How to Debug Inclusivity Bugs? An Empirical Investigation of Finding-to-Fixing with Information Architecture

Mariam Guizani
guizanimm@oregonstate.edu
Oregon State University
Oregon, USA

Igor Steinmacher
igorsteinmacher@gmail.com
Northern Arizona University
Arizona, USA

Jillian Emard
emardj@oregonstate.edu
Oregon State University
Oregon, USA

Abrar Fallatah
fallataa@oregonstate.edu
Oregon State University
Oregon, USA

Margaret Burnett
burnett@engr.orst.edu
Oregon State University
Oregon, USA

Anita Sarma
sarmaa@oregonstate.edu
Oregon State University
Oregon, USA

ABSTRACT

Background: Although some previous research has found ways to find inclusivity bugs (biases in software that introduce inequities among cognitively diverse individuals), little attention has been paid to how to go about fixing such bugs. We hypothesized that Information Architecture (IA)—the way information is organized, structured and labeled—may provide the missing link from finding inclusivity bugs in information-intensive technology to fixing them.

Aims: To investigate whether Information Architecture provides an effective way to remove inclusivity bugs from technology, we created Why/Where/Fix, an inclusivity debugging paradigm that adds inclusivity fault localization via IA.

Method: We conducted a qualitative empirical investigation in three stages. (Stage 1): An Open Source (OSS) team used the Why (which cognitive styles) and Where (which IA) parts to guide their understanding of inclusivity bugs in their OSS project’s infrastructure. (Stage 2): The OSS team used the outcomes of Stage One to produce IA-based fixes (Fix) to the inclusivity bugs they had found. (Stage 3): We brought OSS newcomers into the lab to see whether and how the IA-based fixes had improved equity and inclusion across cognitively diverse OSS newcomers.

Results: Information Architecture was a source of numerous inclusivity bugs. The OSS team’s use of IA to fix these bugs reduced the number of inclusivity bugs participants experienced by 90%.

Conclusions: These results provide encouraging evidence that using IA through Why/Where/Fix can help technologists to address inclusivity bugs in information-intensive technologies such as OSS project infrastructures.

KEYWORDS

Diversity, Information Architecture, Open Source Software

1 INTRODUCTION

Although in recent times diversity initiatives have become common, sometimes we forget why diversity is important to so many organizations. Besides social justice reasons, what many organizations hope to gain from diverse backgrounds (cultural, ethnic, education, gender, etc.) is diversity of information and of thought [42]—i.e., cognitive diversity. Diversity’s accompanying diversity of thought has been shown to have many positive effects to organizations, including better ability to innovate, better reputation as ethical corporate citizens, and a better “bottom line” for businesses [29, 40, 42]. However, efforts to support diversity rarely consider either cognitive diversity or inclusivity of technology environments.

In this paper, we consider these aspects together: how to increase support for cognitive diversity within technology environments, especially information-heavy ones. The setting for our investigation is an information-heavy environment that is particularly challenged in attracting diverse populations: Open Source Software (OSS) communities. Prior research has investigated inclusivity issues affecting OSS [5, 12, 26, 32, 36, 45, 47, 58], but has not focused on how to debug OSS projects’ technology/infrastructure to improve support for cognitive diversity.

A debugging perspective suggests that OSS practitioners who want to improve inclusivity of their project’s infrastructure will need three capabilities. (1) First, they need to find “inclusivity failures” (analogous to testing [1]). Since the failure is about inclusivity (not about producing a wrong output), OSS practitioners will also need to be able to discern why the observed phenomenon is considered an inclusivity failure. (2) Second, the practitioners will need to tie an inclusivity failure to where the “inclusivity fault(s)” occur (analogous to fault localization [3]); so that (3) the inclusivity faults can be fixed to stop the associated inclusivity failure from occurring. In this paper, we term this set of inclusivity debugging capabilities as “Why/Where/Fix,” and investigate the efficacy of supporting it, especially the “Where” capability.

Debugging requires a definition of a bug, and we derive our definition from the testing community’s notion of a software failure. According to Ammann and Offutt “Failure is defined as external, incorrect behavior with respect to the requirements or [...] expected behavior” [1]. Building upon this definition, our requirement is inclusivity across diverse cognitive styles, so we define inclusivity failures/bugs as user-visible features or system workflows that do not equitably support users with a diversity of cognitive styles. As with Ammann/Offutt’s definition, an inclusivity bug is a barrier but not necessarily a “show-stopper”. That is, if groups of users eventually complete their tasks but disproportionately experience barriers along the way (e.g., confusion, missteps, workarounds), these too are inclusivity bugs.

Regarding finding such inclusivity bugs and the “Why” of them, we leverage GenderMag [11], a validated inspection method [39, 60] with a dual gender/cognitive focus. GenderMag integrates finding an inclusivity bug with its “Why”, because using GenderMag includes
identifying cognitive mismatches that pinpoint which users disproportionately run into barriers using a system. In this paper’s investigation, we worked with an OSS team who drew upon GenderMag to detect inclusivity bugs in their project’s technology infrastructure.

After finding a bug, the next step in debugging is to figure out what and where a bug’s causes are, referred to as “faults” in SE literature. According to Avizienis et al. [3] a fault is the underlying cause of an error, a condition that may lead to a failure; and fault localization is the act of identifying the locations of faults. Building upon these definitions, we define an inclusivity fault as the user-facing components (e.g., UI elements, user-facing documentation, workflow) of the system that produced an inclusivity bug; and inclusivity fault localization as the process of identifying the locations of these faults in these user-facing components.

