The authors analyzed 67 studies to address the joining process, contribution barriers, motivation, retention, and abandonment. The study also highlighted gaps in mentoring newcomers, finding starting tasks, and identifying factors influencing developer participation and engagement. Steinmacher et al. [5] identified and aggregated 20 studies that provided evidence of barriers newcomers face when onboarding to OSS projects. The study highlighted the most studied barriers and shows that successful contributions require domain knowledge, technical skills, and social interaction, emphasizing the importance of community receptivity, simple code, and organized documentation.

Some literature reviews focused on diversity and inclusion aspects in software engineering that can influence software development. Trinkenreich et al. [95] examined women's participation in OSS projects, focusing on their demographics, motivations, types of contributions, challenges, and the proposed strategies to address those challenges. The study reveals a significant gender disparity in OSS, with women representing only about 10% of participants. Gender biases exist in various aspects, such as differential acceptance rates for pull requests based on gender identification. Women also face social challenges, including a lack of peer parity, non-inclusive communication, a toxic culture, impostor syndrome, and bias in peer review. Considering the need for more diversity in software projects, our study emphasizes the importance of examining and improving current software solutions for onboarding. Additionally, Rodríguez-Pérez et al. [87] conducted an SLR to understand the relationship between perceived diversity aspects (gender, age, race, and nationality) in software engineering. The authors analyzed 131 previous studies to identify factors influencing diverse developers' engagement and permanence in software engineering, methods used to improve perceived diversity in teams, and limitations of previous studies. The study highlights gaps in the current literature and emphasizes the need for future action in addressing perceived diversity in software engineering.

Pedreira et al. [109] conducted a mapping study focusing on the potential benefits of gamification to the Software Engineering (SE) field. The study findings highlight that gamification can be a promising field that can help improve software engineers' daily engagement and motivation in their tasks. The authors also observed that the adoption of gamification in SE is going more slowly than in other domains such as marketing, education, or mobile applications. This trend is similar to our findings on only three software solutions that adopted gamification elements to improve onboarding. Furthermore, Darejeh and Salim [131] conducted an SLR to thoroughly examine gamification solutions addressing user engagement issues across various software categories. Their findings highlighted gamification as a viable approach for enhancing user engagement and performance. Most gamification solutions aim to motivate users to contribute more content to software, encourage active software usage, and improve the software's appeal to induce behavior change. Moreover, their results show a limited focus on motivating users to effectively utilize software content, addressing learning challenges, and integrating users' real identities within the software environment.

8. Conclusion

In this paper, we conducted an SLR analyzing 32 primary studies to investigate the software solutions proposed in the literature to enhance the onboarding processes for newcomers in software projects. The proposed software solutions for onboarding focused on recommendation systems using web-based implementations, and the impact of those software solutions involves personal, interpersonal, technical, and process aspects. Moreover, laboratory experiments were the most common research strategy for evaluation. Concerning diversity, software solutions for onboarding mainly consider newcomers' backgrounds and experience levels.

We recognize that various project domains may exhibit distinct characteristics and requirements during the onboarding process, and the software solutions found in our SLR may not apply equally to all project domains. As a future work opportunity, exploring onboarding solutions tailored to different project domains is essential, allowing for a more nuanced understanding of the unique scenarios. Moreover, as future work, we aim to investigate the diversity and inclusion aspects of onboarding and propose inclusive software solutions that contribute to the diversity and inclusion of more users in software projects. Additionally, we aim to explore how large language models (LLMs) can be used to enhance onboarding processes for newcomers and evaluate their impacts on newcomers' activities.

CRediT authorship contribution statement

Italo Santos: Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Formal analysis, Data curation. **Katia Romero Felizardo:** Writing – review & editing, Writing – original draft. **Igor Steinmacher:** Writing – review & editing, Writing – original draft. **Marco A. Gerosa:** Writing – review & editing, Writing – original draft.

Declaration of competing interest

The authors 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 is available at https://zenodo.org/records/10211339.

Acknowledgments

The National Science Foundation (NSF) partially supports this work under grant numbers 2236198, 2247929, 2303042, and 2303612. Katia Romero Felizardo is funded by a research grant from the Brazilian National Council for Scientific and Technological Development (CNPq), Grant 302339/2022-1.

References

[1] M. Azanza, A. Irastorza, R. Medeiros, O. Díaz, Onboarding in software product lines: concept maps as welcome guides, in: IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training, ICSE-SEET, IEEE, 2021, pp. 122–133.

[2] A. Labuschagne, R. Holmes, Do onboarding programs work? in: IEEE/ACM 12th Working Conference on Mining Software Repositories, MSR, IEEE, 2015, pp. 381–385.

[3] R. Pham, S. Kiesling, L. Singer, K. Schneider, Onboarding inexperienced developers: struggles and perceptions regarding automated testing, Softw. Qual. J. 25 (4) (2017) 1239–1268.

