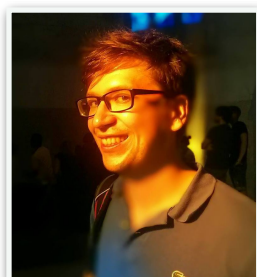


Oops, My Tests Broke the Build: An Explorative Analysis of Travis CI with GitHub



Moritz Beller, Georgios Gousios, Andy Zaidman

2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)

発表者 吉富 楓雅

目次

- 背景とリサーチクエスチョン
- Travis CIとは
- 研究方法
- 結果
- 結論

要約

継続的インテグレーション（CI）は、現代のソフトウェア開発の最も効率の良い方法となっている。

しかしCIベースのソフトウェア開発で一般的に行われるベストプラクティスに関する洞察が不足している。

本研究では特に

- ①CIプロセスに対するテストは中核なのか
 - ②プログラム言語がテストにどの程度影響を及ぼすか
 - ③さまざまな統合環境が重要かどうか
 - ④CIのテストがIDEのローカルテストの代理として機能するか
- についての調査を行う。

背景と リサーチクエスチョン

継続的インテグレーション

- 継続的インテグレーション(CI)は継続的に開発への貢献度を検証するソフトウェアエンジニアリングプラクティスである
 - * 自動ビルドツールを用いる
 - ▶ XP(Extreme Programming)の1つとして提案された
- 本論文では特にTravis CIを用いる
- 以前からの研究によって作成されたCIについての5つの命題を元にリサーチクエスチョンを作成する

命題

- CIは普及しており，包括的な品質保証活動である
 - * OSSプロジェクトでの使用はどのくらい普及しているのか？
- テストはCIの中核を成している
 - * 以前の研究ではCIのテストフェーズを考慮していない
- CIのテストはプログラム言語に依存する
 - * ある言語でのCI観測は他の言語に一般化できない可能性がある

命題

- 異なる環境においてのテストの統合は貴重である
 - * 異なる環境でのビルドと統合は時間とリソースを大量に消費するため、1環境での統合に比べて付加価値を提供する必要がある
 - ▶ しかし現在はこれを裏付けるデータが存在しない
- CIでのテストは開発者に迅速なフィードバックを提供することである
 - * しかし416人のソフトウェア開発者が統合開発環境(IDE)でどのように研究を行ったかについての研究では「テストパラドックス」について説明できなかった
 - * また、CI環境がローカル環境でテストを実行するのに適しているのかが不明

