Is It Safe to Uplift This Patch?

An Empirical Study on Mozilla Firefox

Marco Castelluccio, Le An, and Foutse Khomh

Outline

Background

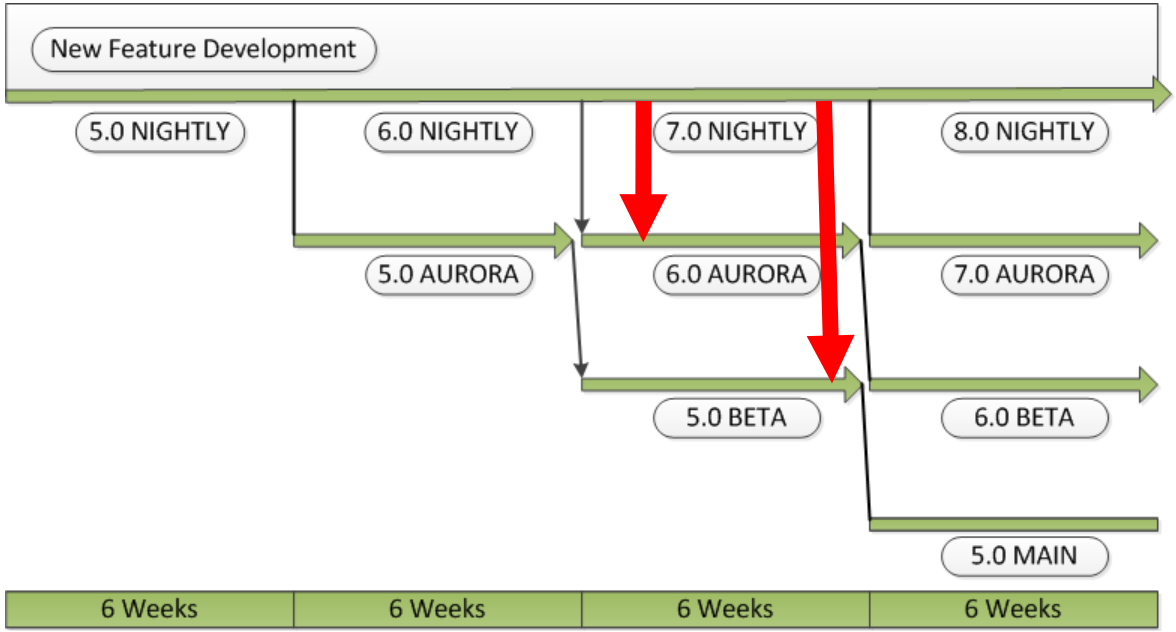
Related studies

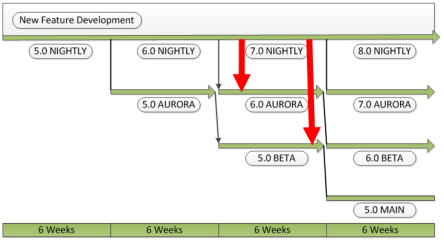
Case study design

Case study results

Summary

Mozilla Release Process

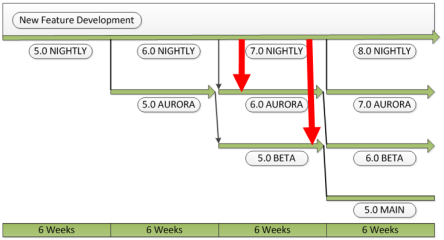




Patch Uplift@Mozilla

- **Release managers** (who are independent and different from reviewers) are responsible for deciding which patches can be uplifted.

- **The more a channel is stable**, the higher is the bar for approval of uplift requests.



Excerpt of Patch Uplift Rules

- **Aurora:** no new features; no disruptive refactorings; no massive code changes; no string changes, and must be accompanied by automated tests.

- **Beta:** the patch should be reproducible by QA; should have been verified on Aurora/Nightly first; and should not contain changes to the user-visible strings.

- **Release:** possible uplifts are fixes for major top crashes, security issues, functional regressions with a very broad impact.