[4] A. Rastogi, S. Thummalapenta, T. Zimmermann, N. Nagappan, J. Czerwonka, Ramp-up journey of new hires: do strategic practices of software companies influence productivity? in: 10th Innovations in Software Engineering Conference, ISEC, 2017, pp. 107–111.

[5] 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. (IST) 59 (2015) 67–85.

[6] G. Viviani, G.C. Murphy, Reflections on onboarding practices in mid-sized companies, in: IEEE/ACM 12th International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE, IEEE, 2019, pp. 83–84.

[7] T.N. Bauer, B. Erdogan, Organizational socialization: The effective onboarding of new employees., in: APA handbooks in psychology, American Psychological Association, Washington, DC, US, 2011, pp. 51–64.

[8] A.J. Ko, Mining the mind, minding the mine: grand challenges in comprehension and mining, in: 26th Conference on Program Comprehension, ICPC, 2018, pp. 1–1.

[9] S.E. Sim, R.C. Holt, The ramp-up problem in software projects: a case study of how software immigrants naturalize, in: 20th International Conference on Software Engineering, ICSE, IEEE, 1998, pp. 361–370.

[10] M. Zhou, A. Mockus, Developer fluency: achieving true mastery in software projects, in: 18th ACM SIGSOFT International Symposium on Foundations of Software Engineering, FSE, 2010, pp. 137–146.

[11] L.M. Berlin, Beyond program understanding: a look at programming expertise in industry, ESP 93 (744) (1993) 6–25.

[12] J. Buchan, S.G. MacDonell, J. Yang, Effective team onboarding in agile software development: techniques and goals, in: ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM, IEEE, 2019, pp. 1–11.

[13] A. Ju, H. Sajnani, S. Kelly, K. Herzig, A case study of onboarding in software teams: tasks and strategies, in: IEEE/ACM 43rd International Conference on Software Engineering, ICSE, IEEE, 2021, pp. 613–623.

[14] K. Rollag, S. Parise, R. Cross, Getting new hires up to speed quickly, MIT Sloan Manage. Rev. (2005).

[15] A. Forte, C. Lampe, Defining, understanding, and supporting open collaboration: Lessons from the literature, Am. Behav. Sci. 57 (5) (2013) 535–547.

[16] I. Steinmacher, T. Conte, M. Gerosa, D. Redmiles, Social barriers faced by newcomers placing their first contribution in open source software projects, in: 18th ACM Conference on Computer Supported Cooperative Work & Social Computing, CSCW, 2015.

[17] 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.

[18] 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: 16th International Symposium on Open Collaboration (OpenSym), 2020, pp. 1–14.

[19] 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, ACM Hum.-Comput. Interact. 5 (CSCW2) (2021) 1–30.

[20] I. Steinmacher, M.A. Gerosa, D. Redmiles, Attracting, onboarding, and retaining newcomer developers in open source software projects, in: Workshop on Global Software Development in a CSCW Perspective, 2014.

[21] I. Steinmacher, T.U. Conte, C. Treude, M.A. Gerosa, Overcoming open source project entry barriers with a portal for newcomers, in: 38th International Conference on Software Engineering, ICSE, 2016, pp. 273–284.

[22] B. Trinkenreich, M. Guizani, I. Wiese, A. Sarma, I. Steinmacher, Hidden figures: Roles and pathways of successful oss contributors, ACM Hum.-Comput. Interact. 4 (CSCW) (2020) 1–22.

[23] M. Zhou, A. Mockus, What make long term contributors: willingness and opportunity in oss community, in: 34th International Conference on Software Engineering, ICSE, IEEE, 2012, pp. 518–528.

[24] L. Pradel, Quantifying the ramp-up problem in software projects, in: 20th International Conference on Evaluation and Assessment in Software Engineering, EASE, 2016, pp. 1–4.

[25] S. Balali, I. Steinmacher, U. Annamalai, A. Sarma, M. Gerosa, Newcomers' barriers... is that all? an analysis of mentors' and newcomers' barriers in OSS projects, Comput. Support. Coop. Work (CSCW) (2018).

[26] R. Britto, D.S. Cruzes, D. Smite, A. Sablis, Onboarding software developers and teams in three globally distributed legacy projects: a multi-case study, J Soft.: Evol. Process 30 (4) (2018) e1921.

[27] F. Fagerholm, A.S. Guinea, J. Borenstein, J. Münch, Onboarding in open source projects, IEEE Softw. 31 (6) (2014) 54–61.

[28] V.R. Basili, Evolving and packaging reading technologies, J. Syst. Softw. 38 (1) (1997) 3–12.

[29] G. Canfora, M. Di Penta, R. Oliveto, S. Panichella, Who is going to mentor newcomers in open source projects? in: ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering, FSE, 2012, pp. 1–11.

[30] D. Cubranic, G.C. Murphy, Hipikat: recommending pertinent software development artifacts, in: 25th International Conference on Software Engineering, ICSE, IEEE, 2003, pp. 408–418.

[31] B.A. Kitchenham, D. Budgen, P. Brereton, Evidence-based Software Engineering and Systematic Reviews, vol. 4, CRC Press, 2015.

