Impact of switching bug trackers: a case study on a medium-sized open source project

Théo Zimmermann (with Annalí Casanueva Artís)

πr^2 project-team, Inria
&
IRIF, University of Paris

Friday, 12 April 2019

The Coq interactive proof assistant

- Software to formally verify *mathematical proofs* and *programs*.
- Created in 1984 by Gérard Huet and Thiery Coquand.
- Developed at Inria ever since.
- ACM Software System Award 2013.

Development team

Currently about 10 people (5 Inria researchers, one postdoc, 2 PhD students, 2 engineers), mostly in three French cities:

- Paris area (Paris Diderot Univ., Inria Saclay, Mines ParisTech);
- Sophia-Antipolis (Inria project-team Marelle);
- Nantes (Inria project-team Galinette).

Development team

Currently about 10 people (5 Inria researchers, one postdoc, 2 PhD students, 2 engineers), mostly in three French cities:

- Paris area (Paris Diderot Univ., Inria Saclay, Mines ParisTech);
- Sophia-Antipolis (Inria project-team Marelle);
- Nantes (Inria project-team Galinette).

An **ever increasing number of contributors** (61 in upcoming 8.10 release, 93 in the last year, over 200 in total).

Comparing bug tracking systems

The bug tracker used by an open source project is a fundamental piece of the development infrastructure.

Comparing bug tracking systems

The bug tracker used by an open source project is a fundamental piece of the development infrastructure.

Bugzilla is one of the most known bug trackers. It is **used by major open source projects** (Mozilla, Linux, Eclipse...) and has many features.

GitHub issues, provide less features but:

- **no switching** between code, pull requests and issues;
- Markdown formatting;
- comment editing;
- less heavyweight, more **modern feel** and look;
- **no need to create a new account** for opening an issue (if you already have a GitHub account).

Questions

What was the impact of Coq's bug tracker switch from Bugzilla to GitHub?

- Impact on **level** of bug tracking activity (RQ1)
- Impact on **quality** of bug tracking activity (RQ2)
- Impact on the **audience** of the bug tracker (RQ3)

Migration tool

We reused a tool by Andriy Berestovskyy to import existing reports using GitHub's API (from an XML dump of Bugzilla's data).

Its main feature is to preserve issue numbers when possible.