For Why/Where/Fix’s “Where”, we devised an inclusivity fault localization approach based on Information Architecture (IA) [35]. A project’s IA is its “blueprint” for the structure, arrangement, labeling, and search affordances of its information content, and is especially pertinent in information-rich environments [51]. Although substantial research exists on how Information Architectures can support usability, navigation, and understandability [18, 21, 27, 33, 48], research has not considered how different Information Architectures do or do not support populations with diverse cognitive styles, or how IA can be used for inclusivity fault localization.

To use IA to tie together the above “Why” and “Where” foundations to point to the fixes, we supplemented the GenderMag process for finding inclusivity bugs with a mechanism by which evaluators specified any IA elements (the faults) implicated in the inclusivity bugs found along the way. Thus, the Why/Where/Fix process is: find the bugs using cognitive styles, which contribute the Why (using GenderMag), enumerate the implicated IA elements involved in the bug (Where), and change those IA elements (Fix).

Our empirical investigation of IA’s effectiveness in such a debugging process took place in three stages. In Stage One (Why → Where), we worked with an OSS team who used GenderMag to detect cognitive inclusivity bugs in their project’s infrastructure, to investigate RQ1: Is IA implicated in inclusivity bugs? If so, how? In Stage Two (Where → Fix), the OSS team changed the project infrastructure’s IA using what they had learned in Stage One, which enabled us to investigate RQ2: Can practitioners use IA to fix inclusivity bugs? If so, how? In Stage Three (Lab Participants), we brought OSS newcomers into the lab to investigate whether the inclusivity bugs the team found in RQ1 actually arose with the OSS newcomers; and whether the team’s IA changes from RQ2 aiming to fix the inclusivity bugs actually decreased the inclusivity bugs those newcomers experienced.

The primary contributions of this paper are:

1. The first work to empirically investigate an inclusivity debugging paradigm (Why/Where/Fix) with a fault localization component.
2. The first work to empirically investigate whether Information Architecture can itself be the cause of inclusivity bugs.
3. The first work to investigate ways OSS projects can change their infrastructures’ Information Architecture to fix inclusivity bugs.

2 BACKGROUND AND RELATED WORK

2.1 Information Architecture

The term “Information Architecture” was first coined in the mid-70’s as a way of “making the complex clear” [61]. This paper follows the definition of Morville and Rosenfeld [35], often referred to as the “bible” of IA. That work defines IA as a set of four component systems (Figure 1).

The first is the Organization System (Org), analogous to the architectural arrangement of a building’s “rooms”, has an organization scheme OrgScheme and an organization structure OrgStruct. The organization scheme is the way content is arranged or grouped (e.g., alphabetical, by task, by topic, etc.) An architect chooses the scheme according to the situations they want the Information Architecture to support, such as alphabetical (OrgScheme-Alpha) to support exact look-ups, or task-based (OrgScheme-Task) to facilitate high priority tasks. The organization structure defines the relationship between content groups (e.g., hierarchical (OrgStruct-Hierarchy)).

Second, the Navigation System (Nav), analogous to adding doors and windows to a building, enables users to traverse the information groupings and structure. Some of the navigation system is embedded in the information content (e.g., contextual links (Nav-ContextualLink)), while others are supplemental (e.g., site maps). Third, the Labeling System (Label) adds signposts (also known as “cues” in Information Foraging literature [43]) to the “doors”, such as the labels on contextual links (Label-ContextualLink), headers (Label-Header), cues/keywords (Label-IndexTerm), etc. Fourth, the Search System, when provided, supplements the rest of the Information Architecture, to enable users to retrieve information using a particular term or phrase.

While the majority of IA research has focused on the design and evaluation of websites, some have explored other domains also. For example, IA has been used in the design of usable security tools [14], as the basis of a semantic web structuring tool [7, 8, 16], to investigate the accessibility, use and reuse of information across multiple devices [37], to evaluate different information visualization tools [30] and screen-reader navigation for mobile applications [20, 62].

One body of research has compared IA to other attributes of information sites. For example, Aranyi et al. qualitatively analyzed end users’ verbalization as they evaluated a news website; the actual content and its IA were found to be the main problems [2]. Petri

![Figure 1: IA's four component systems [35]. The organization and navigation systems have subsystems (underlined). *s mark IA (subsystems and elements used in this paper.](image-url)
and Power’s study likewise found prominent IA problems when evaluating six government websites, with IA accounting for about 9% of the bugs both users and experts reported [41].

Other IA research has evaluated the usability of different subsets (organizational vs. labeling schemes) of IA. For example, Gullikson et al. evaluated the IA of an academic website and reported that although participants were satisfied with the content of the site, they found its (IA) labeling to be confusing [22], and were especially dissatisfied with the IA’s organization system. Resnick and Sanchez found that user-centric labels significantly improved user performance and satisfaction as compared to user-centric organization, which only improved performance if labels were of low quality [46]. Similarly, others have found navigation success depends more on the quality of labels than the structure of a page [34, 52].

