

Do Small Code Changes Merge Faster? A Multi-Language Empirical Investigation

Gunnar Kudrjavets
University of Groningen
Groningen, Netherlands
g.kudrjavets@rug.nl

Nachiappan Nagappan
Meta Platforms, Inc.
Menlo Park, CA, USA
nnachi@fb.com

Ayushi Rastogi
University of Groningen
Groningen, Netherlands
a.rastogi@rug.nl

In Proceedings of MSR '22: Proceedings of the 19th International Conference on Mining Software Repositories (MSR 2022) pp.

発表者：坂本

背景

▶ 継続的デプロイ

- コードの変更のストリームを本番環境にデプロイする開発手法
- メインブランチから頻繁に変更を取得してプッシュする
→ マージの衝突を防ぎ、タイムリーなデプロイが可能
- スピードが大切な開発現場で広く採用されている

▶ コードベロシティ

- コードの変更が本番環境にマージされるまでの速度
- コードの特性, プロジェクトの特性, 社会的要因などが影響^[9,12,32]
- プルリクエスト作成者が制御できるものはごくわずか

[9] Olga Baysal, Oleksii Kononenko, Reid Holmes, and Michael W. Godfrey. 2016. Investigating Technical and Non-Technical Factors Influencing Modern Code Review. Empirical Software Engineering 21, 3 (01 Jun 2016), 932–959. <https://doi.org/10.1007/s10664-015-9366-8>

[12] Amiangshu Bosu and Jeffrey C. Carver. 2014. Impact of Developer Reputation on Code Review Outcomes in OSS Projects: An Empirical Investigation. In Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (Torino, Italy) (ESEM '14). Association for Computing Machinery, New York, NY, USA, Article 33, 10 pages. <https://doi.org/10.1145/2652524.2652544>

[32] Michael Hilton, Timothy Tunnell, Kai Huang, Darko Marinov, and Danny Dig. 2016. Usage, Costs, and Benefits of Continuous Integration in Open-Source Projects. In Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering (Singapore, Singapore) (ASE 2016). Association for Computing Machinery, New York, NY, USA, 426–437. <https://doi.org/10.1145/2970276.2970358>

先行研究

▶ コードサイズ

- 開発者はサイズに関する要因がコードレビューの時間と決定に対して最も重要[36]
- 大きなパッチはレビューが難しく，読むのに時間がかかる
- 企業のプロジェクトとOSSプロジェクトで変更サイズは違う [64,63,61,75,30]

プルリクエストのサイズは「小さい」または「妥当」を推奨しているが，その指針は曖昧

[36] Oleksii Kononenko, Olga Baysal, and Michael W. Godfrey. 2016. Code Review Quality: How Developers See It. In Proceedings of the 38th International Conference on Software Engineering (ICSE '16). Association for Computing Machinery, Austin, Texas, 1028–1038. <https://doi.org/10.1145/2884781.2884840>

[64] Caitlin Sadowski, Emma Söderberg, Luke Church, Michal Sipko, and Alberto Bacchelli. 2018. Modern Code Review: A Case Study at Google. In Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP '18). Association for Computing Machinery, Gothenburg, Sweden, 181–190. <https://doi.org/10.1145/3183519.3183525>

[63] ChuckRossi,ElisaShibley,ShiSu,KentBeck,TonySavor,andMichaelStumm. 2016. Continuous Deployment of Mobile Software at Facebook (Showcase). In Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE 2016). Association for Computing Machinery, Seattle, WA, USA, 12–23. <https://doi.org/10.1145/2950290.2994157>

[61] Peter C. Rigby and Christian Bird. 2013. Convergent Contemporary Software Peer Review Practices. In Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering (ESEC/FSE 2013). Association for Computing Machinery, Saint Petersburg, Russia, 202–212. <https://doi.org/10.1145/2491411.2491444>

[75] Peter Weißgerber, Daniel Neu, and Stephan Diehl. 2008. Small Patches Get In!. In Proceedings of the 2008 international working conference on Mining software repositories (MSR '08). Association for Computing Machinery, Leipzig, Germany, 67–76. <https://doi.org/10.1145/1370750.1370767>

