Issue Discussion Annotation

Issue Annotation Methodology. We qualitatively analyzed the issue report discussions to identify the resolution-related activities performed by various stakeholders (developers, project managers, reporters, *etc.*) to resolve the issues. To investigate the issue report discussions, we adopted a rigorous approach by employing an iterative multi-coder *open coding* methodoloy [154] where all the issue reports were analyzed by at least two researchers. We detail the annotation procedure here.

Goal: The goal of the issue annotation was to construct a *code catalog* that accurately captured the Firefox issue resolution process. To that end, we aimed to accurately identify: (1) *themes* or *codes* related to issue resolution activities performed and discussed by Mozilla stakeholders in issue reports (*e.g.,* reproduction attempts or a code review), and (2) problem types described in the issues (*e.g.,* crashes and UI issues), which we call *problem categories*.

Annotators: Seven researchers from our research lab conducted the issue annotation process, including two authors of this paper. The team comprised one professor and six Ph.D. students. All seven annotators have 1-9 years of research experience (particularly in qualitative text analysis) and five of them have 1-4 years of industry experience.

Annotation Tool: We used the Hypothesis annotation tool [151] to directly annotate the web pages of the issue reports. The tool allowed us to collaboratively assign *codes* to text snippets in the issue threads, modify the assigned codes, and discuss the annotations. The annotation data, including the codes, annotated text snippets, and meta-information was exported to JSON files, found in our replication package [152].

Annotation Unit: The annotation procedure aimed to code *text snippets* in issue comments. The minimal unit of annotation was complete sentences; however, the annotators were allowed to annotate multiple sentences, paragraphs, or even entire comments if the textual content, as a whole, described information relevant to issue resolution activities. This decision was made because issue reports often contain comments with different sentences describing various resolution activities. For example, a single comment may contain some sentences describing reproduction attempts and some other sentences may describe an analysis of the problem's root cause. Our sampled 384 issues contain 13.4 (9) comments, 30.27 (16) paragraphs, and 56.73 (25) sentences on average (median). A single textual snippet was allowed to be coded with one or more codes if it matched the codes' definition and application rules from our *code catalog*.

Code Catalog and Coding Guidelines: During the whole annotation process, we maintained a *code catalog*: a shared Google spreadsheet that was accessible to all the annotators to create and update the annotation codes and problem categories. The catalog included a list of codes, code descriptions, rules to apply the codes, and text snippets from annotated issues to illustrate the codes. The code catalog also included a list of problem categories, with detailed definitions, and issues that reported those categories. We also maintained a shared Google document with detailed guidelines of the annotation procedure, coding rules, and necessary resources for annotating the issues (official Mozilla documentation to get familiar with Firefox's resolution process, a glossary of annotation terminology, documentation about the Hypothesis tool, *etc.*). Both the code catalog and coding guidelines were built from scratch and developed by all the annotators incrementally and collaboratively.

Annotation Procedure: We employed an iterative muti-coder open coding methodology [154] to annotate issue resolution activities and the problem categories in the issue discussions:

- *Iterative multi-coder annotation:* The 384 issues were distributed evenly among the seven annotators, who examined, annotated, and validated the issue comments in iterative sessions, each session handling a batch of 30-50 issues. Each issue report was annotated and validated by two different annotators. The first annotator assigned codes to text snippets in the comments. A second annotator reviewed these annotations to verify their accuracy and identify any missed content. Reconciliation sessions between the annotators were held to discuss and resolve disagreements. The annotator roles alternated across batches: each person either annotated issues from scratch or reviewed annotations made by another team member. We instructed the annotators to annotate small sets of issues with breaks in between to avoid fatigue. These strategies aimed to minimize possible human mistakes and biases.

  We now detail the annotation procedure for an issue report. The first annotator carefully reviewed the issue, including attached patches and files, linked commits and issues, and available metadata (*e.g.,* commentators, issue flags, and status), to better understand and evaluate the content. Based on the content's meaning and the codes in the catalog, the annotator assigned one or more codes to the text snippets in the comments. The first annotator also identified the problem category by understanding the reported issue. The second annotator critically reviewed the first researcher's annotations, including the problem category, by following the methodology followed by the first annotator. The second annotator made necessary comments on the first author's annotations, added more annotations if necessary, or marked incorrect annotations. After each batch was processed, both annotators engaged in a discussion session to discuss the misunderstandings and disagreements to reach a consensus and finalize the annotation of the conflicting issues.

- *Code creation and refinement:* Two researchers annotated the first batch of 30 issues, creating an initial set of codes by annotating text snippets representing a stakeholders' activity to resolve the issue (designing a potential solution, implementing a proposed solution, requesting a code review, *etc.*) and identifying problem categories. This initial set of codes was created with complete and clear code definitions, examples from the annotated issues, and rules to apply them. Discussion sessions were carried out to consolidate the initial code set, merge codes with similar meanings or split codes that were too general, and update the catalog and annotated text snippets accordingly. The annotation of this first batch allowed us to create the coding guidelines document with the necessary resources to understand the issues and general annotation rules.

  Before annotating the remaining issue batches, we engaged in training the other four annotators by instructing them to review the coding guidelines and engaging in tutorial sessions illustrating the annotation of a few issues from the first batch and answering their questions. During the annotation of the remaining batches, the code catalog was collectively updated, and changes—such as new codes, code merges, and renames—were agreed upon and promptly communicated among the annotators. When the code catalog was updated, relevant coded issue text was revisited, and annotations were adjusted as needed. Constant communication and discussion among the annotators were conducted via online Zoom meetings and Slack discussions. This was essential to ensure an accurate code catalog was constructed and consistency between the catalog and the annotated content.

