

A

# **CONCEPTUAL REPLICATION OF CONTINUOUS INTEGRATION PAIN POINTS**

A  
CONCEPTUAL REPLICATION  
OF  
CONTINUOUS INTEGRATION  
PAIN POINTS



**DAVID  
WIDDER**



MICHAEL  
HILTON



CHRISTIAN  
KÄSTNER



BOGDAN  
VASILESCU

# **WHAT WILL I TALK ABOUT?**

# **WHAT WILL I TALK ABOUT?**

## **I. WHAT IS CONTINUOUS INTEGRATION? WHY DO PEOPLE USE IT?**

# **WHAT WILL I TALK ABOUT?**

- I. WHAT IS CONTINUOUS INTEGRATION?  
WHY DO PEOPLE USE IT?**
- 2. WHY DO A REPLICATION?**

# **WHAT WILL I TALK ABOUT?**

- I. WHAT IS CONTINUOUS INTEGRATION?  
WHY DO PEOPLE USE IT?**
- 2. WHY DO A REPLICATION?**
- 3. OUR MIXED METHOD**

# **WHAT WILL I TALK ABOUT?**

- 1. WHAT IS CONTINUOUS INTEGRATION?  
WHY DO PEOPLE USE IT?**
- 2. WHY DO A REPLICATION?**
- 3. OUR MIXED METHOD**
- 4. HIGHLIGHTS OF FINDINGS:  
WHAT REPLICATES + WHAT DOESN'T?**

# **WHAT IS CONTINUOUS INTEGRATION?**

# **WHAT IS CONTINUOUS INTEGRATION?**



# WHAT IS CONTINUOUS INTEGRATION?



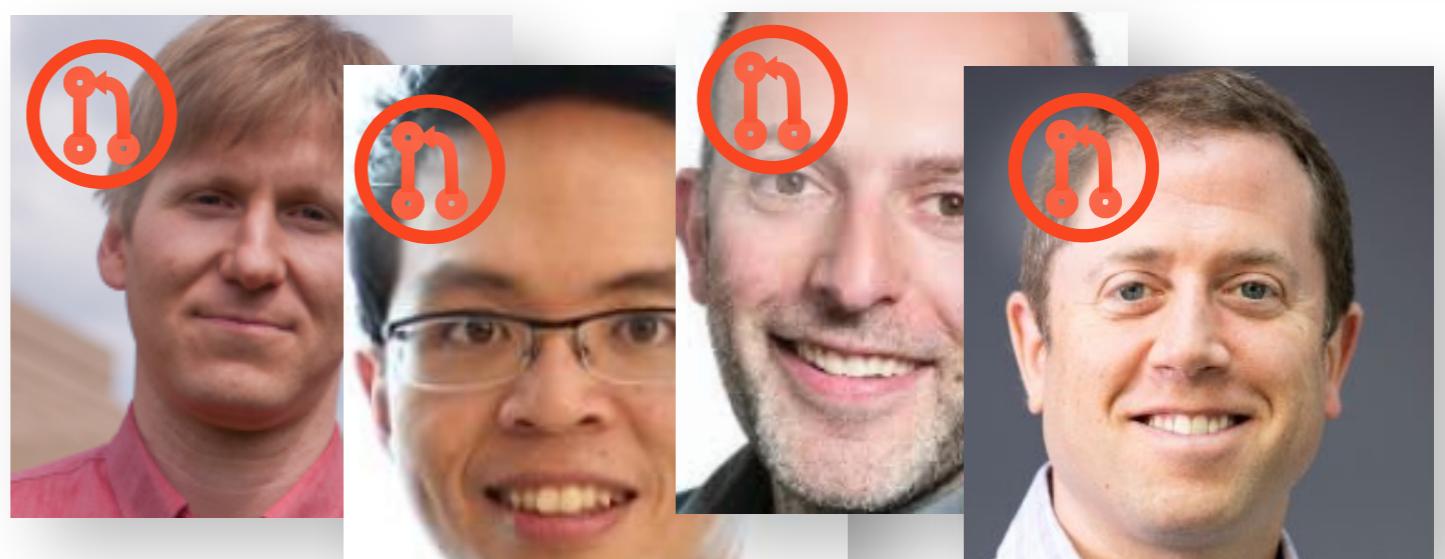
# WHAT IS CONTINUOUS INTEGRATION?



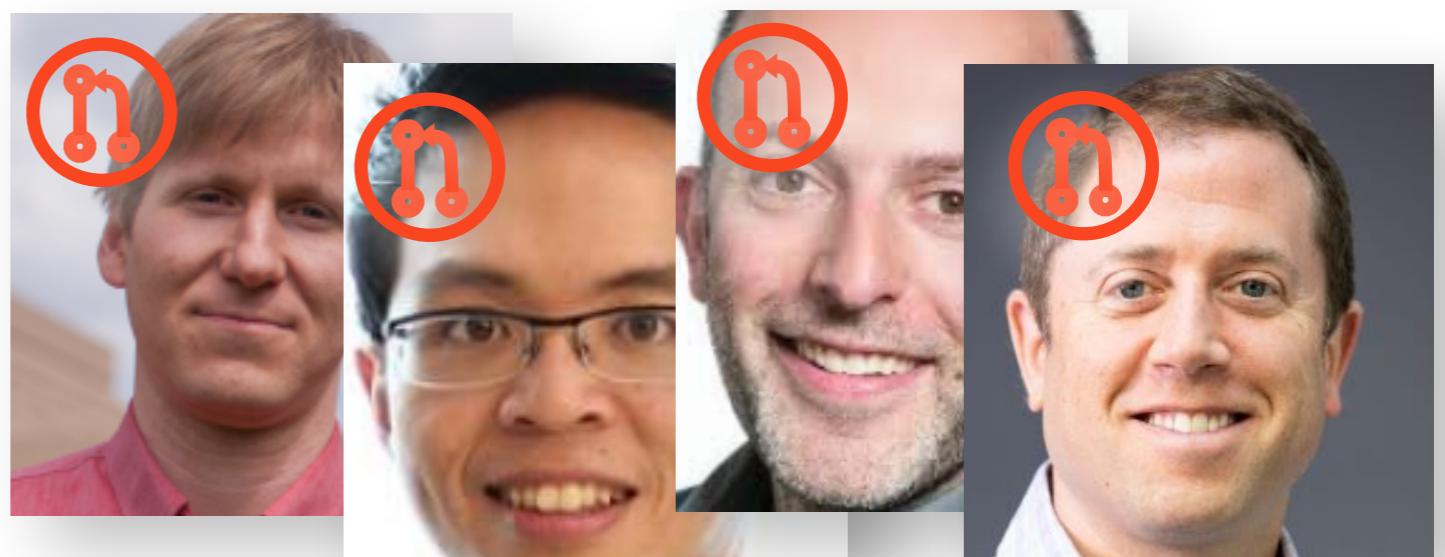
# WHAT IS CONTINUOUS INTEGRATION?



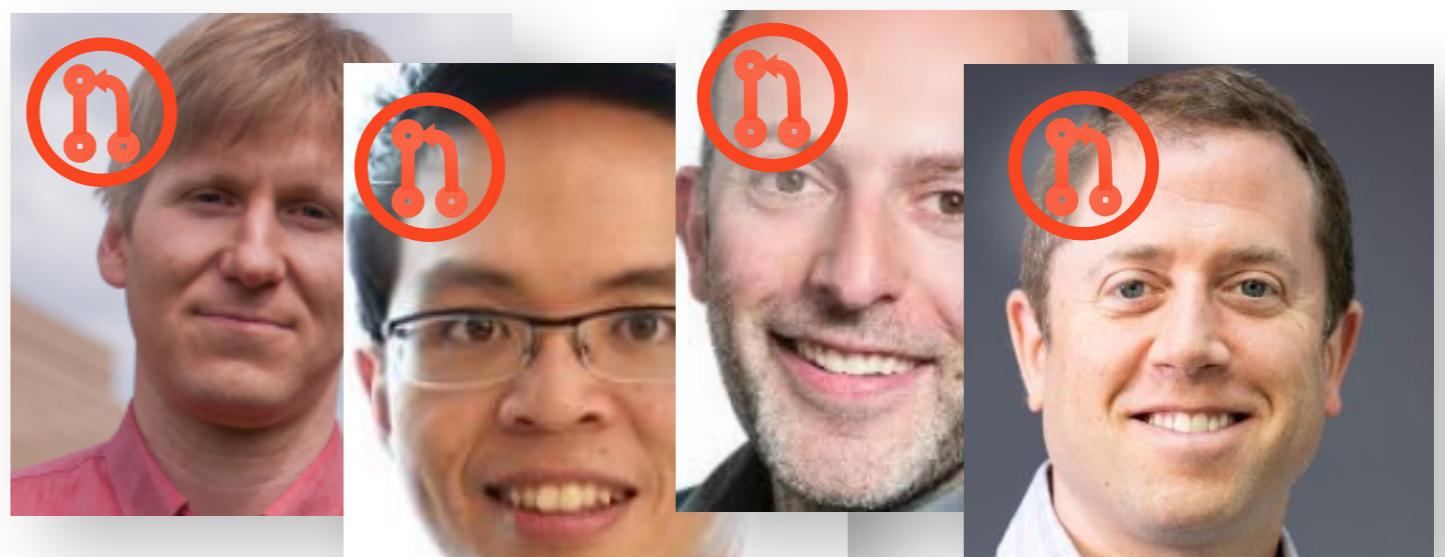
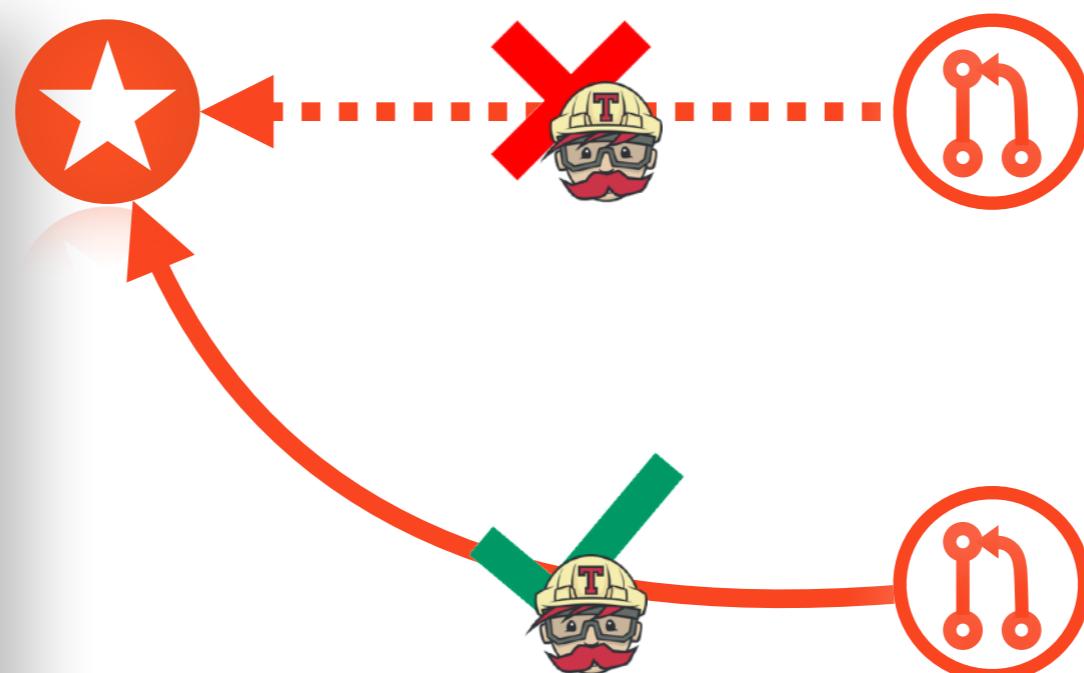
# WHAT IS CONTINUOUS INTEGRATION?



