Designing for Cognitive Diversity: Improving the GitHub Experience for Newcomers

Italo Santos*, João Felipe Pimentel*, Igor Wiese†, Igor Steinmacher*, Anita Sarma‡ and Marco A. Gerosa*
*Northern Arizona University, Flagstaff, AZ, USA
†Federal University of Technology, Campo Mourao, PR, Brazil
‡Oregon State University, Corvallis, OR, USA
Email: italo_santos@nau.edu, joao.pimentel@nau.edu, igor@utfpr.edu.br,
igor.steinmacher@nau.edu, anita.sarma@oregonstate.edu, marco.gerosa@nau.edu

***Abstract*—Social coding platforms such as GitHub have become *defacto* environments for collaborative programming and open source. When these platforms do not support specific cognitive styles, they create barriers to programming for some populations. Research shows that the cognitive styles typically favored by women are often unsupported, creating barriers to entry for woman newcomers. In this paper, we use the GenderMag method to evaluate GitHub to find cognitive style-specific inclusivity bugs. We redesigned the "buggy" GitHub features through a web browser plugin, which we evaluated through a between-subjects experiment (n=75). Our results indicate that the changes to the interface improve users' performance and self-efficacy, mainly for individuals with cognitive styles more common to women. Our results can inspire designers of social coding platforms and software engineering tools to produce more inclusive development environments.**

***General Abstract*—Diversity is an important aspect of society. One form of diversity is cognitive diversity—differences in cognitive styles, which helps generate a diversity of thoughts. Unfortunately, software tools often do not support different cognitive styles (e.g., learning styles), disproportionately impacting those whose styles are not supported. These individuals pay a cognitive "tax" each time they use the tools. In this work, we found "inclusivity bugs" in GitHub, a social coding platform. We then redesigned these buggy features and evaluated them with users. Our results show that the redesign makes it easier for the group of individuals whose cognitive styles were unsupported in the original design, with the percentage of completed tasks rising from 67% to 95% for this group.**

keywords: open source, diversity and inclusion, human factors, cognitive styles, human-computer interaction.

I. Introduction

Open Source Software (OSS) projects play an important role in improving inclusion in workforce development, where contributors join projects to learn new skills [1], showcase their skills [2], or improve their career paths [3]. Successful participation in OSS projects also helps newcomers gain visibility among their peers [4], [5], benefits society by developing a product used by many users [6], and improves their chances of achieving professional success [7], [5].

However, newcomers to OSS face several challenges [8], and these challenges differently affect underrepresented populations, including those whose cognitive styles are ill-supported by the project's information landscape [9], [10]. The consequences of these challenges to underrepresented populations may include a steeper learning curve, lack of community support, and obstacles to figuring out how to start contributing, all of which add to the diversity imbalance in OSS [11]. Social diversity has been shown to positively affect productivity, teamwork, and quality of contributions [12], [13]. On the other hand, low diversity has unfortunate effects: (i) OSS projects miss out on the benefits of a more expansive set of contributors and the diversity of thought that these potential contributors could bring; (ii) minorities miss out on the learning and experience opportunities that OSS projects provide; and (iii) minorities miss out on job opportunities when recruiters use OSS contributions to make hiring decisions [14], [15]. Although the lack of diversity in OSS has been well-documented for years, there is limited progress in closing this gap [11], [16], [17].

Past work [9], [10] has shown that the way information is provided in OSS 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 information architecture of OSS project pages (e.g., project description pages and descriptions of issues in the issue tracker) usually appeal to those who have high self-efficacy and are motivated by individual pursuits such as intellectual stimulation, competition, and learning technology for fun. According to Burnett et al. [18], these pursuits cater to characteristics associated with men, which can neglect women and other contributors who may have different motivations and personal characteristics (see also [19], [20]). This lack of support for diverse user characteristics leads to inclusivity bugs [21], [22]—software behaviors that disproportionately disadvantage a particular group of users of that software.

In our study, we investigate inclusivity bugs in the GitHub platform that affect newcomers to this platform. Inclusivity bugs in the platform can have far-reaching impacts on thousands of OSS projects (as of today, more than 200 Million repositories are hosted on GitHub). The following research questions guided our investigation:

Research Question 1

What inclusivity bugs does GitHub pose for newcomers trying to make their first contribution?

Research Question 2

What are the effects of fixing those inclusivity bugs?

We analyzed four tasks newcomers often perform to make their first pull request on GitHub and found inclusivity bugs in all of them. We redesigned the impacted interface to address the identified bugs and implemented a browser plugin to change the platform interface based on our redesign (we do not have access to change GitHub itself). We evaluated the original and the redesigned interface through a between-subject user study with 75 participants.

Our main goal is to mitigate cognitive barriers newcomers face due to inclusivity bugs. As we show in this paper, GitHub, a platform newcomers use to contribute to OSS, creates barriers for users with different characteristics, disproportionately impacting those from underrepresented groups. These barriers may discourage newcomers and add to the existing diversity gaps, as these tools and infrastructure are the main channels through which OSS newcomers interact with the community. This paper provides insights into how newcomers' performance can be improved when their cognitive styles are supported. Providing adequate support for diverse cognitive styles can help improve the overall community diversity.

II. Related Work

This section discusses work related to newcomers' onboarding in OSS, diversity and bias in OSS, and cognitive styles.

Newcomer's Onboarding: Previous work has investigated OSS contribution challenges [8], [23], [24], [25], [26]. Steinmacher et al. [8] conducted a mixed-method study and identified 58 barriers faced by newcomers. Researchers have also investigated specific types of challenges. For example, toxic environments have been studied in the literature [27], [28], [29], which evidenced situations in which OSS project members were unfriendly, unhelpful, or elitist [30]. Jensen et al. [25] analyzed the speed at which emails sent by newcomers are answered, the role played by gender or nationality in the kinds of answers newcomers receive, and the reception newcomers face. A better understanding of the barriers enables communities and researchers to design and produce tools and conceive strategies to better support newcomers [31]. Our work complements existing literature by focusing on making social coding platforms more inclusive by supporting the onboarding of newcomers with different cognitive styles.

Diversity/Bias in OSS: Low diversity in OSS is a concern raised by different studies in the literature when considering gender [11], [22], [27], [32], language [30], and location [30]. Past work has shown that diverse teams are more productive [13]. However, minorities face challenges in becoming a part of an OSS community [11]. Most OSS communities function as meritocracies [33], in which minorities report experiencing "imposter syndrome" [13]. These competitive settings have been known to discourage minorities such as women in OSS [34], [35]. Participant observation of OSS contributors found that "men monopolize code authorship and simultaneously de-legitimize the kinds of social ties necessary to build mechanisms for women's inclusion" [36]. Generally, cultures that describe themselves as meritocracies tend to be male-dominated ones that women experience as unfriendly [37]. In our work, we aim to reduce the bias found in social coding platforms used by a wide range of users to support them regarding their different cognitive styles to interact with OSS projects.

Cognitive styles: Research has shown that developers have different cognitive styles [38] and motivation [1], and that cognition plays an essential role in software engineering activities [39]. For example, more women are task-oriented, whereas more men are motivated to learn a new technology for fun [9], [10], [40]. These differences in cognitive styles may negatively impact how women and men contribute to OSS, and it mainly happens when OSS projects and the underlying infrastructure support certain cognitive styles (e.g., selective information processing or learning by tinkering) and impede others (e.g., comprehensive information processing or process-oriented learning). Our work considers a variety of cognitive styles to propose changes to GitHub to support diverse newcomers.

III. Research Method

We followed a three-step method, as illustrated in Figure 1: (1) we conducted a GenderMag analysis, which has been extensively used to detect gender biases in commercial and OSS products [10], [41], [42], [43], [44]; (2) we proposed fixes to the GitHub-related inclusivity bugs and developed a browser plugin to implement these changes in the GitHub interface; and (3) we conducted an experiment to compare the original GitHub interface with the interface enriched by the plugin.