[32] Ø. Hauge, C. Ayala, R. Conradi, Adoption of open source software in software-intensive organizations–a systematic literature review, Inf. Softw. Technol. 52 (11) (2010) 1133–1154.

[33] Y. Park, C. Jensen, Beyond pretty pictures: examining the benefits of code visualization for open source newcomers, in: 5th IEEE International Workshop on Visualizing Software for Understanding and Analysis, VISSOFT, IEEE, 2009, pp. 3–10.

[34] M. Burnett, S.D. Fleming, S. Iqbal, G. Venolia, V. Rajaram, U. Farooq, V. Grigoreanu, M. Czerwinski, Gender differences and programming environments: across programming populations, in: Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, 2010, pp. 1–10.

[35] A.-M. Cazan, E. Cocoradă, C.I. Maican, Computer anxiety and attitudes towards the computer and the internet with Romanian high-school and university students, Comput. Hum. Behav. 55 (2016) 258–267.

[36] A. Singh, V. Bhadauria, A. Jain, A. Gurung, Role of gender, self-efficacy, anxiety and testing formats in learning spreadsheets, Comput. Hum. Behav. 29 (3) (2013) 739–746.

[37] I. Santos, J.F. Pimentel, I. Wiese, I. Steinmacher, A. Sarma, M.A. Gerosa, Designing for cognitive diversity: improving the GitHub experience for newcomers, in: IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Society, ICSE-SEIS, 2023, pp. 1–12.

[38] R. Kaur, K.K. Chahal, M. Saini, Understanding community participation and engagement in open source software projects: a systematic mapping study, J. King Saud Univ. - Comput. Inf. Sci. 34 (7) (2022) 4607–4625.

[39] L. Heimburger, L. Buchweitz, R. Gouveia, O. Korn, Gamifying onboarding: how to increase both engagement and integration of new employees, in: International Conference on Social and Occupational Ergonomics, AHFE, Springer, 2020, pp. 3–14.

[40] C. Stanik, L. Montgomery, D. Martens, D. Fucci, W. Maalej, A simple nlp-based approach to support onboarding and retention in open source communities, in: IEEE International Conference on Software Maintenance and Evolution, ICSME, IEEE, 2018, pp. 172–182.

[41] J. Wang, A. Sarma, Which bug should I fix: helping new developers onboard a new project, in: 4th International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE, 2011, pp. 76–79.

[42] J. Biolchini, P.G. Mian, A.C.C. Natali, G.H. Travassos, Systematic review in software engineering, System Engineering and Computer Science Department COPPE/UFRJ, Technical Report ES 679 (05) (2005) 45.

[43] T. Dyba, T. Dingsoyr, G.K. Hanssen, Applying systematic reviews to diverse study types: An experience report, in: First International Symposium on Empirical Software Engineering and Measurement, ESEM, IEEE, 2007, pp. 225–234.

[44] A. Valente, M. Holanda, A.M. Mariano, R. Furuta, D. Da Silva, Analysis of academic databases for literature review in the computer science education field, in: 2022 Ieee Frontiers in Education Conference (Fie), IEEE, 2022, pp. 1–7.

[45] A.-W. Harzing, S. Alakangas, Google scholar, scopus and the web of science: a longitudinal and cross-disciplinary comparison, Scientometrics 106 (2016) 787–804.

[46] A. Carrera-Rivera, W. Ochoa, F. Larrinaga, G. Lasa, How-to conduct a systematic literature review: A quick guide for computer science research, MethodsX 9 (2022) 101895.

[47] B. Kitchenham, L. Madeyski, D. Budgen, How should software engineering secondary studies include grey material? IEEE Trans. Softw. Eng. 49 (2) (2022) 872–882.

[48] A. Yasin, R. Fatima, L. Wen, W. Afzal, M. Azhar, R. Torkar, On using grey literature and google scholar in systematic literature reviews in software engineering, IEEE Access 8 (2020) 36226–36243.

[49] P. Brereton, B.A. Kitchenham, D. Budgen, M. Turner, M. Khalil, Lessons from applying the systematic literature review process within the software engineering domain, J. Syst. Softw. 80 (4) (2007) 571–583.

[50] G.C. Diniz, M.A.G. Silva, M.A. Gerosa, I. Steinmacher, Using gamification to orient and motivate students to contribute to OSS projects, in: IEEE/ACM 10th International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE, IEEE, 2017, pp. 36–42.

[51] J. Dominic, J. Houser, I. Steinmacher, C. Ritter, P. Rodeghero, Conversational bot for newcomers onboarding to open source projects, in: IEEE/ACM 42nd International Conference on Software Engineering Workshops, ICSEW, 2020, pp. 46–50.

[52] C. Fu, M. Zhou, Q. Xuan, H.-X. Hu, Expert recommendation in OSS projects based on knowledge embedding, in: International Workshop on Complex Systems and Networks, IWCSN, IEEE, 2017, pp. 149–155.