Despite these rules, **multiple uplifted patches still introduce regressions** in the code…

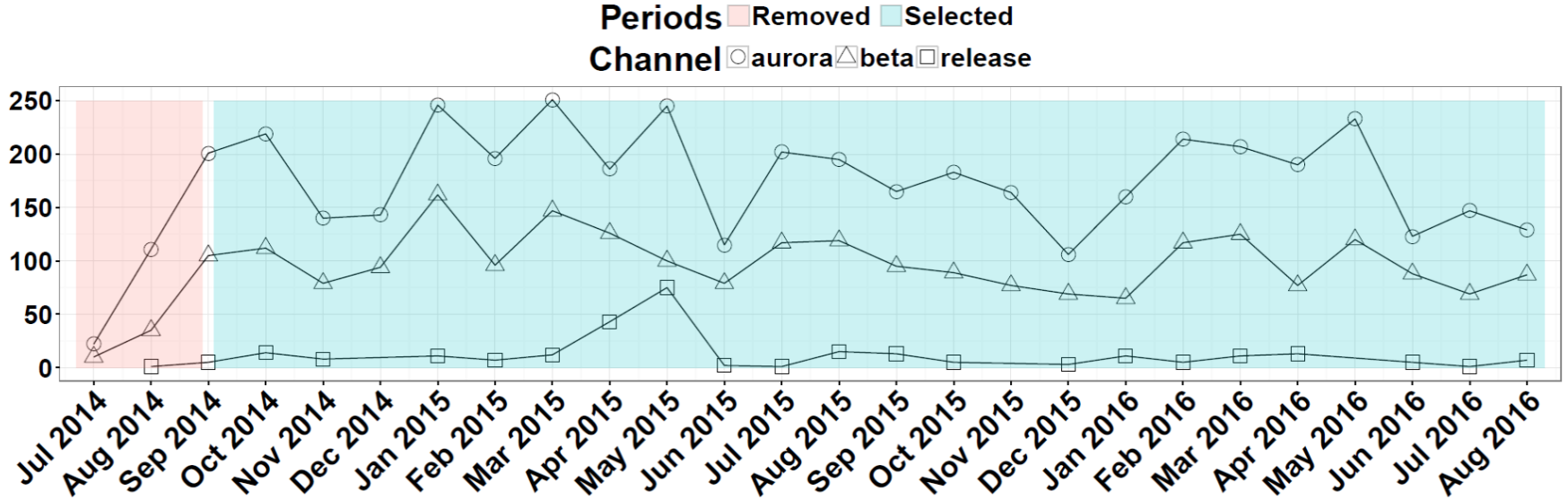
…**it was unclear if–and–how the rules are enforced at Mozilla** and why certain uplifted patches introduce post-release bugs.

Previous Studies on Urgent Patches

- S. Hassan, W. Shang, and A. E. Hassan, *"An empirical study of emergency updates for top android mobile apps"*, EMSE 2016.
- D. Lin, C.-P. Bezemer, and A.E. Hassan,*"Studying the urgent updates of popular games on the steam platform"*, EMSE 2016.
- M. T. Rahman and P. C. Rigby, *"Release stabilization on linux and chrome"*, IEEE Software 2015.

These works focused on their occurrence and impact on users' satisfaction but not specifically on post release bugs.

Case Study System



33,664 issue reports, in which there are

7,267 uplift requests in 17 versions of Firefox

285 to Release; **2,614** to Beta; and **4,368** to Aurora.