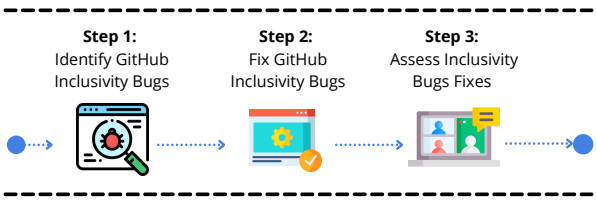

Figure 1. Research method overview.

A. Step 1 - Identifying GitHub Inclusivity Bugs

To identify inclusivity bugs, we used GenderMag [38], a systematic inspection method tool builders can use to evaluate their software for inclusivity bugs. GenderMag is based on research showing that individual differences in cognitive styles (referred to as facets) cluster by gender. The method encapsulates these facets into personas–Abi, Pat, and Tim. Abi and Tim occupy the opposite spectrum of facet values, with the Abi persona aligned with facet values that women tend to favor and Tim embodying facet values typically favored by men. The Pat persona includes a mix of these facet values.

The five facets that GenderMag uses are: (i) **Motivation:** Abis are motivated to use technology for what they can accomplish with it, whereas Tims are often motivated by their enjoyment of technology per se [18], [45], [46]; (ii)

Information processing styles: Abis process new information comprehensively—gathering fairly complete information before proceeding—but Tims use selective information processing—following the first promising information, then backtracking if needed [47], [48]; (iii) **Computer self-efficacy:** relates with a person's confidence about succeeding at a specific task, which influences their use of cognitive strategies, persistence, and strategies for coping with obstacles. Abis have lower computer self-efficacy as compared to their peers; (iv) **Risk aversion:** Abis are risk-averse when trying out new features as compared to Tims [49], [50], which impact their decisions about which feature sets to use; and (v) **Learning: by Process vs. by Tinkering:** Abis prefer process-oriented learning, whereas Tims like to playfully experiment ("tinker") with software features new to them [18], [51], [52]. Each cognitive style has advantages, but either is at a disadvantage when not supported by the software [53].

GenderMag is used by evaluation teams to walk through a use case in the project they are evaluating using Abi, Pat, or Tim personas. At each step of the walkthrough, the team writes down the answers to three questions:

- **SubgoalQ:** Will Abi have formed this subgoal as a step to their overall goal? (Yes/no/maybe, why, facets involved).
- ***ActionQ1:*** Will Abi know what to do at this step? (Yes/no/maybe, why, facets involved).
- ***ActionQ2:*** If Abi does the right thing, will s/he know s/he did the right thing and is making progress toward their goal? (Yes/no/maybe, why, facets involved).

When the answer to any of these questions is negative, the team identifies a potential bug; if the "why" relates to a particular cognitive style, this shows a disproportionate effect on people who have that cognitive style—i.e., an *inclusivity bug*. Thus, a team's answers to these questions become their inclusivity bug report, which they can process and prioritize the same way they would with any other type of bug report.

We selected the Abi and Tim personas [54] as they represent opposite ends of the GenderMag facet ranges. We customized the persona's profile to represent our target users: newcomers looking to make their first contribution using GitHub and have never performed a pull request (PR) before. We identified four use cases (i.e., edit a file, submit a pull request, fork repository, upload a new file) as described in Table I.

Table I
GENDERMAG ANALYSIS USE CASE AND SUBGOALS

Use Case	Subgoals
UC#1 - Submit a pull request	#1.1 - Make a change to a README file
	#1.2 - Submit the pull request
UC#2 - View changed files in PR	#2.1 - Find the changed files in the interface
UC#3 - Request help to solve the pull request	#3.1 - Find an experienced contributor in the project to ask for help to solve the PR
UC#4 - Upload file	#4.1 - Discover how to upload a file
	#4.2 - Request push access to upload file

Given these personas and use cases, 6 members of our research group conducted the GenderMag walkthroughs on GitHub-hosted projects using the procedures defined by Burnett et al. [54]. The group had prior training and experience in conducting GenderMag analysis. As a first step, the group identified the subgoals and actions for each use case. We then performed the GenderMag evaluations for each use case by first using the Abi persona and then another set of evaluations with the Tim persona. We identified 12 inclusivity bugs in different parts of the GitHub interface.

B. Step 2 - Fixing GitHub Inclusivity Bugs

We redesigned the GitHub interface to support the GenderMag facets that were previously unsupported and caused the inclusivity bugs we identified in Step 1. As stated by Guizani et al. [22], the outcomes of GenderMag analysis point not only to inclusivity bugs but also to why the bugs might arise and what specific problem-solving facet(s) are implicated.

As an example of redesign, for UC#1, we identified an issue related to Abi's process-oriented learning style and self-efficacy facets that would affect her ability to edit a file in an OSS project. The redesign focused on Abi's process-oriented learning facet to give explicit guidance on submitting a pull request by leveraging the design principle of "visibility." We did so by: (1) presenting the README file information to users more explicitly through a new tab called home (Figure 2), which highlights the importance of the README file, and (2) including a tooltip to explain that the user can edit the file: *To edit this file, go to the "code" tab above, and select the file you want to edit.* Our proposed solution also addressed Abi's self-efficacy facet by showing that she is on the right track to completing the subgoal (#1.1, make changes to README).

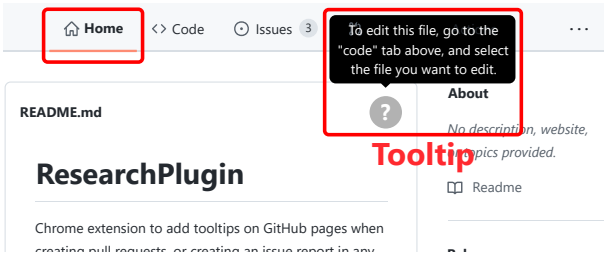

Figure 2. GitHub interface modified by the developed plugin.

Once our research team agreed with the redesign solutions proposed for each issue identified in Step 1, we started the development of a plugin to change GitHub's interface. The plugin was developed as a Chrome extension to change the original GitHub interface. The plugin is developed in JavaScript and uses the GitHub API to collect data about a user in JSON format. It is available on GitHub[1] for anyone interested in using it and making contributions, as well as in the supplementary material[2].

C. Step 3 - Assessing the Inclusivity Bug Fixes

Finally, we conducted an experiment to evaluate how the modified interface changed the user experience for Abis. Even though inclusivity bugs can be fixed in multiple ways, we

[1]https://github.com/NAU-OSL/ResearchPlugin
[2]https://figshare.com/s/4e7724bde0b1d47ecaeb

expected that we would reduce an eventual performance gap between Abis and Tims who use the modified interface, since the modifications were supported by the analytic/theory-based method.

We follow the guidelines provided in [55] to report our experiment. We conducted an experiment to *analyze* how the proposed plugin supports newcomers with different cognitive styles. We compared users using GitHub's original version to users using our GitHub plugin, *for the purpose of* evaluating, *with respect to their* effectiveness in completing the use cases, *from the point of view of the* researchers, *in the context of* the GitHub environment when a newcomer attempts to make their first contribution.

The participants interacted with a copy of a community-based OSS project named JabRef[3]. Participants completed the four use cases used for finding the bugs (Table I):

- UC#1 - Submit a pull request: The newcomer needs to edit a file in the project and submit the changes via a pull request (PR);
- UC#2 - View changed file. In this task, we asked participants to analyze an open pull request and find which files were changed when this pull request was created;
- UC#3 - Request help to solve the PR. The participant needs to find an experienced project contributor and invite them to work together to solve the pull request; and
- UC#4 - Upload a file: The participant should try to upload a new file to the repository.