# WHAT IS CONTINUOUS INTEGRATION?



# WHAT IS CONTINUOUS INTEGRATION?



# **CI IS A BEST PRACTICE**



# CI IS A BEST PRACTICE



**6 REASONS TO USE CI**

One of the main benefits of unit testing is a shorter feedback loop. The less time passes between a developer making an update and discovering that it's not ready and cleaning it up, the less likely it is that the problem will make it to production. Frequent execution of all user tests leaves that interval short. The more automated your tests, the easier it is to run them and with that, the more likely it is that the developers will actually execute them. Always remember:

Unit tests can only provide value, if they are executed often.

**WHY WE NEED CI**

The Problem

Fist, we will take a look at the source of the problem, which lies in the software development cycle. Next, we will cover some of the change conflicts that can take place during that process; and finally we will explore the main factors that can make those problems escalate, followed by an explanation of how continuous integration solves those issues.

The Source of the Problem

Let's take a look at what a traditional software development cycle looks like. Each developer

**WHAT IS CI AND WHY USE IT?**

First of all, what does CI stand for? It means Continuous Integration. Continuous Integration is at its root a practice where you merge the work of all your developers several times a day. Each integration is verified by your automated tests and built (when it needs to be, for example when a developer mostly).

These days however, CI is usually referring to the server part of the process, where you run your tests and build your application automatically.

Why use CI?

CI is really important to move fast, or just to move with the confidence you are not breaking anything. It is easy to break software when you write untested code, and I think it is also as easy to do the same with CI. That is why CI will make sure your code is good. If you have tests, CI will verify that after every commit, all the tests in your app will still pass. This means that you can commit by knowing that nothing got broken. Rather than focus on a regular release cycle, you can now everyone on your team release their code whenever it's ready. We'll cover the ins and outs of continuous deployment in great detail in later posts, but essentially we (and many others) believe that being able to ship early and often is a key competitive advantage in the technology industry.

A CI server will usually tell you if it is ok to "green" a build or if it broke it, giving a "red" build. You don't want to break the build, so you obviously need a CI server.

Rainforest now provides a nice integration with CI. Here are examples of red and green builds:

**WHAT IS CI AND WHY DO YOU NEED IT?**

While not every project is destined to achieve great success, there are software development best practices that can dramatically increase the chance of success of a project and make the development more pleasurable experience. One of such practices is Continuous integration.

It was originally adopted as the extreme programming practice and its main purpose is to prevent integration problems and to avoid "Integration hell".

So let's learn what Continuous Integration is and how it can help you become a better software developer.

**7 REASONS YOU SHOULD USE CI**

1. Run your tests in the real world
2. Increase your code coverage

**WHY CI IS IMPORTANT**

Matthew Setter is a developer and technical writer. He creates web-based applications and technical content that engage developers with platforms, technologies, applications, and tools.

Follow us

Join over 100,000 subscribers to our newsletter

**LOTS OF RESEARCH  
SHOWING CI BENEFITS**



# LOTS OF RESEARCH SHOWING CI BENEFITS

HELPS  
CATCH  
BUGS  
FASTER

RELEASE  
TWICE AS  
OFTEN



FEWER PULL  
REQUESTS  
REJECTED



HIGHER PULL  
REQUEST  
THROUGHPUT

ENCOURAGES  
PEOPLE TO  
WRITE TESTS

**ALSO, LOTS OF  
RESEARCH SHOWING  
CI PAIN POINTS**



# ALSO, LOTS OF RESEARCH SHOWING C| PAIN POINTS



# HARD TO CONFIGURE

# UNSUPPORTED FEATURES

# INSECURE

OO MANY  
OPTIONS

LONG  
BUILDS



**WE BELIEVE IT IS  
TIME TO REVIEW  
AND REPLICATE  
CI PAIN POINTS**

# **WHY REVIEW AND REPLICATE CI PAIN POINTS?**



# **WHY REVIEW AND REPLICATE CI PAIN POINTS?**

- EVALUATE & SYNTHESIZE  
PAST RESEARCH**



# **WHY REVIEW AND REPLICATE CI PAIN POINTS?**



- EVALUATE & SYNTHESIZE  
PAST RESEARCH**
- PROVIDE REPLICATED  
GUIDANCE TO  
PRACTITIONERS**

# **WHY REVIEW AND REPLICATE CI PAIN POINTS?**



- EVALUATE & SYNTHESIZE  
PAST RESEARCH**
- PROVIDE REPLICATED  
GUIDANCE TO  
PRACTITIONERS**
- FOCUS FUTURE RESEARCH  
ON AREAS OF UNCERTAINTY**