Migration tool

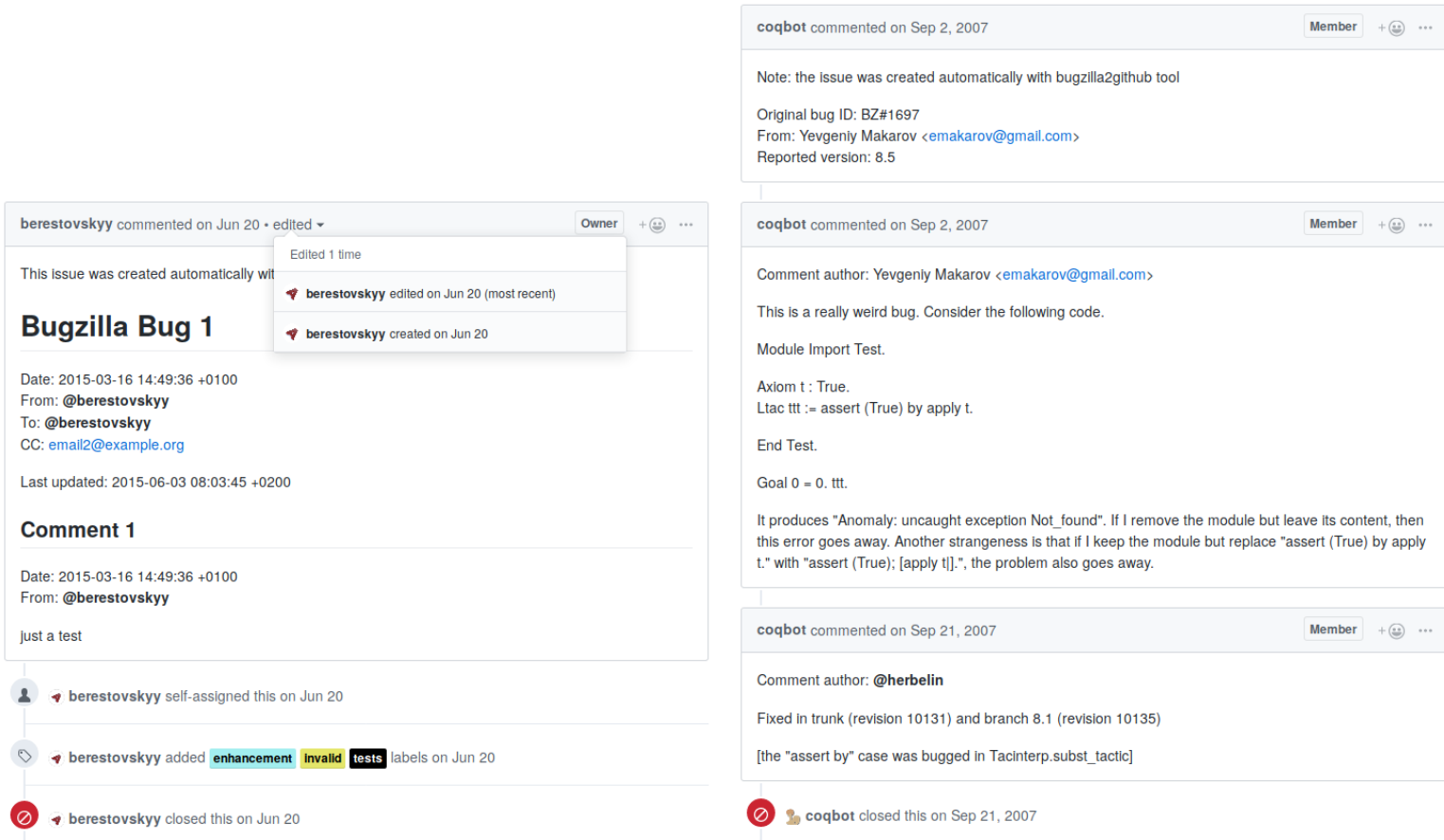
We reused a tool by Andriy Berestovskyy to import existing reports using GitHub's API (from an XML dump of Bugzilla's data).

Its main feature is to preserve issue numbers when possible.

We modified the tool to:

- handle non-consecutive issue numbers;
- save a table of correspondence for renumbered bugs;
- use the GitHub issue import beta API. This API:
  - does not generate notifications for new imported issues;
  - benefits from a much **higher rate limit** (necessary to import thousands of issues);
  - in one request, creates an issue with all its comments and its meta-data. In particular, this supports keeping the comment **original dates**.

Difference between using GitHub's normal REST API and the beta import API



Data extraction

GitHub's API is our **single source** for data extraction (thanks to the preservation of the meta-data).

Preprocessing included removing data from two users: a user who had opened nearly a quarter of all issues, and myself; but also removing artifact migration comments.

Outcome variables

Numbers of:

- issues,
- distinct weekly reporters,
- comments,
- distinct weekly commentators.

Heterogeneous analysis of the impact on developers / non-developers.

Developers defined as people with more than 100 commits.

