The Effects of CI on Software Development

Introduction to the MSR 2017 Challenge

By: Omar Elazhary

Agenda

- Continuous Integration
- The Nitty-Gritty Details
  - Travis CI
  - TravisTorrent
- Possible Research Questions
  - The Effects of CI on Software
  - The Effects of CI on Developers
- Walkthrough: Test Regressions

Agenda

- **Continuous Integration**
- The Nitty-Gritty Details
  - Travis CI
  - TravisTorrent
- Possible Research Questions
  - The Effects of CI on Software
  - The Effects of CI on Developers
- Walkthrough: Test Regressions

Continuous Integration

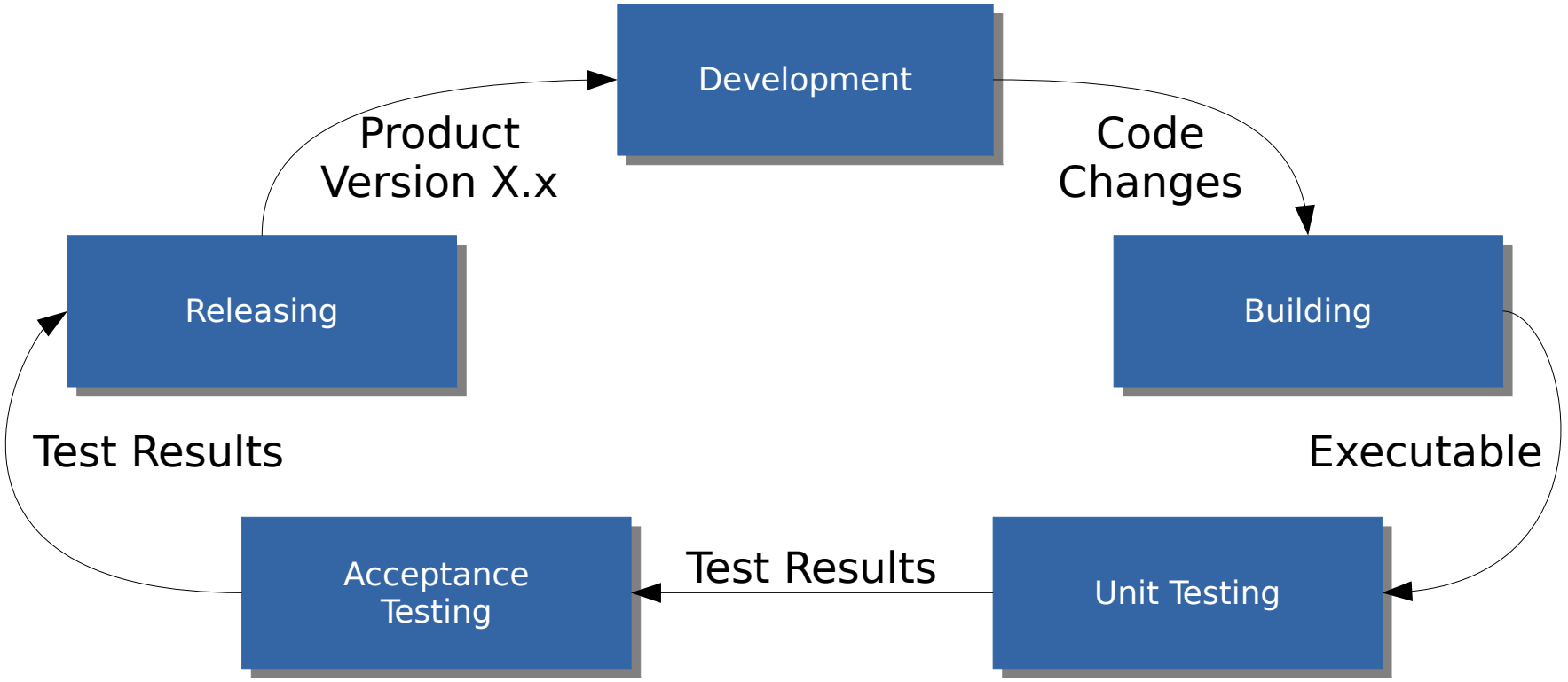
“A set of software engineering practices that speed up the delivery of software by decreasing integration times.”

