Summary of changes

Pierre Depaz - Discursive Strategies in Style Guides Negotiation on GitHub

---

In order to address the feedback from both the reviewers and the editorial committee, I've made changes regarding the length of the article, the methodology and the theoretical backgrounds used, cross-referencing analyses of each repositories, adding screenshots of some of the most illustrative points of the corpus, as well as more minor edits.

Article length

While the character count is slightly below the first version, it nonetheless includes all edits mentioned below. In general, I've shorted the description of the plan, as well as all sections leading up to the methodolgy.

I've also consolidated the section *Source code, understood as [. . .]* (p.4) and condensed the *need for style in programming* section, in order to make the section on GitHub come sooner, by moving it one section earlier.

Minor edits

I added a reference to "backwards compatibility" on page 17.

I added a reference to the "benevolent dictator" re: Linus Torvalds in a footnote. It seemed to me that it was out of scope to expand too much on the role of Torvalds (and other leader figures) in FOSS projects.

I've also added several screenshots of comments through the analyses, to better illustrate the points made. However, I've decided against including more verbatim comments, in order to maintain the length of the article under a reasonable threshold.

Data collection

I clarified some of the metrics I have used for the study of the comments, as well as what was analyzed within these comments:

> The gathered corpus consists of 12780 issues and pull requests, totalling 53673 comments. From these, we discarded the least commented issues, under the assumption that meaningful and diverse discursive interaction takes place after 15 or more comments, which leaves a core corpus of 80 issues for airbnb/javascript, 82 issues for standard/standard and 444 for prettier/prettier1. While these were the main point of focus, this study included cursory browsing of

> lesser issues. As a complementary to the data-heavy studies mentioned above, the approach for this analysis is qualitative: reading through all messages of the most commented issues, evaluating them on multiple dimensions: the lexical field they resort to (emotional, technical, rational, organizational), the subject's stance (speaking as an individual or on behalf of a group), the user's background (shown by their status as either Member, Contributor or User, as well as the other repositories of the corpus they might contribute to). Reaction emojis, used to express passive agreement or disagreement with a message, are considered qualifying markers, not creating new meanings in and of themselves. (p. 9)

Methodology

In terms of clarifying the theoretical background I rely on, I've also made clearer the reliance on not the sociology of organization, but rather the social role of writing in organizations, tying it more clearly to the work of Goody and Frankel. I've also clarified how linters, as intermediary objects, are taken into account in this process.

> Here again we see the role that intermediary objects play in the productive activity of writing source code: as objects that lie in between several elements (source code, compilation process, IDE), several actors (programmers in the same team) and successive stages of a work process (drafting code, reviewing it, and committing it to production) (Vinck & Jeantet, 1995), this study will rely on this concept to highlight the very active role that linters play in these negotiations. (p. 5)

As for methodology of language, I've clarified why I use Habermas's approach, particularly because he introduces a typology of discourses appearing within the context of communicative action.

> Habermas provides a typology of arguments deployed through communicative action (Habermas, 1984). Among these are theoretical discourse (based on logic), practical discourse (based on situational appropriateness), aesthetic criticism and explicative discourse; we will see how effective each of these are, and particular how practical discourse is influenced by the socio-technical environment in which the communication takes place. Additionally, we use here Habermas's notion of a negotiation as a bargain: in which both parties give up something in order to reach satisfaction, and contrast it with discussion, of which the result can be the change in stance of one, or both, parties. (p.9)

Additionally, I've added a footnote to specify that the linguistic elements I borrow from Habermas for analysis can be complemented by Jakobson's *phatic* function of language. (footnote p.12)

In terms of “negotiation vs. discussion”, I’ve developed on the instances in which the discursive action takes these different forms. For airbnb/airbnb:

> shifting the interactions from a negotiation, in which parties reach a compromise, to an explanation, in which the outcome of the exchange is ultimately the alignment of the user to the views of the maintainer (p.13)

and reiterated in the conclusion:

> and therefore shifting a process of negotiation in which all parties are equal, to one in which the maintainer has a larger bargaining power and, more often than not, only accepts to discuss why such a decision has been made, and not whether it should be changed (p.24)

In terms of standard/standard, and prettier/prettier, I added a concluding sentence to clarify that the discussion is about the means, side-stepping a negotiation about the ends

> This shift emphasizes that the most productive issues are those based on a discussion around implementation, rather than a negotiation around style itself. (p.16)
>
> Once again, the negotiation over the stylistic validity of the automated style guide is superseded by dicussions over its technical implementation. (p.20)

I added an assessment of programming style by including studies by Cox 1988 and Fisher 2008, showing that style isn’t commensurated with existing metrics, but that it clearly helps with program comprehension, another factor of productivity (p.4). I also linked the article that is on Prettier’s main page (*Why style matters*), and connected this to the limitations of the study: self-motivation indeed seems to be a big part of this community.

Regarding the limitations of the study, I added a paragraph to clarify that it is indeed a subset, but it is worthwhile to note that choices in taste and tools are widespread amongst the larger software development community.

> It must be noted that a limitation of this study is that the surveyed population might just be a motivated subset of users, putting emphasis on the correct automation of the correct style of their code. While the findings here might not therefore represent the broader software development communities, they nonetheless embody heightened forms of socio-technical interactions, which might only be more faintly manifested in other developers. (p. 10)

Repository analyses

In terms of connecting the different repositories, I’ve added a sentence in the airbnb section about how the phenomenon identified (shift to the technical) foreshadows the same phenomenon in the other repositories:

> This shift, from the stylistic to the technically feasible, is a recurring pattern which we'll also observe under different manifestations in the standard/standard and prettier/prettier projects. (p. 14)

I've also qualified further the description of the different projects by specifying how they resemble, and how they differ from each other (p. 14, p.17), and a qualification of how standard/standard and prettier/prettier engage in the same issue:

> It has been shown that, while standard/standard shares this approach of « no-options », their justification is rather by broadening the scope of what is considered acceptable (standard/standardx, standard/doublestandard), therefore growing the community, and not relying on the fact that only one single project should exist. (p.22)

Final remarks

Please let me know if you would still prefer to have more verbatim quotes included in the article, and if you still consider it too long in its present form. I will try to accomodate it as much as possible without making it too succint.