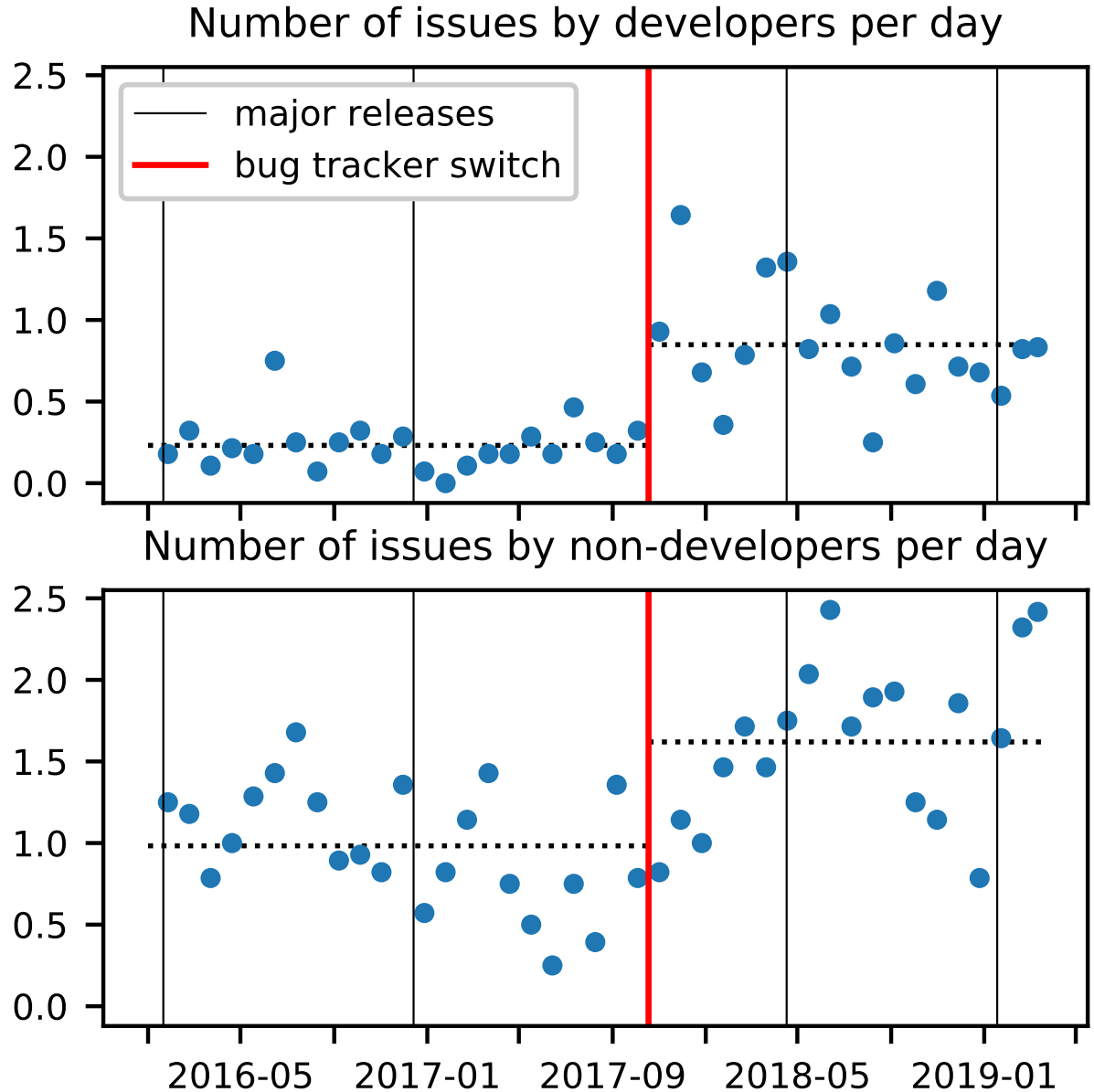


Figure 1: Number of issues per day (averaged by 4-week periods) with release dates (since 2016).