WHAT'S A  
**CONCEPTUAL  
REPLICATION?**

**CURRENT  
CI USERS**



**SWITCHERS &  
ABANDONERS**

# WHAT'S A **CONCEPTUAL REPLICATION?**

**CURRENT  
CI USERS**



**SWITCHERS &  
ABANDONERS**



# **WHY STUDY LEAVERS INSTEAD OF CURRENT USERS?**

# WHY STUDY LEAVERS INSTEAD OF CURRENT USERS?



# CURRENT CI USERS INDUSTRY & OSS



# LIT REVIEW

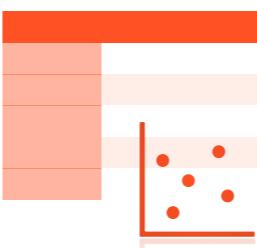


# PAIN POINTS

- # • TRAVISCI LEAVERS • OSS



# **SURVEY & INTERVIEWS**



# LOGISTIC REGRESSIONS



# PAIN POINTS REPLICATED (OR NOT)

# CURRENT CI USERS INDUSTRY & OSS



# LIT REVIEW

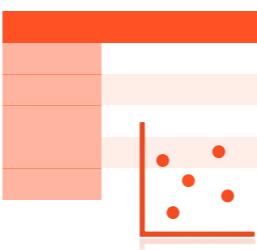


# PAIN POINTS

- # • TRAVISCI LEAVERS • OSS



# **SURVEY & INTERVIEWS**



# LOGISTIC REGRESSIONS

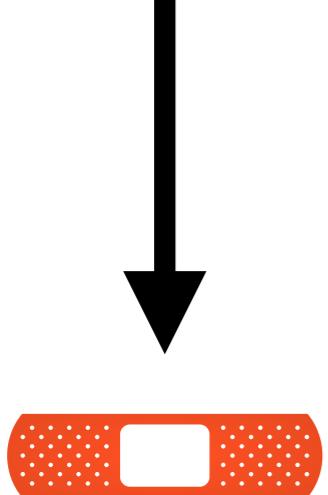


# PAIN POINTS REPLICATED (OR NOT)

# CURRENT CI USERS INDUSTRY & OSS



# LIT REVIEW

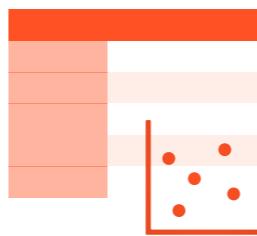


# PAIN POINTS

- # • TRAVISCI LEAVERS • OSS



# **SURVEY & INTERVIEWS**



# LOGISTIC REGRESSIONS



# PAIN POINTS REPLICATED (OR NOT)

# CURRENT CI USERS INDUSTRY & OSS



# LIT REVIEW

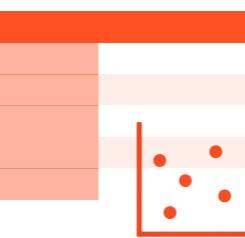


# PAIN POINTS

- # • TRAVISCI LEAVERS • OSS



# **SURVEY & INTERVIEWS**



# LOGISTIC REGRESSIONS



# PAIN POINTS REPLICATED (OR NOT)

# CURRENT CI USERS INDUSTRY & OSS



# LIT REVIEW

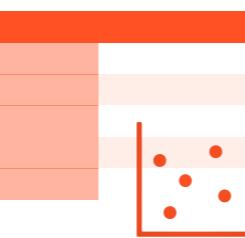


# PAIN POINTS

- # • TRAVISCI LEAVERS • OSS



# **SURVEY & INTERVIEWS**



# LOGISTIC REGRESSIONS



# PAIN POINTS REPLICATED (OR NOT)

# CURRENT CI USERS INDUSTRY & OSS

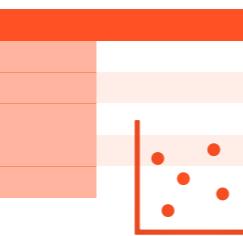
- # • TRAVISCI LEAVERS • OSS



# LIT REVIEW



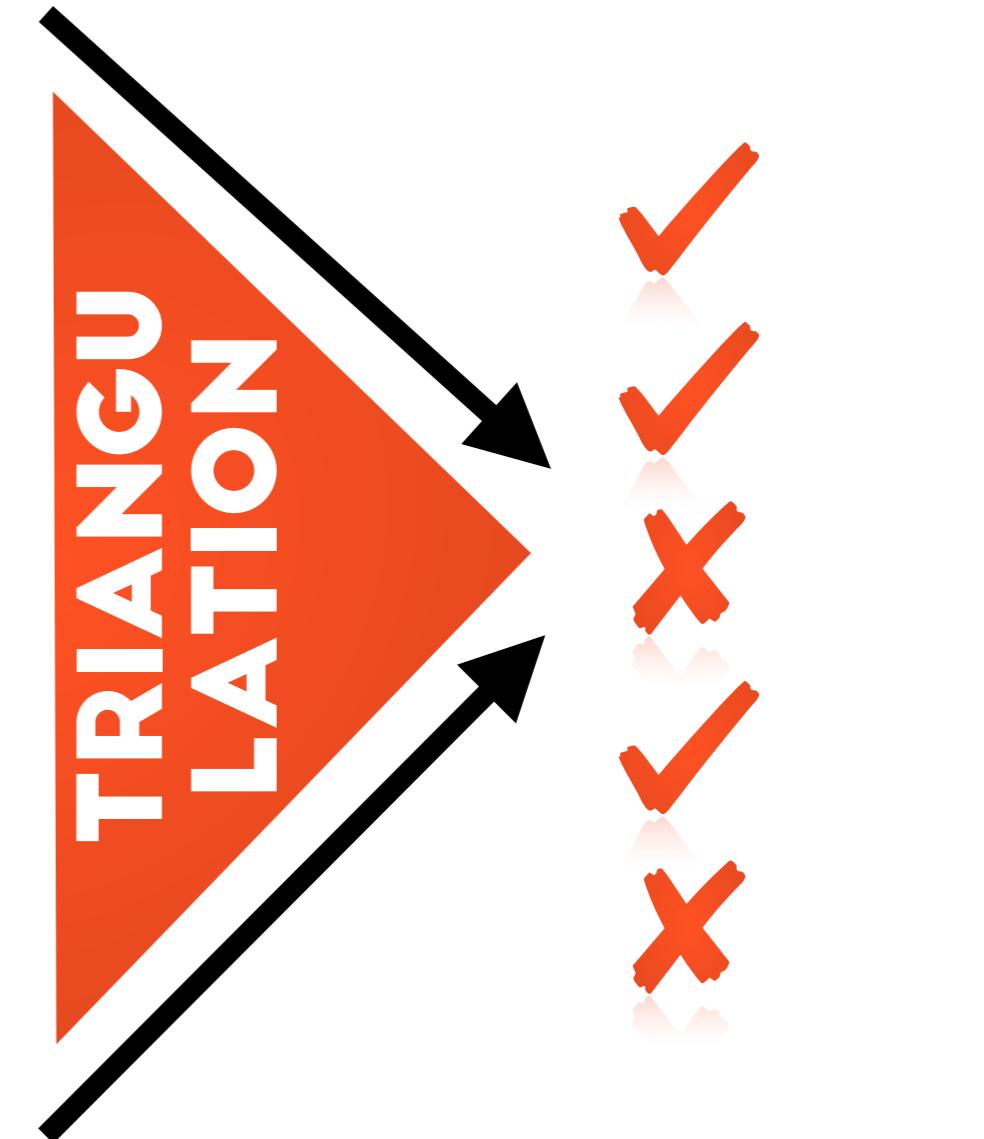
# PAIN POINTS



# LOGISTIC REGRESSIONS



# **SURVEY & INTERVIEWS**



# PAIN POINTS REPLICATED (OR NOT)



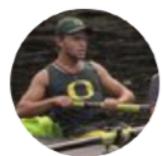
**LIT REVIEW  
37 PAPERS  
35 PAIN POINTS**



## **132 SURVEY RESPONSES**

“Why did your project stop using Travis CI, and what has the CI situation been like since then?”

David  
@davidthe



David Widder  
@davidthewid

Edit profile

✳️✳️✳️ Science in progress! ✳️✳️✳️  
Afternoon spent card sorting ~140 survey  
responses w @michaelhilton  
📸: @b\_vasilescu #mixedmethods



elhilton

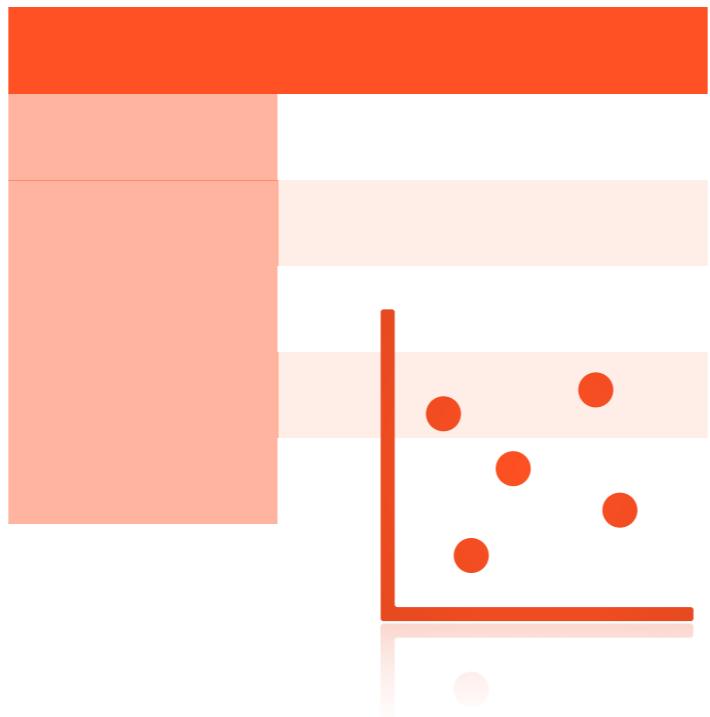


**132 SURVEY  
RESPONSES +  
12 INTERVIEWS**



**132 SURVEY  
RESPONSES +  
12 INTERVIEWS**

# WHY LOGISTIC REGRESSION?



**PEOPLE WHO LEAVE CI  
VS  
THOSE WHO DON'T**

# ABANDONMENT VS SWITCHING



Commits on Aug 7, 2019

[maven-release-plugin] prepare for next development iteration

kohsuke committed 4 days ago X

All checks have failed

1 failing check

[maven-release-plugin] prepare re...

kohsuke committed 4 days ago

X continuous-integration/jenkins/branch — This c... Details

[maven-release-plugin] prepare for next development iteration

# ABANDONMENT VS SWITCHING



Commits on Aug 7, 2019

[maven-release-plugin] prepare for next development iteration

kohsuke committed 4 days ago X

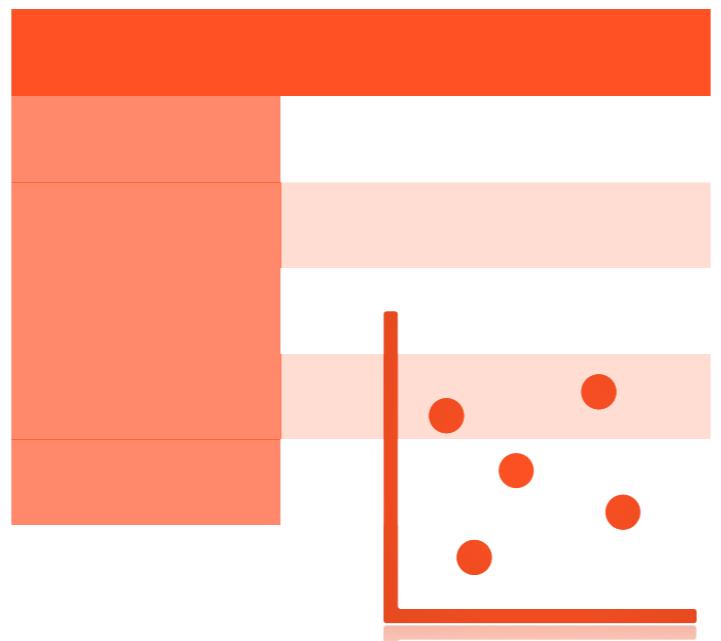
All checks have failed  
1 failing check

[maven-release-plugin] prepare re

kohsuke committed 4 days ago X continuous-integration/jenkins/branch — This c... [Details](#)

