

Why We Refactor Confessions of GitHub Contributors

蘆田 誠人

目次

- 背景
- 関連研究
- 研究結果
- まとめ

背景

- リファクタリングの実践の動機を調査する研究は数少ない
- 開発者がどのような状態でリファクタリングしているか
不透明

関連研究

リファクタリングリスクに関する研究

- API等でリファクタリング後のバグ修正数が増加していることが検証済みである(Kim et al.)
- バグレポートの割合が増加することが示されている(Rachatasumrit et al.)

→ 特定のリファクタリング操作を詳細に調査していない

関連研究

- Tsantalisらが行った研究

リファクタリングを行う前後で関連するソースコードを手動で調査し、
リファクタリングの可能性を明らかに

→ リファクタリングを行った開発者にリファクタリングをする理由
をインタビュー

関連研究

- RefactoringMinerツール

2つのバージョンのソースの違いを分析

→ バージョンごとに実行されたリファクタリングを検索

本研究ではツールが検出した12の有名なリファクタリングタイプを
分析する

RefactoringMinerの精度

- RefactoringMinerで検出したリストと既知のリファクタリングの比較
 - ✓ 2人の学生が変更前変更後で120のインスタンスを選択

Table 1: RefactoringMiner Recall and Precision

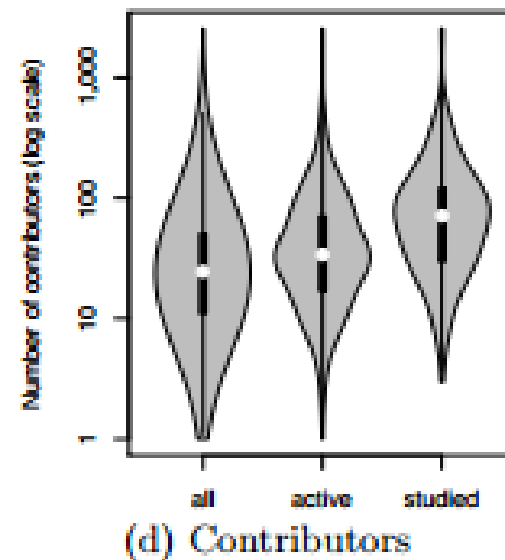
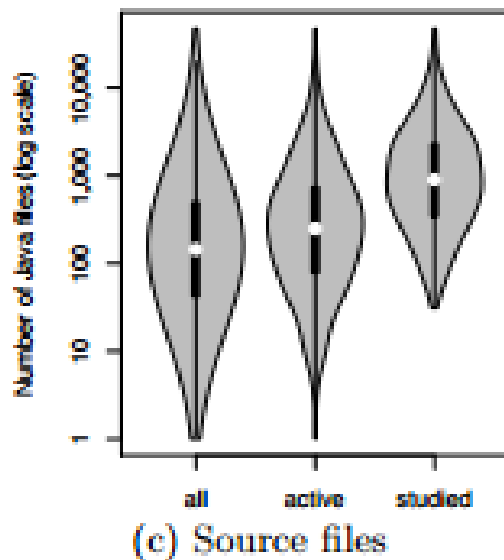
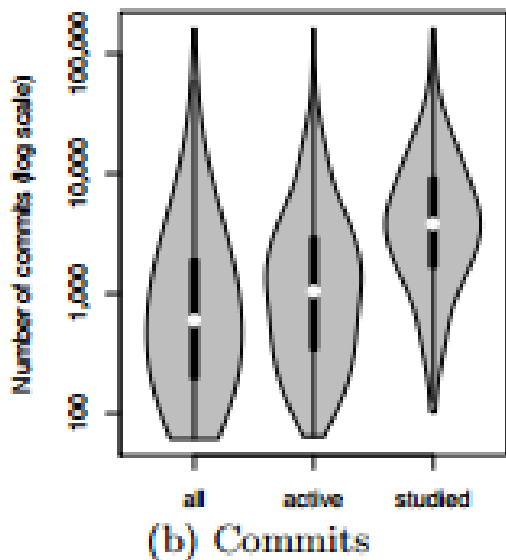
Refactoring	TP	FP	FN	Recall	Prec.
EXTRACT METHOD	15	2	0	1.00	0.88
INLINE METHOD	13	0	2	0.87	1.00
PULL UP ATTRIBUTE	15	0	0	1.00	1.00
PULL UP METHOD	12	0	3	0.80	1.00
PUSH DOWN ATTRIBUTE	15	0	0	1.00	1.00
PUSH DOWN METHOD	13	0	2	0.87	1.00
MOVE ATTRIBUTE	15	0	0	1.00	1.00
MOVE METHOD	13	0	2	0.87	1.00
Total	111	2	9	0.93	0.98