[53] M. Guizani, T. Zimmermann, A. Sarma, D. Ford, Attracting and retaining OSS contributors with a maintainer dashboard, in: ACM/IEEE 44th International Conference on Software Engineering: Software Engineering in Society, ICSE-SEIS, 2022, pp. 36–40.

[54] H. He, H. Su, W. Xiao, R. He, M. Zhou, GFI-bot: automated good first issue recommendation on GitHub, in: 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE, 2022, pp. 1751–1755.

[55] H. Kagdi, M. Hammad, J.I. Maletic, Who can help me with this source code change? in: IEEE International Conference on Software Maintenance, ICSM, IEEE, 2008, pp. 157–166.

[56] R. Medeiros, O. Díaz, Assisting mentors in selecting newcomers' next task in software product lines: A recommender system approach, in: Advanced Information Systems Engineering: 34th International Conference, CAiSE, Springer, 2022, pp. 460–476.

[57] L. Nagel, O. Karras, J. Klünder, Ontology-based software graphs for supporting code comprehension during onboarding, in: 47th Euromicro Conference on Software Engineering and Advanced Applications, SEAA, IEEE, 2021, pp. 158–165.

[58] A. Sarma, M.A. Gerosa, I. Steinmacher, R. Leano, Training the future workforce through task curation in an OSS ecosystem, in: 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, FSE, 2016, pp. 932–935.

[59] L.P. Serrano Alves, I.S. Wiese, A.P. Chaves, I. Steinmacher, How to find my task? chatbot to assist newcomers in choosing tasks in OSS projects, in: Chatbot Research and Design: 5th International Workshop, CONVERSATIONS, Springer, 2022, pp. 90–107.

[60] I. Steinmacher, I.S. Wiese, M.A. Gerosa, Recommending mentors to software project newcomers, in: Third International Workshop on Recommendation Systems for Software Engineering, RSSE, IEEE, 2012, pp. 63–67.

[61] C. Toscani, D. Gery, I. Steinmacher, S. Marczak, A gamification proposal to support the onboarding of newcomers in the flosscoach portal, in: 17th Brazilian Symposium on Human Factors in Computing Systems, IHC, 2018, pp. 1–10.

[62] W. Xiao, H. He, W. Xu, X. Tan, J. Dong, M. Zhou, Recommending good first issues in GitHub OSS projects, in: 44th International Conference on Software Engineering, ICSE, 2022, pp. 1830–1842.

[63] H. Yin, Z. Sun, Y. Sun, G. Huang, Automatic learning path recommendation for open source projects using deep learning on knowledge graphs, in: IEEE 45th Annual Computers, Software, and Applications Conference, COMPSAC, IEEE, 2021, pp. 824–833.

[64] D. Ford, N. Shrestha, T. Zimmermann, Reboc: recommending bespoke open source software projects to contributors, in: IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), IEEE, 2022, pp. 1–5.

[65] C. Liu, D. Yang, X. Zhang, B. Ray, M.M. Rahman, Recommending GitHub projects for developer onboarding, IEEE Access 6 (2018) 52082–52094.

[66] F. Santos, I. Wiese, B. Trinkenreich, I. Steinmacher, A. Sarma, M.A. Gerosa, Can i solve it? identifying apis required to complete OSS tasks, in: IEEE/ACM 18th International Conference on Mining Software Repositories, MSR, IEEE, 2021, pp. 257–346.

[67] S. Minto, G.C. Murphy, Recommending emergent teams, in: Fourth International Workshop on Mining Software Repositories, MSR-ICSEW, IEEE, 2007, p. 5.

[68] Y. Malheiros, A. Moraes, C. Trindade, S. Meira, A source code recommender system to support newcomers, in: IEEE 36th Annual Computer Software and Applications Conference, COMPSAC, IEEE, 2012, pp. 19–24.

[69] C. Yang, Q. Fan, T. Wang, G. Yin, H. Wang, Repolike: personal repositories recommendation in social coding communities, in: 8th Asia-Pacific Symposium on Internetware (Internetware), 2016, pp. 54–62.

[70] Y. Zhou, J. Wu, Y. Sun, Ghtrec: a personalized service to recommend GitHub trending repositories for developers, in: IEEE International Conference on Web Services, ICWS, IEEE, 2021, pp. 314–323.

[71] A.S.M. Venigalla, K. Boyalakuntla, S. Chimalakonda, GitQ-towards using badges as visual cues for GitHub projects, in: 30th IEEE/ACM International Conference on Program Comprehension, ICPC, 2022, pp. 157–161.

[72] X. Sun, W. Xu, X. Xia, X. Chen, B. Li, Personalized project recommendation on GitHub, Sci. China Inf. Sci. 61 (2018) 1–14.

[73] A. Sarma, L. Maccherone, P. Wagstrom, J. Herbsleb, Tesseract: interactive visual exploration of socio-technical relationships in software development, in: IEEE 31st International Conference on Software Engineering, ICSE, IEEE, 2009, pp. 23–33.