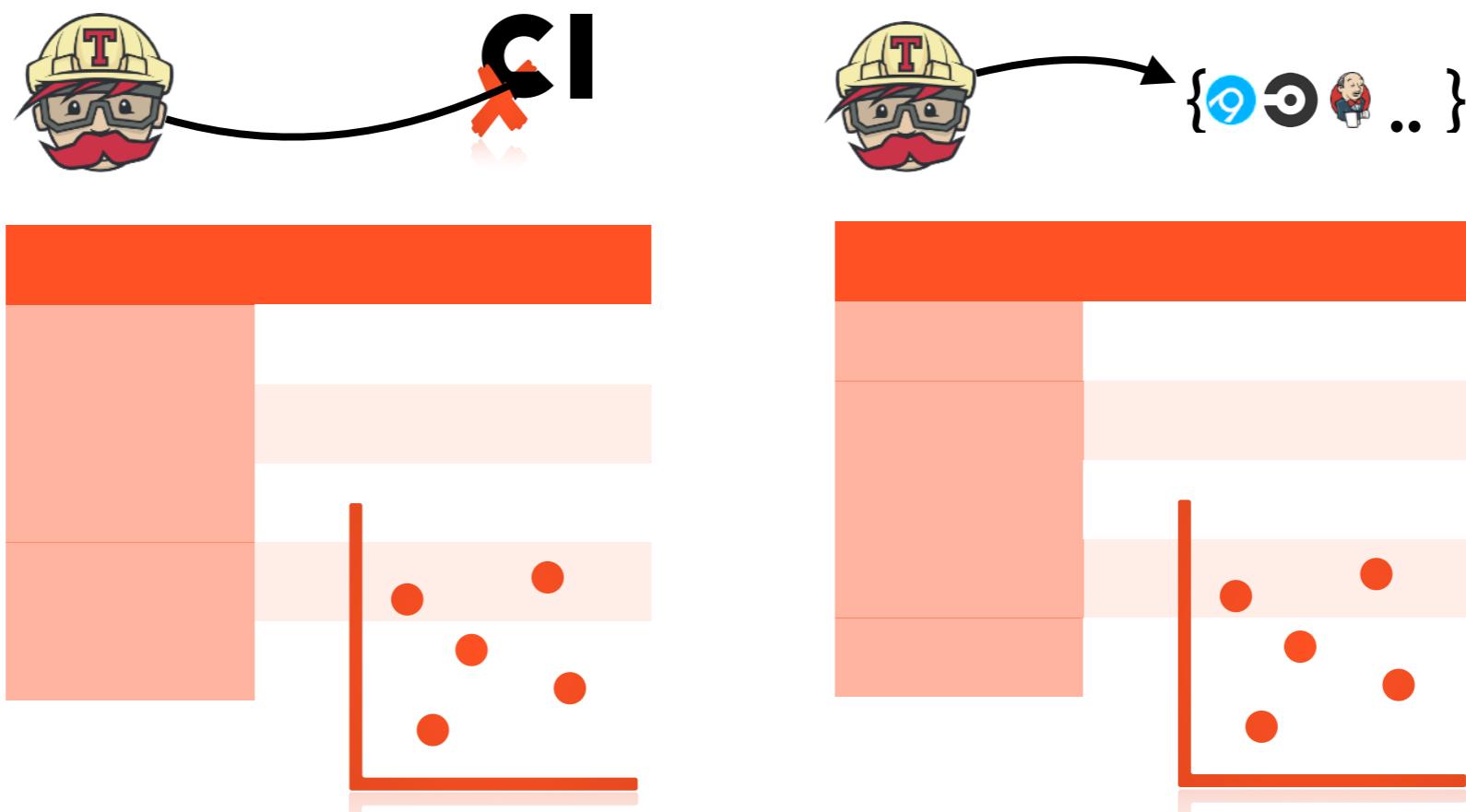
[maven-release-plugin] prepare for next development iteration

# ABANDONMENT VS SWITCHING



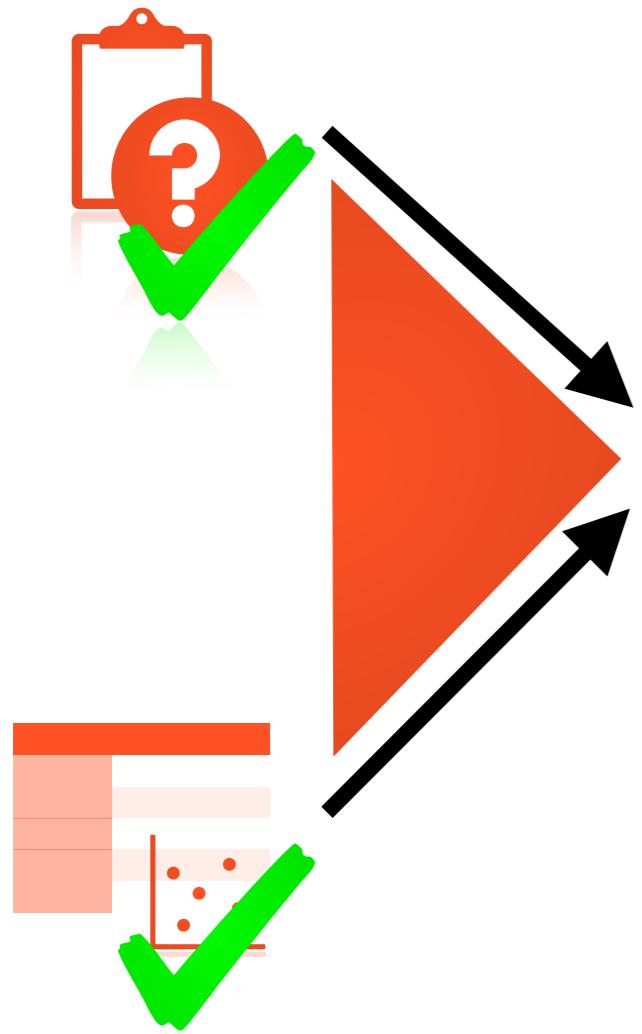
**2 LOGISTIC  
REGRESSIONS,  
6,239 REPOS**

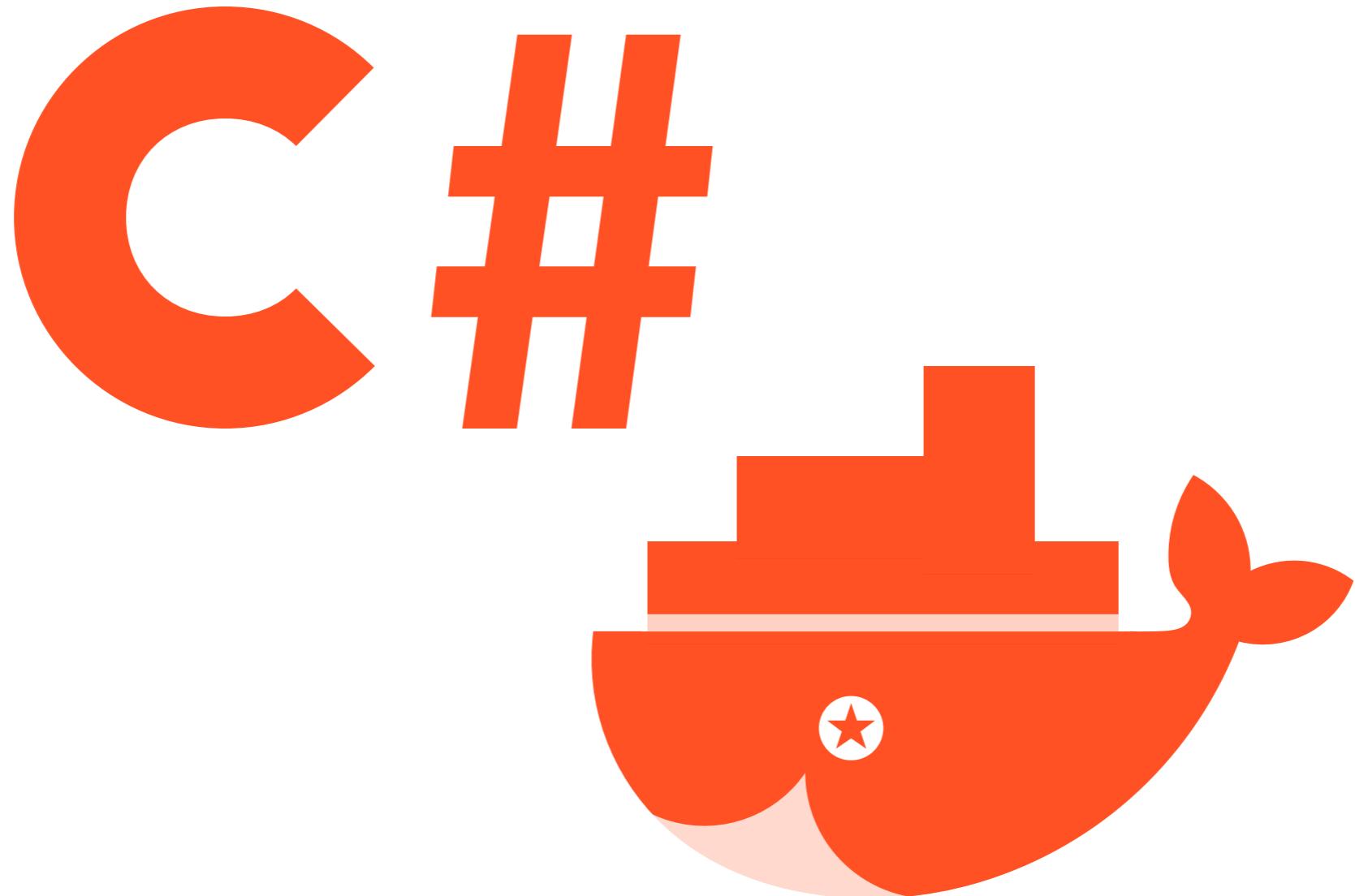
# ABANDONMENT VS SWITCHING



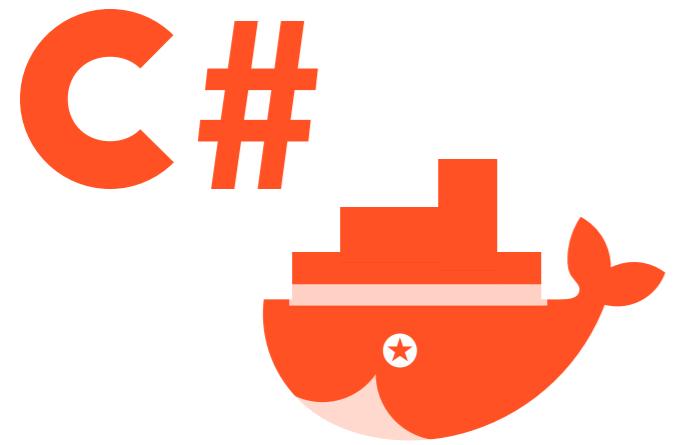
**2 LOGISTIC  
REGRESSIONS,  
6,239 REPOS**