Of particular interest is IA research on supporting diverse populations. Lachner et al. used IA to promote cultural diversity and used Hofstede et al. power distance cultural dimension [25] to design and evaluate culturally-specific collaborative Q&A websites [28]. Accessibility and IA has been studied by others. Swierenga et al. showed that IA’s organization and labeling system create barriers for visually impaired and low-vision individuals [57]. A multitude of research [4, 17, 49, 50, 59, 62] has investigated IA auditory systems for designing and evaluating accessible websites for low-vision users. Ghahari et al., for example, showed how topic- and list-based aural navigation strategies can enhance user’s navigation effectiveness and efficiency [49]. However, we cannot locate any research on how IA can support cognitive diversity.

2.2 Diversity and the GenderMag Method

GenderMag, a method used to find and fix inclusivity bugs, provides a dual lens—gender- and cognitive-diversity—to evaluate workflows. It considers five dimensions (“facets” in GenderMag) of cognitive styles (Table 1), each backed by extensive foundational research [11, 56]. Each facet has a range of possible values. A few values within each facet’s range are brought to life by the three GenderMag multi-personas: “Abi”, “Pat”, and “Tim.” Abi’s facets are statistically more common among women than other people, Tim’s are statistically more common among men, and Pat has a mix of Abi’s/Tim’s facets plus a few unique ones.

Each persona is a “multi-”persona [24] in that their demographics can be customized to match those of the system’s target audience. For example, any gender can be assigned to any of them, any photo(s) can be inserted, any pronoun can be integrated (e.g., she/her, he/him, they, ze, etc.), any educational background, etc. Note that even when Abi, Pat, and Tim are assigned identical demographics, each represents a cognitively different subset of a system’s target users, because each has a different combination of facet values. Figure 2 shows portions of the OSS team’s customization of Abi, which they used in Stage One.

Evaluation teams, such as the OSS team in this paper, use GenderMag to walk through a use-case in the project they are evaluating using Abi, Pat, or Tim. At each step of the walkthrough, the team writes down the answers to three questions: (1) whether <Persona> would have the subgoal the project owners hoped for and why, (2) whether <Persona> would take the action the project owners hoped for and why, and (3) if <Persona> did take the hoped-for action, would they know they did the right thing and were making progress toward their goal, and why. When the answer to any of these questions is negative, it 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

<table>
<thead>
<tr>
<th>Table 1: The GenderMag cognitive facet values for each persona. The research behind each facet is enumerated in [11].</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Facet</strong></td>
</tr>
<tr>
<td>Motivations</td>
</tr>
<tr>
<td>Self-Efficacy</td>
</tr>
<tr>
<td>Attitude Toward Risk</td>
</tr>
<tr>
<td>Information Processing</td>
</tr>
<tr>
<td>Learning Style</td>
</tr>
</tbody>
</table>

**Abi (Abigail/Abishek)**

*Abi is a second-year engineering student... She is comfortable with the technologies she uses regularly... She is interested in branching out to the world of open source... but their software systems are new to her... She likes Math...*

*Abi’s facets are listed and described here*

**Figure 2:** Portions of the OSS team’s Abi persona. The photo(s) and blue text are customizable; the black text is not. Abi’s facets (gray block) are as per Table 1. (The supplemental document [15] includes the full Abi persona used in Stage One.)
We conducted a qualitative empirical investigation to analyze whether and not the UI of the hosting platform (e.g., GitLab, GitHub). They ultimately selected 6 bugs (last column of Table 2).

Along the way, Team F had noticed some general usability bugs not related to any cognitive facet. To prevent these from influencing the prototype, Team F fixed these bugs and brought the project up to GitHub’s recommended content standards [38], resulting in the prototype we call the Original version.

### Table 2: The four use-cases and associated bugs. Team F provided these use-cases, which were important to their project.

<table>
<thead>
<tr>
<th>Use-Case</th>
<th>Descriptions</th>
<th>Bugs</th>
</tr>
</thead>
<tbody>
<tr>
<td>U1-Find</td>
<td>Finding an issue to work on</td>
<td>Bug 1 &amp; 2</td>
</tr>
<tr>
<td>U2-Document</td>
<td>Contribute to the documentation</td>
<td>Bug 3</td>
</tr>
<tr>
<td>U3-FileIssue</td>
<td>File an issue</td>
<td>Bug 4</td>
</tr>
<tr>
<td>U4-Setup</td>
<td>Set up the environment</td>
<td>Bug 5 &amp; 6</td>
</tr>
</tbody>
</table>

#### 3.2 Stage 2 (Team F, RQ2): Where → Fix

Team F then derived fixes for each of these 6 bugs by changing the IA elements they had identified as the probable causes of the bugs, without loss of support for the supported facets. (Note that nobody on Team F had HCI training, so the only HCI resource they could draw upon in deriving their fixes was what they had learned from their Stage One analysis.) We refer to the “fixed” version of Project F as the DiversityEnhanced version.

#### 3.3 Stage 3: Lab participants (RQ1+RQ2)

We then brought OSS newcomers into the lab to investigate: (1) whether OSS newcomers trying to use the Original version would run into the bugs Team F had found in the Original version, and (2) whether the IA fixes Team F had derived for the DiversityEnhanced version would actually improve support for cognitively diverse OSS newcomers.

We recruited the OSS newcomers from a large US university. Our recruiting criteria were people with no prior experience contributing to OSS projects. All 31 respondents came from a variety of science and engineering majors. Because the investigation focuses only on cognitive diversity (not on disabilities), we did not seek out participants with any particular cognitive style or with a disability. Because none of the experimental tasks required programming, we did not collect their programming experience.