[74] C. Wohlin, Guidelines for snowballing in systematic literature studies and a replication in software engineering, in: 18th International Conference on Evaluation and Assessment in Software Engineering, EASE, 2014, pp. 1–10.

[75] J. Corbin, A. Strauss, Techniques and procedures for developing grounded theory, in: Basics of Qualitative Research, 3rd ed., Sage, Thousand Oaks, CA, USA, 2008, pp. 860–886.

[76] F. Heimerl, S. Lohmann, S. Lange, T. Ertl, Word cloud explorer: text analytics based on word clouds, in: 47th Hawaii International Conference on System Sciences, HICSS, IEEE, 2014, pp. 1833–1842.

[77] M. Robillard, R. Walker, T. Zimmermann, Recommendation systems for software engineering, IEEE Softw. 27 (4) (2009) 80–86.

[78] D. Moody, The "physics" of notations: toward a scientific basis for constructing visual notations in software engineering, IEEE Trans. Softw. Eng. (TSE) 35 (6) (2009) 756–779.

[79] L.C. Kats, R.G. Vogelij, K.T. Kalleberg, E. Visser, Software development environments on the web: a research agenda, in: ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software (Onward!), 2012, pp. 99–116.

[80] S.K. Lo, Q. Lu, C. Wang, H.-Y. Paik, L. Zhu, A systematic literature review on federated machine learning: from a software engineering perspective, ACM Comput. Surv. 54 (5) (2021) 1–39.

[81] A. Agrawal, J.S. Gans, A. Goldfarb, Artificial intelligence adoption and system-wide change, J. Econ. Manag. Strategy (2023).

[82] T.P. Nagarhalli, V. Vaze, N. Rana, A review of current trends in the development of chatbot systems, in: 6th International Conference on Advanced Computing and Communication Systems, ICACCS, IEEE, 2020, pp. 706–710.

[83] F. Nayebi, J.-M. Desharnais, A. Abran, The state of the art of mobile application usability evaluation, in: 25th IEEE Canadian Conference on Electrical and Computer Engineering, CCECE, IEEE, 2012, pp. 1–4.

[84] K.-J. Stol, B. Fitzgerald, Guidelines for conducting software engineering research, in: Contemporary Empirical Methods in Software Engineering, Springer, 2020, pp. 27–62.

[85] K.-J. Stol, B. Fitzgerald, The ABC of software engineering research, ACM Trans. Soft. Eng. Methodol. (TOSEM) 27 (3) (2018) 1–51.

[86] K.A. Jehn, G.B. Northcraft, M.A. Neale, Why differences make a difference: A field study of diversity, conflict and performance in workgroups, Adm. Sci. Q. 44 (4) (1999) 741–763.

[87] G. Rodríguez-Pérez, R. Nadri, M. Nagappan, Perceived diversity in software engineering: a systematic literature review, Empir. Softw. Eng. 26 (2021) 1–38.

[88] K. Blincoe, O. Springer, M.R. Wrobel, Perceptions of gender diversity's impact on mood in software development teams, IEEE Softw. 36 (5) (2019) 51–56.

[89] S.K. Horwitz, I.B. Horwitz, The effects of team diversity on team outcomes: a meta-analytic review of team demography, J. Manag. (2007).

[90] B. Vasilescu, D. Posnett, B. Ray, M.G. van den Brand, A. Serebrenik, P. Devanbu, V. Filkov, Gender and tenure diversity in GitHub teams, in: ACM CHI Conference, 2015.