We conducted a pilot study with five researchers outside our group to collect feedback about the instruments (questionnaires and use case definitions) and study design. The pilot study helped to improve our instruments. We used an iterative process to apply the necessary changes after each pilot session. This resulted in more detailed scripts and documentation about the use cases. We ran new pilot sessions until we reached a consensus that the instruments were reliable enough to start the actual study. A replication package with this material is available online (see the previous subsection). The replication package also includes the developed GitHub plugin and installation instructions.

We recruited 75 undergraduate students from diverse STEM majors from 5 distinct universities in the US and Brazil. The majority of participants were pursuing Computer Science majors. Our recruiting criteria were students who knew how to program but had never opened a pull request on GitHub, so previous experiences with the interface would not bias them. We opted to recruit undergraduate students for our study because the literature mentions that educators have been using OSS to train students, and these students are potential OSS project contributors [56]. We asked the students if they had previous experience with GitHub and OSS. Some of them responded that they had used GitHub once (Plugin = 10 and Control = 7), but when we questioned about what they had used GitHub for, they said that they just created the account but never contributed to any project, so they fit our criteria (never opened a pull request). We also asked about their experience with OSS, and a few participants answered that they had some experience (Plugin = 4 and Control = 3). When we asked what kind of experience they had, they informed us that they had studied OSS concepts in previous courses in college.

We used a between-subject design to balance participants in the original version (Control group) and GitHub plugin version (Plugin group) by GenderMag facets [57], [44]. We used GenderMag's questionnaire to assess participants' facets with 9-point Likert items [58]. We ended up with different numbers of participants between the two treatments, as some of the participants were a no-show, Table II).

Unfortunately, we had a small sample of women participants (18 vs. 57) due to the gender distribution of students in the classes we recruited from. We attempted to balance the participants in each treatment based on their cognitive facets, achieving an almost equal distribution of Abis (37) and Tims (38) across the treatment groups. Table II presents the participants' characteristics in each group.

In the beginning, we conducted each user session one participant at a time with a facilitator and an observer. The participants were asked to perform the four use cases described in Table I. We collected audio recordings and observation notes from the sessions and qualitatively analyzed participants' data. We conducted those individual sessions with 50% of our participants. Then we decided to optimize the data collection by conducting the experiment with students from two classes where we provided an online questionnaire with all the instructions they had to follow to participate in the experiment. A researcher was present the whole time to assist the students in case they needed help or had any questions.

We performed a quantitative analysis by collecting the percentage of use cases completed by participants in each group and applied a self-efficacy survey to measure newcomers' confidence in using GitHub.

Table II
NUMBER OF PARTICIPANTS IN THE EXPERIMENT

	Subjects	Facets		Gender	
		Tim	Abi	Man	Woman
Control	**36**	**18**	**18**	**30**	**6**
Plugin	**39**	**20**	**19**	**27**	**12**
Total	**75**	**38**	**37**	**57**	**18**

We also administered a questionnaire in which participants provided their self-perception about their ability to complete use cases using GitHub, i.e., self-efficacy to complete specific tasks. The questionnaire was based on the work of Bandura [59] and had 5 items. Participants answered those questions before and after the experiment using a 5-point Likert scale ranging from strongly disagree to strongly agree (with a neutral option). The goal was to capture the students' self-perceived efficacy about the use case before and after they attempted executing it. The items were prefixed with "I am confident that I can:" followed by: (i) ... use GitHub to contribute to projects; (ii) ... open a pull request using the GitHub web interface; (iii) ... change a file and submit the changes to the project using GitHub; (iv) ... find someone to

[3]https://github.com/JabRef/jabref

Table III
GITHUB INCLUSIVITY BUGS AND PROPOSED FIXES

Use Case	# Bugs	Bug Description	GenderMag Facets	Bug Fixes
#1 Submit pull request	1	Difficulty in finding Readme file to edit;	- Learning: Process vs. Tinkering; - Computer self-efficacy.	- Add "Home" link to the navbar to highlight the importance of the Readme File in the repository hierarchy. This link presents this file's content and includes a tooltip to explain that the user can edit the file.
	2	After clicking to edit the file, difficulty in finding the options to edit the file;	- Learning: Process vs. Tinkering; - Computer self-efficacy; - Attitude Towards Risk.	- Include a progress bar, to indicate the steps of the workflow related to this task; - Include a tooltip to explain what happens in case the user changes the original filename.
	3	Difficulty in understanding the commit form;	- Learning: Process vs. Tinkering; - Computer self-efficacy; - Attitude Towards Risk.	- Put tooltips and field labels explaining form fields to help the user understand the importance of informing a commit message.
	4	Difficulty in understanding the workflow after the file is edited;	- Motivations; - Learning: Process vs. Tinkering; - Information Processing Style.	- We have the progress bar, to indicate that this step is important to complete the task; - Include a tooltip to explain the conflict message that appears; - Include a tooltip to explain the code that is related to the changes.
	5	Lack of feedback indicating if the creation of the pull request was successful;	- Learning: Process vs. Tinkering; - Computer Self-Efficacy; - Information Processing Style.	- After the click on the create pull request button, redirect to a page with a success message and a progress bar showing that the pull request is completed; - Include a tooltip to explain what does the "Close pull request" button do.
#2 View changed files	6	Difficulty in understanding the workflow after the user opens a pull request;	- Learning: Process vs. Tinkering; - Computer Self-Efficacy.	- Include a tooltip to highlight the navbar that describes some actions that can be made in the pull request.
#3 Request help to solve the PR	7	Difficulty in finding the option to mention another contributor;	- Learning: Process vs. Tinkering; - Computer Self-Efficacy.	- Include a tooltip in the @ symbol icon to say "Use it to mention a contributor."
	8	Lack of feedback about the action of mentioning another contributor;	- Information Processing Style; - Computer Self-Efficacy.	- Add a confirmation message to let the user know that the mentioned contributor will receive a notification and may help the newcomer in this issue.
#4 Upload file	9	Difficulty in understanding the steps needed to upload a file;	- Information Processing Style; - Attitude Towards Risk; - Motivations.	- Change the message to inform that it is necessary to fork; - Make the fork button green to highlight that it is enabled.
	10	Lack of feedback indicating if the action of forking the repository is completed;	- Information Processing Style; - Computer Self-Efficacy.	- Add a success message to the page that appears after the click on the fork button.
	11	Difficulty in understanding the commit form;	- Learning: Process vs. Tinkering; - Motivations.	- Include tooltips explaining the form fields to make the newcomer understand the importance of informing a commit message.
	12	Lack of feedback indicating if the action of uploading the file is completed;	- Information Processing Style; - Learning: Process vs. Tinkering;	- Add a success message to the repository page that appears after the click on the commits changes button.

help me using the GitHub web interface; and (v) ... submit a new file to a project using GitHub.

In addition to the quantitative analysis, we qualitatively analyzed participants' comments to the open questions of the survey following open coding procedures [60]. We asked participants after each use case to explain any difficulties they experienced in accomplishing the task and what in the interface helped them. Our goals were to understand (i) students' difficulties in using the original and the modified interfaces; and (ii) what in the interfaces helped students the most to complete each use case. The analysis was performed by two authors and validated by a third author. The analysis took around one month.

For our study, we considered the following variables: (i) the **dependent variables** comprise the successful completion of each use case by the participants (Y/N), and (ii) the **independent variables** are the use of the Plugin and the GenderMag facets (whether the participant is Tim- or Abi-like).

IV. RESULTS

A. Discovering and Fixing inclusivity bugs on GitHub

We answer RQ1 based on the results of the GenderMag evaluation of the GitHub interface, which uncovered 12 inclusivity bugs. Table III summarizes the inclusivity bugs, associated GenderMag facets, and how we fixed them. The fixes leveraged the design principles of visibility and feedback, along with the tenet of clarity of instructions and reduction of information load where appropriate. The specific UI design changes were inspired by successful fixes to inclusivity bugs as compiled in the GenderMag design catalog[4]. The parts of the GitHub interface where these bugs were found can be accessed in the supplementary material[5].