Participants filled out a cognitive facet questionnaire [9, 19, 60] (provided in our supplemental document [15]) in which participants answered to Likert-scale items about their cognitive styles. We used the questionnaire responses to select 18 of these respondents focusing on sampling a wide range of cognitive styles (Figure 3). Of the 18 selected participants, 8 identified as women, 9 identified as men, and one participant declined to specify their gender.

We assigned participants to the Original or DiversityEnhanced treatments, balancing the cognitive styles between the treatments based on the participants’ cognitive facet questionnaire responses. Because facet values are relative to one’s peer group, the median response for each facet served to divide closer-to-Abi facet values from closer-to-Tim facet values. This produced identical facet distributions (Figure 4) for both groups.
We audio-recorded each participant as they talked aloud while working on the use-cases (presented earlier in Table 2). We transcribed the recordings, and counted how often the participants encountered one of the 6 bugs that Team F had attempted to fix.

Also, to enable comparing their in-situ reactions to their cognitive facet questionnaire responses, we used the facets to code what participants verbalized when they encountered these bugs. For example, we coded P2-O’s verbalization “...this leads me to a page with the bare minimum of instructions... I have no idea where to go from here” as “learning style: process-oriented”, which aligned with their questionnaire response. To ensure reliability of the coding, two researchers independently coded 20% of the data and calculated IRR using the Jaccard index. Jaccard, a measure of “consensus” interrater reliability [55], is useful when multiple codes per segment are used, as in our case. The consensus level was 90.2%. Given this level of consensus, the researchers split up coding the remainder of the data.

At the end of the session, participants filled out a subset of the System Usability Scale (SUS) survey [6] (supplemental document [15]).

4 RESULTS

We begin with “whether” answers to both research questions—for RQ1, whether Information Architecture was implicated in the inclusivity bugs, and for RQ2, whether Team F’s IA fixes increased inclusivity for OSS newcomers.

As Table 3 shows, both answers were yes. Regarding RQ1, with the Original version, OSS newcomers ran into inclusivity bugs in the Information Architecture 20 times. Regarding RQ2, Team F’s inclusivity fixes to the IA reduced the number of inclusivity bugs experience in the DiversityEnhanced version to only 2. In total, Team F’s IA fixes cut the number of bugs participants experienced by 90% (Table 3).

Figure 3: Number of participants with more Abi facets (left half, orange) or more Tim (right half, blue). For example: the first column says that 1 participant had 5 Abi facets and no Tim facets. Table 1 explains Abi, Tim, and their facets.

Figure 4: Number of participants with Abi (bottom, orange) vs. Tim (top, blue) facets who used the Original (columns 1-5) vs. DiversityEnhanced (columns 6-10) versions of the OSS project. (The two distributions are identical.)

To answer the how aspects of our RQs, Table 4 summarizes, for each bug, Team F’s Why analyses (first column) of the cognitive facets involved in the bug, their Where analyses to localize the faults to IA elements (second column), and their IA Fixes (third column). The following sections discuss them in depth.

4.1 Bug 1 & 2 in Depth: Issues with the “issue list”

The first two rows in Table 4 show how Team F addressed Bug 1 & 2, the IA-based inclusivity bugs that Team F identified in Stage One in the context of use-case U1-Find (finding a task to work on). As Table 4 shows, for Bug 1, Team-F predicted that Abi-like newcomers would face problems in understanding the process of finding an issue. Team F’s Stage One why analysis (Table 4 row 1 col. 1) pointed out that the lack of information about finding an issue could be problematic to comprehensive information processors, risk averse, or process-oriented newcomers. As Stage Three Participant 1 using the Original version later put it:

P1-O: “I just feel like I wouldn’t have enough to go on.”

Team F localized the fault (where, Table 4’s row 1 col. 2) to the IA’s link labeling (Label-ContextualLink) and to the absence of keywords (Label-IndexTerm), which could lead newcomers to follow wrong link(s) and never obtain the kind of information they were seeking.

Once a newcomer was past Bug 1, Team F predicted that the Issue List provided little information to enable newcomers to select an issue appropriate to their skills (Bug 2). Team F’s why analysis showed that this bug would be particularly pertinent to newcomers with a comprehensive information processing style, low self-efficacy, or risk aversion.

Team F localized the fault behind Bug 2 (IA where) to the issue list’s nondescriptive titles, uninformative descriptions, and limited labeling. Team F realized that, with this IA, the Issue List gave little indication as to whether an issue would fit a newcomer’s skill level (Label-IndexTerm, Label-Header). Stage Three proved Team F to be right: Bug 1 & 2 did affect several participants (Figure 5):

P1-O: “...I don’t really know...I would say if I had to fix [an issue from the issue list], I’d probably just ask someone for help.”

To fix Bug 1 (Table 4 rol 1 col. 3), Team F made several changes to the IA. They created better cues for the link to the contribution guidelines by changing its label (Label-ContextualLink) from the file name (“contributing.md”) to “contributing guidelines” and including additional keywords about what to expect from the link. They also modified the IA of the “contributing.md” to point out specific task-oriented instructions for finding an issue (OrgScheme-Task) including a header (Label-Header)—“Find an issue” (Fig 6), a link to the “issue list” (Nav-ContextualLink, Label-ContextualLink), and

Table 3: The number of participants who ran into the bug(s), out of the 9 participants per group.