リサーチクエスチョン

- RQ1

- * Travis CIはGitHubでどのくらい一般的に使用されているのか？

- RQ2

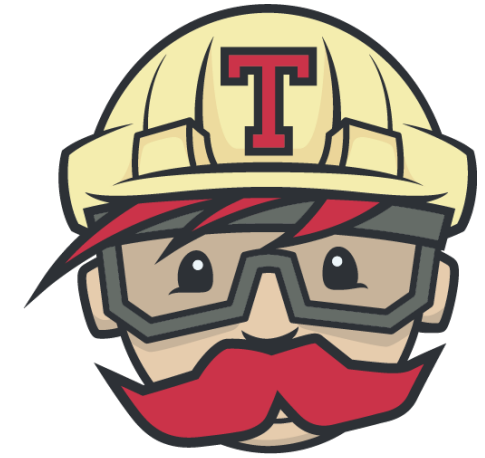
- * どのようにテストはCIの中核を成しているのか？

- RQ3

- * テストはビルド結果にどのように影響するのか？

Travis CIとは

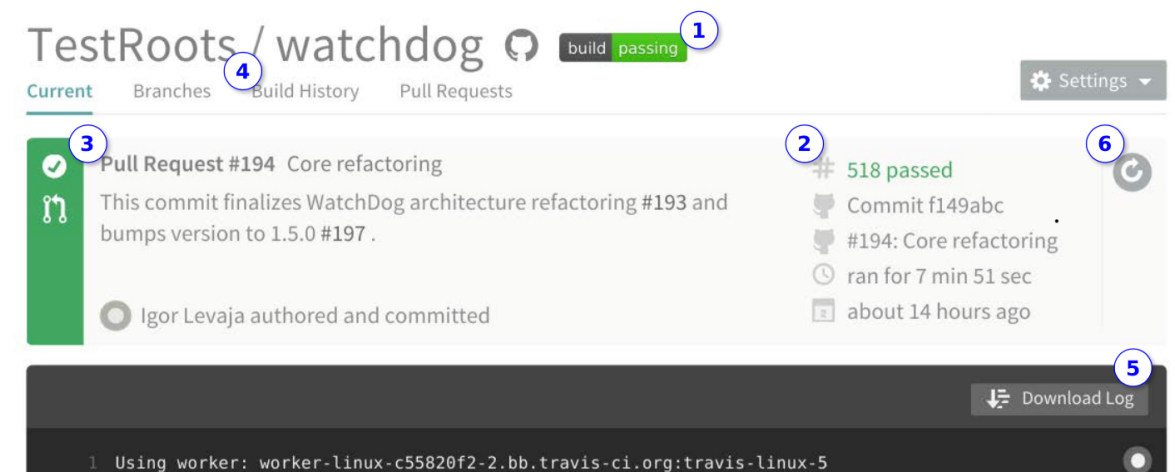
Travis CI



- オープンソースのビルドサービス
 - * Githubとの緊密な連携によりプロジェクトにCIの手順を構築して実行できる
- 2010年にオープンソースプロジェクトとしてスタート, 2012年に会社化した
- さまざまなプログラム言語に対応している
 - * Java, C, C++, Python, R, Visual Basic など

Travis CIの使用

- 1:プロジェクト名及びマスタートランチのビルドステータス
- 2:ビルドをトリガーしたGitのコミット, ビルドステータス, ビルド時間など
- 3:プル要求によるビルドのトリガー
- 4:このプロジェクトの実行履歴を取得
- 5:ビルド実行時のログファイル
- 6:ビルドの再続行をトリガー

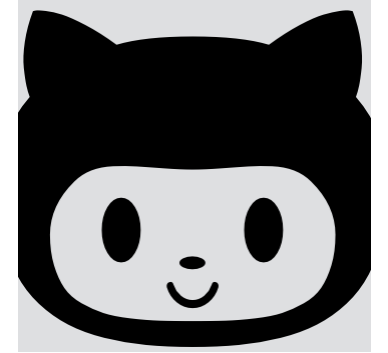


ビルドのセットアップ

- Travis CI対応のGitHubリポジトリのコミットがプッシュされる度に、ブランチまたはプルリクエストの最新のコミットがGitHubウェブフックを介してTravisCIに送信される
 - * Travis CIはこれを自動受信してその後ビルドが構築される
- 現在は単一言語のビルドのみを提供
 - * Javaなどの複数のバージョンでの構築には対応できる
- ビルド実行が一部で不成功な場合、ビルド全体が不成功であると判断する

研究方法

ソースと使用言語



- GHTORRENTデータベースを使用
 - * 最も適切なものを選択するためのフィルタリング方法を調査, 適用
- テストツールの出力を調べる必要があるため, プロジェクトの検索をJavaとRubyに限定, それ以外は除外
 - * 利用可能な自動テストツールが数多く存在する
- Travis CIと連携されていないものも除外した
 - * Travis CIビルドログとGHTORRENTデータベースからビルド情報を抽出して分析, TRAVISTORRENT内の両方のデータソースを結合

ツール

- TRAVISTORRENTの作成
 - * <http://travistorrent.testroots.org>
- TravisPoker
 - * GitHubでTravisCIを使用するプロジェクトの数を調査するために実装