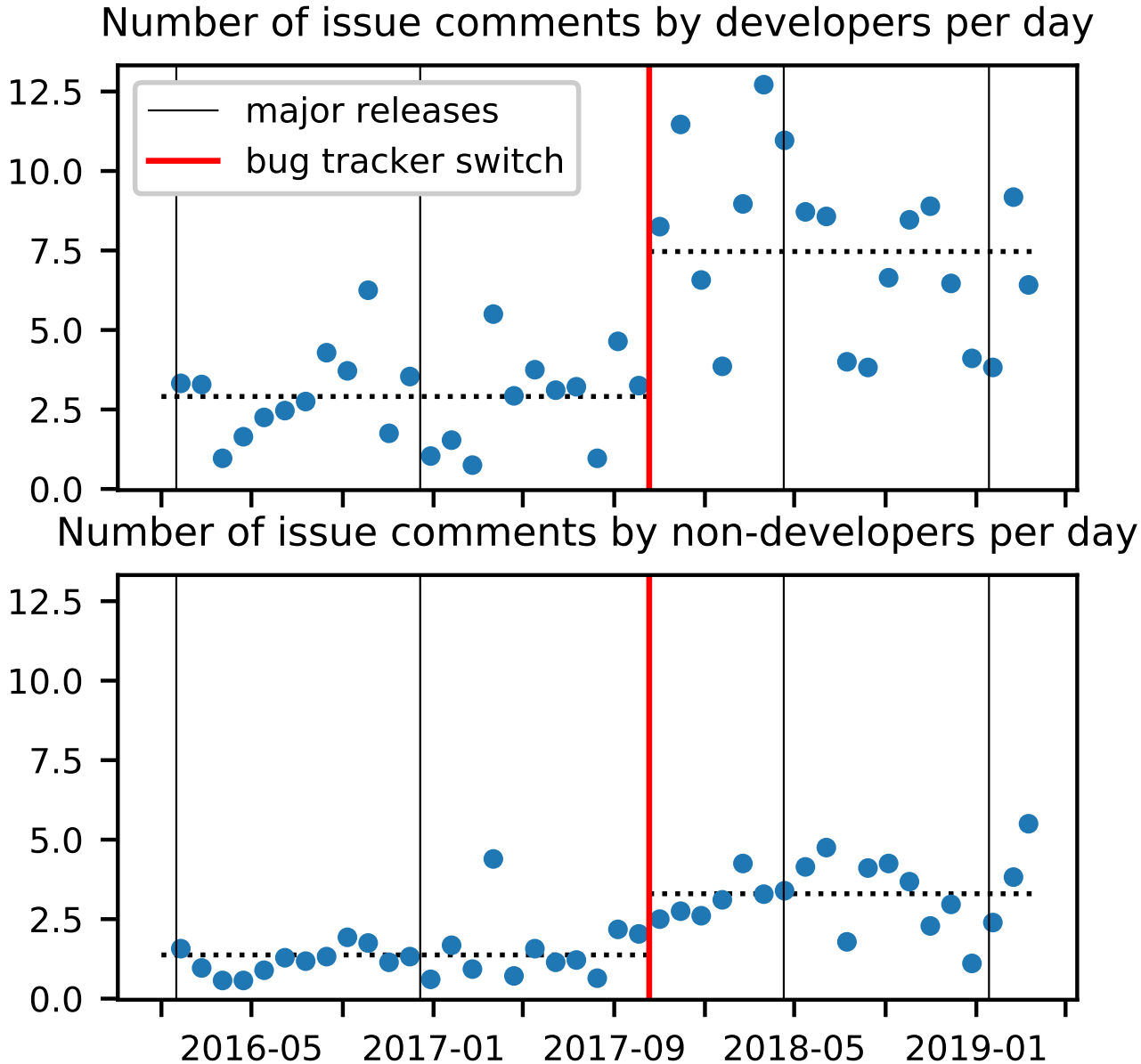


Figure 2: Number of comments per day (averaged by 4-week periods) with release dates (since 2016).