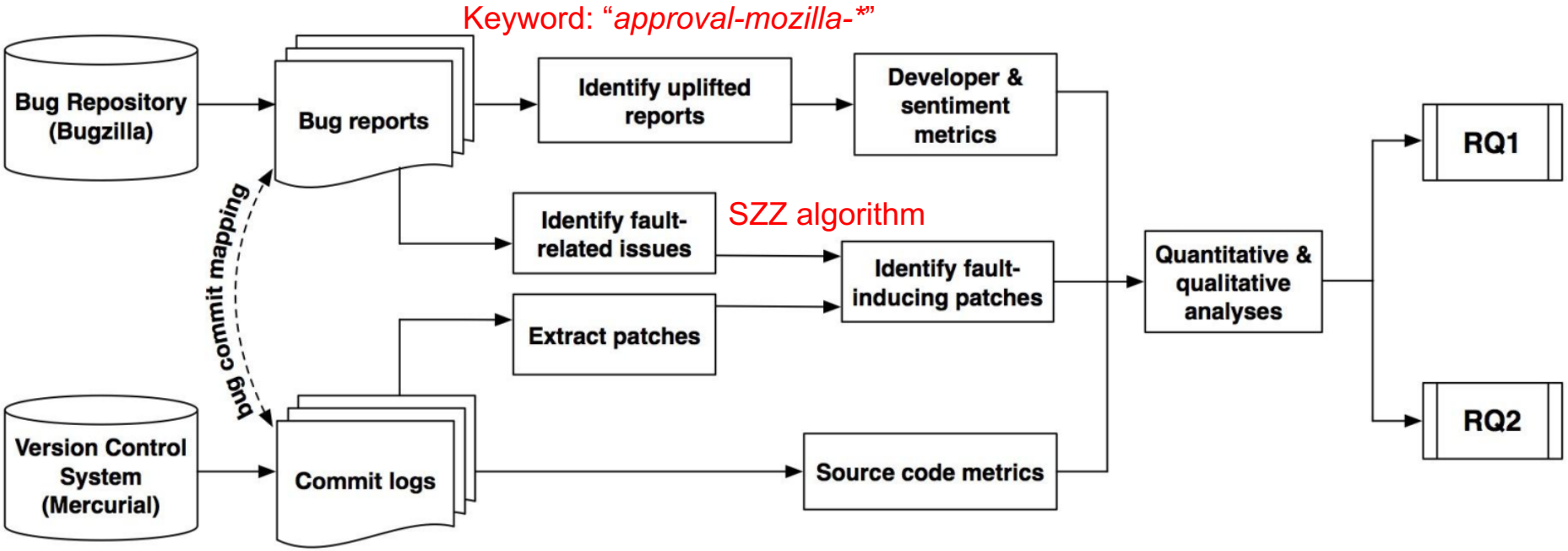
Research Questions

RQ1: What are the characteristics of patches that are uplifted?

RQ2: What are the characteristics of uplifted patches that introduced faults in Mozilla Firefox?

We answer both questions quantitatively and qualitatively…

Analysis Approach



Quantitative Analysis - Metrics

Experience & participation
developer experience, reviewer experience, comment number, comment words, review duration

Uplift process
landing delta, response delta, release delta

Sentiment
developer sentiment, module owner sentiment

Code complexity
patch size, test patch size, prior changed times, LOC, McCabe, function number, max. Nesting, comment ratio, module number

Social network analysis
PageRank, betweenness, closeness

Quantitative Analysis - Statistical Tests

- **We use the *Mann-Whitney U test* (α = 0.05) to compare the characteristics of**
  - uplifted patches vs. patches that were not uplifted (RQ1),
  - faulty vs. clean uplifted patches (RQ2).

- **We use *Cliff's delta effect size* to measure the magnitude of differences.**

Qualitative Analysis - RQ1

- Based on a random sampling, we manually examine and classify the *reasons why developers uplift patches* in the Beta and Release channels.

- We interview Mozilla release managers on the following questions:

1) Which factors do you take into account when deciding about an uplift?

2) Are there differences in how you handle uplifts in different channels, and what are the differences?

3) How do you decide which developers you can trust?

Qualitative Analysis - RQ2

- Based on a random sampling, we manually examine and classify the *root causes of uplifted patches that lead to faults* in the Beta and Release channels.

- We interview Mozilla release managers on the following question:

  - What are the characteristics of the fault-introducing patches that you are not currently taking enough into account but could be considered in the future?