Sean Stolberg

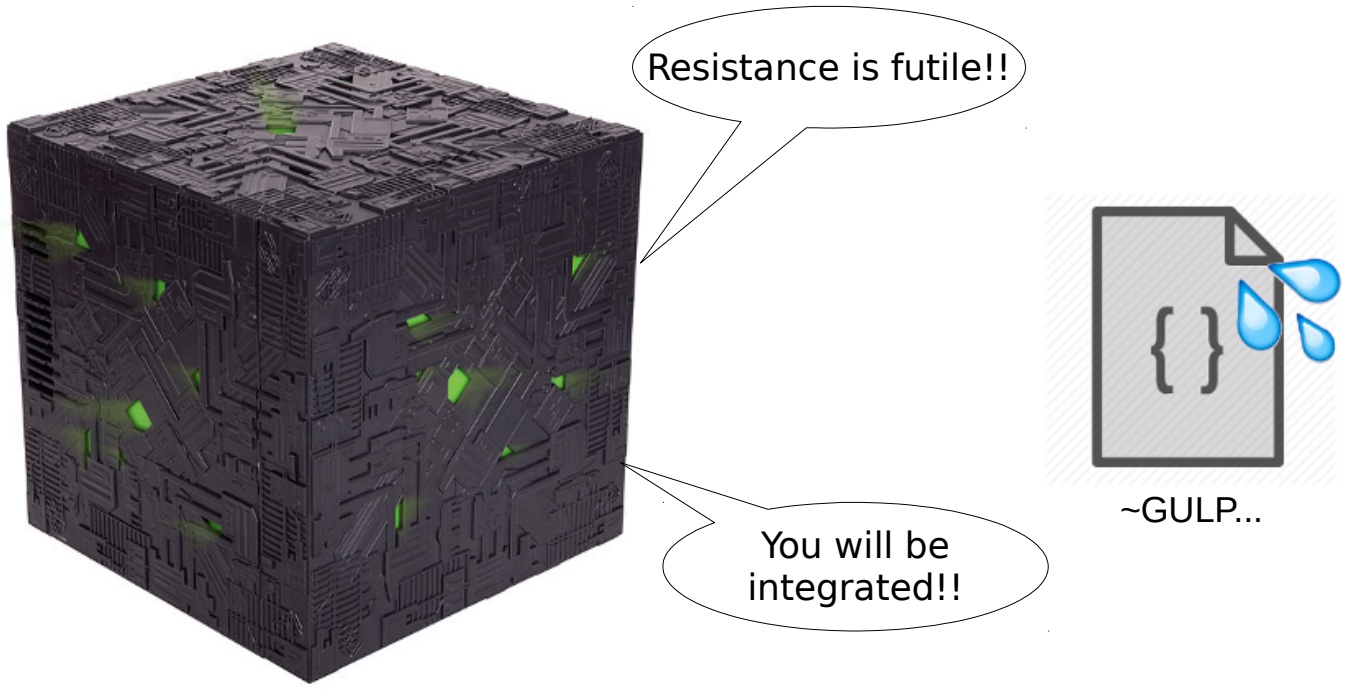
“A software development practice where members of a team integrate their work frequently, usually each person integrates at least daily.”

Martin Fowler

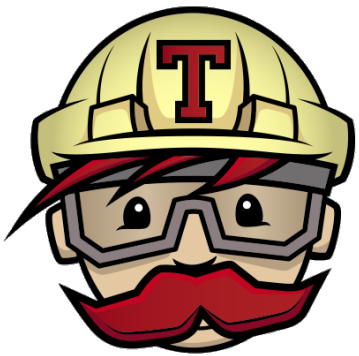
Integration Cycle



Develop. Build. Test. Repeat.