# TRIANGULATED RESULTS

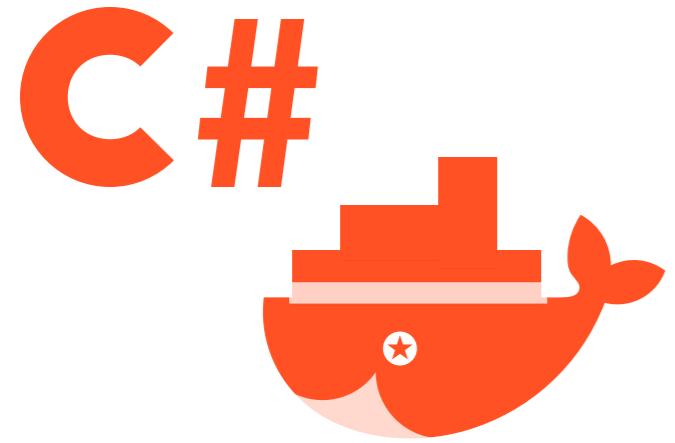




# UNSUPPORTED TECHNOLOGIES

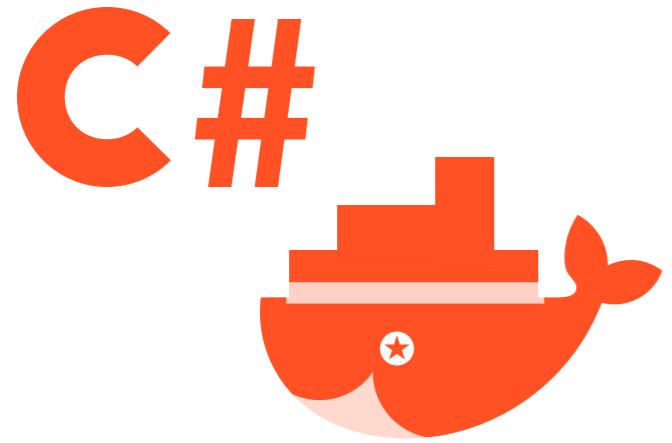


# UNSUPPORTED TECHNOLOGIES



# UNSUPPORTED TECHNOLOGIES

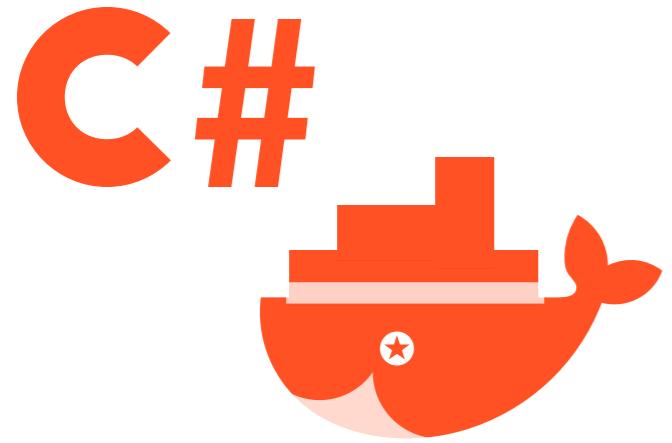
“I switched to Wercker, a container based CI pipeline, which means it can execute any scripts with much more flexibility.” (P89)



# UNSUPPORTED TECHNOLOGIES

“I switched to Wercker, a container based CI pipeline, which means it can execute any scripts with much more flexibility.” (P89)

**NEEDING DOCKER:**  
**9.5X INCREASE IN SWITCHING,**  
**5.25X INCREASE IN ABANDONING**

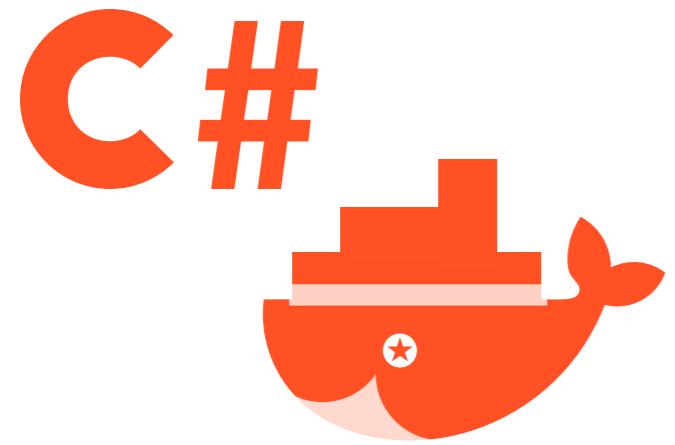


# UNSUPPORTED TECHNOLOGIES

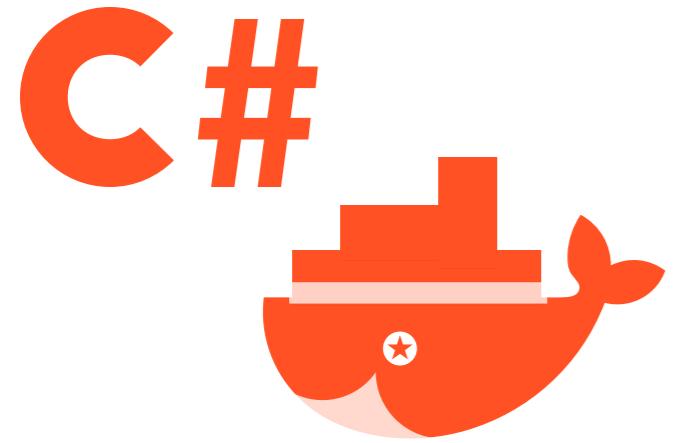
“I switched to Wercker, a container based CI pipeline, which means it can execute any scripts with much more flexibility.” (P89)

**NEEDING DOCKER:**  
**9.5X INCREASE IN SWITCHING,**  
**5.25X INCREASE IN ABANDONING**

**LACKING LANGUAGE SUPPORT:**  
**1.5X INCREASE IN SWITCHING AND**  
**ABANDONING**



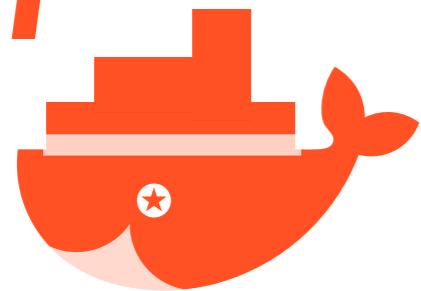
# UNSUPPORTED TECHNOLOGIES



# UNSUPPORTED TECHNOLOGIES

**NEEDING DOCKER AFFECTS  
NEW TRAVISCI USERS ONLY**

C#



# UNSUPPORTED TECHNOLOGIES

NEEDING DOCKER AFFECTS  
**NEW TRAVISCI USERS ONLY**

WHEN CAN PEOPLE HACK  
THROUGH TECHNOLOGY  
SUPPORT CHALLENGES, AND  
WHEN ARE THE **NOT** ABLE TO?



**LACK  
OF TESTS**



# LACK OF TESTS



# LACK OF TESTS

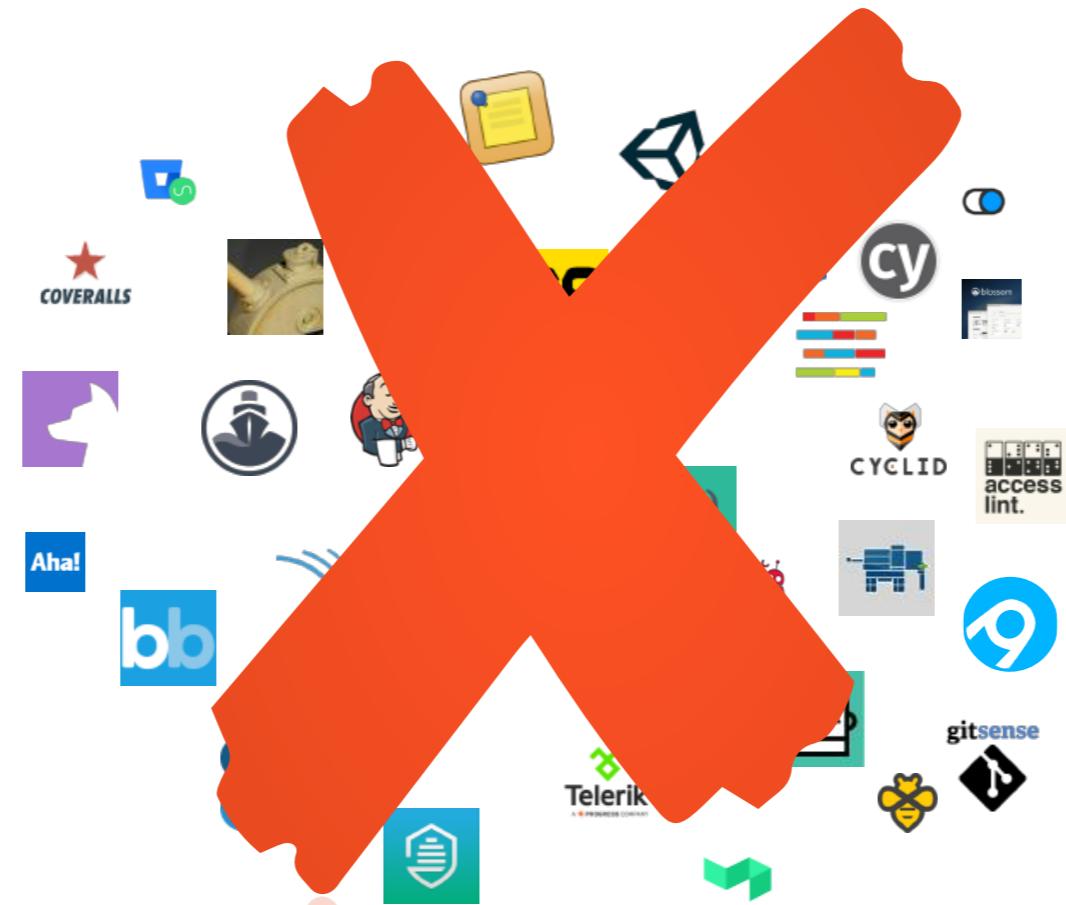
“The goal [of adopting Travis] was to ‘force’ myself to add some real tests (to have green Travis badge again!) but this failed so far :)" (P111)



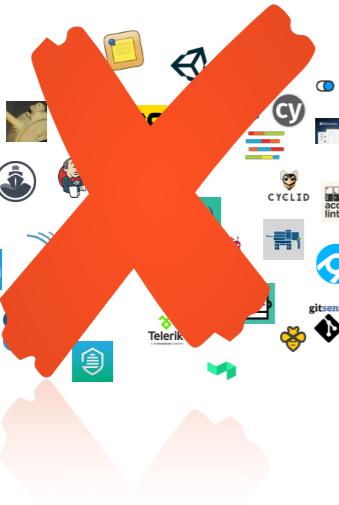
# LACK OF TESTS

“The goal [of adopting Travis] was to ‘force’ myself to add some real tests (to have green Travis badge again!) but this failed so far :)" (P111)