[30] Georgios Gousios, Martin Pinzger, and Arie van Deursen. 2014. An Exploratory Study of the Pull-Based Software Development Model. In Proceedings of the 36th International Conference on Software Engineering (ICSE 2014). Association for Computing Machinery, Hyderabad, India, 345–355. <https://doi.org/10.1145/2568225.2568260>

目的とRQ

▶ 目的

- 開発者がコードベロシティを直接制御できる範囲での解決
- 実用的なコードベロシティを向上する方法を提案

▶ RQ

- RQ1：プルリクエストのサイズ，構成，およびマージまでの速度はどのような特徴を持つか
- RQ2：プルリクエストのサイズと構成はマージまでの速度にどのような関係があるか
- RQ3：コンテキストはプルリクエストのサイズと構成がマージまでの速度にどのような影響を与えるか

データセット

最も人気のある言語**10**種類

- 最もフォークされた**100**のプロジェクトをGitHubから選定
- プロジェクトごとにマージされたプルリクエストの数のリスト



最もマージされたプルリクエストのあるプロジェクトを
言語ごとに**10**個選定



マージまでの時間がゼロのものは除外

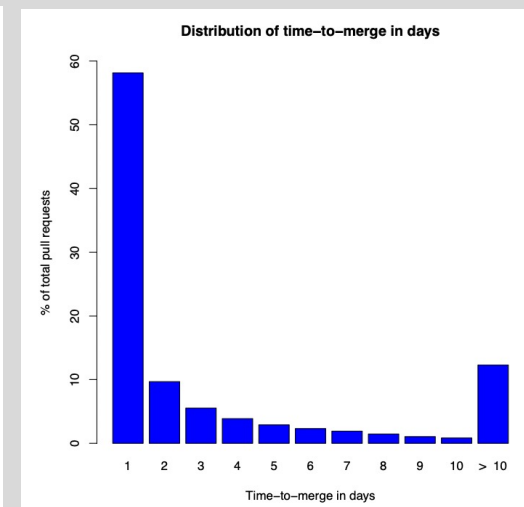
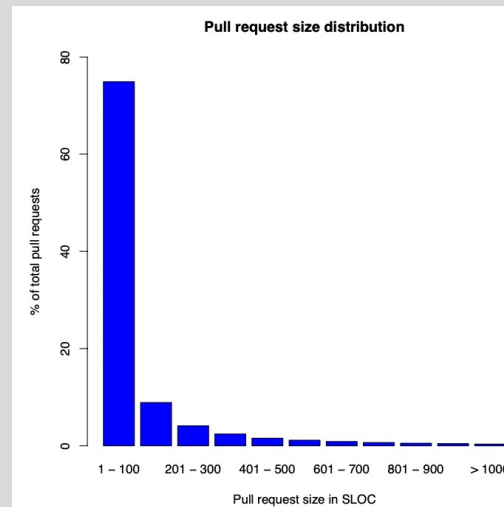


826,259件のプルリクエスト

分析

RQ1(プルリクエストのサイズ, 構成, マージまでの時間を特徴づけるものは何か)

- ▶ 変更されたSLOCの最頻値は1（データセットの11%が1）
 - ほとんどのプルリクエストは比較的サイズが小さい
- ▶ 半数のプルリクエストは数時間でマージされる
- ▶ 追加が修正の3倍多い
- ▶ 削除は修正の1.5倍多い



分析

RQ2(プルリクエストのサイズと構成がマージまでの時間に与える関係とは)

▶ マージまでの時間と以下の2つでスピアマン相関係数を計算

1. プルリクエストのサイズ

- ・ 相関は弱い($r_s = 0.26, p < .001$)

2. 合計サイズに対する変更タイプの比率(追加, 削除, 修正)

- ・ 追加($r_s = 0.18, p < .001$)
- ・ 削除($r_s = 0.06, p < .001$)
- ・ 修正($r_s = -0.14, p < .001$)
→ 修正比率ではわずかな負の相関