Various Tools



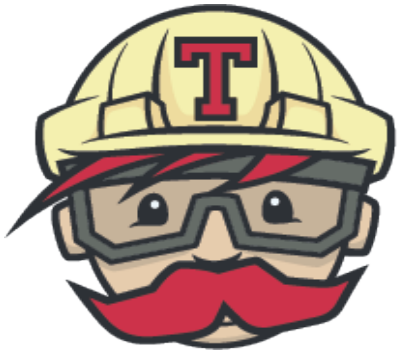
Agenda

- Continuous Integration
- **The Nitty-Gritty Details**
  - Travis CI
  - TravisTorrent
- Possible Research Questions
  - The Effects of CI on Software
  - The Effects of CI on Developers
- Walkthrough: Test Regressions

Agenda

- Continuous Integration
- **The Nitty-Gritty Details**
  - **Travis CI**
  - TravisTorrent
- Possible Research Questions
  - The Effects of CI on Software
  - The Effects of CI on Developers
- Walkthrough: Test Regressions

Travis CI



Travis CI

- Open source distributed build service
- Connected to github via web hooks
- Great for OSS developers!!

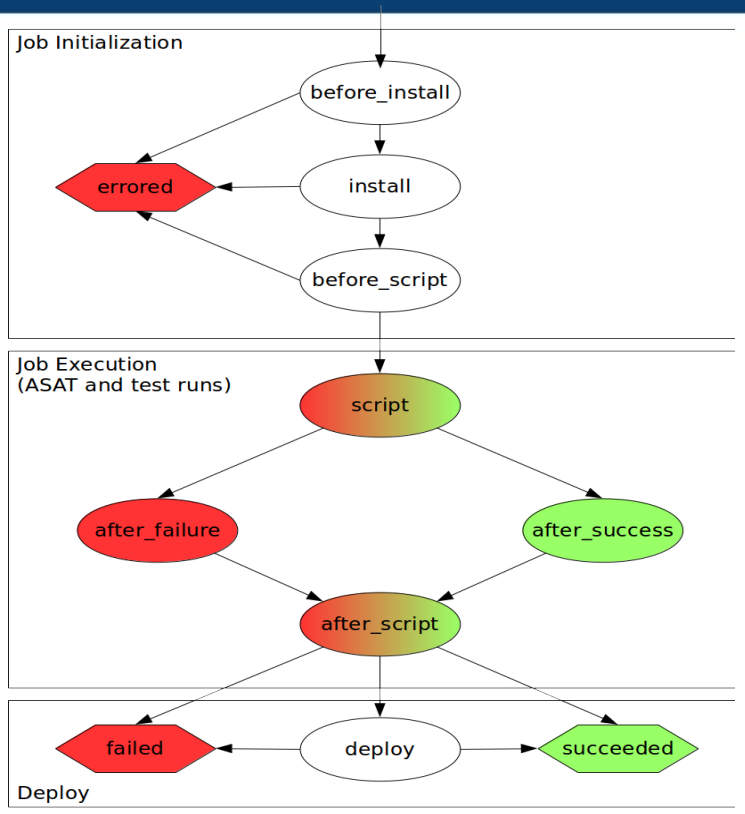
Build Setup

- Favors Newer Changes

IMPORTANT
FOR LATER!!