  To mitigate agreement by chance, the codes assigned to the text snippets were reviewed by the second annotator, including those without disagreement. We avoided defining "umbrella" codes by creating complete definitions for each code and by assessing the appropriateness of codes during reconciliation of the annotators' results such that codes that were too similar could be merged and codes that were overbroad could be broken into multiple, more specific codes. This was done when the annotators observed multiple codes frequently being used to describe the same resolution activity or when a code was being applied to text snippets with different meanings, respectively.

Disambiguation Techniques. We adopted the following techniques to handle ambiguities present in issue comments:

1. **Comprehensive Code catalog:** We developed a comprehensive code catalog that includes the code definition, rules to apply them, and real examples from annotated issues. The catalog was updated iteratively and collectively to ensure an accurate coding catalog via recurrent communication and discussion among the annotators.

2. **Inspecting Referenced Files and Running External Searches:** Annotators were instructed to inspect the attached patches or other files to clearly understand the problem reported in the issue report and the comments. The annotators were allowed to search the web for Mozilla terminology, components, technologies, and any other confusing information conveyed in the issues comments.

3. **Assigning Multiple Codes:** Annotators were allowed to assign multiple codes to the same textual snippet if it satisfies the criteria of one or more codes. This decision was made to handle ambiguous situations, which were discussed during the reconciliation sessions.

4. **Discussion Sessions:** In-depth discussion sessions among annotators helped them to clarify their misunderstanding about the issue, issue comments, or codes in the code catalog. If two annotators could not resolve any ambiguous case, a third annotator reviewed the issue/comment and helped to resolve the ambiguity.

Mitigating Potential Biases. For issue annotation, our goal was to accurately identify themes or codes related to issue resolution activities performed and discussed by Firefox stakeholders in issue reports. To that end, our coding methodology was fundamentally inductive, designed to mitigate possible biases and assess the correctness of our catalog, by:

1. **Performing multi-coder iterative coding:** Each issue was coded/reviewed by at least two annotators. One researcher first read and understood the issue title and description and every issue comment, creating and/or assigning codes to text phrases that signified a well-defined resolution activity. A second researcher followed the same procedure on the issue, considering the codes assigned to the issue content by the first coder and making comments about any discrepancies (incorrect or missed coded content). The coding was performed in batches of 30-50 issues and at the end of each batch, there were one or more discussion sessions to solve discrepancies (involving a third coder when needed). The iterative nature of the coding helped mitigate human biases as issues were coded because it

allowed to: (1) homogenize the understanding of new and existing codes, including merged or split ones, and how to apply them, (2) refine the definitions, rules, and examples of the codes in the catalog, and (3) update the catalog incrementally, retroactively updating the coded text in case of catalog changes.

2. **Critically reviewing and discussing issue coding:** The second researcher always carefully and critically reviewed the coding of the first researcher. The multiple discussion sessions at the end of each iteration were essential to ensure coding accuracy and refine the catalog. Coding, reviewing, and discussion needed to be incremental and iterative to effectively address the diversity of reported problems (17 problem categories), the variety of discussed information kinds (22 resolution activity-related codes with 27 other types of codes), and the length of the discussion threads (50% of the issues have 10 or more comments).

3. **Maintaining a detailed code catalog and coding guidelines:** The catalog was accessible by all the coders at any moment via a web spreadsheet. The catalog includes detailed code definitions, real examples of coded text from the issues, and specific rules to apply them (*e.g.,* an attached patch, attached file, or commit messages indicate a IMPLEMENTATION), which facilitated annotation. Before making any changes to the catalog (merging/splitting codes, code definition and example changes, *etc.*) there was also discussion and agreement among coders. The coded text in all coded issues was updated with any catalog change. This was possible because the Hypothesis annotation tool [151] keeps traceability between the coded text and the codes and allows us to update the codes and coded text easily. We also built a document with specific guidelines about the coding process and how to apply some of the codes. The catalog and guidelines will be found in our replication package [152].

4. **Alternating roles during coding:** The annotator alternated roles across coding iterations (*i.e.,* across batches): they performed coding from scratch or reviewed the coding of another researcher. Also, the issues were distributed across different (coder, reviewer) pairs. This coder/reviewer diversity aimed to address possible biases stemming from having the same two people coding or reviewing.

5. **Maintaining constant communication and taking breaks:** A Slack channel was used for instant communication about the coding. Zoom meetings were also used for communication and discussions. We instructed the coders to not code for more than one hour to avoid fatigue and possible mistakes in the coding.

We believe that the iterative in-depth coding, reviews, and discussions between coders and the comprehensive coding catalog contributed significantly to ensure coding quality and mitigate biases.

Annotation Results and Inter-Coder Agreement. The annotation procedure resulted in 3,707 annotated textual snippets in 2,574 issue comments across the 356 issue reports. The annotators agreed on 3,438 annotations with an agreement rate of ≈93% and a Cohen's kappa of 0.92, which indicates high overall agreement [153]. Common sources of disagreement (observed in 269 of 3,707 text snippets) included misunderstandings due to ambiguous comments or unclear code definitions. If both annotators were unable to reach an agreement (which included code catalog and annotation updates), a third annotator reviewed the issue to resolve the conflict.

While we report agreement measurements, we clarify that our annotation methodology was fundamentally inductive, designed to mitigate possible biases and assess the correctness of our catalog. We believe that the iterative in-depth coding, reviews, and discussions between annotators as well as the shared code catalog and coding guidelines contributed significantly to ensuring annotation quality and bias mitigation.

Bibliography

[151] The hypothesis web annotation tool. `https://web.hypothes.is`, 2024.

[152] Online replication package. `https://tinyurl.com/repl-pack`, 2024.

[153] J. Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

[154] D. Spencer. *Card sorting: Designing usable categories*. Rosenfeld Media, 2009.