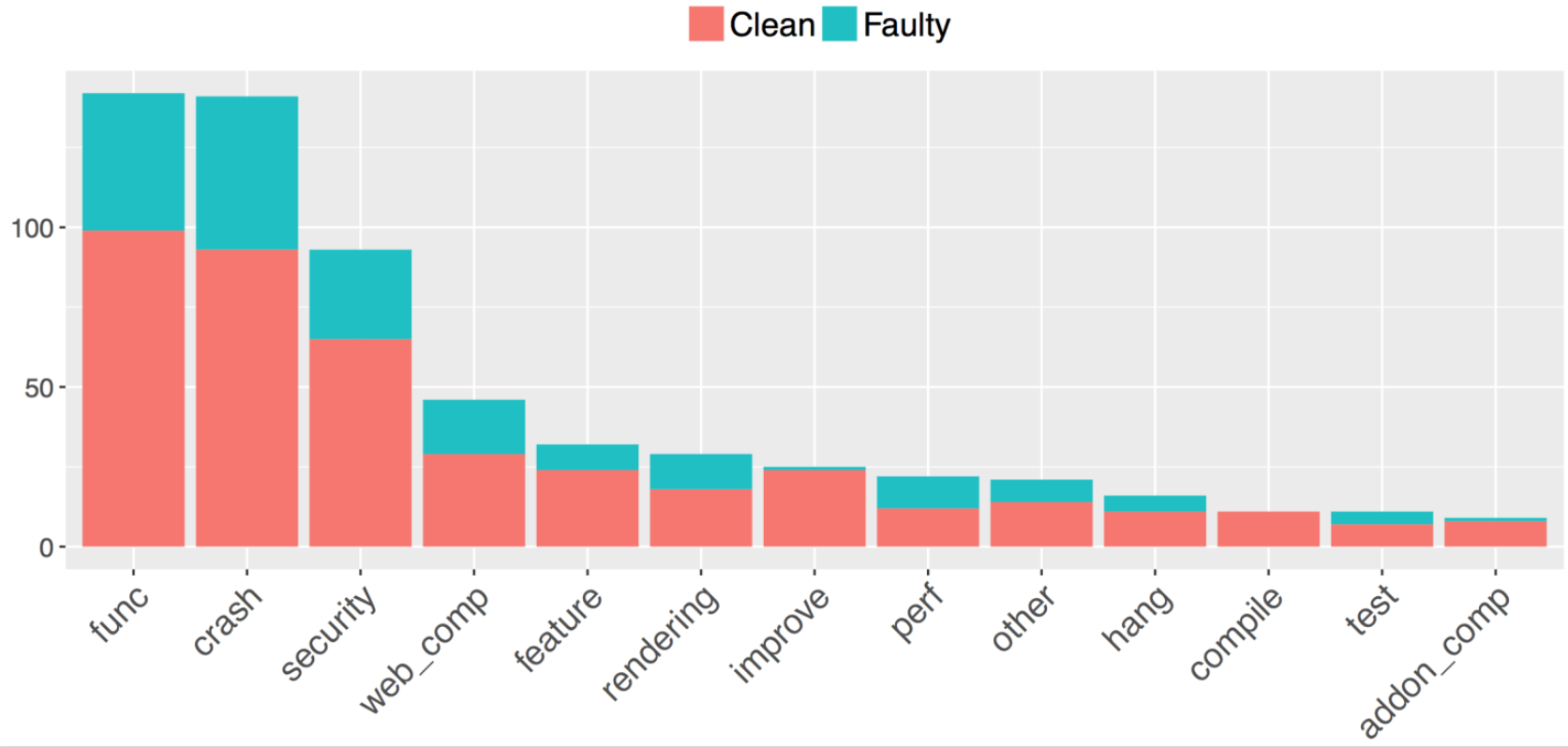
RQ1: What are the characteristics of patches that are uplifted?

For all of the Aurora, Beta, and Release channels, uplifted patches have significant shorter response delta (with small effect size) than other patches.