Build Life-cycle

- Build Status:
  - Started
  - Cancelled
  - Errored
  - Failed
  - Passed

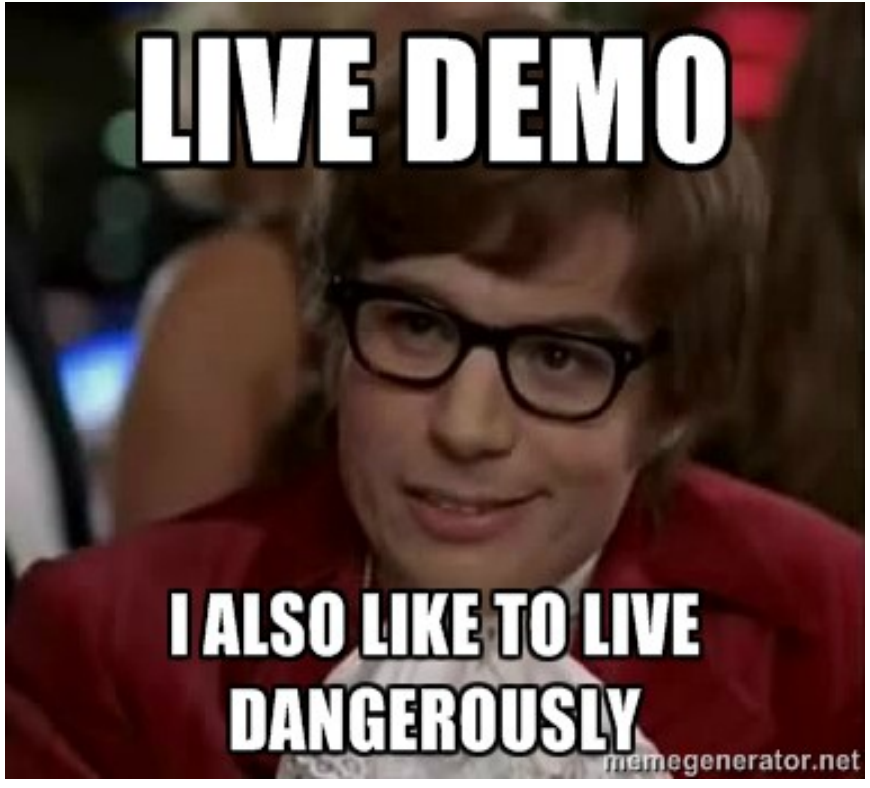


Build States from [3]

Other Important Information

- Build Environments
- Jobs

What else???



Agenda

- Continuous Integration
- **The Nitty-Gritty Details**
  - Travis CI
  - **TravisTorrent**
- Possible Research Questions
  - The Effects of CI on Software
  - The Effects of CI on Developers
- Walkthrough: Test Regressions

TravisTorrent

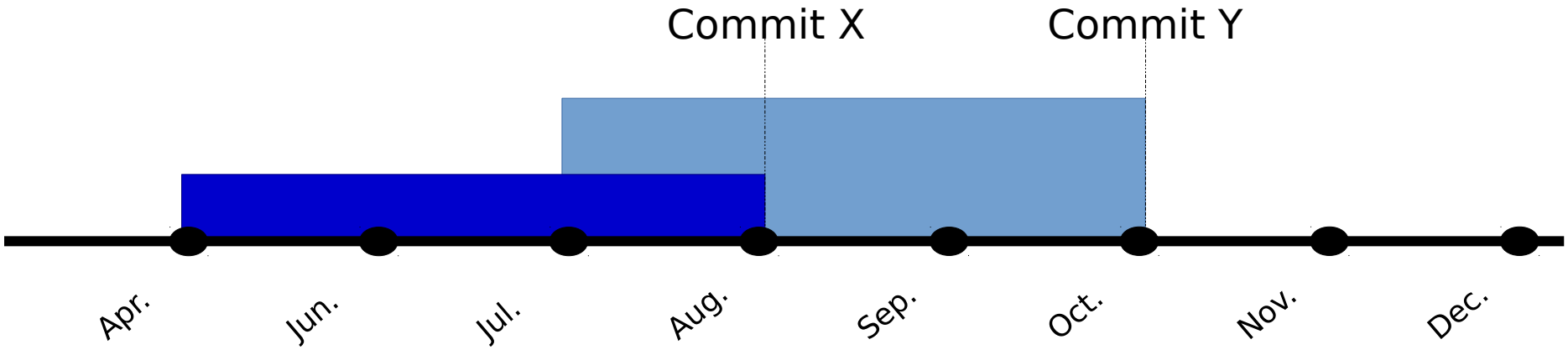
- A Collection of Stats about:
  - Github projects
  - Their builds on Travis CI
- Build Origin