ツール

- TravisHarvesterの実装
 - * プロジェクトのTravisCIビルド履歴に関する詳細情報を集約
 - ▶ GitHubプロジェクト名を入力, 履歴の各ビルドに関する統計をCSVファイルで収集
 - * 全てのジョブのビルドごとにビルドログをダウンロードし, CSVファイルと共に保存
- また, AmazonAWSサーバにビルドログをアーカイブすることも依頼

結果

RQ1:



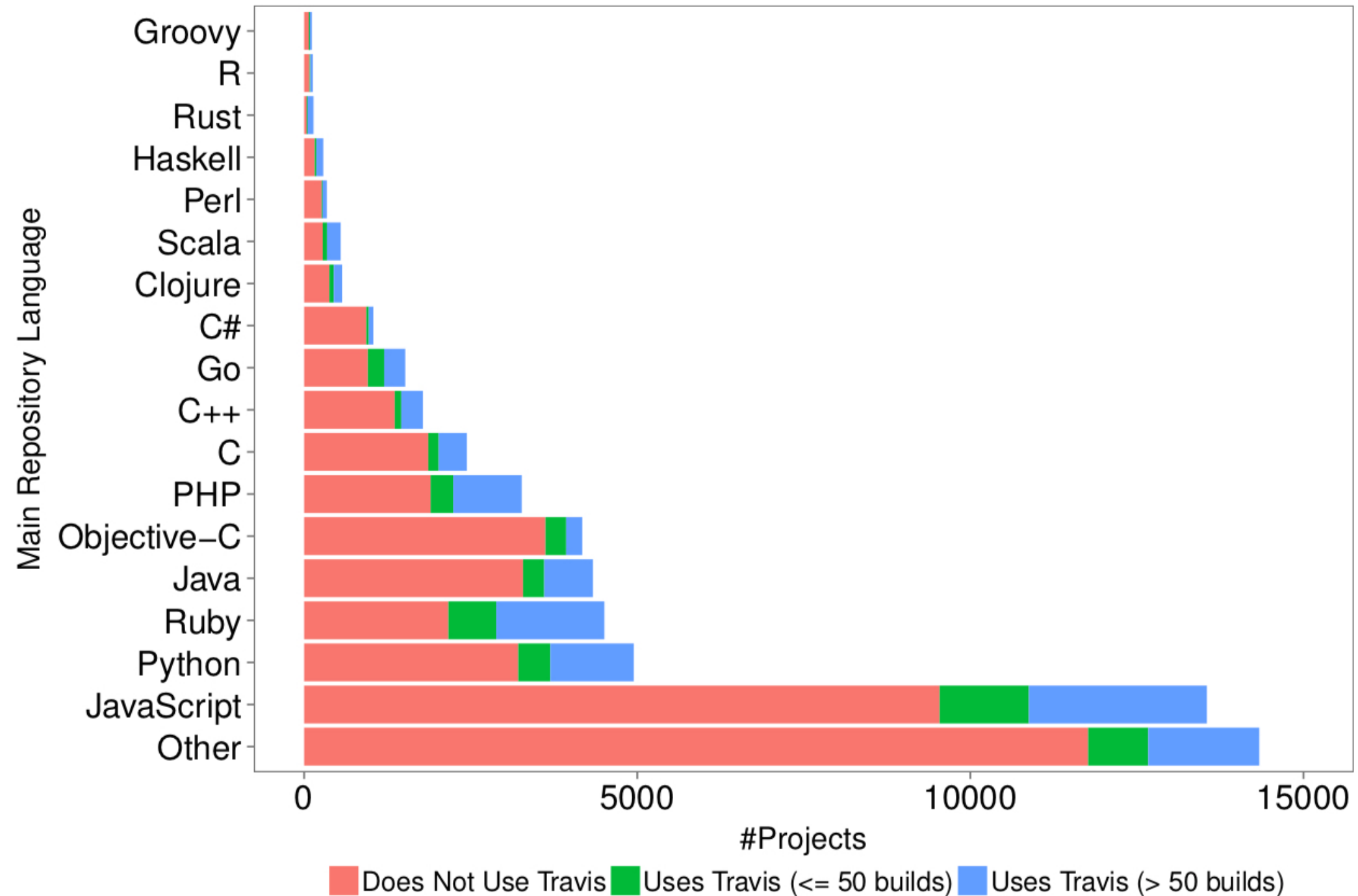
Travis CIはGitHubでどのくらい一般的に使用されているのか？

- TRAVISPOKERを使用して実行されたTravisCIビルドの数を収集
 - * 58,032件のプロジェクトを抽出
 - ▶ 123の多彩な言語が使用されていた
 - ▶ しかし大部分はTravisCIでサポートされていない