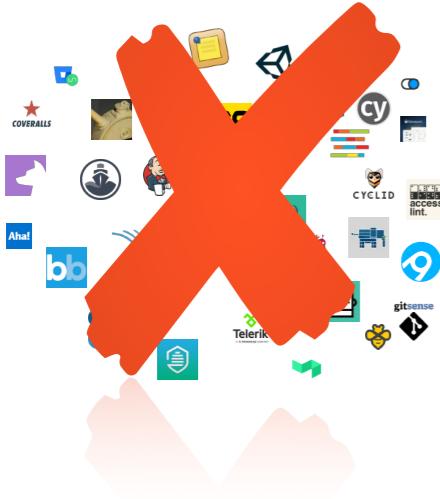
**1% INCREASE IN TESTS =  
16% LOWER CHANCE OF ABANDONING  
(NO EFFECT ON SWITCHING)**



# CI CONSISTENCY

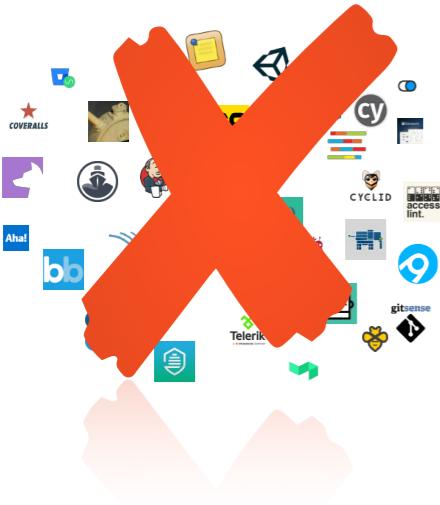


# CI CONSISTENCY



# CI CONSISTENCY

“I had some open source projects running in TravisCI and some in CircleCI. I just wanted to consolidate the project to one place and I’m sorry to say that at that time TravisCI lost the battle.” (P57)



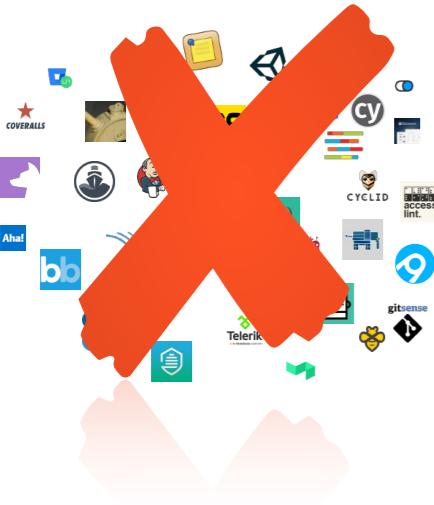
# CI CONSISTENCY

“I had some open source projects running in TravisCI and some in CircleCI. I just wanted to consolidate the project to one place and I’m sorry to say that at that time TravisCI lost the battle.” (P57)

**EXPOSURE TO LEAVERS:  
~1.5X INCREASE IN CHANCES  
OF SWITCHING & ABANDONING**

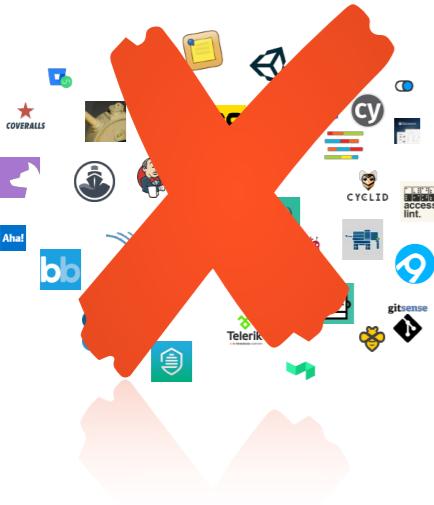


# CI CONSISTENCY



# CI CONSISTENCY

**PEOPLE'S PAST EXPERIENCE IS  
VERY IMPORTANT IN  
DETERMINING BEHAVIOR ON  
FUTURE PROJECTS!**

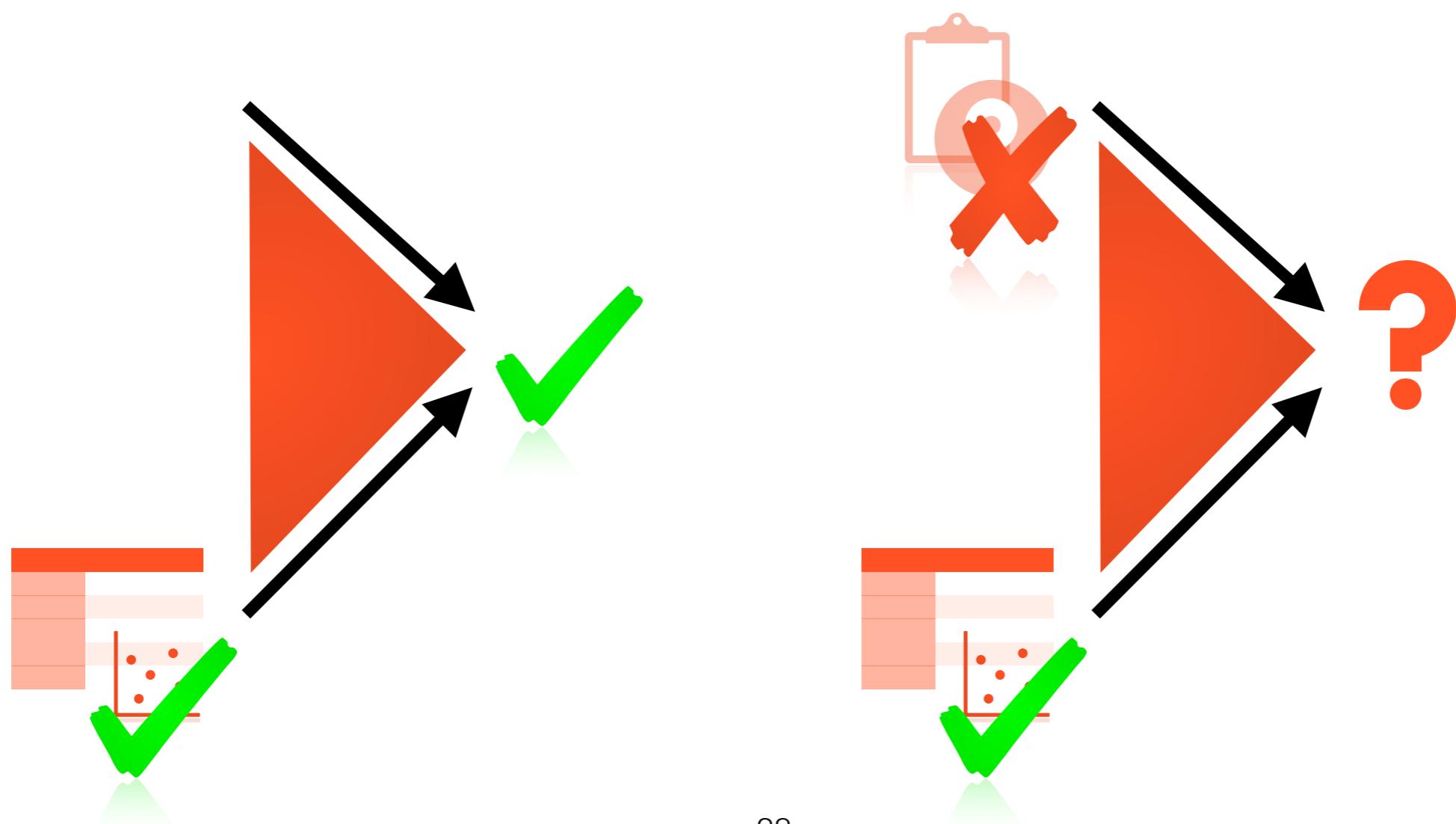


# CI CONSISTENCY

**PEOPLE'S PAST EXPERIENCE IS  
VERY IMPORTANT IN  
DETERMINING BEHAVIOR ON  
FUTURE PROJECTS!**

**BUT NOT OBSERVED IN PAST  
LITERATURE, SO  
NEEDS REPLICATION.**

# SINGLE METHOD OR CONFLICTING RESULTS





**LONG BUILD  
TIMES**



# LONG BUILD TIMES



# LONG BUILD TIMES

“We stopped using Travis CI because it was too slow for us.” (P125)



# LONG BUILD TIMES

“We stopped using Travis CI because it was too slow for us.” (P125)

**LONGER BUILD TIMES ASSOCIATED WITH A  
DECREASED CHANCE OF SWITCHING AND  
ABANDONING.**



# LONG BUILD TIMES

“We stopped using Travis CI because it was too slow for us.” (P125)

**LONGER BUILD TIMES ASSOCIATED WITH A  
DECREASED CHANCE OF SWITCHING AND  
ABANDONING.**

**PRESENCE OF **VERY** LONG BUILDS ASSOCIATED  
WITH INCREASED RISK OF ABANDONING.**



# LONG BUILD TIMES



# LONG BUILD TIMES

**SO, WHEN DO LONG  
BUILDS STOP PROVIDING  
VALUE, AND START  
BECOMING ANNOYING?**

# FULL RESULTS IN THE PAPER

## A Conceptual Replication of Continuous Integration Pain Points in the Context of Travis CI

David Gray Widder  
dwidder@cmu.edu  
Carnegie Mellon

Michael Hilton  
mhilton@cmu.edu  
Carnegie Mellon

Christian Kästner  
Carnegie Mellon

Bogdan Vasilescu  
vasilescu@cmu.edu  
Carnegie Mellon

### ABSTRACT

Continuous integration (CI) is an established software quality assurance practice, and the focus of much prior research with a diverse range of methods and populations. In this paper, we first conduct a literature review of 37 papers on CI pain points. We then conduct a conceptual replication study on results from these papers using a triangulation design consisting of a survey with 132 responses, 12 interviews, and two logistic regressions predicting TRAVIS CI abandonment and switching on a dataset of 6,239 GitHub projects. We report and discuss which past results we were able to replicate, those for which we found conflicting evidence, those for which we did not find evidence, and the implications of these findings.

### CCS CONCEPTS

• Software and its engineering → Software maintenance tools.