A Typical Tuple

- Git commit
- Project name
- Is it a PR?
- Language
- Team size
- Number of comments
- Churns, additions and deletions
- File types
- Test stats
- Build ID
- Build status
- Build duration
- Job ID
- Test stats
  - Did tests run?
  - Numbers (ran, failed, ...etc.)
- Setup duration
- Pure build duration
- CI latency

Oddities

- The Time Dimension
  - Reflects 3 months



Oddities

- Granularity
  - Build?
  - Job?
- Numbers don’t add up:
  - 2640824 rows
  - 1800389 unique jobs

	gh_project_name text	git_branch text	tr_build_id bigint	tr_job_id bigint
219	abarisain/dmix	1_07_b1	16735991	16735990
220	abarisain/dmix	1_07_b1	16735991	16735992
221	abarisain/dmix	1_07_b2	19716617	19716614
222	abarisain/dmix	1_07_b2	19716617	19716618
223	abarisain/dmix	1_07_b3	19810733	19810604
224	abarisain/dmix	1_07_b3	19810733	19810734
225	abarisain/dmix	1_07_b4	20018068	20018069
226	abarisain/dmix	1_07_b4	20018068	20018076

Performance

- MySQL 5.7: Appalling performance (might need to investigate further)
- PostgreSQL is reasonable
- R can load the CSV file
  - In-Memory
  - What if I want to use GHTorrent as well?

Agenda

- Continuous Integration
- The Nitty-Gritty Details
  - Travis CI
  - TravisTorrent
- **Possible Research Questions**
  - The Effects of CI on Software
  - The Effects of CI on Developers
- Walkthrough: Test Regressions

Agenda

- Continuous Integration
- The Nitty-Gritty Details
  - Travis CI
  - TravisTorrent
- **Possible Research Questions**
  - **The Effects of CI on Software**
  - The Effects of CI on Developers
- Walkthrough: Test Regressions

The Effect of CI on Software

- Does the use of CI lead to higher-quality products?[6]
- Does CI lead to fewer test regressions?
- Do CI-enabled projects switch to a continuous delivery process, or do they release by hand?
- Do multiple integration environments lead to fewer defects?

Agenda

- Continuous Integration
- The Nitty-Gritty Details
  - Travis CI
  - TravisTorrent
- **Possible Research Questions**
  - The Effects of CI on Software
  - **The Effects of CI on Developers**
- Walkthrough: Test Regressions

The Effects of CI on Developers

- Does a broken build negatively affect developer productivity?
- Successful build as an objective[5][6] vs. breaking and fixing builds often.
- Do broken builds result in fewer outside contributions?

Agenda

- Continuous Integration
- The Nitty-Gritty Details
  - Travis CI
  - TravisTorrent
- Possible Research Questions
  - The Effects of CI on Software
  - The Effects of CI on Developers
- **Walkthrough: Test Regressions**

Walkthrough – Test Regressions

- TravisTorrent doesn’t really specify what type of test gets executed
- Test attributes include:
  - tr_tests_ok
  - tr_tests_fail
  - tr_tests_run
  - tr_tests_skipped
  - tr_failed_tests
  - tr_tests_ran
  - tr_tests_failed

- We need to identify test regressions on our own

Defining Test Regressions

- *“Selective retesting of a system or component to verify that modifications have not caused unintended effects and that the system or component still complies with its specified requirements.”* - as cited in the SWEBOK [4]
- So... how does this translate to TravisTorrent?
  - A test ran and failed due to new changes introduced by the latest commit
  - Previous build executed normally without problems
  - *Assumption:* New commit introduced functionality without tests

Detour – Granularity??