しかし,
変更タイプの比率の相関係数
の強さは無視できる

分析

RQ3(コンテキストはサイズと構成とマージまでの時間の関係に影響があるか)

プログラミング言語

所属(企業かOSSか)

プルリクエストが作成された曜日



影響があるか調査

分析

RQ3(コンテキストはサイズと構成とマージまでの時間の関係に影響があるか)

▶ 言語ごとのSLOCとマージまでの時間の記述的統計量(表4)

Table 4: Time-to-merge (hours) and SLOC per pull request.
N = total number, M = mean, Mdn = median, SD = standard deviation.

Language	N	Time-to-merge			SLOC		
		M	Mdn	SD	M	Mdn	SD
C	45,555	154	15	654	313	21	5,603
C++	136,308	190	19	769	721	21	14,574
C#	82,387	161	22	639	1416	40	23,422
Java	64,495	173	17	886	425	43	9,169
JavaScript	72,510	140	11	654	307	15	6,904
PHP	82,074	194	16	809	550	15	13,795
Python	137,365	228	21	978	229	21	7,476
Ruby	78,814	183	10	961	187	13	9,096
Shell	14,212	545	15	2,715	173	8	1,811
TypeScript	112,539	122	12	523	979	35	33,703

▶ マージまでの時間とサイズの中央値は2～3倍の差がある

分析

RQ3(コンテキストはサイズと構成とマージまでの時間の関係に影響があるか)

- ▶ クラスカル・ワリス検定（順位を用いたノンパラメトリック手法）
 - 少なくとも1つの言語のペアで、マージまでの時間の平均順位の間有意差がある
 - サイズについても同様の結果
- ▶ ボンフェローニ補正を用いたダン検定
 - 4つのペアを除くすべての言語間でマージまでの時間に差異
 - サイズについても同様の結果

分析

RQ3(コンテキストはサイズと構成とマージまでの時間の関係に影響があるか)

- ▶ 企業の開発者はOSS開発者とは別のプレッシャーがあるかも
 - 締切に追われる, など



- ▶ マンホイットニーのU検定
 - サイズ：企業のプロジェクト > OSSプロジェクト
 - マージまでの時間：OSSプロジェクト > 企業のプロジェクト
- ▶ 企業のプロジェクトとOSSプロジェクトにおいて,
 - マージまでの時間とサイズの有意な相関係数の差はない
 - 異なるタイプのコード変更の比率とマージまでの時間はほとんど一緒

分析

RQ3(コンテキストはサイズと構成とマージまでの時間の関係に影響があるか)

▶ 企業のエンジニアは週末に仕事をしないだろう

- 金曜日に提出されたプルリクエストはマージまでに時間がかかる

Table 6: Spearman correlation coefficient r_s ($p < .001$) between time-to-merge in hours and total SLOC in a pull request by pull request creation day. N = total number.

Day created	N	r_s
Monday	132,488	0.26
Tuesday	145,908	0.27
Wednesday	146,326	0.27
Thursday	143,626	0.27
Friday	131,587	0.27
Saturday	65,845	0.25
Sunday	60,479	0.28

▶ 週末は提出されるプルリクエストが半減する

▶ 異なるタイプのコード変更比率とサイズの中央値には、曜日による有意差はない

まとめ

- ▶ マージまでの時間とサイズと構成の間に有意な相関はない
 - プルリクエストの作成曜日, 言語, 所属に依存しない
- ▶ 企業のプロジェクトの方がOSSプロジェクトよりもマージまでの時間が短い
 - しかし, プルリクエストのサイズ, 構成に特徴は見つからない



- ▶ サイズや構成を変えてもコードベロシティに影響はない
 - 個人の特性, 対人関係をメトリクスにするとどうか?

所感

- ▶ 所属や曜日を分析に使うのはおもしろい
 - マージまでの時間に影響がないのは意外だった
- ▶ 結局じゃあコードベロシティを向上させる方法は？