### KEYWORDS

Continuous integration, open source software, replication

### ACM Reference Format:

David Gray Widder, Michael Hilton, Christian Kästner, and Bogdan Vasilescu. 2019. A Conceptual Replication of Continuous Integration Pain Points in the Context of Travis CI. In *Proceedings of the 27th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '19), August 26–30, 2019, Tallinn, Estonia*. ACM, New York, NY, USA, 12 pages. <https://doi.org/10.1145/3338906.3338922>

### 1 INTRODUCTION

Continuous integration (CI) has enjoyed tremendous popularity as a quality assurance mechanism during software development, by automating the execution of builds, tests, and other tasks. CI adoption was primarily driven by practitioners,<sup>1</sup> but research has shown that CI practices have a positive effect on software quality and productivity [28, 69, 76].

Despite the widespread adoption of CI, it has long been established by contingency theory [50, 66] that a single “universal best practice” is unlikely, whatever the actual practice. Moreover, for CI specifically, the literature abounds with studies (we counted 37 papers; Section 3) that each touch on some CI pain points. For example, research has shown that it can take significant effort to

<sup>1</sup>[www.martinfowler.com/articles/continuousIntegration.html](http://www.martinfowler.com/articles/continuousIntegration.html)

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).  
*ESEC/FSE '19, August 26–30, 2019, Tallinn, Estonia*  
© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.  
ACM ISBN 978-1-4503-5572-8/19/08.. \$15.00  
<https://doi.org/10.1145/3338906.3338922>

set up and customize CI infrastructure [27, 34], and reports from CI systems require effort to process and can cause unwanted interruptions [39], especially without developer buy-in and in the presence of frequent false positives from flaky tests and platform instabilities [37]. Bad experiences or frustration with a specific CI tool can turn developers away from CI as a practice, even when more customized tool solutions exist [86].

Given the number of studies, conducted using a multitude of methods, on diverse populations, we argue that it is the right time for a *thorough review of the pain points and context mismatches that turn people away from CI*. This can help practitioners adopt CI with realistic expectations and in a way that fits their needs, and researchers and tool builders focus on the most severe CI barriers.

In this paper we *review the CI literature* from the perspective of pain points to adoption and usage, and perform a mixed-methods *conceptual replication* [32, 65] of previously observed findings, on a new population (GitHub open-source developers using TRAVIS CI), and using a robust study design. As particular strengths of our study design, we note: (1) the mixed qualitative (survey with 132 developers and interviews with 12; Sec. 4) and quantitative (large-scale multivariate statistical modeling of trace data from 6,239 projects; Sec. 5) analyses, which enable us to *triangulate* our results; and (2) the focus on CI *leavers* (rather than current CI users), i.e., those who either switched the TRAVIS CI tool or abandoned the CI practice altogether, which, similarly to customer exit surveys in market research [70], enable us to identify the most acute of TRAVIS CI pain points, since they caused users to leave.

Our main results (Sec. 6), confirming past literature, are that many developers find troubleshooting build failures difficult, desire consistency in CI tools across their projects, find it difficult to use CI with complex tool setups including Docker or to use CI with unsupported languages, find long build times annoying, and find CI less useful without enough tests.

In summary, we contribute: (1) a literature review of general CI pain points; (2) an analysis of 132 survey responses about reasons for abandoning or switching TRAVIS CI; (3) regression models on a dataset of 6,239 GitHub TRAVIS CI projects, testing observations from literature; and (4) a discussion of results and implications.

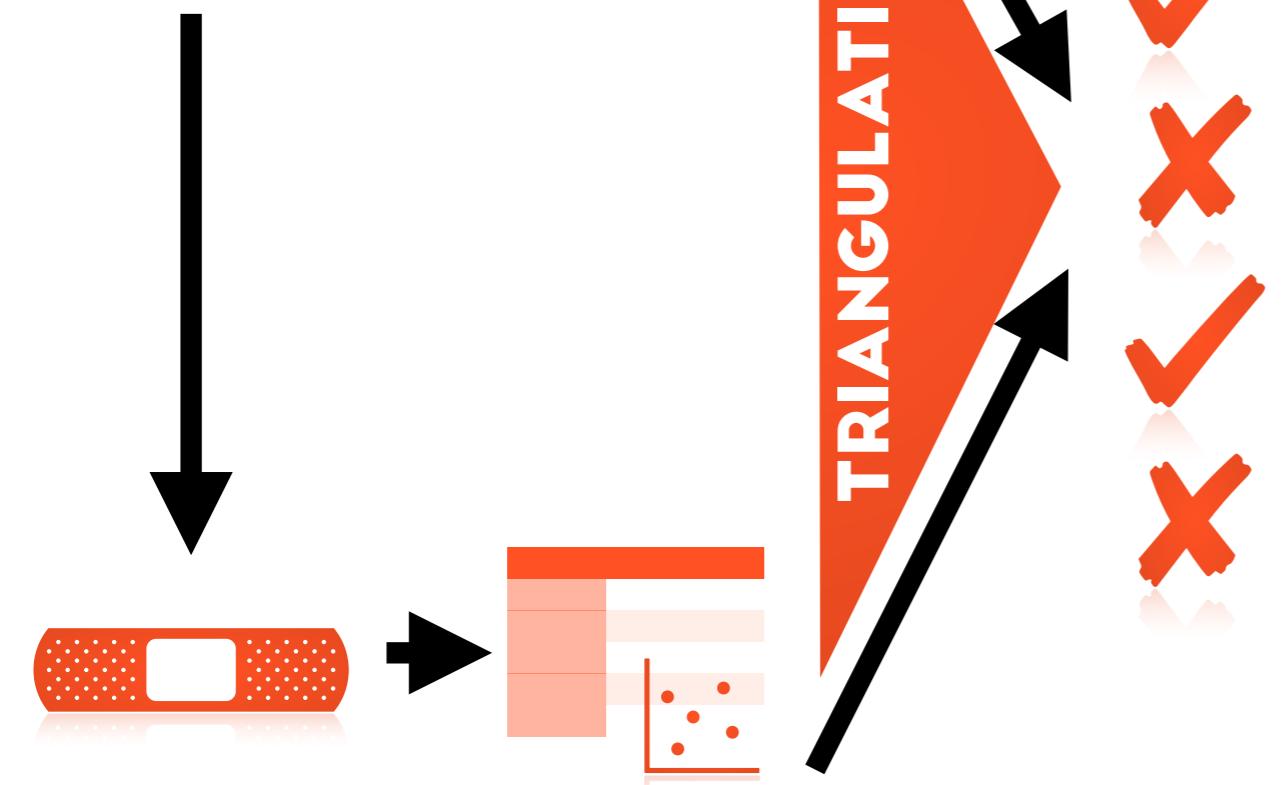
### 2 STUDY DESIGN

What are the major pain points that turn people away from CI? To answer this research question, we conduct a *conceptual replication* [32, 65], i.e., we attempt to corroborate observations from past research using a different experimental design, on a different population. The importance of replication studies in software engineering is increasingly recognized.<sup>2</sup> Our *conceptual replication*, as opposed to an *exact replication*, represents a more robust design:

<sup>2</sup>E.g., see the ROSE (Recognizing and Rewarding Open Science in Software Engineering) panel at FSE 2018: <https://tinyurl.com/y4m2uzsp>