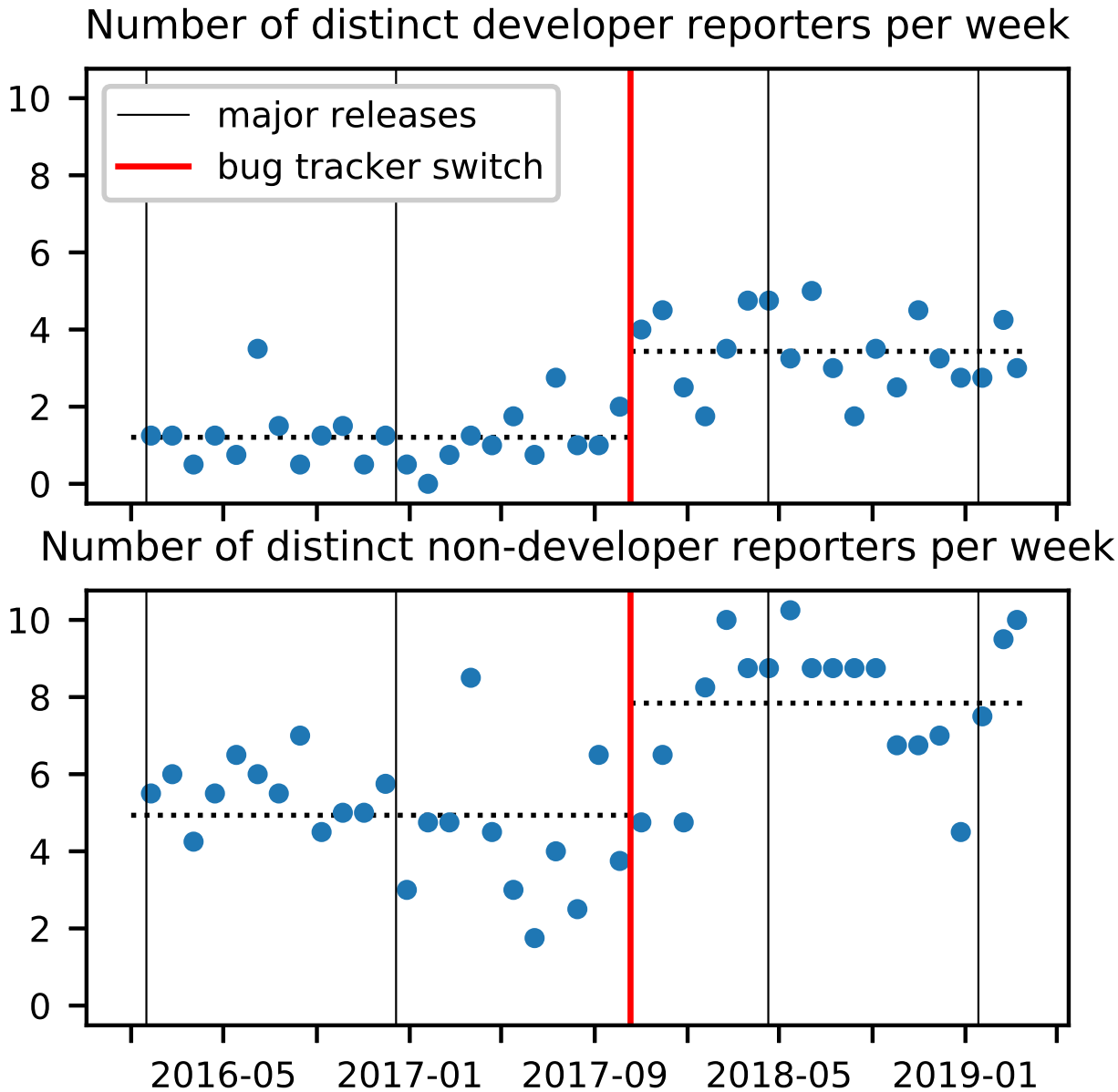


Figure 3: Number of weekly distinct reporters (averaged by 4-week periods) with release dates (since 2016).