- 16,159件のプロジェクトでTravisCIを少なくとも構築(使用率27.8%)

RQ1:

Travis CIはGitHubでどのくらい一般的に使用されているのか？

TravisCIを集約したプロジェクトの数



RQ2

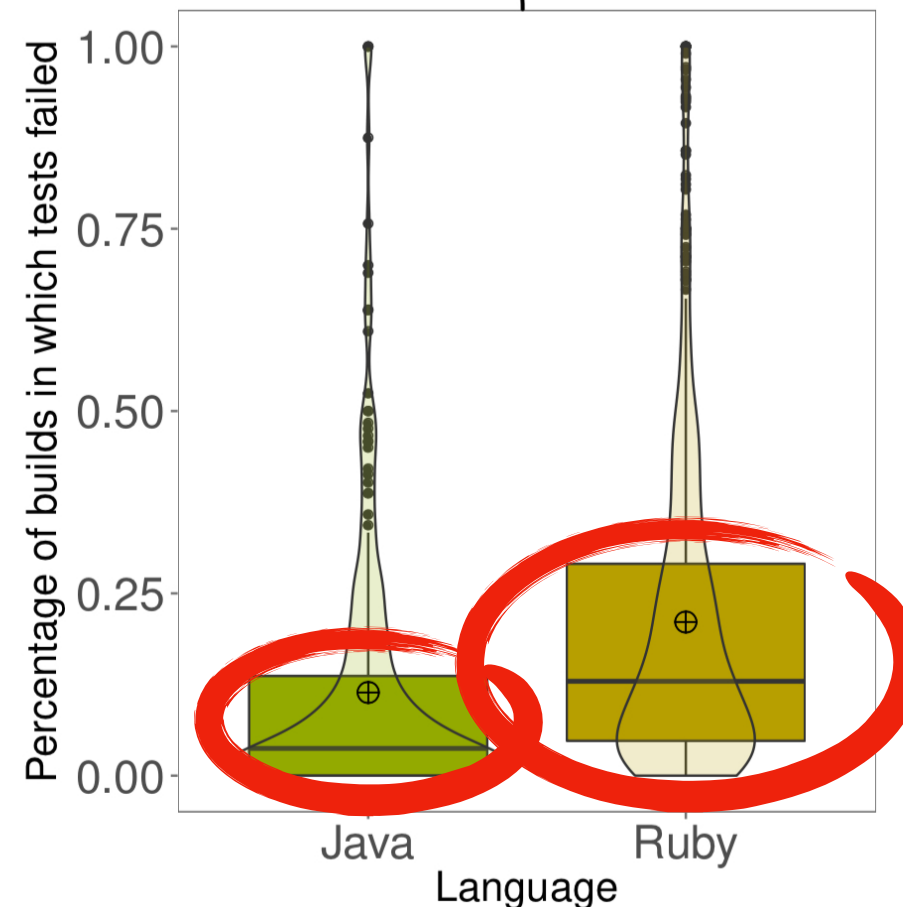
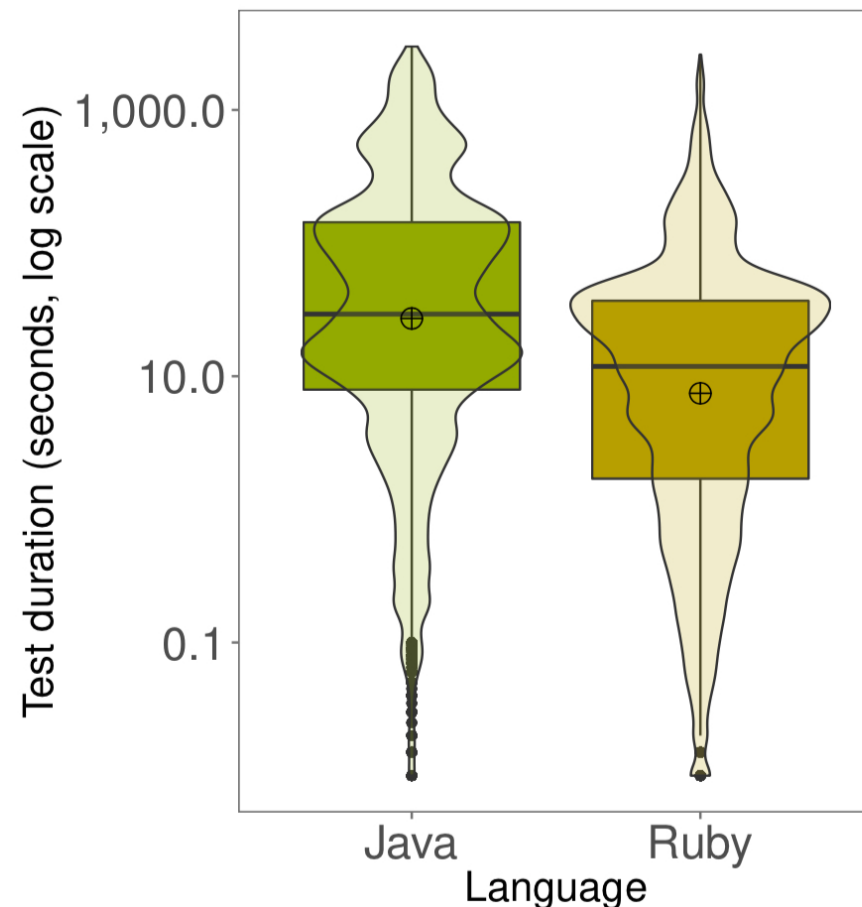
どのようにテストはCIの中核を成しているのか？