In UC#1 - Submit pull request, we investigated the GitHub interface that an average user interacts with to edit a file and open a pull request. This use case involved five inclusivity bugs. Among the reported bugs, we found Abi would have difficulty understanding the workflow (what to do next) after the file was edited (Bug #4). Abis are comprehensive information

[4]https://gendermag.org/dc/

[5]https://figshare.com/s/4e7724bde0b1d47ecaeb

processors and process-oriented learners; in this interface, they would not have all the information needed to complete the task and are unlikely to tinker to figure out how to complete it. To address this bug, we proposed: (1) A progress bar indicating the steps of the workflow (improved feedback), allowing Abis to know upfront the process needed to complete the use case; and (2) a tooltip (improved visibility) to explain what happens when the file is edited to provide additional information if Abis need it (Figure 3). Instructions as a tooltip reduce clutter and do not disadvantage tinkerers like Tim.

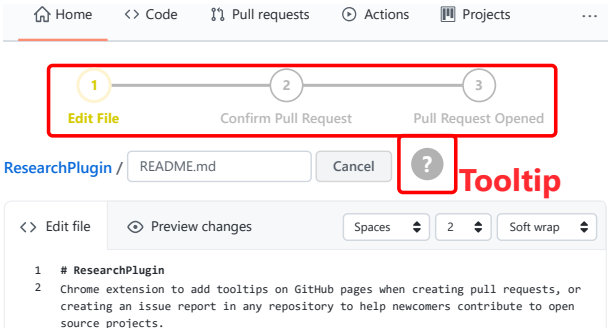

Figure 3. UC#1 / Bugfix #2 - plugin interface: inclusion of progress bar and tooltip.

UC#2, View changed files, included one inclusivity bug (Bug #6), where Abi has difficulty understanding what to do next after opening the pull request. In GitHub, after a user opens a pull request, they are directed to a different page, which does not inform what can be done on that page. On reaching this page, Abis, who are process-oriented learners with lower self-efficacy, would be lost, not knowing what to do next. They would not know if they were progressing towards their goal and would be unlikely to tinker around to figure out how to close the pull request. Our solution adds a tooltip to the navbar that describes some actions that can be made on the pull request page (improved visibility) (Figure 4).

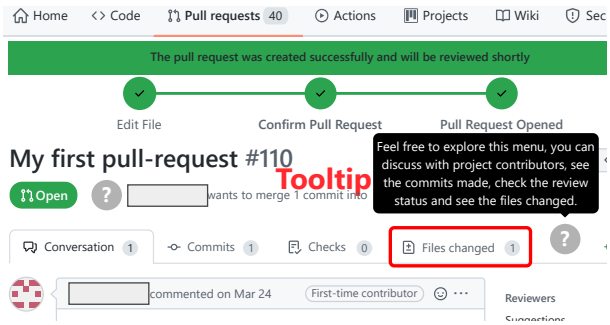

Figure 4. UC#2 / Bugfix #6 - plugin interface: inclusion of tooltip to guide users.

In UC#3 - Request help to solve the PR, we found 2 inclusivity bugs that could affect users' performance with Abi's cognitive style. The pull request interface is not straightforward. Once the user opens the pull request, it is not clear that it is possible to mention someone in the comment box to ask for help. This lack of information affects users with Abi's facets of learning by process and computer self-efficacy. To address this bug, we included a tooltip in the @ symbol icon to display "*Use @ to mention a contributor to help*," as illustrated in Figure 5.

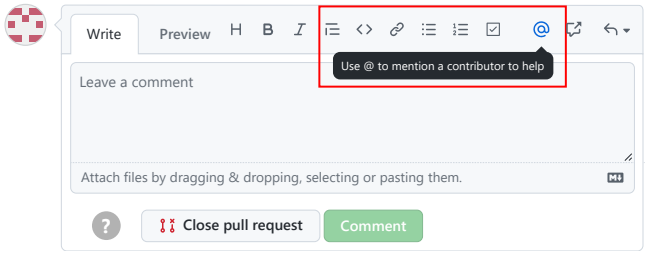

Figure 5. UC#3 / Bugfix #7 - plugin interface: inclusion of tooltip to guide users.

Moreover, after the mention is made, the GitHub interface does not give any feedback about what happens next, affecting comprehensive information processors such as Abi. This can impair Abis' ability to continue with the pull request given their lower self-efficacy, where such users are likely to blame themselves and quit. Even if they asked for help, they would be unsure if the mentioned developer would receive a notification to help them. To fix this bug, we proposed the addition of a confirmation message (improved feedback) to the top of the page informing that: *The mentioned user will receive a notification and may help you to work on the pull request*, as illustrated in Figure 6.

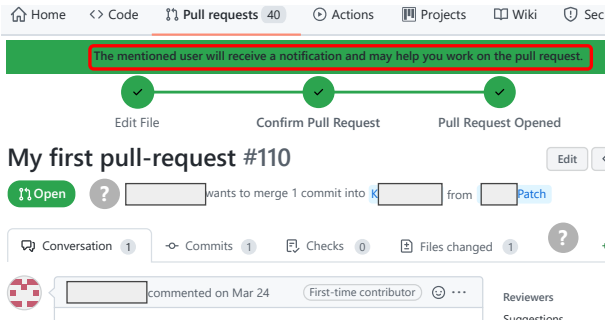

Figure 6. UC#3 / Bugfix #8 - plugin interface: inclusion of confirmation message to provide feedback to users.

In UC#4 - Upload a file, to upload an image to an OSS project, the user needs to have push access to it. For this use case, we found 4 inclusivity bugs. The major bug is related to the second subgoal: it is not possible to upload a file because the newcomer does not have a repository fork nor push access to the original repository. The interface only presents the message that the user needs to have push access to the repository but no direction about how to do it. This bug impacts Abi's facets of comprehensive information processing style, risk averseness, and task-oriented motivations. We proposed the following fixes to address this bug: we changed the message to give better feedback informing the user that it is necessary to fork the repository and made the fork button green to highlight that it is enabled on the page. The new message states *In order to upload files, click the fork button in the upper right* (see Figure 7).

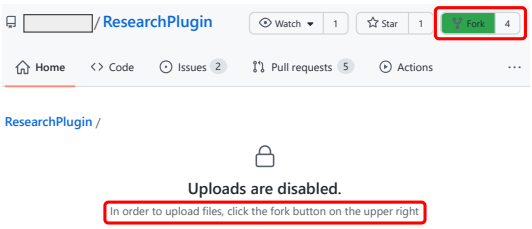

Figure 7. UC#4 / Bugfix #9 - plugin interface: change of message and color of the fork button.

Research Question 1

What inclusivity bugs does GitHub pose for newcomers trying to make their first contribution?

Answer: We found 12 inclusivity bugs after applying the GenderMag inspection method in four use cases a newcomer may perform. These bugs are generally correlated with two or more GenderMag facets. We used the principles of improving visibility, feedback, and reducing information overload in designing the fixes.

B. *Effects of removing GitHub inclusivity bugs*

Impact on completion rates. In RQ2, we investigate how our redesign in Step 1 impacted Abis and Tims. Table IV presents the number of participants who correctly completed the tasks, comparing the treatment groups and the different persona facets. We evaluated the effectiveness of both groups in completing the tasks using the *Chi-Square test* to check the independent relationship between the two categorical variables [61] (see Table V).

For UC#1, there are no statistical differences between Abis and Tims between the treatment groups (Control vs. Plugin). All participants in both treatments had high success rates. Tims had 100% completion rates in both treatments. Abis in the Plugin group performed better than the Control group (94.7% vs. 83.3%), but the difference is not statistically significant. This reflects that UC#1 was a simple enough use case, with the majority of Abis able to overcome the inclusivity bugs (Bug #1 to Bug #5) to complete the task. Recall, inclusivity bugs need not be show stoppers, but they add an additional cognitive tax every time a user faces them.