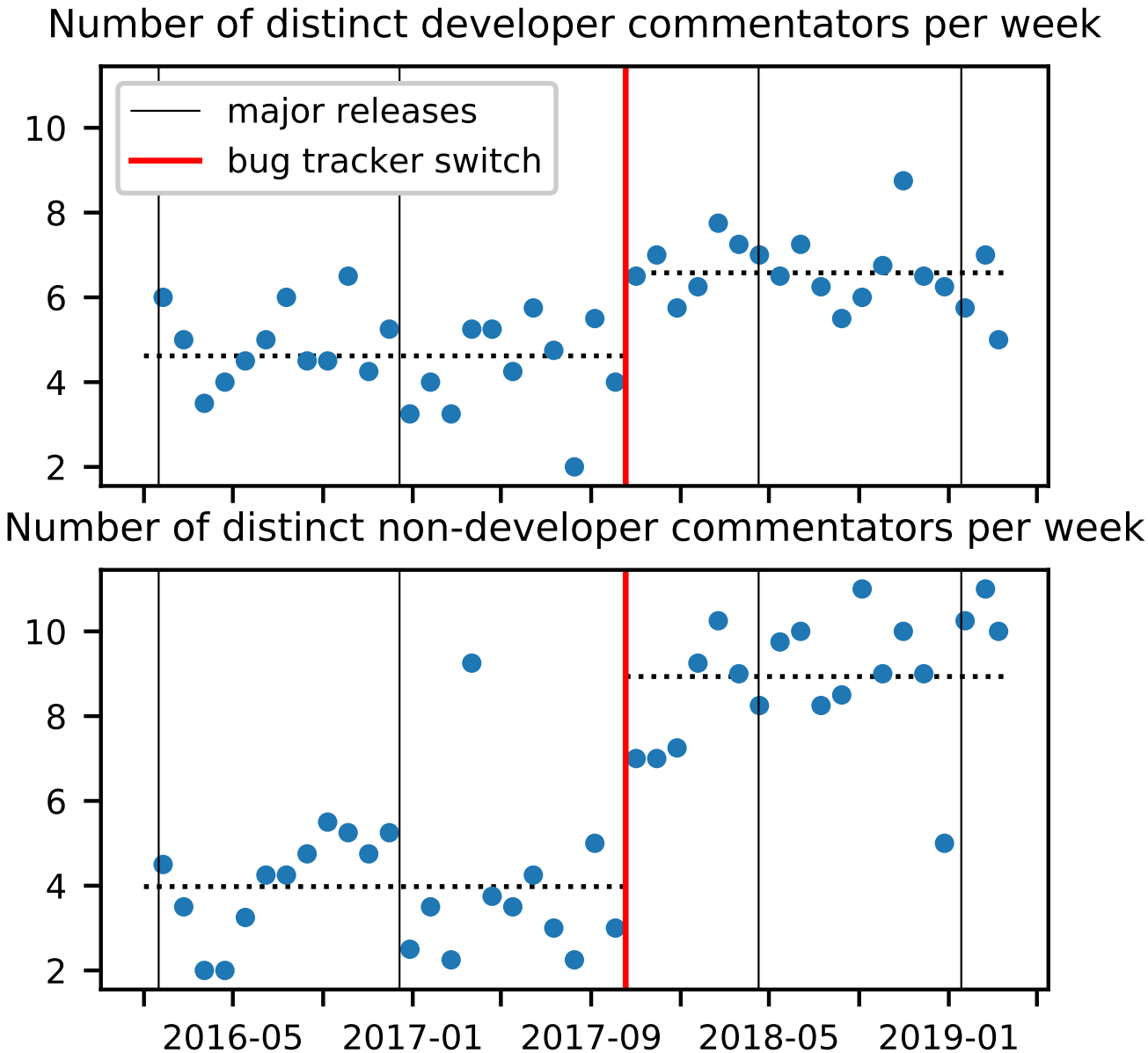


Figure 4: Number of weekly distinct commentators (averaged by 4-week periods) with release dates (since 2016).

Estimated regression

$$\textit{Number of issues}_{t<0} = \gamma_0 + \gamma_1 \times \textit{Relative date}_t + \epsilon_p$$

$$\textit{Number of issues}_{t\geq 0} = (\gamma_0 + \gamma_2) + (\gamma_1 + \gamma_3) \times \textit{Relative date}_t + \epsilon_t$$

Coefficients γ_2 and γ_3 are the estimates of interest.

