design, discussion, and dissent in open bug reporting

Andrew J. Ko

Parmit K. Chilana

dub
Information School
UNIVERSITY of WASHINGTON
if you want software to serve your needs, here are your options...
if you want software to serve your needs, here are your options...

**do it yourself**

write it yourself

test it yourself

use it yourself
if you want software to serve your needs, here are your options...

do it yourself

write it yourself

test it yourself

use it yourself

closed software

decide to buy

call support

blog mean things about them
if you want software to serve your needs, here are your options...

do it yourself

write it yourself
test it yourself
use it yourself

closed software
decide to buy
call support
blog mean things about them

open software
become a contributor
write tests
persuade developers
open software is perhaps the most accommodating to user involvement and influence but how open?

here’s one story...
on April 30th, 2004, a developer in Greece filed a bug titled “History of Firefox is lost frequently without crash.”
9 months later, several Firefox contributors begin to diagnose the crash.

they concluded that it affected too few users to be worth the time to diagnose and fix.
another month later in February 2005, a small group of Mozillazine users found the bug report and began to write complaints about it. They provided a formal list of 15 patterns they’d noticed about what caused the crash, this prompted the developers resume work.
throughout 2005, more users found the report and described their experiences with the crash.

in December, the developers began to get annoyed; they began citing the Mozilla etiquette guidelines, asking people to calm down.
on Christmas Eve, 2005 one developer declared that the crash would not be a priority: Firefox 2.0 would have a new history mechanism, making the current problem moot. Users felt slighted and a debate ensued about whether the bug was serious enough to warrant attention.
after 4 months of silence, two years after the initial report, the debate began anew...

**users demanded** that the 1.x release of Firefox be fixed, regardless of the 2.0 plans
after 4 months of silence, two years after the initial report, the debate began anew...

**users demanded** that the 1.x release of Firefox be fixed, regardless of the 2.0 plans.
two years after the original report the developer originally assigned to fix the bug posted a patch and the bug was fixed.

users win!
two years after the original report
the developer originally assigned to
to fix the bug posted a patch and the
bug was fixed.

users win!
when debates like this arise...

what is debated?
what kinds of arguments to participants make?
how does the medium of bug report help and hinder these arguments?
what kinds of impact do users have?

this was just **one** report of **600,000** Mozilla reports from the last 10 years...
the dataset

three complete bug repositories downloaded in August 2009

selected because they have active, open bug report repositories

Firefox Linux Kernel Facebook Apps
the sample

we ranked reports by contentiousness

 operationalized as frequency of personal pronouns (I, me, you, we, us, them, etc.)

randomly selected 100 reports from the top 300, accounting for ~1,000,000 words

frequency distribution of personal pronouns across all three repositories
we categorized each report by what quality the reporter wanted improved. Reports mostly concerned functionality, usability, flexibility, and features.
the analysis

we segmented each report into a series of utterances. Utterances were inductively analyzed for subjects, resulting in 6 categories:

- **scope** of discussion
- **idea** for solution
- **dimension** of solution space
- **rationale** for idea
- **process** for decision
- **decision** for the report
the results

both authors coded each utterance category

we then analyzed the collection of arguments made in each set, considering the role of the medium
“idea” utterances

idea utterances explored the design space of a feature, illustrating solutions that emphasized different software qualities.

text was a limiting for many of the inherently visual or interactive ideas, but contributors worked around it with ASCII art:

```
<table>
<thead>
<tr>
<th>Please Wait</th>
</tr>
</thead>
<tbody>
<tr>
<td>Firefox is clearing your history, this may take a few minutes</td>
</tr>
<tr>
<td>[</td>
</tr>
</tbody>
</table>
```
“dimension” utterances

ideas engendered discussion of what aspects of a solution could be varied to influence software qualities such as usability or performance of a particular design.

“can we get away with having UI that’s only visible to people who can clearly distinguish red from black?

“Does the user expect the favicon to remain stored, or to be updated when the site icon changes?

“Are security-savvy people going to get phished?
“scope” utterances

developers specified constraints on what solutions should and should not be discussed

Better presentation of URI elements would help http, but those were shouted down in another bug. We need to revisit that and gather hard evidence to better support that type of presentation, but that's way way way out of scope for this release.

many visitors were unaware of these scoping constraints, since they were overlooked in earlier comments
“rationale” utterances

the most common type of utterance was rationale for design ideas.

we analyzed the rhetorical devices employed to defend it
rhetorical devices

we identified 10 rhetorical devices in use

each rationale’s device was categorized with this scheme

the most common devices were not evidence-based

device #
anecdote 176
speculation 114
generalization 112
hyperbole 87
pragmatism 73
impact 71
logic 46
connotation 46
tradeoff 45
authority 37

frequency of rhetorical device usage in the sample
**anecdote**

“For example, a teacher at the high school I went to [...] could be seen to do things that, well, are looked upon badly by technical people. For one, she could be seen numerous times just clicking "OK" on a dialog, and shortly after asking why thing weren't working.

**speculation**

“Joe user won’t have the knowledge, nor the need to acquire it, and is unlikely to shoot himself in the foot using a feature he doesn’t know exists.
generalization

“For 99.9 percent of Firefox’s installed base, they DO NOT CARE about having multiple versions installed side by side. I know that IE doesn’t give users this option, so to the lay person [...] they will not miss the ability [...] because they had no idea you could do that.

hyperbole

“You mentioned this before and I still think it is rather unacceptable to leave such a critical bug in released versions. It renders Firefox 1.x rather almost unusable - I know I am exaggerating a bit, but I guess you get the point.
“process” utterances

some comments addressed the discussion itself and how it should unfold.

“I understand the frustration and want to reassure all of you that it is high on our bug priority list. The solutions are not exactly trivial so bear with us while we figure it out.

“[...] but from here on out, can we refrain from spamming the bug with more comments along the lines of "it happens to me". That is the purpose of the voting mechanism in bugzilla. PLEASE read a bug fully before posting a new comment to it, and only then if there is information to add, and you aren't repeating something that is already there.
“decisions” utterances

three decision patterns emerged:

63% involved brief technical discussions between developers with little dissent

19% involved clashes between the impact of an issue and constraints on addressing it

18% involved clashes between the original intent of a feature and its desired use

overall, developers welcomed participation, but wanted visitors to follow the process, follow etiquette guidelines, and refrain from treating the report like a discussion board.
summary

what is debated?

developers: solutions, solution spaces, and impact, constrained by prior and future intent

uses: egocentric characterizations of impact

what kinds of arguments to participants make?

anecdotal and speculative ones

mostly appeals to “logic” and authority

how does the medium of bug report help and hinder these arguments?

it makes visitors think the report is a discussion thread, when it is actually not
implications

bug reporting is an accidental, but occasionally effective form of user involvement. Such involvement is often messy and inflammatory.

Open software projects need new ways for users to contribute their perspectives collectively, to facilitate more evidence-based decision by developers.

questions?