[91] J. Marlow, L. Dabbish, J. Herbsleb, Impression formation in online peer production: activity traces and personal profiles in GitHub, in: Conference on Computer Supported Cooperative Work, ACM, 2013.
[92] L. Singer, F. Figueira Filho, B. Cleary, C. Treude, M.-A. Storey, K. Schneider, Mutual assessment in the social programmer ecosystem: an empirical investigation of developer profile aggregators, in: Conference on Computer Supported Cooperative Work, 2013.
[93] D. Ford, A. Harkins, C. Parnin, Someone like me: how does peer parity influence participation of women on stack overflow? in: IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), IEEE CS, 2017.
[94] G. Robles, L.A. Reina, J.M. González-Barahona, S.D. Domínguez, Women in free/libre/open source software: the situation in the 2010s, in: 12th IFIP Advances in Information and Communication Technology (OSS), Springer, 2016.
[95] B. Trinkenreich, I. Wiese, A. Sarma, M. Gerosa, I. Steinmacher, Women's participation in open source software: a survey of the literature, ACM Trans. Soft. Eng. Methodol. (TOSEM) (2022).
[96] B. Dagenais, M.P. Robillard, Recommending adaptive changes for framework evolution, ACM ACM Trans. Softw. Eng. Methodol. (TOSEM) 20 (4) (2011) 1–35.
[97] Z. Kotti, R. Galanopoulou, D. Spinellis, Machine learning for software engineering: a tertiary study, ACM Comput. Surv. 55 (12) (2023) 1–39.
[98] K. Meinke, A. Bennaceur, Machine learning for software engineering: Models, methods, and applications, in: IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion), 2018, pp. 548–549.
[99] J. Zheng, L. Williams, N. Nagappan, W. Snipes, J.P. Hudepohl, M.A. Vouk, On the value of static analysis for fault detection in software, IEEE Trans. Softw. Eng. 32 (4) (2006) 240–253.
[100] A. Nederlof, A. Mesbah, A.V. Deursen, Software engineering for the web: the state of the practice, in: 36th International Conference on Software Engineering, 2014, pp. 4–13.
[101] V. Goduguluri, T. Kilamo, I. Hammouda, Kommgame: a reputation environment for teaching open source software, in: 7th IFIP Advances in Information and Communication Technology, OSS, Springer, 2011, pp. 312–315.
[102] B. Morgan, C. Jensen, Lessons learned from teaching open source software development, in: 10th IFIP Advances in Information and Communication Technology, OSS, Springer, 2014, pp. 133–142.
[103] D.M. Nascimento, K. Cox, T. Almeida, W. Sampaio, R.A. Bittencourt, R. Souza, C. Chavez, Using open source projects in software engineering education: a systematic mapping study, in: IEEE Frontiers in Education Conference, FIE, IEEE, 2013, pp. 1837–1843.
[104] S. Deterding, D. Dixon, R. Khaled, L. Nacke, From game design elements to gamefulness: defining "gamification", in: 15th International Academic MindTrek Conference: Envisioning Future Media Environments (MindTrek), 2011, pp. 9–15.
[105] J. Bell, S. Sheth, G. Kaiser, Increasing student engagement in software engineering with gamification, in: 4th International Workshop on Social Software Engineering, SSE, 2012, pp. 1–2.
[106] A.P.O. Bertholdo, M.A. Gerosa, Promoting engagement in open collaboration communities by means of gamification, in: HCI International 2016–Posters' Extended Abstracts: 18th International Conference, Springer, 2016, pp. 15–20.
[107] A. Bartel, G. Hagel, Gamifying the learning of design patterns in software engineering education, in: IEEE Global Engineering Education Conference, EDUCON, IEEE, 2016, pp. 74–79.
[108] D. Dicheva, C. Dichev, G. Agre, G. Angelova, Gamification in education: a systematic mapping study, J. Educ. Technol. Soc. (JSTOR) 18 (3) (2015) 75–88.
[109] O. Pedreira, F. García, N. Brisaboa, M. Piattini, Gamification in software engineering–a systematic mapping, Inf. Soft. Technol. (IST) 57 (2015) 157–168.
[110] I. Qureshi, Y. Fang, Socialization in open source software projects: a growth mixture modeling approach, Organ. Res. Methods 14 (1) (2011) 208–238.
[111] A. Begel, B. Simon, Novice software developers, all over again, in: Fourth International Workshop on Computing Education Research, ICER, 2008, pp. 3–14.
[112] S. Easterbrook, J. Singer, M.-A. Storey, D. Damian, Selecting empirical methods for software engineering research, Guide Adv. Empir. Softw. Eng. (2008) 285–311.
[113] S.H. Padala, C.J. Mendez, L.F. Dias, I. Steinmacher, Z.S. Hanson, C. Hilderbrand, A. Horvath, C. Hill, L.D. Simpson, M. Burnett, et al., How gender-biased tools shape newcomer experiences in OSS projects, IEEE Trans. Soft. Eng. (TSE) (2020).
[114] A. Bosu, K.Z. Sultana, Diversity and inclusion in open source software (OSS) projects: where do we stand? in: ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM, IEEE, 2019, pp. 1–11.
[115] M. Guizani, I. Steinmacher, J. Emard, A. Fallatah, M. Burnett, A. Sarma, How to debug inclusivity bugs? a debugging process with information architecture, in: ACM/IEEE 44th International Conference on Software Engineering: Software Engineering in Society, ICSE-SEIS, 2022.
[116] M.-A. Storey, A. Zagalsky, F. Figueira Filho, L. Singer, D.M. German, How social and communication channels shape and challenge a participatory culture in software development, IEEE Trans. Soft. Eng. (TSE) (2016).
[117] B. Kitchenham, S. Charters, et al., Guidelines for performing systematic literature reviews in software engineering version 2.3, Engineering 45 (4ve) (2007) 1051.
[118] S. Jalali, C. Wohlin, Systematic literature studies: database searches vs. backward snowballing, in: ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM, 2012, pp. 29–38.
[119] A. Ampatzoglou, S. Bibi, P. Avgeriou, M. Verbeek, A. Chatzigeorgiou, Identifying, categorizing and mitigating threats to validity in software engineering secondary studies, Inf. Softw. Technol. 106 (2019) 201–230.
[120] A.A. Khan, A. Ahmad, M. Waseem, P. Liang, M. Fahmideh, T. Mikkonen, P. Abrahamsson, Software architecture for quantum computing systems—A systematic review, J. Syst. Softw. 201 (2023) 111682.
[121] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, A. Wesslén, Experimentation in Software Engineering, Springer Science & Business Media, 2012.
[122] D.M. Cable, F. Gino, B.R. Staats, Reinventing employee onboarding, MIT Sloan Manage. Rev. (2013).
[123] H.J. Klein, B. Polin, K. Leigh Sutton, Specific onboarding practices for the socialization of new employees, Int. J. Sel. Assess. 23 (3) (2015) 263–283.
[124] N. Talya, D. Bauer, Onboarding new employees: Maximizing success, 2014.
[125] P. Gregory, D.E. Strode, H. Sharp, L. Barroca, An onboarding model for integrating newcomers into agile project teams, Inf. Softw. Technol (IST) 143 (2022) 106792.
[126] I. Santos, I. Wiese, I. Steinmacher, A. Sarma, M.A. Gerosa, Hits and misses: Newcomers' ability to identify skills needed for oss tasks, in: IEEE SANER, IEEE, 2022, pp. 174–183.
[127] I. Steinmacher, I. Wiese, A.P. Chaves, M.A. Gerosa, Why do newcomers abandon open source software projects? in: 6th International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE, IEEE, 2013, pp. 25–32.
[128] F. Fronchetti, I. Wiese, G. Pinto, I. Steinmacher, What attracts newcomers to onboard on OSS projects? tl; dr: Popularity, in: 15th IFIP Advances in Information and Communication Technology, OSS, Springer, 2019, pp. 91–103.
[129] 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: 48th Hawaii International Conference on System Sciences, HICSS, IEEE, 2015, pp. 5299–5308.
[130] V. Wolff-Marting, C. Hannebauer, V. Gruhn, Patterns for tearing down contribution barriers to FLOSS projects, in: 12th International Conference on Intelligent Software Methodologies, Tools and Techniques (SoMeT), IEEE, 2013.
[131] A. Darejeh, S.S. Salim, Gamification solutions to enhance software user engagement—a systematic review, Int. J. Hum.-Comput. Interact. 32 (8) (2016) 613–642.