Issues

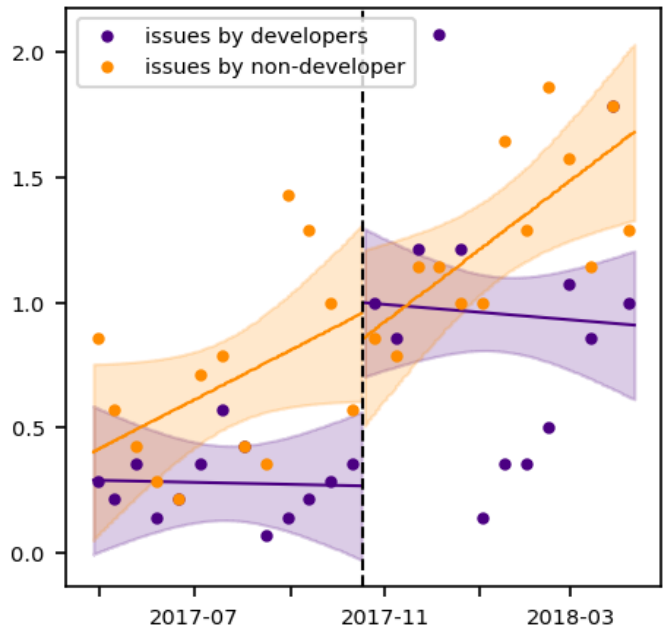


Figure 5: Number of issues per day before and after the switch (with fitting lines and confidence intervals from the regression results, and points corresponding to average values over two-week periods).

The jump in the issue creation rate (γ_2) is statistically significant for developers (with $p < 0.01$).

Comments

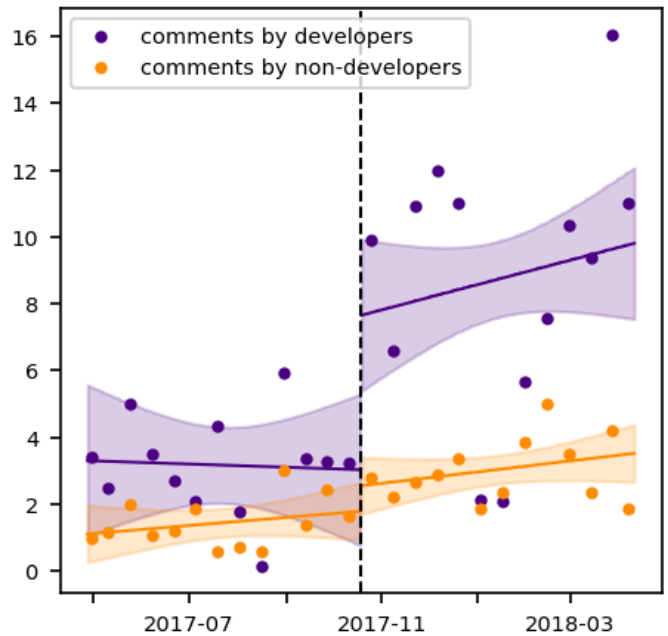


Figure 6: Number of comments per day before and after the switch (with fitting lines and confidence intervals from the regression results, and points corresponding to average values over two-week periods).

The jump in the commenting rate (γ_2) is statistically significant for developers (with $p < 0.05$).

Reporters

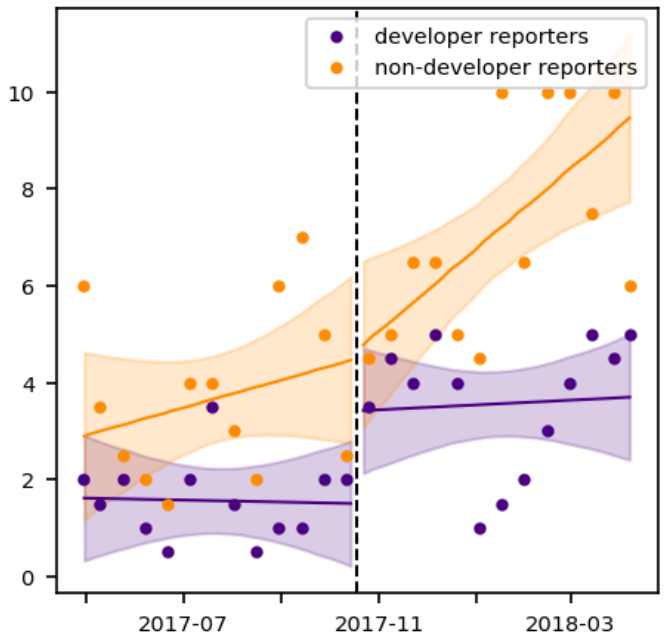


Figure 7: Number of weekly distinct reporters before and after the switch (with fitting lines and confidence intervals from the regression results, and points corresponding to average values over two-week periods).