For UC#2 and UC#3 in the Control group, Abis performed worse than Tims by about 33%, with the difference being statistically significant (p-value < 0.05). However, there is no difference when we compare the Abis and Tims in the Plugin group. Both Abis and Tims have a 100% completion rate for UC#2 and 95% for UC#3. This suggests that our redesign helped Abis overcome barriers to completing these tasks.

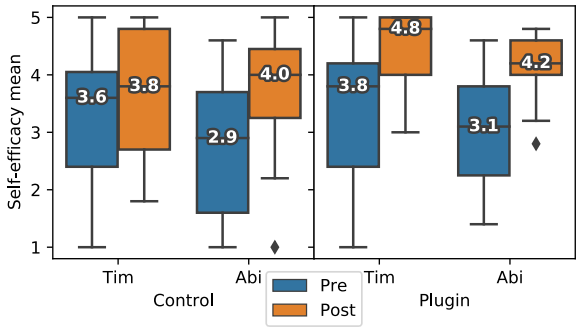

Figure 8. Self-efficacy results.

All participants struggled to complete UC#4 in the Control group; Abis' completion rate was 27.7% as compared to Tims' 33.3%. The redesign helped both Abis and Tims, with Abis' improvements at 61.7% and Tims' at 66.6%. The improvements in the Plugin group compared with the Control group were statistically significant (p-value < 0.001). This result highlights that designing an interface to improve the experience of one underserved population can help make the software better for the larger population.

Impact on self-efficacy. Figure 8 presents the results of the self-efficacy questionnaire that participants filled out at the beginning ('pre') and end ('post') of the study, disaggregated by treatment groups ('Control' and 'Plugin') and per persona ('Tim' and 'Abi').

At the beginning of the experiment ('pre'), Abis (2.9) had a lower self-efficacy as compared the Tims (3.6). After performing the experiment tasks ('post'), both types of participants gained confidence. Given these participants had never interacted with GitHub to submit a pull request before, it is expected that they were not confident in completing the tasks at the start of the experiment. But, after completing use cases (UC#1 to UC#3), their self-efficacy improved. It is heartening to note that the failure to complete UC#4, the last use case, did not dampen those participants' starting self-efficacy.

The improvement in participants' self-efficacy was larger ('pre' vs. 'post') in the Plugin group for both Abis and Tims. We calculated the Wilcoxon signed-rank test, a frequently used nonparametric test for paired data (e.g., pre- and post-treatment measurements) [62], which indicates that the difference in improvement ('pre' vs. 'post') between the Control and Plugin groups is significant for both types of participants; improve-

Table IV
NUMBER OF TASKS COMPLETED OR FAILED BY PARTICIPANTS

	UC#1		UC#2		UC#3		UC#4		All Use Cases	
	Completed	Failed	Completed	Failed	Completed	Failed	Completed	Failed	Completed	Failed
Control	33	3	34	2	28	8	11	25	106	38
Plugin	38	1	39	0	37	2	37	2	151	5
Abi - Control	15	3	17	1	11	7	5	13	48	24
Tim - Control	18	0	17	1	17	1	6	12	58	14
Abi - Plugin	18	1	19	0	18	1	17	2	72	4
Tim - Plugin	20	0	20	0	19	1	20	0	79	1

Table V
EFFECTIVENESS OF TASKS COMPLETED AND COMPARISON AMONG GROUPS.

UC	Abi		Tim		Differences			
	Control	Plugin	Control	Plugin	Abi-C x Tim-C	Abi-P x Tim-P	Abi-P x Abi-C	Tim-P x Tim-C
#1	83.3%	94.7%	100%	100%	-	-	-	-
#2	61.1%	100%	94.4%	100%	⇓ -33.3% *	-	-	-
#3	61.1%	94.7%	94.4%	95%	⇓ -33.3% *	-	⇑ +36.6% *	-
#4	27.7%	89.4%	33.3%	100%	-	-	⇑ +61.7% **	⇑ +66.6% **

(* p≤0.05; ** p≤0.01)

ment for Abis has p-value $= 0.005$, and Tims has p-value < 0.001. We calculated Cliff's delta effect size measure [63] to calculate the magnitude of these differences among the Plugin group. The effect size of improvement for Tims 'pre' vs. 'post' is large (delta = 0.682), as well as for Abis 'pre' vs. 'post' improvements (delta = 0.542).

Impact of the proposed interface on participant experiences. The questionnaire that participants filled out after every task (Section III-C) confirms that the control group participants faced more challenges. In the following, we discuss participants' reflections on what their difficulties when performing the experiment tasks and how the Plugin design helped them.

In UC#1, the main challenge Tims faced in the Control group was the difficulty of finding the editor and the README files (Bug #1 of Table III). P44 mentioned "*Finding the README file, definitely, because I didn't know where to look for all these files, I didn't think it would be like in the middle of those files.*" The plugin solved this problem by presenting the README file information to users more explicitly (improving visibility) through a new tab called home. None of the participants in the Plugin group mentioned finding the README to be a problem.

The Abis in the Plugin group mentioned that the improved visibility of features in the redesigned interface (button colors, tooltips) helped them complete UC#1. The tooltips allowed comprehensive information processors to gather the necessary information before starting the task. It also improved their self-efficacy by letting participants know they were on the right path. Indeed, P29 mentioned that "*the tooltip guides me into the execution of the task*".

The redesigned interface, however, did not help Abi-like participants in figuring out how to edit the file and save it (Bug #4). P1 said, "*Starting the Edit process was really hard. And once you have a little computer knowledge and you actually get into the Edit tab, you can look at the various files you want to edit and then go through the process*". This comment highlights Abi's risk-averseness when having to use new features.

In UC#2, a difficulty that both Abi- and Tim-like participants faced in the Control group was finding the changed files in the pull request interface (Bug #6). Tims and Abis in the Plugin group mentioned that the changed files in the navigation menu (improved feedback) and the tooltips (improved visibility) helped them to complete the task. This is an example of how a solution designed to help one class of users (Abis) helps a broader population (also Tims).

The main difficulty in UC#3 was finding out how to request help. Some participants reported that their first idea was directly contacting the experienced user. P43 mentioned: "*I thought there would be a way that I could just like leave them a personal message and ask for help rather than posting. It [comment in the interface] looks like a public comment.*" Other participants tried to contact the user directly by going to their GitHub profile page and looking for a direct message option, which GitHub does not offer. A majority did not realize they could use the '@' button in the panel to direct their comments to a specific contributor.

Participants in the Plugin group used the tooltip associated with the mention icon ('@') to figure out this feature. The improved visibility of the '@' icon helps process-oriented learners, who would be hesitant to tinker around the interface to find and use the '@' button. With this fix, an Abi participant mentioned that the task was intuitive (P40): "*Once I recognized that I needed to do this task as well, it was pretty intuitive.*".

In UC#4, participants in the Control group faced more difficulty figuring out how to obtain push access: one Abi participant and ten Tims mentioned having that difficulty. Only three of these ten participants overcame this challenge and successfully completed the task. None of the participants mentioned this challenge in the Plugin group. Abis in the Plugin group mentioned that the improved visibility afforded by the green fork button and the feedback message was helpful. P26 said: "*interface messages when trying to upload the file helps a lot*". Tims in the Plugin group said the same, exemplified by P5: "*So when I went back, I saw that the fork was highlighted in like the same green color. (...) It really just puts me back in the right direction*".

Research Question 2

What are the effects of fixing those inclusivity bugs?

Answer: A GenderMag-inspired redesign of the GitHub interface removed the task completion gaps between Abi and Tim participants for UC#2 and UC#3. For UC#4, the redesign significantly improved task completion for both Abi and Tim participants.