Primary studies

[PS01] M. Azanza, A. Irastorza, R. Medeiros, O. Díaz, Onboarding in software product lines: concept maps as welcome guides, in: IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training, ICSE-SEET, IEEE, 2021, pp. 122–133.
[PS02] G. Canfora, M. Di Penta, R. Oliveto, S. Panichella, Who is going to mentor newcomers in open source projects? in: ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering, FSE, 2012, pp. 1–11.
[PS03] D. Cubranic, G.C. Murphy, Hipikat: recommending pertinent software development artifacts, in: 25th International Conference on Software Engineering, ICSE, IEEE, 2003, pp. 408–418.
[PS04] G.C. Diniz, M.A.G. Silva, M.A. Gerosa, I. Steinmacher, Using gamification to orient and motivate students to contribute to OSS projects, in: IEEE/ACM 10th International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE, IEEE, 2017, pp. 36–42.
[PS05] J. Dominic, J. Houser, I. Steinmacher, C. Ritter, P. Rodeghero, Conversational bot for newcomers onboarding to open source projects, in: IEEE/ACM 42nd International Conference on Software Engineering Workshops, ICSEW, 2020, pp. 46–50.
[PS06] C. Fu, M. Zhou, Q. Xuan, H.-X. Hu, Expert recommendation in OSS projects based on knowledge embedding, in: International Workshop on Complex Systems and Networks, IWCSN, IEEE, 2017, pp. 149–155.
[PS07] M. Guizani, T. Zimmermann, A. Sarma, D. Ford, Attracting and retaining OSS contributors with a maintainer dashboard, in: ACM/IEEE 44th International Conference on Software Engineering: Software Engineering in Society, ICSE-SEIS, 2022, pp. 36–40.
[PS08] H. He, H. Su, W. Xiao, R. He, M. Zhou, GFI-bot: automated good first issue recommendation on GitHub, in: 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE, 2022, pp. 1751–1755.
[PS09] H. Kagdi, M. Hammad, J.I. Maletic, Who can help me with this source code change? in: IEEE International Conference on Software Maintenance, ICSM, IEEE, 2008, pp. 157–166.

[PS10] R. Medeiros, O. Díaz, Assisting mentors in selecting newcomers' next task in software product lines: A recommender system approach, in: Advanced Information Systems Engineering: 34th International Conference, CAiSE, Springer, 2022, pp. 460–476.

[PS11] L. Nagel, O. Karras, J. Klünder, Ontology-based software graphs for supporting code comprehension during onboarding, in: 47th Euromicro Conference on Software Engineering and Advanced Applications, SEAA, IEEE, 2021, pp. 158–165.

[PS12] A. Sarma, M.A. Gerosa, I. Steinmacher, R. Leano, Training the future workforce through task curation in an OSS ecosystem, in: 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, FSE, 2016, pp. 932–935.

