

Lessons from Eight Years of Operational Data from a Continuous Integration Service

An Exploratory Case Study of CircleCI

Keheliya Gallaba

Centre for Software Excellence

Huawei Canada

Kingston, Canada

keheliya.gallaba@huawei.com

Maxime Lamothe

Polytechnique Montréal

Montréal, Canada

maxime.lamothe@polymtl.ca

Shane McIntosh

University of Waterloo

Waterloo, Canada

shane.mcintosh@uwaterloo.ca

In Proceedings of 44th International Conference on Software Engineering (ICSE), 2022,
pp. 1330 - 1342

背景 : CI (継続的インテグレーション)

- ▶ コード変更をメインブランチに頻繁にマージするプロセス
 - マージ前に自動でビルド・テストを行う
- ▶ 開発者に迅速なフィードバックを提供
- ▶ 自分の変更が他の変更ときれいに統合されているか検証可能



開発者の生産性向上
ソフトウェア品質の改善[42]

- ▶ CIプロバイダ
 - **CircleCI**, Travis CI など

背景 : CIビルド

- ▶ ソースコードへの変更が
統合可能かを示すシグナルの生成を期待
→ しかし, ビルドの結果は必ず決定的なものではない
- ▶ シグナルを生成せずに失敗するビルドがある
 - ユーザによる取り消し, 設定エラーなど. . .
- ▶ 全てのビルドでシグナルを生成することが理想
→ シグナルが生成されないと, 利用者からの信用がなくなる

背景：指標

▶ CIパイプラインの性能を追跡するための指標が必要

1. ビルド時間

- ビルド時間は長すぎても短すぎてもいけない

2. MTTR (Mean Time to Recovery：平均復旧時間)

- ビルドが失敗から成功に変わる時間間隔の平均

3. 成功率

- シグナルを生成するビルドのうち、ビルドに成功した割合

4. 処理能力

- 一定期間内に実行されたビルドの回数

対象 & 目的

▶ CircleCI

- CircleCIが5つの中で最もインストールされている
- ビルド結果が8年間にわたって公開されている



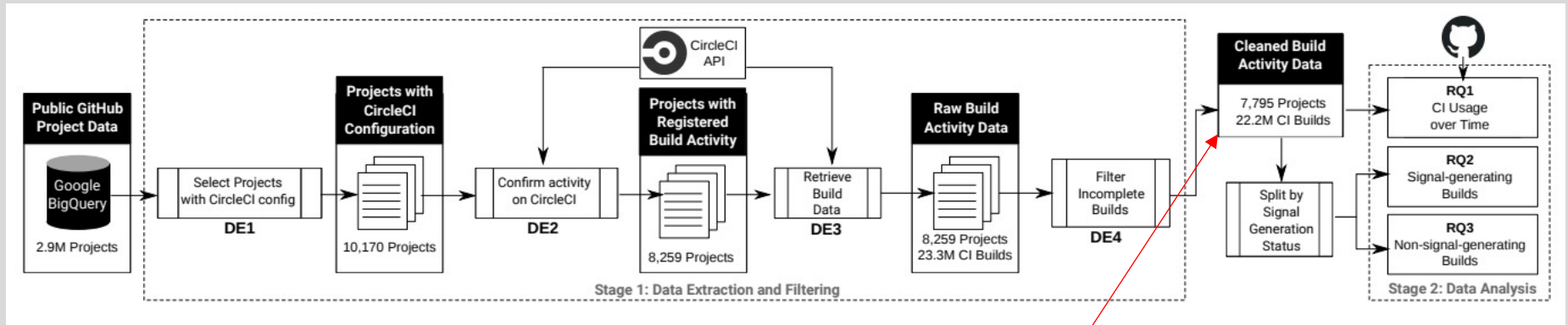
CircleCIを分析の対象に決定

▶ 目的

- CIプロバイダ視点で,
4つの指標を使用からCI利用率の変化の伸びを特徴づける
- RQ1:CircleCIの利用状況は時間とともにどのように変化するか
- RQ2:シグナル生成中のビルドはどのような処理がされているか
- RQ3:シグナルが生成されずに終了するビルドはなぜ存在するか