V. DISCUSSION

A decade of research on gender HCI has found that individual differences in how people problem solve—how they think when interacting with a software—cluster by gender. Past research has shown that current software and documentation embed inclusivity bugs—bugs that disproportionately affect a subset of users whose cognitive styles are unsupported by the software. These inclusivity bugs result in an additional cognitive tax every time a user faces the bug, which can add up to create barriers to participation.

In our study, we investigated to what extent the GitHub interface embedded inclusivity bugs and how these inclusivity bugs impacted users' performance with different cognitive styles (i.e., Abis and Tims). After applying the GenderMag method on GitHub, we found 12 inclusivity bugs that affect the Abi persona.

Alignment with past research. Our findings are similar to that of past GenderMag research identifying inclusivity bugs in OSS projects. Padala et al. [9] found Information Processing, Self-efficacy, and Learning Style facets favored by Abi to be the most frequent facets that were not supported by OSS projects, and the lack of support of these facets was instrumental in causing the top reported barriers to contribute. More specifically, they found that: (1) comprehensive information processors would feel disoriented because of insufficient upfront information provided in the project README. In our study, Abi-like participants also reported feeling lost in the Control group; (2) participants with lower computer self-efficacy were worried about completing the task and described a lack of knowledge of the technologies as a reason for it. These findings also appear in our results—participants in the Control group felt scared by the GitHub interface; (3) process-oriented learners were hampered by a lack of clear instructions on how to contribute. We observed that Abi-like participants in the Control group also got stuck completing some of the tasks because of a lack of instructions on how to use many of the GitHub features.

Fixing these inclusivity bugs not only helps Abi-like users, whose facets were used to redesign the software but can also make the software better for the larger population. Vorvoreanu et al. [44] in their work found that a redesign of their software to fix the inclusivity bugs found via GenderMag helped women do better (who had twice the failure rate as men in 'pre-fix' version), removing the gender gap in the 'post-fix' version. Moreover, both men and women participants had fewer failures. We found similar results, where redesigning the GitHub interface to accommodate Abi-like users also helped the Tim-like participants in our study (66.67% improvement among Tims in UC#4).

Cognitive diversity bugs can become gender-bias bugs. Past research using GenderMag has shown that the inclusivity bugs created when Abis' cognitive styles are unsupported also become gender-bias bugs because individual differences in how people problem solve cluster by gender [51], [46], [18], [38]. In our data set, we see that the distribution of Tim facets aligned more closely with the distribution of men. We had 63% men who had a majority of Tim facets compared to those who had a majority of Abi facets. In our study, perhaps due to the small sample size and our recruitment pool, we had an equal distribution of Abis and Tims among the women participants.

The need to make OSS tools and technology inclusive. OSS has a severe gender diversity imbalance, with the percentage of women ranging around 10%. One of the challenges women face is a lack of sense of belonging, which may make them less inclined to share their opinions with the rest of the team. We noticed such reticence among our women participants in opining about their difficulties. In contrast, the men (the majority comprised of Tim) in the study felt more empowered to talk about the challenges they faced and suggest how they would improve the GitHub interface. One reason for this difference in behavior can be because women tend to have lower computer self-efficacy than men within their peer sets [9]. This can affect their behavior with technology [64], [46], [19], [65], indicating that women feel less comfortable sharing their opinions and are inclined to think that it is their fault for not being able to use a certain technology. By making the OSS tools and technology more inclusive, we can break the barriers that Abi-like users, typically women, face when using the tools and technology, which can add to their feelings of not belonging [66] and impostor syndrome [11].

Making GitHub inclusive of varied cognitive skills is important for OSS to attract newcomers. Making GitHub inclusive will remove additional barriers newcomers face when their cognitive styles are not supported by the tool [9]. When the gap between newcomers' skills and those needed to accomplish the task is too broad, it demotivates newcomers, causing them to drop out [67], [68]. This can particularly impact students who are still developing their skills and have limited time and experience when first contributing to an OSS project.

VI. Implications

Implications for social coding platforms. For the designers and developers of GitHub and other social coding platforms, our results highlight the importance of developing software that encompasses the diversity of users. Social coding platforms can insert inclusivity biases that are crosscutting to a large number of projects. Social coding platform designers should consider newcomers' cognitive styles to understand how they process information or use the technology itself and how they can accomplish tasks to help them reach their main goals. A more inclusive design means including more users by making it easier for them to contribute to OSS projects.

Implications for Maintainers of OSS projects. Our work reports inclusivity bugs newcomers can face and what part of a task they can get stuck on. Maintainers can use this information to consider how they could mitigate these challenges. One suggestion would be to provide more information in the README/Contributing.md files. We also hope our work can foster and ignite the interest in OSS communities to investigate and remove inclusivity bugs in the different tools and technology they use.

Implications for newcomers (Abis and Tims). Our results are important for newcomers. We showed the difficulties they face, where they struggle most, and how the interface can help them. Abis, who notoriously have low self-efficacy, should be aware that the interface was not designed for their cognitive style, and poor performance is a reflection of the tool failing them and not a reflection on their self-worth or capability. Tims should be aware that developers with diverse cognitive styles exist and respect the differences.

Implications for educators. Familiarizing students with the OSS contribution process is becoming more common [69].

Contributing to a real project helps students gain real-life experience and allows them to add this experience to their resume, which aids them in securing jobs. Our results highlight that based on their cognitive styles, some students can face more challenges when interacting with the GitHub platform. Educators should understand those challenges and teach students how to overcome them. They can also explore other ways to facilitate students' learning of the GitHub platform.

VII. Limitations

Our investigation also has threats to validity and limitations. We focused our analysis on finding inclusivity bugs for newcomers based on GenderMag Abi's persona. We followed the guidelines suggested by Hilderbrand et al. [70] and focused on this persona because its facet values tend to be more undersupported in software than the other personas [22], [41]. However, fixing problems from only this persona's perspective could leave non-Abi newcomers less supported. This was a clear trade-off that could impact Tims, for example. However, the results from out experiment that include both Tim and Abi personas, showed that the performance of the Tim participants also improved for some tasks.

Despite our best efforts to recruit women for the experiment, there is a gender imbalance in the sample. At the same time that having more women would be important for the gender balance perspective, we would lose in terms of representativeness of the population of interest. Still, although the number of women is lower than men, we have almost the same amount of Abi (37) and Tim (38) participants. Nevertheless, this paper aims to investigate the cognitive facets, and some men also present facets associated with Abi's persona.

In the GenderMag analysis, we carefully conducted the walkthroughs on GitHub following the procedures described by Burnett et al. [38]. We had different meetings to review the GenderMag analysis and solutions proposed to fix the inclusivity bugs and the members of our research group had previous experience in conducting GenderMag analysis. Another concern is that the GenderMag method only relies on participants' gender, though that is not the case. Vorvoreanu et al. [44] states that the keys to more inclusive software lie not in someone's gender but in the facet values themselves. As this answer makes clear, GenderMag can be used to find and fix inclusiveness issues without ever speaking of gender.

We recruited 75 undergraduate students from diverse STEM majors from 5 different universities in the US and Brazil. Most participants were pursuing Computer Science majors. We acknowledge that the sample is not representative of the population under analysis. But, we decided to not seek for generalization, but to understand the phenomenon in a controlled environment that would generate initial evidence to be further investigate. Therefore, future studies may investigate whether newcomers from different countries or with education levels to compare the results.

Regarding the plugin development and evaluation, we ran tests during the development to assert its usability and correctness. However, the plugin could have different behaviors depending on the browser. To mitigate this threat, we made available a pre-configured computer in case the plugin did not behave as we expected during the experiment.

We collected the time participants spent completing the tasks. However, the high number of participants that did not complete the tasks made it hard to compare the time differences between groups. Future studies with larger samples may help to investigate time differences.

Concerning the qualitative analysis, we are aware that data interpretation can lead to bias. To mitigate subjectivity, we employed two researchers who independently coded the answers and conducted meetings to discuss and resolve conflicts. Still, this qualitative piece was important to collect the feedback from the users during their activity. We chose to provide this more subjective understanding to complement and enrich our results, instead of collecting only objective data.