<table>
<thead>
<tr>
<th>Bug ID</th>
<th>Original</th>
<th>DiversityEnhanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bug 1 &amp; 2</td>
<td>9/9</td>
<td>1/9</td>
</tr>
<tr>
<td>Bug 3</td>
<td>2/9</td>
<td>0/9</td>
</tr>
<tr>
<td>Bug 4</td>
<td>0/9</td>
<td>0/9</td>
</tr>
<tr>
<td>Bug 5 &amp; 6</td>
<td>9/9</td>
<td>1/9</td>
</tr>
<tr>
<td>Total bugs</td>
<td>20</td>
<td>2</td>
</tr>
</tbody>
</table>
Table 4: For each use-case’s bug(s), excerpts from Team F’s Stage One analysis, the Bug’s Why’s (facets impacted), Where’s (IA involved), and their Stage Two IA fixes.

<table>
<thead>
<tr>
<th>Bug</th>
<th>Bug's Why: Facets</th>
<th>Bug's Where: IA involved</th>
<th>Bug's Fixes and IA elements changed</th>
</tr>
</thead>
</table>
| Bug 1 | “[referring to the issue list] ...would want to read a bit more about issues to be certain of what to do next” Facets: Info, Risk, Learn | “... may click [the wrong link]...” IA: Label-ContextualLink, Label-IndexTerm | - In README.md:  
- Label-IndexTerm: added cue/keyword to guide to “contributing guidelines” for finding an issue.  
- Label-ContextualLink: changed a link label to clarify what it leads to.  
- In contributing.md:  
- Label-ContextualLink, Label-ContextualLink: added a link to the “issue list”.  
- Label-IndexTerm: added cues/keywords to guide issue choice.  
- OrgScheme-Task, Label-Header: added a header following a task-based organization scheme.  
- Other: added more information.  
See Figure 6 |
| Bug 2 | “...just from the titles she is not getting as much info as she wants...not a good enough description, might think of giving up” Facets: Info, SE, Risk | “...labels will help, but there aren’t labels for every issue...like ‘good for newcomer’. Headings are missing info, should be a bit more detailed” IA: Label-IndexTerm, Label-Header | - In the issue list:  
- Label-IndexTerm: added labels to aid issue selection.  
- Label-Header: improved issue headers to be more descriptive.  
- Other: improved issues’ descriptions.  
See Figure 7 |
| Bug 3 | “[The instructions are] all about technical contributions, nothing about documentation changes...” [So] she may think that she needs to do all the technical setup before editing the README (which is a lot)” Facets: Motiv, Learn, SE, Risk | “README and contribute files may confuse her. The README is here but there is no clear indication [cue/keyword] of what she needs to do to change the file.” IA: Label-IndexTerms | - In README.md:  
- Label-IndexTerms: added cue/keyword to guide to “contributing guidelines” for documentation contributions.  
- In contributing.md:  
- Label-IndexTerm: added cues/keywords to guide a documentation contribution.  
- Nav-ContextualLink: linked to additional information.  
- OrgScheme-Task, Label-Header: added a header that followed a task based organization scheme.  
- Other: added more information.  
| Bug 4 | “...nothing clearly says that filing an issue is part of contributions. No clear instruction about what she needs to do...it doesn’t say where to find the issue list” Facets: Info, SE, Risk | “...doesn’t say where to find the issue list...Maybe adding an indication [cue/keyword] or a link would be helpful.” IA: Label-IndexTerm, Label-ContextualLink | - In README.md:  
- Label-IndexTerm: added cue/keyword to “contributing guidelines” for filing an issue.  
- In contributing.md:  
- Label-IndexTerm: added cues/keywords about filing an issue.  
- Nav-ContextualLink, Label-ContextualLink: added link to the “issue list”.  
- OrgScheme-Task: reorganized the section to follow a topic-based organization scheme.  
| Bug 5 | “...nothing that explicitly says set up the env...She would read through step 0 and think it’s not for mac [OS].” Facets: Info, SE, Risk | “...no hint [cue/keyword] about how to set up the environment in the readme... More about Ubuntu and Linux and not about Windows and Mac...maybe this file needs to be more high level.” IA: Label-Index Term, OrgScheme, OrgStruct | - In README.md:  
- Label-IndexTerm: added cue/keyword to “contributing guidelines” for setting up the environment.  
- In contributing.md section “Help us with code”:  
- OrgStruct-Hierarchy: restructured section with an extra layer of abstraction.  
- Nav-ContextualLink, Label-ContextualLink: added links to instructions per OS.  
- OrgScheme-Topic: reorganized the section to follow a topic-based organization scheme.  
See Figure 10 |
| Bug 6 | “...No explanation about the different things to install and where to install them”. Facets: Info, Motiv, Learn, SE, Risk | “sees all this code and does not know where and how to run it. Maybe a hint [cue/keyword] and copying and pasting the code would be helpful.” IA: Label-IndexTerm | - In OS instruction sub-pages:  
- Label-IndexTerm: added cues/keywords about where to execute commands.  
- Other: added additional explanation about each command.  

additional keywords (Label-IndexTerm) to add support for process-oriented and risk-averse newcomers.

Team F fixed Bug 2 (Table 4 row 2 col. 3) with improved issue headers and labels (Label-Header, Label-IndexTerm). The labels signaled attributes of the open issues in the project (Figure 7). Team F also rewrote some of the issue descriptions to support newcomers with a comprehensive information processing style.

In Stage Three, the participants showed that Bug 1 & 2 were pervasive; all participants using the Original version faced problems related to Bug 1 and/or 2 (Figure 5). This raises the question of whether the bugs were inclusivity bugs, i.e., disproportionately affected people with particular cognitive styles.