リポジトリの選択

●GitHubリポジトリの選択

フォークでないGitHubで人気のある1000のJavaリポジトリを選択

より多くの保守活動を行っているリポジトリ → 748



アプローチ

- 関連研究はコミットを全く考慮せずに行われていた
- 本研究ではコミットを考慮して精度を計測

リファクタリングの審査

- 調査期間中に748のプロジェクトを監視

→ 2241個のインスタンスが検出

→ 実際にリファクタリングが行われていたか手動で調査

リファクタリングの審査

○

Table 2: Refactoring activity

Refactoring	TP	FP	Prec.	Com.	Proj.
EXTRACT METHOD	468	135	0.78	312	136
MOVE CLASS	432	512	0.46	85	60
MOVE ATTRIBUTE	129	44	0.75	45	38
RENAME PACKAGE	105	0	1.00	25	24
MOVE METHOD	99	48	0.67	40	31
INLINE METHOD	58	67	0.46	44	36
PULL UP METHOD	33	1	0.97	18	17
PULL UP ATTRIBUTE	23	1	0.96	11	11
EXTRACT SUPERCLASS	22	11	0.67	18	16
PUSH DOWN METHOD	16	1	0.94	6	6
PUSH DOWN ATTRIBUTE	15	1	0.94	7	7
EXTRACT INTERFACE	11	8	0.58	10	9
Total	1411	830	0.63	539	185

- コミットに変化があれば
リファクタリング誤検出が増える
- Table1よりも精度が下がっている

リファクタリングの自動化

- リファクタリング自動化についてのインタビュー

222人の開発者が回答

Table 5: Manual vs. automated refactoring

Modification	Occurrences	
Manual	95	
Automated	66	
Not answered	48	
Partially automated	13	

なぜ手動で行うのか

●リファクタリングを自動化しない理由をインタビュー

29人の開発者が回答

手動で行う理由	人数
複雑なリファクタリングの分析を信頼できない	10
手動リファクタリングが簡単だからそもそも不要	8
IDEが対応していない	6
使っているIDEのリファクタリング機能に精通していない	3
リファクタリングツールがあることを知らなかった	2

タイプ別の自動化率

- リファクタリングを行ったタイプ別自動化率

256の作業数进行分析

Table 6: Refactoring automation by type

Refactoring Type	Occurrences	Automated %
EXTRACT METHOD	118	29%
MOVE CLASS	36	50%
MOVE ATTRIBUTE	21	24%
MOVE METHOD	19	26%
INLINE METHOD	17	18%
RENAME PACKAGE	12	58%
EXTRACT SUPERCLASS	9	11%
PULL UP METHOD	9	11%
PULL UP ATTRIBUTE	7	14%
EXTRACT INTERFACE	3	0%
PUSH DOWN ATTRIBUTE	3	33%
PUSH DOWN METHOD	2	0%

まとめ

- リファクタリングツールはコミットの変更が複雑になると精度が落ちる
- 自動化ができるものの55%の開発者が手動で行っている
 - ✓ ツールを信頼できない
 - ✓ 必要ない
- パッケージの名前変更やクラスの移動を自動的に行っている割合が高い