VIII. Conclusion

Making software products usable to people regardless of their differences has practical importance. If a project's development tools or products fail to achieve inclusiveness, not only does its adoption fall but so does the involvement of underrepresented populations in the teams themselves [71], [10]. In this work, we found 12 inclusivity bugs in the GitHub interface for four tasks that are common for OSS newcomers. These bugs mainly affect users with cognitive styles that are more common to women—defined in the Abi persona [38]. We proposed fixes to the inclusivity bugs, implemented them in a plugin that changed the GitHub interface, and evaluated them through a between-subject experiment with 75 newcomers.

We found that Abi participants in the Control group (regular GitHub) underperformed Tim participants in some use cases, with Abis in the Control group completing only 67% of the tasks. Implementing the fixes based on the GenderMag analysis reduced these differences and improved the performance of Abi participants to 95%, indicating that the redesign improved GitHub's usability and learnability. In one of our use cases, both Tim and Abi participants faced challenges, and the bug fixes implemented in the plugin significantly helped both participants (66% improvement). We also noticed an overall increase in the self-efficacy perception for both Abi- and Tim-like participants in the Plugin group, highlighting how solving inclusivity bugs for minorities can also help the majority population.

In future work, we plan to use our results to continue exploring the inclusivity barriers in tools and infrastructure to improve newcomers' performance and make tools and projects more friendly for those who want to engage in OSS projects.

Acknowledgment

This work is partially supported by the National Science Foundation under grant numbers 1900903, 1901031, 2236198, 2235601, CNPq #313067/2020-1, CNPq/MCTI/FNDCT #408812/2021-4, and MCTIC/CGI/FAPESP #2021/06662-1. We also thank the students for participating in our study and Zachary Spielberger for helping develop the plugin.

REFERENCES

[1] M. Gerosa, I. Wiese, B. Trinkenreich, G. Link, G. Robles, C. Treude, I. Steinmacher, and A. Sarma, "The shifting sands of motivation: Revisiting what drives contributors in open source," in *ICSE 2021*. IEEE, 2021.

[2] G. Von Krogh, S. Haefliger, S. Spaeth, and M. W. Wallin, "Carrots and rainbows: motivation and social practice in open source software development," *MIS Q*, 2012.

[3] C. Jergensen, A. Sarma, and P. Wagstrom, "The onion patch: migration in open source ecosystems," in *19th ESEC/FSE 2011*. ACM, 2011.

[4] Y. Cai and D. Zhu, "Reputation in an open source software community: antecedents and impacts," *Decision Support Systems*, 2016.

[5] D. Riehle, "How open source is changing the software developer's career." *Computer*, 2015.

[6] E. Parra, S. Haiduc, and R. James, "Making a difference: an overview of humanitarian free open source systems," in *ICSE 2016-Companion*. ACM, 2016.

[7] G. J. Greene and B. Fischer, "Cvexplorer: identifying candidate developers by mining and exploring their open source contributions," in *ASE 2016*. ACM, 2016.

[8] I. Steinmacher, T. Conte, M. Gerosa, and D. Redmiles, "Social barriers faced by newcomers placing their first contribution in open source software projects," in *ACM CSCW 2015*, 2015.

[9] 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 TSE*, 2020.

[10] C. Mendez, H. S. Padala, Z. Steine-Hanson, C. Hilderbrand, A. Horvath, C. Hill, L. Simpson, N. Patil, A. Sarma, and M. Burnett, "Open source barriers to entry, revisited: a sociotechnical perspective," in *ICSE 2018*, 2018.

[11] B. Trinkenreich, I. Wiese, A. Sarma, M. Gerosa, and I. Steinmacher, "Women's participation in open source software: a survey of the literature," *ACM TOSEM*, 2022.

[12] S. K. Horwitz and I. B. Horwitz, "The effects of team diversity on team outcomes: a meta-analytic review of team demography," *Journal of Management*, 2007.

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

[14] J. Marlow, L. Dabbish, and J. Herbsleb, "Impression formation in online peer production: activity traces and personal profiles in GitHub," in *ACM CSCW 2013*. ACM, 2013.

[15] L. Singer, F. Figueira Filho, B. Cleary, C. Treude, M.-A. Storey, and K. Schneider, "Mutual assessment in the social programmer ecosystem: an empirical investigation of developer profile aggregators," in *ACM CSCW 2013*, 2013.

[16] D. Ford, A. Harkins, and C. Parnin, "Someone like me: how does peer parity influence participation of women on stack overflow?" in *VL/HCC 2017*. IEEE CS, 2017.

[17] G. Robles, L. A. Reina, J. M. González-Barahona, and S. D. Domínguez, "Women in free/libre/open source software: the situation in the 2010s," in *OSS Conference 2016*. Springer, 2016.

[18] M. Burnett, S. D. Fleming, S. Iqbal, G. Venolia, V. Rajaram, U. Farooq, V. Grigoreanu, and M. Czerwinski, "Gender differences and programming environments: across programming populations," in *ESEM 2010*. ACM, 2010.

[19] A.-M. Cazan, E. Cocoradă, and C. I. Maican, "Computer anxiety and attitudes towards the computer and the internet with romanian high-school and university students," *Computers in Human Behavior*, 2016.

[20] A. Singh, V. Bhadauria, A. Jain, and A. Gurung, "Role of gender, self-efficacy, anxiety and testing formats in learning spreadsheets," *Computers in Human Behavior*, 2013.

[21] A. Chatterjee, M. Guizani, C. Stevens, J. Emard, M. E. May, M. Burnett, I. Ahmed, and A. Sarma, "Aid: an automated detector for gender-inclusivity bugs in OSS project pages," in *ICSE 2021*. IEEE, 2021.

[22] M. Guizani, I. Steinmacher, J. Emard, A. Fallatah, M. Burnett, and A. Sarma, "How to debug inclusivity bugs? a debugging process with information architecture," in *ICSE-SEIS 2022*, 2022.

[23] I. Santos, I. Wiese, I. Steinmacher, A. Sarma, and M. A. Gerosa, "Hits and misses: Newcomers' ability to identify skills needed for oss tasks," in *2022 SANER*, 2022.

[24] C. Hannebauer and V. Gruhn, "On the relationship between newcomer motivations and contribution barriers in open source projects," in *OpenSym'17*. ACM, 2017.

[25] C. Jensen, S. King, and V. Kuechler, "Joining free/open source software communities: an analysis of newbies' first interactions on project mailing lists," in *44th HICSS*. IEEE, 2011.

[26] I. Steinmacher, A. P. Chaves, T. Conte, and M. Gerosa, "Preliminary empirical identification of barriers faced by newcomers to open source software projects," in *SBES 2014*. IEEE CS, 2014.

[27] A. Bosu and K. Z. Sultana, "Diversity and inclusion in open source software (oss) projects: where do we stand?" in *ESEM 2019*, 2019.

[28] G. Prana, D. Ford, A. Rastogi, D. Lo, R. Purandare, and N. Nagappan, "Including everyone, everywhere: understanding opportunities and challenges of geographic gender-inclusion in oss," *IEEE TSE*, 2021.

[29] 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. Gerosa *et al.*, "The long road ahead: ongoing challenges in contributing to large OSS organizations and what to do," *Proc. ACM Hum.-Comput. Interact.*, 2021.

[30] M.-A. Storey, A. Zagalsky, F. Figueira Filho, L. Singer, and D. M. German, "How social and communication channels shape and challenge a participatory culture in software development," *IEEE TSE*, 2016.

[31] S. Balali, I. Steinmacher, U. Annamalai, A. Sarma, and M. Gerosa, "Newcomers' barriers... is that all? an analysis of mentors' and newcomers' barriers in OSS projects," *Computer Supported Cooperative Work (CSCW)*, 2018.