Figure 5 answers this question. Counting up the colored outlines, which show which facets Stage Three participants verbalized when they ran into those bugs, shows that Bug 1 & 2 disproportionately impacted Abi-like facet values: 74% (14/19) of the facets participants...
Figure 5: In Bug 1 & 2, all Original participants ran into bugs (left), but only 1 DiversityEnhanced participant (right). Participant ID numbering is from the most Abi-like to the most Tim-like.

*: facet the fix(es) targeted; circles | squares: the facet values from the participants facet questionnaire for Abi-like and Tim-like facet values respectively; square outline | square outline: participant ID numbering is from the most Abi-like to the most Tim-like.

Figure 6: Bug 1 before the fix, the screen appeared as shown without the call-out, giving little guidance on how to find a suitable issue. The fix added the “Find an issue” process description.

Figure 7: Top: Bug 2 before the fix had only one label (“Bug”). Bottom: The fix added multiple descriptive labels.

Figure 8: For Bug 3, two participants using the Original version ran into problems, but nobody in the DiversityEnhanced treatment did. *, circles, squares: see Figure 5.
The results of Stage Three showed that the changes had positive effects. As Figure 8 shows, although two participants ran into Bug 3 with the Original version, nobody did using the DiversityEnhanced version.

4.3 Bug 4: Where to go to file an issue

For bug 4 (Table 4’s fourth row), Team F decided that, in trying to file an issue (use-case U3-FileIssue), newcomers might not know where to go, especially those who are risk-averse, those with comprehensive information processing styles or relatively low self-efficacy (Table 4 row 4 col. 1). The elements of IA where the team found these problems were in Nav-ContextualLink, Label-IndexTerm, and Label-ContextualLink elements.

However, Team F was wrong—in Stage Three, none of the Original version lab participants ran into Bug 4. The reason was a flaw in Team F’s analysis of this use-case as it related to newcomers’ prior experience. GenderMag analyses are about learnability of a feature set the user does not already know. However, before filing an issue (U3-FileIssue) users have to first review the issue list to “find” if such issue was already reported, and therefore will already be familiar with the “issue list” features. The Stage Three task sequence reflected this prior learning where participants went to the “issue list” in context of an earlier use-case (Finding an issue to work on, U1-Find), as P5-O said:

P5-O: “Since I already spent some time on that issue page [issue list]. That part [filing an issue] was not too hard.”

Still, Stage Three had not yet occurred, and Team F made the IA fixes in Stage Two to fix the bug. As Table 4 row 4 col. 3 shows, they made improvements to Label-IndexTerm, Nav-ContextualLink, and Label-ContextualLink elements, while maintaining the task-based organization scheme (OrgScheme-Task). Participants in Stage Three who used the DiversityEnhanced version experienced no problems.

Thus, the question of whether newcomers would have run into these problems if they had not previously learned the features remains unanswered. However, the question of whether newcomers ran into problems in the changed version is answered: nobody ran into any problems in the DiversityEnhanced version (Table 3).

4.4 Bug 5 & 6: What, where, and how to set up

In use-case U4-Setup, Team F’s analysis revealed Bug 5 (Table 4’s fifth row), namely that newcomers with comprehensive information processing style, low self-efficacy, or risk aversion could run into problems finding the setup instructions for their particular operating system (OS). Team F identified the underlying faults to be the Label-IndexTerm, OrgScheme and OrgStruct, none of which were pointing out where different OS’s setup instructions might be.

Even if a newcomer overcame Bug 5 and found the (right instructions, Team F realized that an OSS newcomer might not necessarily “just know” what each command in the instructions actually did or exactly where to run them (Bug 6: Table 4’s sixth row). As the table shows, Team F’s why analysis suggested that this inculsivity bug could particularly affect a newcomer with any of ABI’s cognitive style values, due in part to the absence of hints with clarifying keywords (e.g., “command line terminal...”) (Label-IndexTerm).

Stage Three’s results confirmed Team F’s predictions: all Original participants ran into one or both of these bugs (Figure 9). Further emphasizing Team F’s prediction, As with the other bugs described so far, when participants ran into the bugs, they verbalized mostly ABI-like facet values: for Bug 5 & 6, 81%(17/21) were ABI-like facet values (orange square outlines left half Figure 9). For example:

P1-O (low-self-efficacy): “I feel like they [the OSS developers] put up barriers because they would want people that really knew what they were doing...”

P1-O (continues): “I’d probably just, like, not work on it.”

The lab participants also pointed out mismatches to cognitive styles like process-oriented learning, comprehensive information processing, and risk-aversion to using commands they did not completely understand:

P1-O: “These instructions aren’t working super good for me... if there was explanations a little more.”

P3-O: “I don’t completely understand... where to move it [a command] or where to put it.”

To address Bug 5, Team F restructured the “Help us with code” section by adding a layer of hierarchy to structurally identify general information about code contributions (OrgStruct-Hierarchy). They also reorganized the section topically by OS type (OrgScheme-Topic) (Figure 10). Moreover, they added keywords (Label-IndexTerm) in the README.md similar to Bug 3’s fix, to more clearly guide newcomers to the right setup instructions for their OS. To fix Bug 6,
Team F added explanations to each step in the instructions, in which they made explicit the reason for each step and the need to use a command line terminal for the commands (Label-IndexTerm).