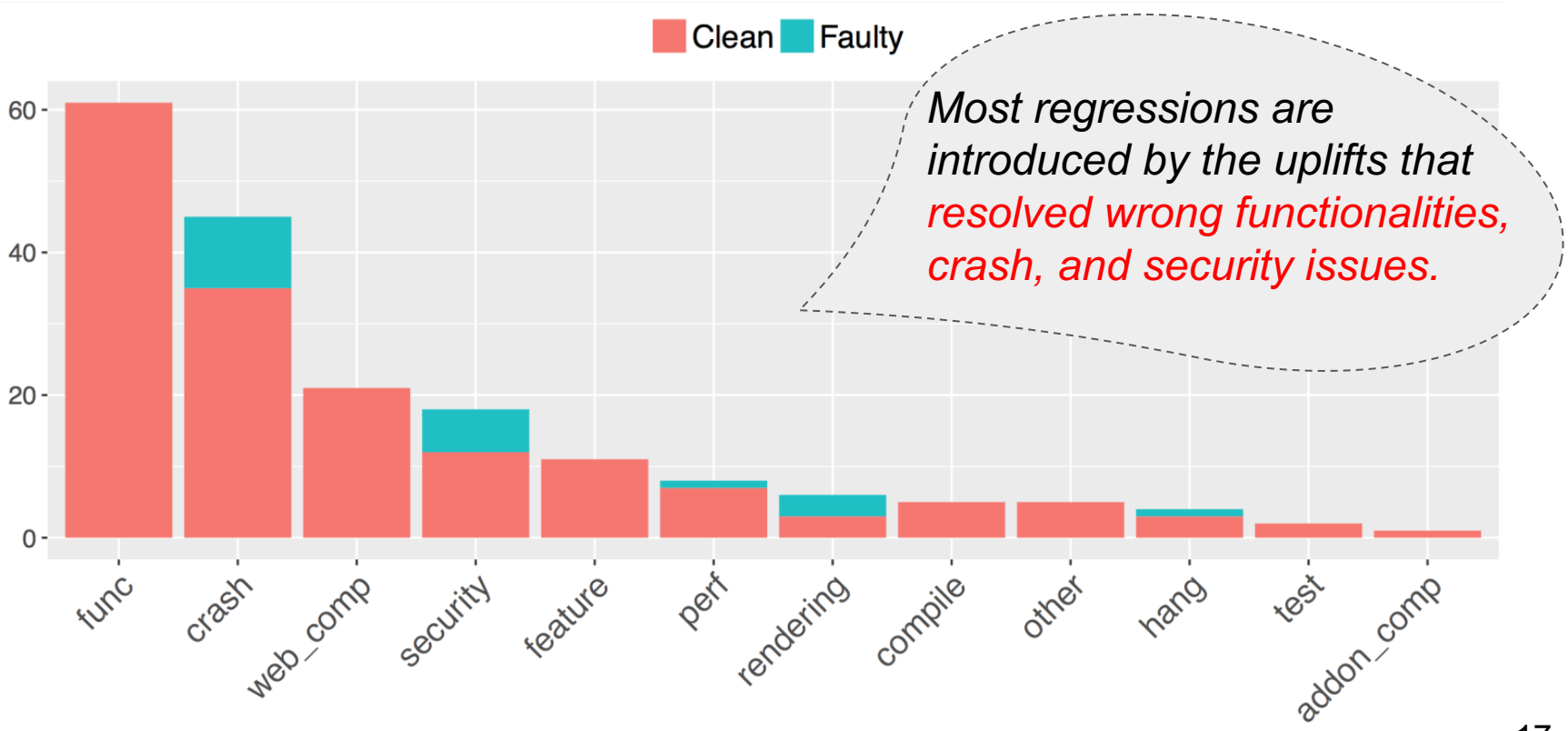
> Release managers' feedback: *"when I reject something, I won't make the call immediately. I will think about it before doing it, in case I change my mind or new facts are coming in the equation"*.

Other results are channel dependent.

Results - RQ1 (uplift reason, Beta channel)



Results - RQ1 (uplift reason, Release channel)



Results - RQ1 (release manager interview)

Q: Which factors do you take into account when deciding about an uplift?

A: *"the importance of the issue, risk associated with the patch, timing of the uplift in stabilization cycles, and verification of the path."*

Q: Are there differences in how you handle uplifts in different channels, and what are the differences?

A: *After the middle point of the Beta cycle, release managers only accept patches fixing high security issues, high volume crashes, severe recent regressions, severe performance issues or memory leaks.*

Results - RQ1 (release manager interview)

Q: How do you decide which developers you can trust?