- So far we assume it’s at the job level
- **Problem:** We want the build level
- Why?
  - If a test fails, the entire build fails
  - We don’t have much information about tests on the job level
- **Solution:** Aggregate!!

Detour – Granularity??

```
-- Squashing the jobs together as a view:

create or replace view vw_tests_per_build as
    select gh_project_name,
        min(tr_build_number) as tr_build_number,
        tr_build_id,
        min(gh_lang) as gh_lang,
        avg(gh_src_churn) as gh_src_churn,
        avg(gh_sloc) as gh_sloc,
        avg(gh_test_churn) as gh_test_churn,
        avg(gh_tests_added) as gh_tests_added,
        avg(gh_tests_deleted) as gh_tests_deleted,
        avg(gh_test_lines_per_kloc) as gh_test_lines_per_kloc,
        avg(gh_test_cases_per_kloc) as gh_test_cases_per_kloc,
        avg(gh_asserts_cases_per_kloc) as gh_assert_cases_per_kloc,
        min(tr_status) as tr_status,
        sum(tr_tests_ok) as tr_tests_ok,
        sum(tr_tests_fail) as tr_tests_fail,
        sum(tr_tests_run) as tr_tests_run,
        sum(tr_tests_skipped) as tr_tests_skipped,
        trim(both from array_to_string(array_agg(tr_failed_tests), ' ')) as tr_failed_tests,
        bool_and(tr_tests_ran) as tr_tests_ran,
        bool_or(tr_tests_failed) as tr_tests_failed
    from travistorrent
    group by gh_project_name,
        tr_build_id
    order by gh_project_name;
```

Defining Test Regressions

- Build β_t
  - Has a status $S(\beta_t)$
  - Has source modification attributes $C_{src}(\beta_t)$
  - Has test modification attributes $C_{test}(\beta_t)$
  - Executed at time t

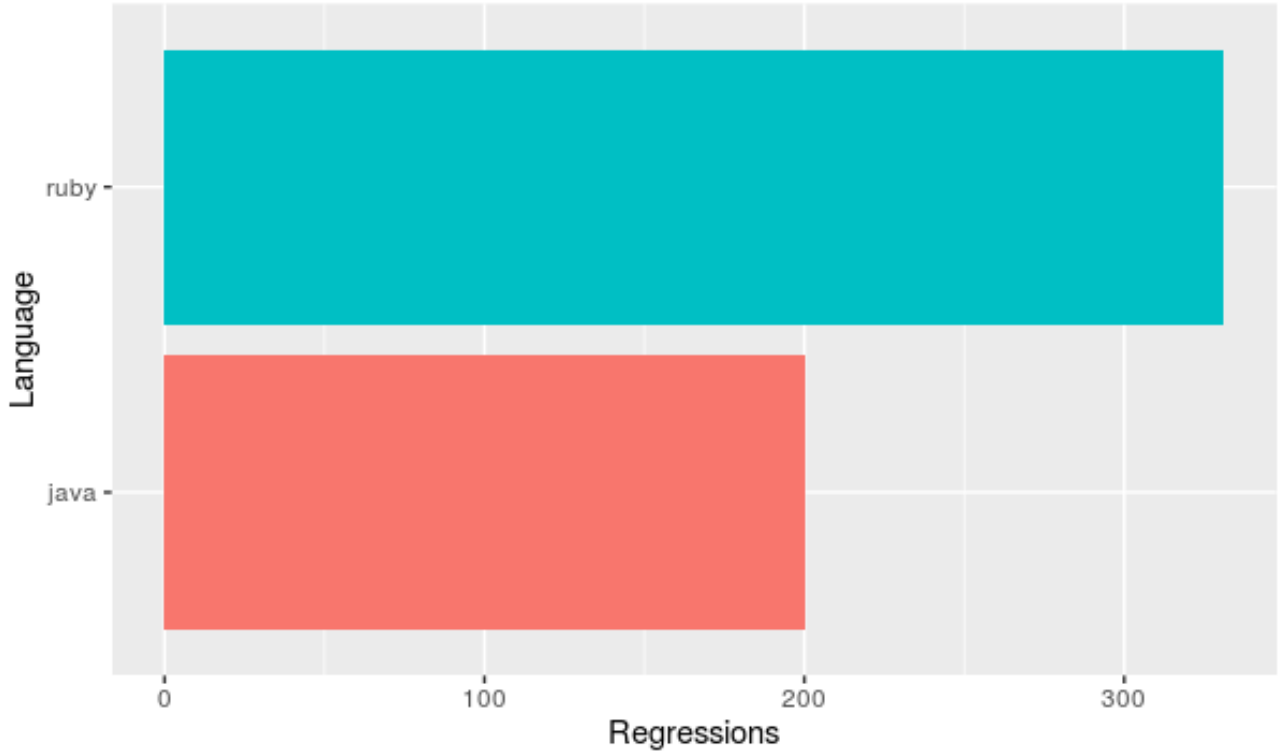
- We detect a failed regression test if:
  - $\wedge$ $S(\beta_t)$ = ‘Failed’
  - $\wedge$ $S(\beta_{t-1})$ = ‘Passed’
  - $\wedge$ $C_{src}(\beta_t) > 0$
  - $\wedge$ $C_{test}(\beta_t) = 0$