アプローチ

▶ データの抽出とフィルタリング



7,795件のプロジェクト

22,238,413件のビルドデータ

- RQ1には全てのビルドデータ
- RQ2, 3にはシグナルを生成するビルドとシグナルを生成しないビルドに分類したデータを使用

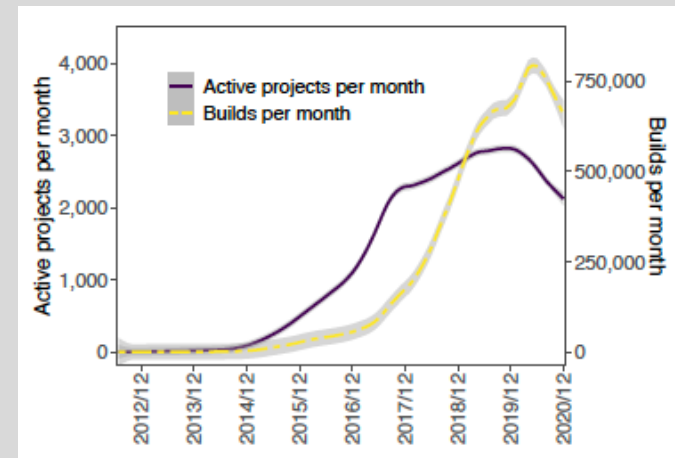
RQ1 (CircleCIの利用状況は時間とともにどのように変化するか)

▶ アプローチ

- CircleCIを利用するプロジェクト数とCircleCI上で実行されたビルド数をプロット

▶ 結果

- 時間の経過とともにプロジェクト数とビルド数の両方が増加
- 2020年以降は成長が停滞
- 2020年中にCircleCIの利用を中止していたプロジェクトのうち14%は他のCIプラットフォームへ移行



RQ1 (CircleCIの利用状況は時間とともにどのように変化するか)