# LIT SURVEYS, REVIEW INTERVIEWS



# PAIN POINTS 2 LOGISTIC REGRESSIONS



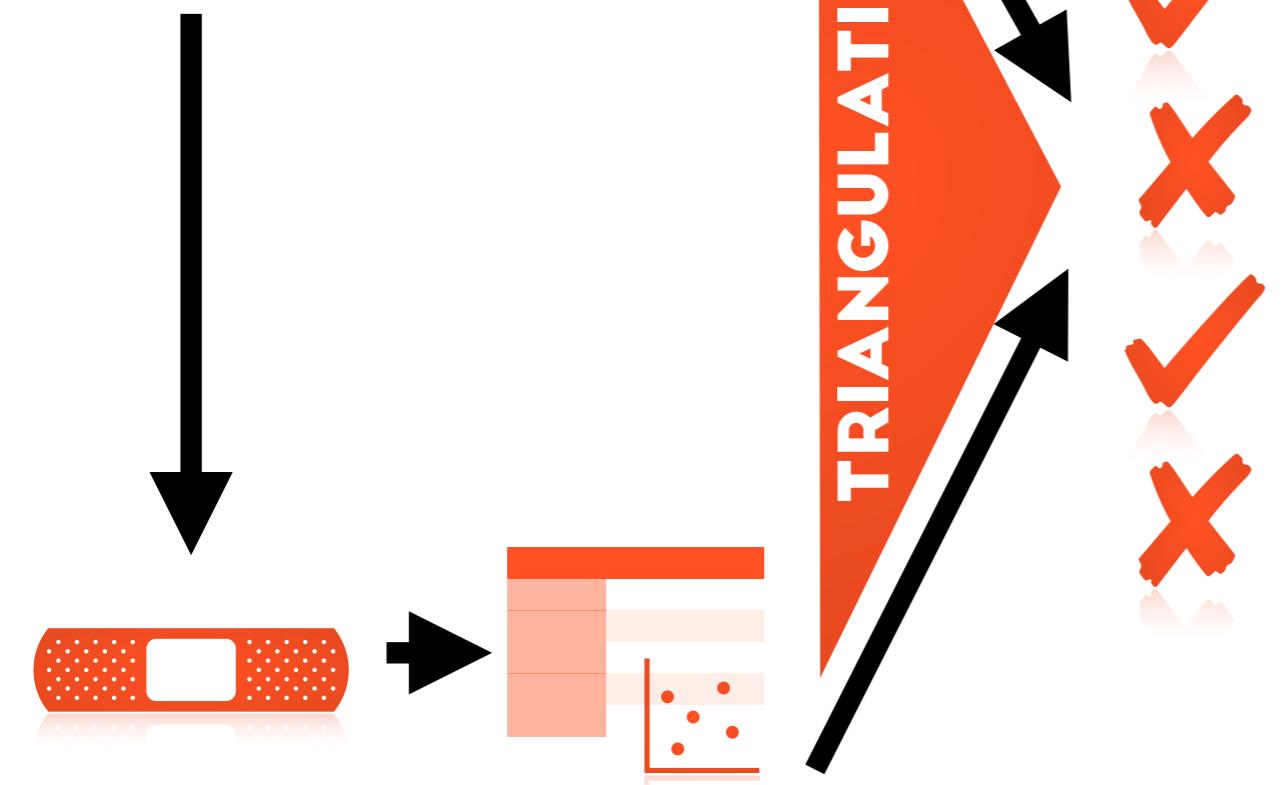
STRUDEL



# DAVID THE WID



# LIT SURVEYS, REVIEW INTERVIEWS



# PAIN POINTS 2 LOGISTIC REGRESSIONS

- 

# **WE WERE UNABLE TO CLEANLY REPLICATE SOME PAIN POINTS, THEY NEED FURTHER STUDY TO VALIDATE THEIR EXISTENCE**



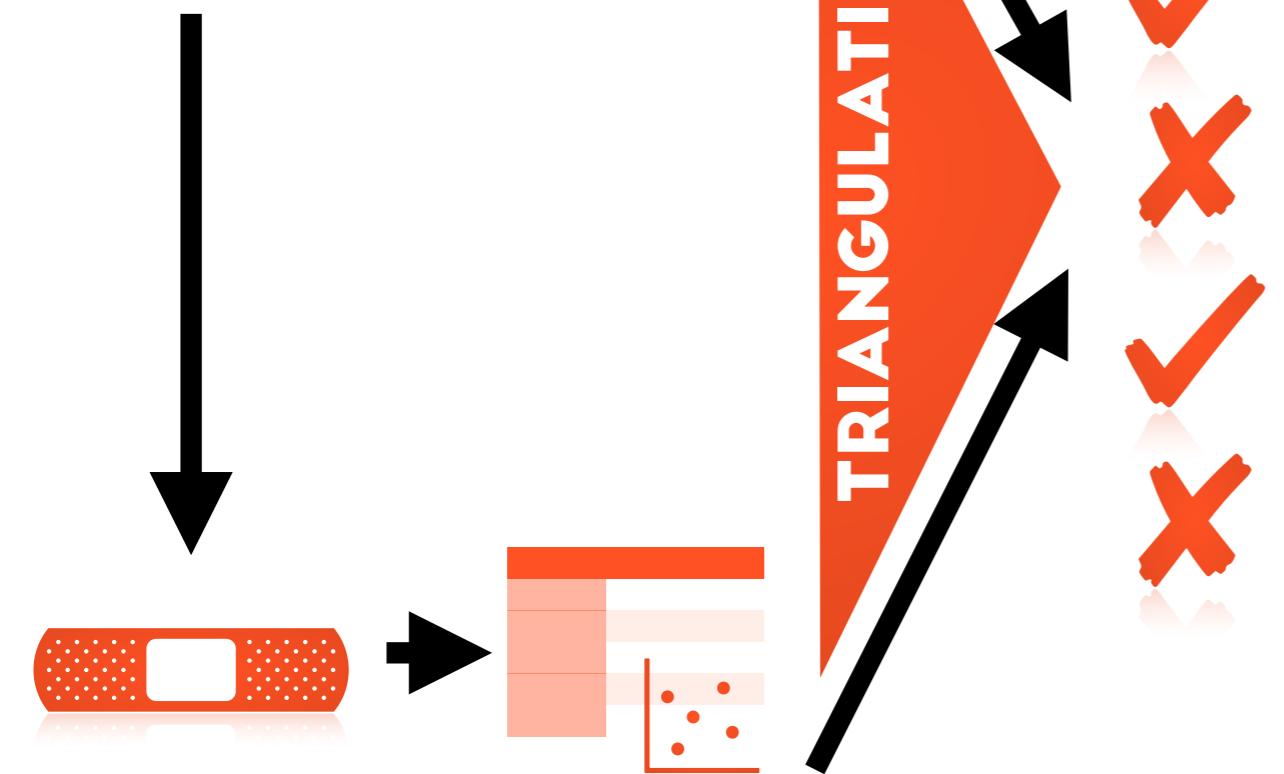
# STRUDEL



# DAVID THE WID



# LIT SURVEYS, REVIEW INTERVIEWS

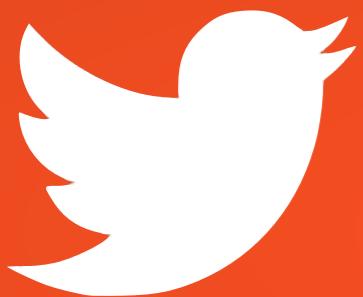


# PAIN POINTS 2 LOGISTIC REGRESSIONS

-  **WE WERE UNABLE TO CLEANLY REPLICATE SOME PAIN POINTS, THEY NEED FURTHER STUDY TO VALIDATE THEIR EXISTENCE**
  -  **WE WERE ABLE TO CLEANLY REPLICATE SOME PAIN POINTS, NOW WE CAN FOCUS ON SOLUTIONS**



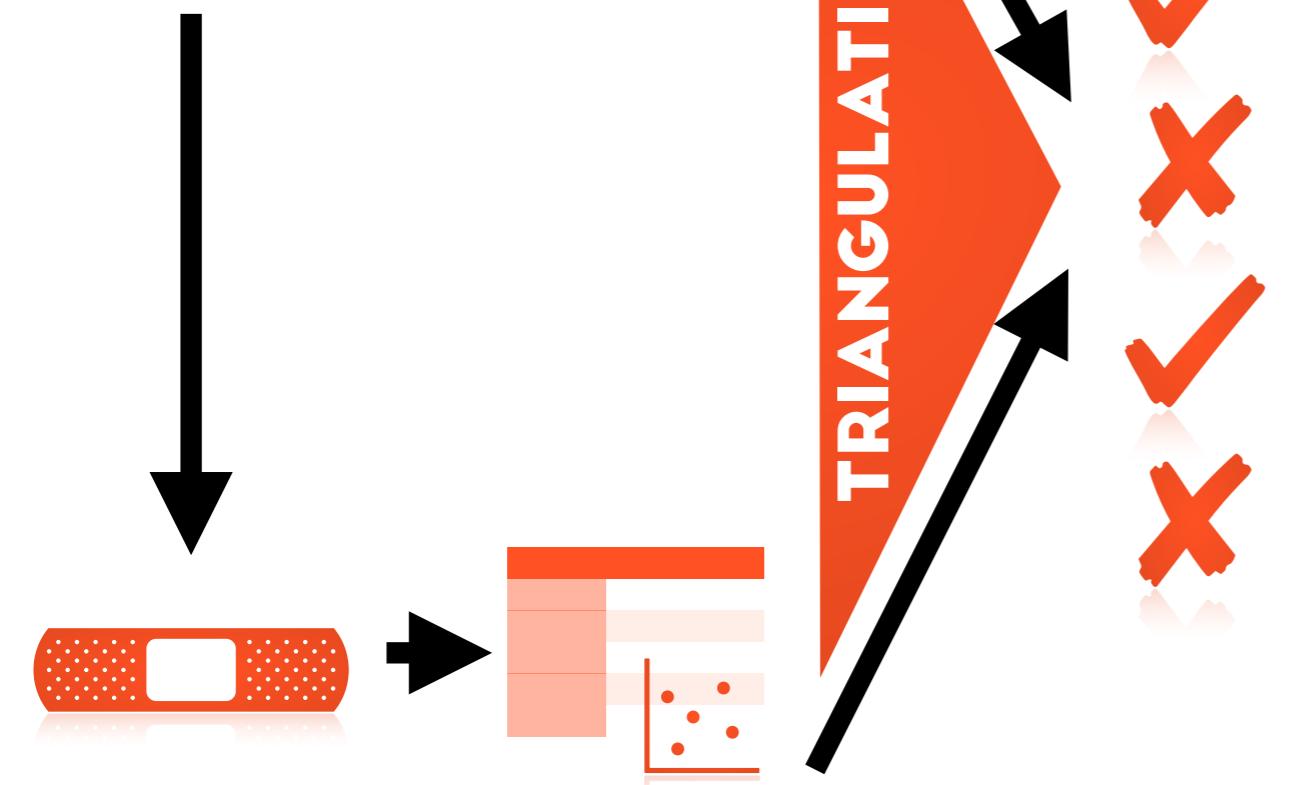
# STRUDEL



# DAVID THE WID



# LIT SURVEYS, REVIEW INTERVIEWS

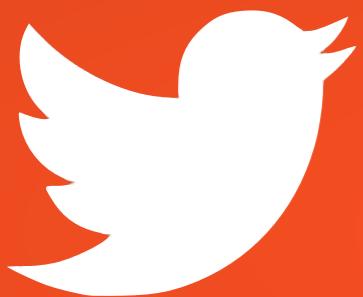


# PAIN POINTS 2 LOGISTIC REGRESSIONS

-  **WE WERE UNABLE TO CLEANLY REPLICATE SOME PAIN POINTS, THEY NEED FURTHER STUDY TO VALIDATE THEIR EXISTENCE**
  -  **WE WERE ABLE TO CLEANLY REPLICATE SOME PAIN POINTS, NOW WE CAN FOCUS ON SOLUTIONS**
  -  **FUTURE WORK SHOULD FOCUS MORE THAN JUST TRAVISCI, AND OUR COMMIT STATUS CONTEXT METHOD CAN HELP**



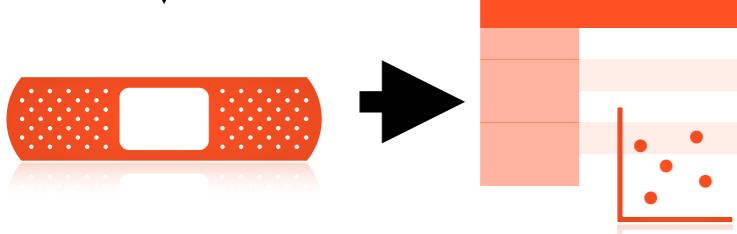
# STRUDEL



# DAVID THE WID



LIT  
REVIEW INTERVIEWS



PAIN  
POINTS

2 LOGISTIC  
REGRESSIONS



- WE WERE UNABLE TO CLEANLY REPLICATE SOME PAIN POINTS, THEY NEED FURTHER STUDY TO VALIDATE THEIR EXISTENCE
- WE WERE ABLE TO CLEANLY REPLICATE SOME PAIN POINTS, NOW WE CAN FOCUS ON SOLUTIONS
- FUTURE WORK SHOULD FOCUS MORE THAN JUST TRAVISCI, AND OUR COMMIT STATUS CONTEXT METHOD CAN HELP
- CI CONSISTENCY: SOCIAL TIES IMPACT A PROJECT'S TOOLING CHOICES



STRUDEL



DAVID THE WID