[PS13] L.P. Serrano Alves, I.S. Wiese, A.P. Chaves, I. Steinmacher, How to find my task? chatbot to assist newcomers in choosing tasks in OSS projects, in: Chatbot Research and Design: 5th International Workshop, CONVERSATIONS, Springer, 2022, pp. 90–107.

[PS14] C. Stanik, L. Montgomery, D. Martens, D. Fucci, W. Maalej, A simple nlp-based approach to support onboarding and retention in open source communities, in: IEEE International Conference on Software Maintenance and Evolution, ICSME, IEEE, 2018, pp. 172–182.

[PS15] I. Steinmacher, T.U. Conte, C. Treude, M.A. Gerosa, Overcoming open source project entry barriers with a portal for newcomers, in: 38th International Conference on Software Engineering, ICSE, 2016, pp. 273–284.

[PS16] I. Steinmacher, I.S. Wiese, M.A. Gerosa, Recommending mentors to software project newcomers, in: Third International Workshop on Recommendation Systems for Software Engineering, RSSE, IEEE, 2012, pp. 63–67.

[PS17] C. Toscani, D. Gery, I. Steinmacher, S. Marczak, A gamification proposal to support the onboarding of newcomers in the flosscoach portal, in: 17th Brazilian Symposium on Human Factors in Computing Systems, IHC, 2018, pp. 1–10.

[PS18] J. Wang, A. Sarma, Which bug should I fix: helping new developers onboard a new project, in: 4th International Workshop on Cooperative and Human Aspects of Software Engineering, CHASE, 2011, pp. 76–79.

[PS19] W. Xiao, H. He, W. Xu, X. Tan, J. Dong, M. Zhou, Recommending good first issues in GitHub OSS projects, in: 44th International Conference on Software Engineering, ICSE, 2022, pp. 1830–1842.

[PS20] H. Yin, Z. Sun, Y. Sun, G. Huang, Automatic learning path recommendation for open source projects using deep learning on knowledge graphs, in: IEEE 45th Annual Computers, Software, and Applications Conference, COMPSAC, IEEE, 2021, pp. 824–833.

[PS21] D. Ford, N. Shrestha, T. Zimmermann, Reboc: recommending bespoke open source software projects to contributors, in: IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC, IEEE, 2022, pp. 1–5.

[PS22] C. Liu, D. Yang, X. Zhang, B. Ray, M.M. Rahman, Recommending GitHub projects for developer onboarding, IEEE Access 6 (2018) 52082–52094.

[PS23] I. Santos, J.F. Pimentel, I. Wiese, I. Steinmacher, A. Sarma, M.A. Gerosa, Designing for cognitive diversity: improving the GitHub experience for newcomers, in: IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Society, ICSE-SEIS, 2023, pp. 1–12.

[PS24] F. Santos, I. Wiese, B. Trinkenreich, I. Steinmacher, A. Sarma, M.A. Gerosa, Can i solve it? identifying apis required to complete oss tasks, in: IEEE/ACM 18th International Conference on Mining Soft- ware Repositories, MSR, IEEE, 2021, pp. 346–257.

[PS25] S. Minto, G.C. Murphy, Recommending emergent teams, in: Fourth International Workshop on Mining Software Repositories, MSR-ICSEW, IEEE, 2007, p. 5.

[PS26] L. Heimburger, L. Buchweitz, R. Gouveia, O. Korn, Gamifying onboarding: how to increase both engagement and integration of new employees, in: International Conference on Social and Occupational Ergonomics, AHFE, Springer, 2020, pp. 3–14.

[PS27] Y. Malheiros, A. Moraes, C. Trindade, S. Meira, A source code recommender system to support newcomers, in: IEEE 36th Annual Computer Software and Applications Conference, COMPSAC, IEEE, 2012, pp. 19–24.

[PS28] C. Yang, Q. Fan, T. Wang, G. Yin, H. Wang, Repolike: personal repositories recommendation in social coding communities, in: 8th Asia-Pacific Symposium on Internetware (Internetware), 2016, pp. 54–62.

[PS29] Y. Zhou, J. Wu, Y. Sun, Ghtrec: a personalized service to recommend GitHub trending repositories for developers, in: IEEE International Conference on Web Services, ICWS, IEEE, 2021, pp. 314–323.

[PS30] A.S.M. Venigalla, K. Boyalakuntla, S. Chimalakonda, Gitq-towards using badges as visual cues for GitHub projects, in: 30th IEEE/ACM International Conference on Program Comprehension, ICPC, 2022, pp. 157–161.

[PS31] X. Sun, W. Xu, X. Xia, X. Chen, B. Li, Personalized project recommendation on GitHub, Sci. China Inf. Sci. 61 (2018) 1–14.

[PS32] A. Sarma, L. Maccherone, P. Wagstrom, J. Herbsleb, Tesseract: interactive visual exploration of socio-technical relationships in software development, in: IEEE 31st International Conference on Software Engineering, ICSE, IEEE, 2009, pp. 23–33.