- 調査したプロジェクトの81%がCIプロセスの一環としてテストを実施
 - * ビルドレベルでは, サンプルのビルドの96%が少なくとも1回のテスト実行を行った
- しかしテストは必ずしも必須のステップではない
 - * Javaプロジェクトのうちの31%と, Rubyプロジェクトのうちの12.5%ではテストを実施していなかった

RQ3

テストはビルド結果にどのように影響するのか？

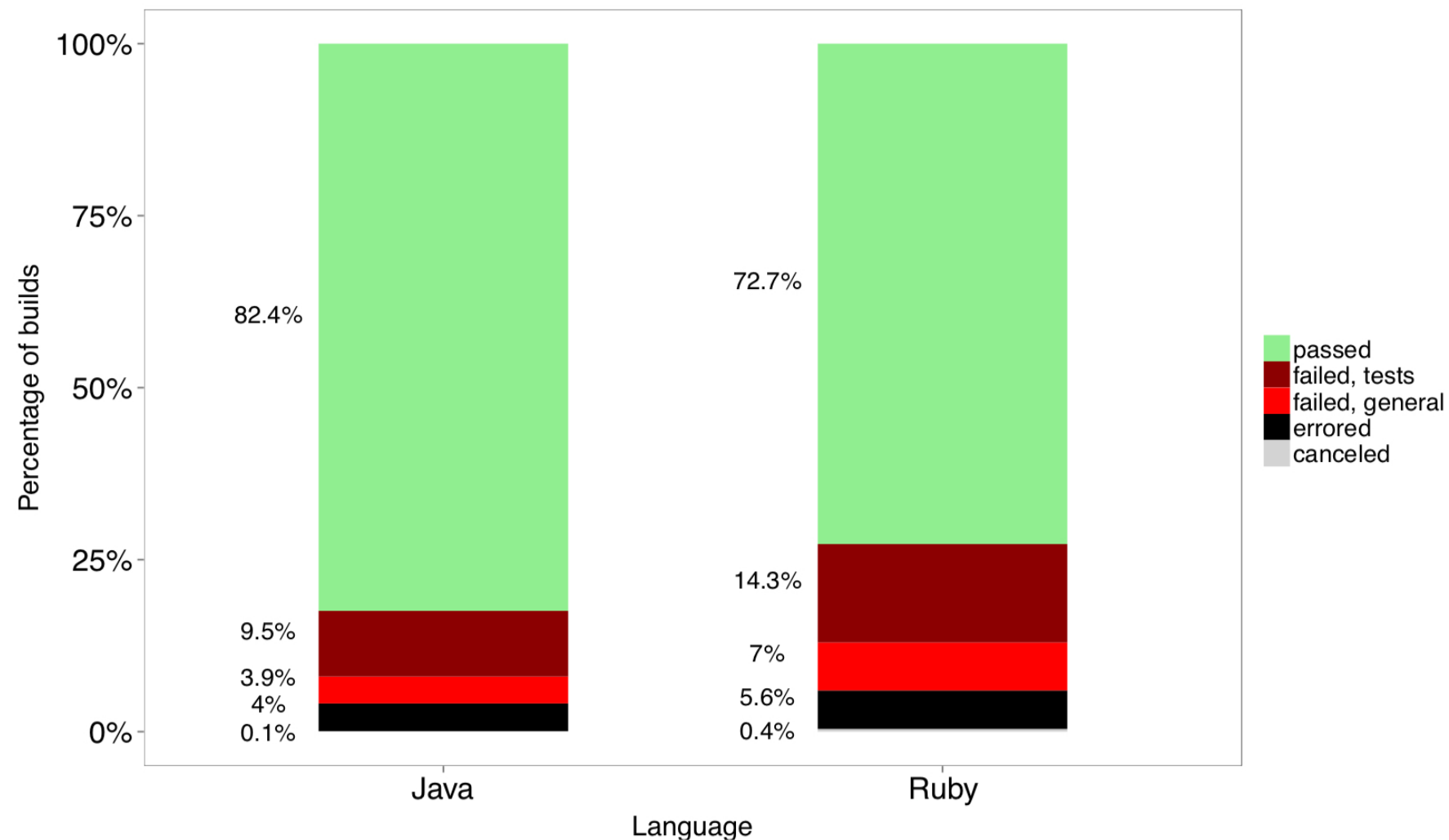
- ビルドと最低1回失敗したテストの比率のビーンプロット
 - * ウィルコクソンの順位和検定を使用
 - * Rubyのビルドのテスト失敗率がJavaプロジェクトよりも大幅に高くなっている



RQ3

テストはビルド結果にどのように影響するのか？

- テストを実行した1,108プロジェクトのビルド結果の集計
 - * 2言語のビルド結果の分布はほぼ同じ
 - * テスト失敗は破壊されたビルドが大半の原因である



結論

結論

- 全てのプロジェクトのうち3分の1がTravisCIを使用していた
 - * 比較的普及していることが判明
- 失敗したテストの大半はビルドが壊れていることが原因である
- TravisCIを使用しているプロジェクトのほとんどは最低1回のテストを実行している
 - * しかし必ずしも必須という訳ではない