Team F’s IA fixes paid off: both Abi-like and Tim-like participants improved and the number of participants who ran into problems decreased from 9 to 1, an 89% improvement (Figure 9). Further, although none of the Original participants completed the task successfully, all participants using the DiversityEnhanced version were able to complete the task—even P14-D, who at first ran into a problem, but overcame it and eventually succeeded.

5 DISCUSSION

5.1 The IA Fixes: Equity and Inclusion

As the results sections have shown, the IA fixes that differentiated the DiversityEnhanced version from the Original version led to a 90% reduction in the bugs that Team F had found to be inclusivity bugs (Section 4’s Table 3). However, this leaves unanswered whether these fixes actually contributed to the goals of making the project’s infrastructure (1) more equitable and (2) more inclusive. For example, equitability could be achieved by helping one group at the expense of another, but that would not achieve inclusivity. Team F’s goal was to do both.

First we consider equity. A dictionary definition of equity is “the quality of being fair and impartial” [44]. We measured equity analyzing the lab participants’ data, because the participants covered an almost equal number of Abi and Tim facets (recall Figure 4: 22 Abi facet values and 23 Tim facet values in each treatment). Thus, if the lab participants’ number of “Abi facets” affected by a bug was greater than the number of “Tim facets”, or vice-versa, we conclude that the bug was inequitable in the ways it affected the participants.

By this measure, we noticed that Bugs 1 & 2 in the Original version were inequitable: together they affected 14 of participants’ Abi facets (orange outlines for Figure 5’s Original version), compared to only 5 Tim facets (black outlines). Applying the same measure to the DiversityEnhanced version shows that, although the DiversityEnhanced version was still slightly inequitable—two of participants’ Abi facet inequities (2 orange outlines), and zero Tim facet inequities—it was less inequitable than the Original version. Applying the same measures to Bug 3 (Figure 8 - Original: 5 Abi/1 Tim; DiversityEnhanced: 0 Abi/0 Tim) and to Bugs 5 & 6 (Figure 9 - Original: 17 Abi/4 Tim; DiversityEnhanced 2 Abi/1 Tim) also show that the IA fixes likewise reduced the inequities. Thus, we can conclude that the IA fixes did make Project F’s infrastructure more equitable for these use-cases.

Inclusion can be computed using a different measure on the same data. According to the dictionary, inclusion is “the action or state of including or of being included within a group or structure” [44]. Applying this definition to being included by a bug fix, we will conclude that the bug fix was inclusive if the number of lab participants’ facets affected by a bug decreased from the Original version to the DiversityEnhanced version for participants’ Abi facets and for participants’ Tim facets.

Applying this measure to Bugs 1 & 2 (Figure 5) reveals that, after the fix, participants’ Abi facets affected by decreased by 12 (from 14 facets affected to 2). Likewise, participants’ Tim facets affected decreased by 5 (from 5 facets affected to 0). Since the number of participants’ facets affected decreased for participants’ Abi facets and for participants’ Tim facets, we conclude that the fixes improved inclusivity for these use-cases. Applying the same measures to Bug 3 (Figure 8 - Abi: 5 Original/0 DiversityEnhanced, Tim: 1 Original/0 DiversityEnhanced) and Bugs 5 & 6 (Figure 9 - Abi: 17 Orig/2 DivEnhanced, Tim: 4 Orig/1 DivEnhanced) shows that they also improved inclusivity for these use-cases. Table 5 summarizes inclusivity results across all facets for these use-cases. As the table shows, for every bug and every facet value, participants’ Abi-facets and Tim-facets all ran into fewer barriers in the DiversityEnhanced version.

5.2 What about gender?

In some prior literature (e.g., [60]), analyses of these cognitive styles have revealed gender differences. That was also the case for our Stage Three participants’ cognitive styles. The participants displayed a range of facet values, but as in other studies, women’s facet values tended more “Abi-wards” than the other participants’ (Figure 11). These results agree with previous literature that explain how these facets tend to cluster by gender [11]. These results also, when taken together with Figure 5, Figure 8, and Figure 9, show that most of the facets affected by the bugs were those of the women participants.

However, the SUS usability ratings did not differ much by gender. First, as Table 6 shows, the SUS scores of participants who used the Original project were equally low across gender, which may

Table 5: Inclusivity summary: Team F’s IA fixes’ effects on the Abi-like facet values (top) and the Tim-like facet values (bottom) were all positive, showing that the IA fixes increased the inclusivity of the prototype across all cognitive styles.

<table>
<thead>
<tr>
<th>Bug ID</th>
<th>Motiv</th>
<th>SE</th>
<th>Risk</th>
<th>Info</th>
<th>Learn</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bug 1 &amp; 2</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Bug 3</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Bug 4</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Bug 5 &amp; 6</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

Figure 11: # of women (orange), men (black), and decline-to-specify (gray) with each combination of facets (from facet questionnaire), using the same x-axis scheme (from 5 Abi facets to 5 Tim facets) as Figure 3. Note that the right half of the graph contains only 1 of the 8 women participants.
suggest that the Original had a long way to go from everyone’s perspective. Second, the SUS scores for participants who used the DiversityEnhanced project were much higher across gender, adding to the body of evidence (e.g., [31, 60]) that designing for often-overlooked populations (here, Abi) can benefit everyone.