Defining Test Regressions

```
-- Detecting and gathering failed regression tests:

select curr_build.*
from    vw_tests_per_build as curr_build
    inner join
        vw_tests_per_build as prev_build
    on
        curr_build.gh_project_name = prev_build.gh_project_name
    and curr_build.tr_build_number = (prev_build.tr_build_number + 1)
    and curr_build.tr_status       = 'failed'
    and prev_build.tr_status       = 'passed'
where   (       curr_build.gh_src_churn             >  0
            or  curr_build.gh_sloc                  != prev_build.gh_sloc
        )
    and (       curr_build.gh_test_churn            = 0
            and curr_build.gh_tests_added           = 0
            and curr_build.gh_tests_deleted         = 0
            and curr_build.gh_test_lines_per_kloc   = prev_build.gh_test_lines_per_kloc
            and curr_build.gh_test_cases_per_kloc   = prev_build.gh_test_cases_per_kloc
            and curr_build.gh_assert_cases_per_kloc = prev_build.gh_assert_cases_per_kloc
            and curr_build.tr_tests_ran             = true
            and curr_build.tr_tests_failed          = true
        );
```

Which can lead to...



Next...

- Figure out how to define test regressions without TravisTorrent data for projects without CI
- Explore other aspects of TravisTorrent
- Combine TravisTorrent with GHTorrent for more comprehensive analyses
- ... The sky is the limit!!

Questions?

References

1. Stolberg, Sean. "Enabling agile testing through continuous integration." In *Agile Conference, 2009. AGILE'09.*, pp. 369-374. IEEE, 2009.
2. Fowler, Martin, and Matthew Foemmel. "Continuous integration." *Thought-Works) http://www. thoughtworks. com/Continuous Integration. Pdf* (2006): 122.
3. Beller, Moritz, Georgios Gousios, and Andy Zaidman. *Oops, my tests broke the build: An analysis of Travis CI builds with GitHub*. No. e1984v1. PeerJ Preprints, 2016.
4. Bourque, Pierre, and Richard E. Fairley. *Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0.* IEEE Computer Society Press, 2014.
5. Miller, Ade. "A hundred days of continuous integration." In *Agile, 2008. AGILE'08. Conference*, pp. 289-293. IEEE, 2008.
6. Holck, Jesper, and Niels Jørgensen. "Continuous integration and quality assurance: A case study of two open source projects." *Australasian Journal of Information Systems* 11, no. 1 (2003).

Thank You!!