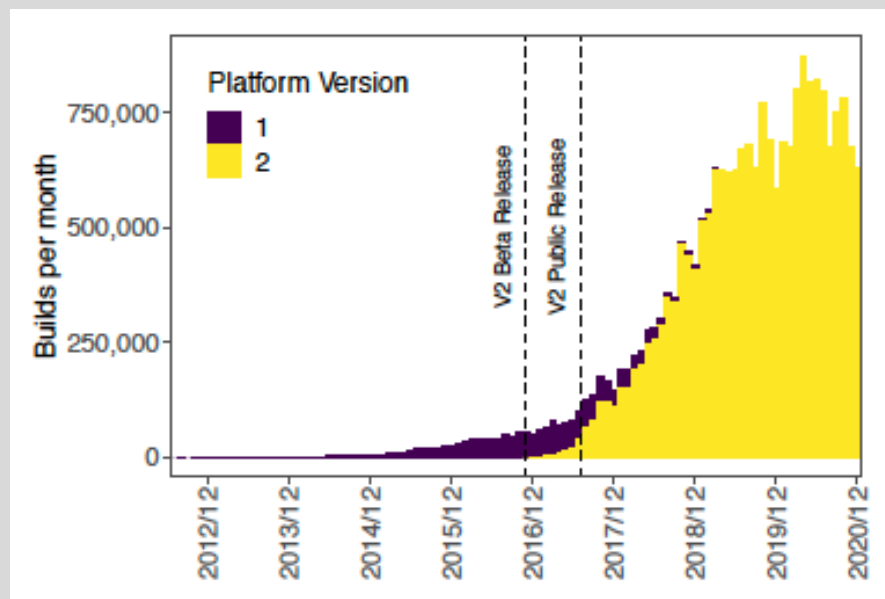
▶ アプローチ

- 2種類のCircleCIプラットフォーム (バージョン違い) でのビルド数の推移を観察

▶ 結果

- version 2からDockerイメージのサポート, SSHアクセスなど様々な機能が追加

version 2のリリースが急速な普及に大きな影響を与えた可能性



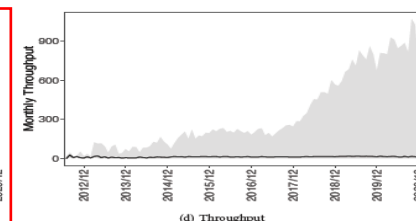
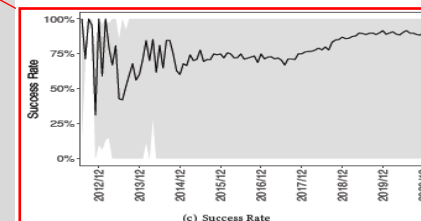
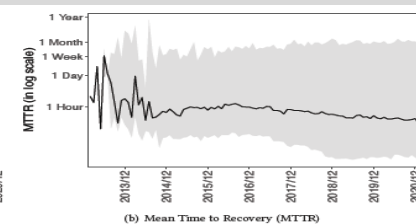
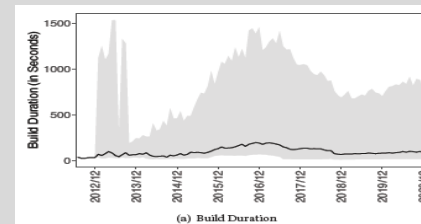
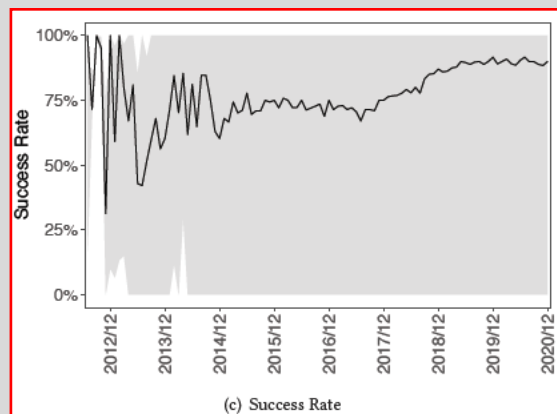
RQ1 (CircleCIの利用状況は時間とともにどのように変化するか)

▶ アプローチ

- CircleCIの4つの指標と時間経過を調査

▶ 結果

- ビルドの成功率はプロジェクトによって大きなばらつき
→しかし、成功率の中央値は期間後半にいくにつれて上昇
- CircleCIの調査では、メインブランチ以外ブランチの成功率が低い
→新たな変更(追加)がビルド失敗の原因



RQ2 (シグナル生成中のビルドはどのような処理がされているか)

▶ アプローチ

- シグナルを生成したビルドの各ステップの開始と終了時間を取得
- 各ステップ複数のアクションタイプも同時に取得



ビルドにおける各アクションタイプの実行時間を計算

Scott-Knott Effect Size Difference (ESD) テストを適用

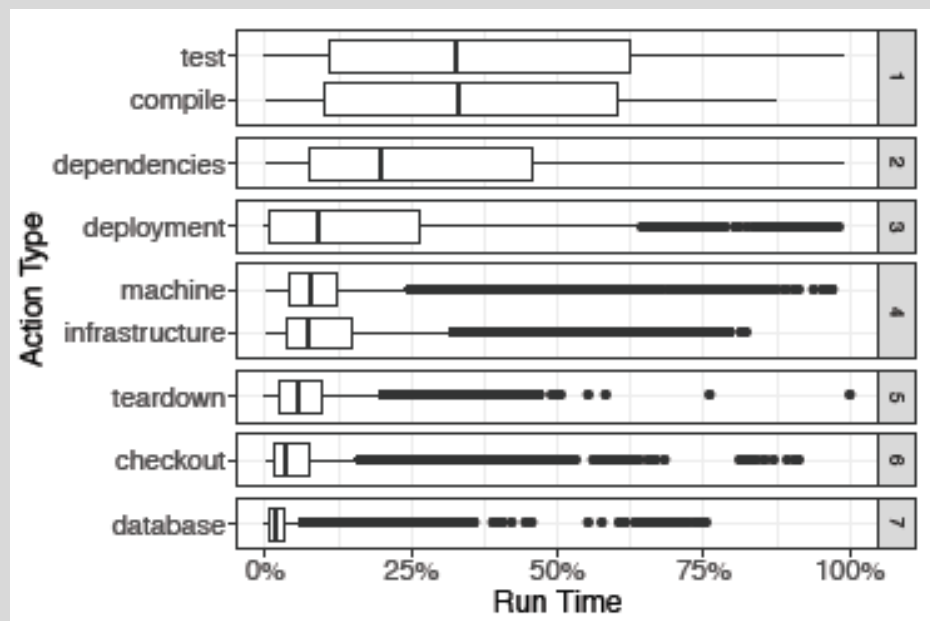
→各ビルドステップのアクションタイプをタイプごとのビルド時間の割合に基づいてクラスタリング