A: Release managers mentioned:

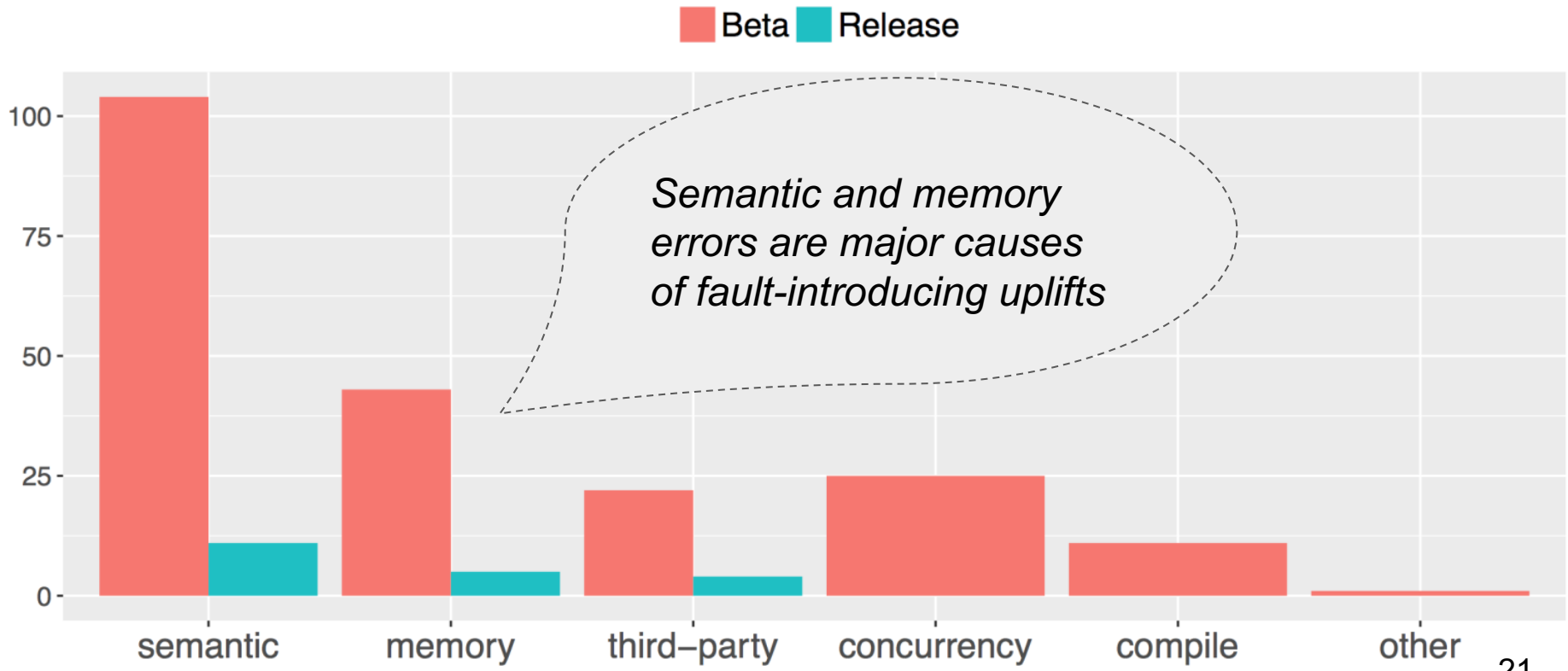
- *“When they seem really overconfident or aren’t telling me the whole story I lose some trust”*
- *“Some developers are taking a lot of risks, some other less and are super reactive to fix potential fallout”*

RQ2: What are the characteristics of uplifted patches that introduced faults in Mozilla Firefox?

- On all three channels, fault-introducing uplifts have significantly larger patch size (with a small effect size) than clean uplifts.

- Other results are channel dependent.

Results - RQ2 (root causes of faulty uplifts)



Results - RQ2 (release manager interview)

Q: What are the characteristics of the fault-introducing patches that you are not currently taking enough into account but could be considered in the future?

A: All the release managers agreed that it would be beneficial for them to have more detailed information about the complexity of the targeted patches and more information about the history of the components involved in these patches.

Summary

- Patch uplift allows to promote features or bug fixes directly from development channel to a stabilization channel.
- Patch uplift sometimes lead to faults.
- Software organizations can apply (or enhance their effort of using) static analysis tools to prevent memory-related faults.
- Reviewers and release managers should carefully inspect uplifts that address: wrong functionality, crashes, vulnerabilities, and incompatibilities.