5.3 The Facet Questionnaire as a Measuring Instrument

As a few other researchers have also done [23, 60], we used the cognitive facet questionnaire (Section 3.3) to collect the lab participants’ facet values. We also collected facet values from a second source: participants’ verbalization during their tasks. These two sources of data enabled us to consider the consistency of the questionnaire’s responses with the facets that actually arose among the participants—a form of validation.

The data comparing participants’ facet questionnaire responses with their actual in-situ facet occurrences were detailed earlier in Figure 5, Figure 8, and Figure 9. Outline colors depict the in-situ facet occurrences that arose; the shape’s fill color depicts the participant’s questionnaire response for that facet. (No outline color simply means no evidence arose in-situ about that facet.) Thus, when an outline color matches the shape’s fill color (questionnaire response), then the questionnaire captured that participant’s facet value reasonably well for the situations that arose.

Overall, 78% of participants’ in-situ facet verbalizations aligned with their facet questionnaire responses. Since facet values can be somewhat situational, we would have been surprised if the match had been 100%. These results are encouraging that the facet questionnaire was a reasonable measure of participants’ facet values.

6 THREATS TO VALIDITY

As with any empirical research, our investigation has threats to validity. In this section, we explain threats related to our investigation and ways we guarded against them.

During Stage One, Team F reported the issues found in their project from the perspective of one type of newcomer based on GenderMag’s Abi persona. Past research has suggested using the Abi persona first [23], since Abi’s facet values tend to be more under-supported in software than those of the other personas (e.g., [10]). However, fixing problems from only this persona’s perspectives could leave non-Abi-like newcomers less supported than before. We mitigated this risk by empirically evaluating the fixes with both Abi-like and Tim-like newcomers. That said, some cognitive facets are not considered at all by GenderMag personas, such as memory or attention span, which could be particularly pertinent to people with even mild cognitive disorders. Our investigation did not account for those types of cognitive facets.

Another threat is that our investigation is based on four use-cases in a single OSS project, which may not generalize to other use-cases, other OSS projects, or other information-rich environments. The relatively small number of participants (18 in total), which was necessary for tractability of qualitative analysis, also threatens generalizability. In addition, our Stage Three investigation was designed as a between-subject study—in which each participant uses only one version of the system—to avoid learning effect and participant fatigue. This design choice could lead to uncontrolled differences between the two participant groups. To partially mitigate this threat, we used participants’ facet questionnaire responses to assign them to treatments with identical facet distributions (recall Fig 4).

In Stage Three, the identical sequence of the tasks (use-cases), which reflects the workflow common for OSS contributions [54], may have created learning effects that could have influenced the results. Finally, our comparison of facet questionnaire results against verbalizations had only partial data available, since we coded facets from only participants’ verbalizations when they encountered a bug, and P5-O’s audio for Bug 1 & 2 were corrupted, so we only had observation notes for that participant.

Threats like these can be addressed only by additional studies across a spectrum of empirical methods that isolate particular variables and establish the generality of findings over different types of OSS projects, populations, and other information rich-environments.

7 CONCLUSION

This paper has empirically investigated the impacts information architecture can have in creating inclusivity bugs in an OSS project’s technology infrastructure. The “whether” aspects of our RQ1 results revealed that IA can indeed cause inclusivity bugs in technology. In our investigation, the newcomer participants ran into IA-related inclusivity bugs 20 times (Table 3). Our RQ2 “whether” results also revealed that IA can be part of the solution. In our investigation, Team-F’s IA fixes reduced the number of inclusivity bugs the participants experienced by 90% (Table 3).

Team F’s hows of the above results lay in the fault localization capabilities IA brought to the “Why-Where-Fix” paradigm:

- **IA and where’s:** In Stage One, Team F was able to localize the IA where’s behind the inclusivity bugs they identified (Section 4 and Table 4).
- **IA and fixes:** In Stage Two, Team F fixed the faults they had localized in Stage One, by changing the IA in the ways detailed in Section 4 and summarized in Table 4. The participants in Stage Three showed that Team F’s IA fixes helped across the cognitive diversity range of the newcomers in our investigation (Tables 3 and 5).

Key to these results is that these inclusivity fixes lay not in supporting one population at the expense of another, and not in “compromising” to give each population a little less than they need. Rather, as Table 5 illustrated, the fixes produced positive effects across diverse cognitive styles. These results provide encouraging evidence that the Why-Where-Fix paradigm’s IA-based approach to localizing inclusivity faults may provide a concretely practical and an effective way to increase the equity and inclusion of information-rich environments like OSS projects.

<p>| Table 6: Participants’ SUS rating scores. (Maximum possible for the subset we used: 32.) |
|-----------------------------------------|-----------------|-----------------|</p>
<table>
<thead>
<tr>
<th></th>
<th>Original</th>
<th>DiversityEnhanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Men’s Average</td>
<td>12 (6 Men)</td>
<td>19 (3 Men)</td>
</tr>
<tr>
<td>Women’s Average</td>
<td>12 (3 Women)</td>
<td>22 (5 Women)</td>
</tr>
<tr>
<td>Gender-not-stated</td>
<td>N/A</td>
<td>32</td>
</tr>
<tr>
<td>Overall Average</td>
<td>12</td>
<td>22</td>
</tr>
</tbody>
</table>

Mariam Guizani, Igor Steinmacher, Jillian Emard, Abrar Fallatah, Margaret Burnett, and Anita Sarma


[51] Louis Rosenfeld, Peter Morville, and Jorge Arango. 2015. Information Architecture: For the Web and Beyond. O'Reilly Media, Inc.