ヘビーユーザのシグナル生成ビルド中に費やされる時間が一般ユーザとは異なるかを調査

RQ2 (シグナル生成中のビルドはどのような処理がされているか)

▶ 結果

- ソースコードのコンパイルとテストの実行の割合が大きい
- ビルド時にコンパイルやテストの時間を短縮することはプロバイダにとって大きなメリット



RQ2 (シグナル生成中のビルドはどのような処理がされているか)

▶ アプローチ

- 月間ビルド時間が多いプロジェクト上位1%と、それ以外に分けて観察

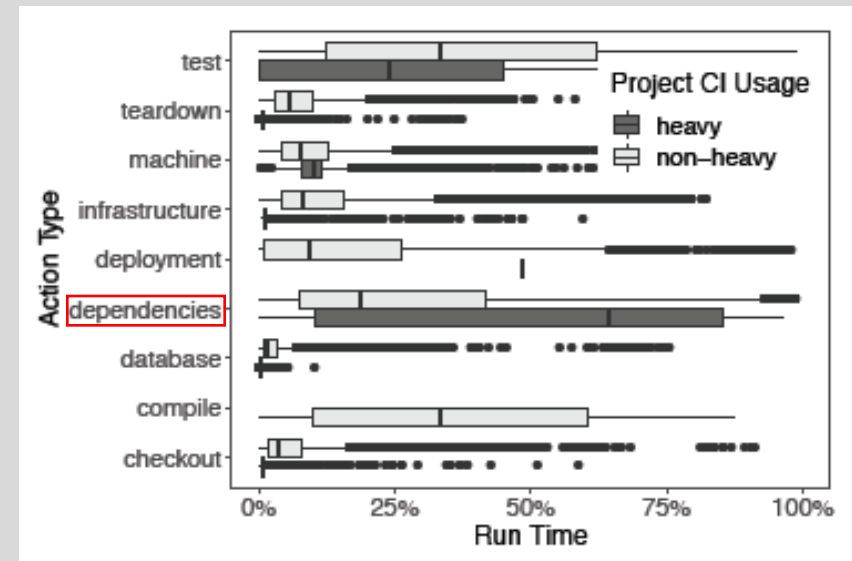
▶ 結果

- デプロイとコンパイル以外の全てのアクションタイプで有意差あり

ビルド後に環境をキャッシュし、
その環境を以降のビルドで使う



以降の依存関係のダウンロードが不要



RQ3 (なぜシグナルを出せないビルドがあるのか)

▶ アプローチ

- ランダムに選択したサンプルに対してシグナルが生成されない理由を調査

▶ 結果

CircleCIのサポートサービスが使用できない

設定エラー

ユーザによる取り消し

Category	#	%
Availability issues	371,647	1.59%
Orphaned builds	359,857	1.54%
Github unreachable	797	0.00%
Infrastructure fail	10,993	0.05%
Configuration Errors	31,238	0.13%
Missing or outdated configuration file	30,065	0.13%
Unsupported XCode version	26	0.00%
Github missing CircleCI SSH key	23	0.00%
User Cancellation	785,877	3.37%
Timed out	17,917	0.08%
Other	485,345	2.08%

ユーザがビルドを中止する理由の分析
タイムアウトしたビルドを特定する手法の開発

などの可能性

所感

- ▶ CIプロバイダの視点で研究されていて、
読んでいて面白かった

補足

Table 1: Distribution of Build Outcome. The global percentage of each category is shown in brackets.

	Outcome	#	%
Signal-generating	Success	18,831,874	80.72%
	Failed	2,806,795	12.03%
	Sub Total	(21,638,666)	92.75%
Non-signal-generating	Canceled	850,033	3.64%
	Infrastructure fail	10,993	0.05%
	Timeout	17,917	0.08%
	No tests	36,184	0.16%
	Null	776,964	3.33%
	Sub Total	(1,692,024)	7.25%