The jump in the number of distinct weekly reporters (γ_2) is statistically significant for developers (with $p < 0.05$).

Distinct commentators

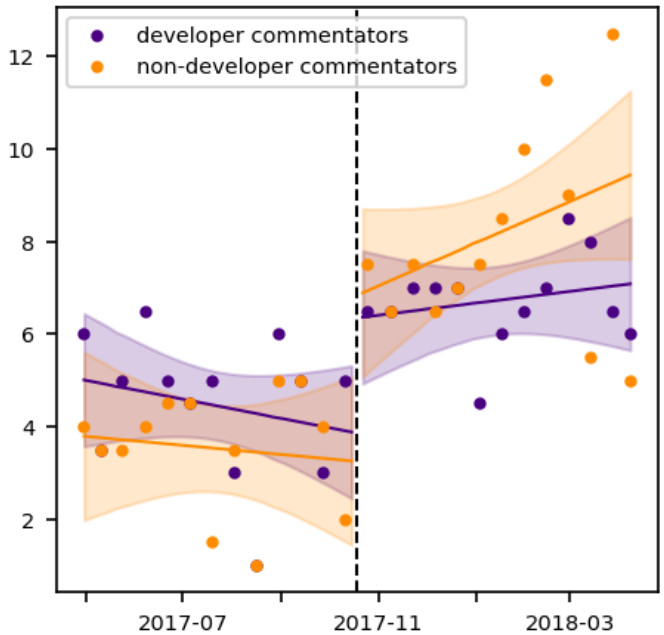


Figure 8: Number of weekly distinct commentators before and after the switch (with fitting lines and confidence intervals from the regression results, and points corresponding to average values over two-week periods).

The jump in the number of distinct weekly commentators (γ_2) is statistically significant for developers (with $p < 0.05$) and for non-developers (with $p < 0.01$).

Method

- 9 semi-structured interviews;
- done after the quantitative study;
- questions about motivations and risks, personal impact, interpretation of results, etc.

Impact on level of bug tracking activity (RQ1)

Quantitative results: more activity from developers, and more developers are frequently active.

Impact on level of bug tracking activity (RQ1)

Quantitative results: more activity from developers, and more developers are frequently active.

Interpretation:

- more **pleasant** to use (more bugs are actually reported);
- GitHub notification system **attracts attention**.

Impact on quality of bug tracking activity (RQ2)

- Experienced users can produce **better reports**;
- more **novice** users, that sometimes produce lower quality reports;
- users are more **reactive** when asked additional info;
- easier to discuss **cross-project** issues.

Impact on the audience of the bug tracker (RQ3)

Quantitative results: more non-developer commentators and more new commentators.

Impact on the audience of the bug tracker (RQ3)

Quantitative results: more non-developer commentators and more new commentators.

Interpretation:

- GitHub is more popular and **accessible**;
- some occasional contributors (through PRs) started commenting issues;
- more attention is given to newcomers;
- **more discussion** is happening with non-developers because it is easy and pleasant;
- interested users can **subscribe** to a subset of the activity.

Threats to validity

- Many robustness checks $\rightarrow$ high confidence on **internal validity**.
- Single case study $\rightarrow$ limited confidence on **external validity**.

Contributions

- Improved bug tracker migration **tooling** (already reused by / inspired other projects migrating to GitHub).
- Analysis of mined repository data shows **causal effect** of the switch.

  A Jupyter notebook is available and contains the full analysis pipeline (including data extraction and preprocessing code).
- **Interviews** complete and help intepret the quantitative results.

  Questions and coding of answers are available.
- Introduced state-of-the-art policy evaluation **Regression on Discontinuity** method to the field of software engineering.

Takeaways

When is it useful for a given project to switch bug trackers?

Some hints:

- Are developers using the bug tracker enough?
- Is switching from pull requests to issues painful?
- Do you want more discussion happening with non-developers on the bug tracker?