[32] J. Terrell, A. Kofink, J. Middleton, C. Rainear, E. R. Murphy-Hill, and C. Parnin, "Gender bias in open source: pull request acceptance of women versus men." *PeerJ Prepr.*, 2016.

[33] J. Feller and B. Fitzgerald, "A framework analysis of the open source software development paradigm," in *ICIS 2000*. Association for Information Systems, 2000.

[34] J. B. Miller, *Toward a new psychology of women*. Boston, USA: Beacon Press, 2012.

[35] M. V. Vugt, D. D. Cremer, and D. P. Janssen, "Gender differences in cooperation and competition: the male-warrior hypothesis," *Psychological Science*, 2007.

[36] D. Nafus, "'patches don't have gender': what is not open in open source software," *New Media & Society*, 2012.

[37] S. Turkle, *The second self: computers and the human spirit*. Mit Press, 2005.

[38] M. Burnett, S. Stumpf, J. Macbeth, S. Makri, L. Beckwith, I. Kwan, A. Peters, and W. Jernigan, "Gendermag: A method for evaluating software's gender inclusiveness," *Interacting with Computers*, 2016.

[39] F. Fagerholm, M. Felderer, D. Fucci, M. Unterkalmsteiner, B. Marculescu, M. Martini, L. G. W. Tengberg, R. Feldt, B. Lehtelä, B. Nagyváradi *et al.*, "Cognition in software engineering: A taxonomy and survey of a half-century of research," *ACM Computing Surveys*, 2022.

[40] C. Mendez, A. Sarma, and M. Burnett, "Gender in open source software: what the tools tell," in *1st Int. Workshop on Gender Equality in Software Engineering*. ACM, 2018.

[41] M. Burnett, A. Peters, C. Hill, and N. Elarief, "Finding gender-inclusiveness software issues with gendermag: a field investigation," in *ACM CHI Conference*, 2016.

[42] S. J. Cunningham, A. Hinze, and D. M. Nichols, "Supporting gender-neutral digital library creation: a case study using the gendermag toolkit," in *Int. Conf. on Asian Digital Libraries*. Springer, 2016.

[43] A. Shekhar and N. Marsden, "Cognitive walkthrough of a learning management system with gendered personas," in *4th Conf. on Gender & IT*. ACM, 2018.

[44] M. Vorvoreanu, L. Zhang, Y.-H. Huang, C. Hilderbrand, Z. Steine-Hanson, and M. Burnett, "From gender biases to gender-inclusive design: an empirical investigation," in *ACM CHI Conference*, 2019.

[45] J. Margolis and A. Fisher, *Unlocking the clubhouse: women in computing*. MIT Press, 2002.

[46] M. M. Burnett, L. Beckwith, S. Wiedenbeck, S. D. Fleming, J. Cao, T. H. Park, V. Grigoreanu, and K. Rector, "Gender pluralism in problem-solving software," *Interacting with Computers*, 2011.

[47] R. Riedl, M. Hubert, and P. Kenning, "Are there neural gender differences in online trust? an fmri study on the perceived trustworthiness of ebay offers," *MIS Q*, 2010.

[48] J. Meyers-Levy and B. Loken, "Revisiting gender differences: what we know and what lies ahead," *Journal of Consumer Psychology*, 2015.

[49] T. Dohmen, A. Falk, D. Huffman, U. Sunde, J. Schupp, and G. G. Wagner, "Individual risk attitudes: measurement, determinants, and behavioral consequences," *Journal of the European Economic Association*, 2011.

[50] G. Charness and U. Gneezy, "Strong evidence for gender differences in risk taking," *Journal of Economic Behavior & Organization*, 2012.

[51] L. Beckwith, C. Kissinger, M. Burnett, S. Wiedenbeck, J. Lawrance, A. Blackwell, and C. Cook, "Tinkering and gender in end-user programmers' debugging," in *ACM CHI Conference*, 2006.

[52] J. Cao, K. Rector, T. H. Park, S. D. Fleming, M. Burnett, and S. Wiedenbeck, "A debugging perspective on end-user mashup programming," in *2010 IEEE Symp. on Visual Languages and Human-Centric Computing*, IEEE. IEEE CS, 2010.

[53] A. Chatterjee, L. Letaw, R. Garcia, D. U. Reddy, R. Choudhuri, S. S. Kumar, P. Morreale, A. Sarma, and M. Burnett, "Inclusivity bugs in online courseware: A field study," in *ICER 2022*, 2022.

[54] M. Burnett, S. Stumpf, L. Beckwith, and A. Peters, "The gendermag kit: how to use the gendermag method to find inclusiveness issues through a gender lens," 2018.

[55] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering*. Springer Science & Business Media, 2012.

[56] I. Steinmacher, T. Conte, C. Treude, and M. Gerosa, "Overcoming open source project entry barriers with a portal for newcomers," in *ICSE 2016*, 2016.

[57] A. K. Montoya, "Selecting a within-or between-subject design for mediation: Validity, causality, and statistical power," *Multivariate Behavioral Research*, 2022.

[58] M. M. Hamid, A. Chatterjee, M. Guizani, A. Anderson, F. Moussaoui, S. Yang, I. Escobar, A. Sarma, and M. Burnett, "How to measure diversity actionably in technology." *Equity, Diversity, and Inclusion in Software Engineering: Best Practices and Insights*, 2023.

[59] A. Bandura, "Social cognitive theory of moral thought and action," in *Handbook of Moral Behavior and Development*. Psychology press, 2014.

[60] A. Strauss and J. Corbin, *Basics of qualitative research techniques*. Sage Publications, 1998.

[61] O. Sureiman, "Conceptual model on application of chi-square test in education and social sciences," *Educational Research and Reviews*, vol. 8, no. 15, 2013.

[62] B. Rosner, R. J. Glynn, and M.-L. T. Lee, "The wilcoxon signed rank test for paired comparisons of clustered data," *Biometrics*, 2006.

[63] N. Cliff, "Dominance statistics: ordinal analyses to answer ordinal questions." *Psychological Bulletin*, 1993.

[64] Z. Wang, Y. Wang, and D. Redmiles, "Competence-confidence gap: a threat to female developers' contribution on GitHub," in *ICSE-SEIS 2018*. IEEE, 2018.

[65] A. H. Huffman, J. Whetten, and W. H. Huffman, "Using technology in higher education: the influence of gender roles on technology self-efficacy," *Computers in Human Behavior*, 2013.

[66] B. Trinkenreich, K.-J. Stol, A. Sarma, D. German, M. Gerosa, and I. Steinmacher, "Do i belong? modeling sense of virtual community among linux kernel contributors," in *International Conference on Software Engineering (ICSE 2023)*. IEEE, 2023.

[67] S. Balali, U. Annamalai, H. S. Padala, B. Trinkenreich, M. A. Gerosa, I. Steinmacher, and A. Sarma, "Recommending tasks to newcomers in OSS projects: how do mentors handle it?" in *OpenSym'20*. ACM, 2020.

[68] I. Steinmacher, T. Conte, and M. Gerosa, "Understanding and supporting the choice of an appropriate task to start with in open source software communities," in *HICSS 2015*. IEEE CS, 2015.

[69] G. H. L. Pinto, F. F. Filho, I. Steinmacher, and M. Gerosa, "Training software engineers using open-source software: The professors' perspective," in *CSEE&T*. IEEE, 2017.

[70] C. Hilderbrand, C. Perdriau, L. Letaw, J. Emard, Z. Steine-Hanson, M. Burnett, and A. Sarma, "Engineering gender-inclusivity into software: ten teams' tales from the trenches," in *ICSE 2020*. ACM, 2020.

[71] D. Ford, J. Smith, P. J. Guo, and C. Parnin, "Paradise unplugged: identifying barriers for female participation on stack overflow," in *24th ESEC/FSE 2016*, 2016.