

開発経験の違いが技術的負債の 報告内容と解決時間に与える影響 の分析

オープンソースソフトウェア工学研究室
学籍番号：60246226
氏名：花田 龍汰郎

技術的負債 (Technical Debt, TD) [1]

- 短期的な利益を目的に、最適とは言えない設計や実装を行うことで発生するコストを表すメタファ
- 導入されることで、プログラムの品質低下、新規機能の追加の妨げなどが起こる
- 早期に発見、修正することが必要

[1] Ward Cunningham. The wycash portfolio management system. In 7th Objectoriented Programming Systems, Languages, and Applications (OOPSLA), pp. 29–30, 1992.

Self-Admitted TD (SATD)

- 開発者が認識し文書化，報告されるTD
- SATD-Issue[2]
 - 課題票を用いて報告されるSATD

History Main Service: bad reference to 'window'? #79426

Closed bpasero opened this issue on Aug 19, 2019 · 0 comments

bpasero commented on Aug 19, 2019

This code seems like we could remove it and it was likely never working because `window` is the DOM global `window`:

```
vscode/src/vs/platform/history/common/historyStorage.ts
Line 74 in 582b187
74    } else if (isUriComponents(window)) {
```

bpasero added the **debt** label on Aug 19, 2019

bpasero assigned aeschli on Aug 19, 2019

bpasero mentioned this issue on Aug 20, 2019

Green TSLint #79454

aeschli added this to the **August 2019** milestone on Aug 20, 2019

Assignees
aeschli

Labels
debt

Projects
None yet

Milestone
August 2019

Development
No branches or pull requests

2 participants
aeschli, bpasero

TDを表すラベル

Labels

debt

[2] Laerte Xavier, Fabio Ferreira, Rodrigo Brito, and Marco Tulio Valente. Beyond the code: Mining self-admitted technical debt in issue tracker systems. MSR '20: Proceedings of the 17th International Conference on Mining Software Repositories, pp. 137–146, 2020.

背景

時間的制約により開発者が修正できるTDには限りがある
→TDに優先順位をつけることが重要

問題

先行研究では、TDの検出ツールやリファクタリングツールが数多く提案

しかし、検出されたTDに優先順位がついていない

→開発者はTDを効率的に返済することができていない

背景

どの開発者がどんなTDを修正することができるのかを把握する

→適切なタスク割り当てを行うことができる

しかし，開発経験により報告・解決できるTDの種類や内容を分析した研究は存在しない

目的とアプローチ

目的

- タスク割り当てを用いた**SATD-Issue**の優先順位付け手法の構築へ向けた知見を得ること

アプローチ

- 報告される**TD**の種類と内容
- **TD**の修正に要する時間

開発経験の差によるそれぞれの違いを調査

開発経験の定義

- 開発者の開発経験を，プロジェクト内で開発者がおこなった**Issue**報告の数と定義
- 開発者を開発経験でランク付け
 - 上位25%を開発経験が豊富な開発者（High）
 - 下位25%を開発経験が乏しい開発者（Low）と定義
- 開発経験の違いによる影響について調査

データセット

Githubに存在し，スター数上位1000件の内，
SATD-Issue を扱っている4プロジェクトを対象

各プロジェクトのIssueからclosed状態の
SATDを抽出

表 3.1: データセット

プロジェクト	期間	Issue 数 (件)	SATD-Issue 数 (件)	開発者数 (人)
vscode	2015 年 10 月～2022 年 10 月	124,918	2,020	53,013
influxdb	2013 年 10 月～2022 年 10 月	10,891	120	3,189
saleor	2013 年 2 月～2022 年 10 月	2,913	109	837
server	2016 年 6 月～2022 年 10 月	11,769	89	5,878

リサーチクエスション

- RQ1: 開発経験と報告・解決されるSATD-Issueの数との関係
 1. 開発経験はSATD-Issue の報告数に影響を与えるか？
 2. 開発経験はSATD-Issue の解決数に影響を与えるか？
- RQ2: 開発経験と報告・解決されるSATD-Issueの種類との関係
 1. 開発経験は報告されるSATD-Issue の種類に影響を与えるか？
 2. 開発経験は解決されるSATD-Issue の種類に影響を与えるか？

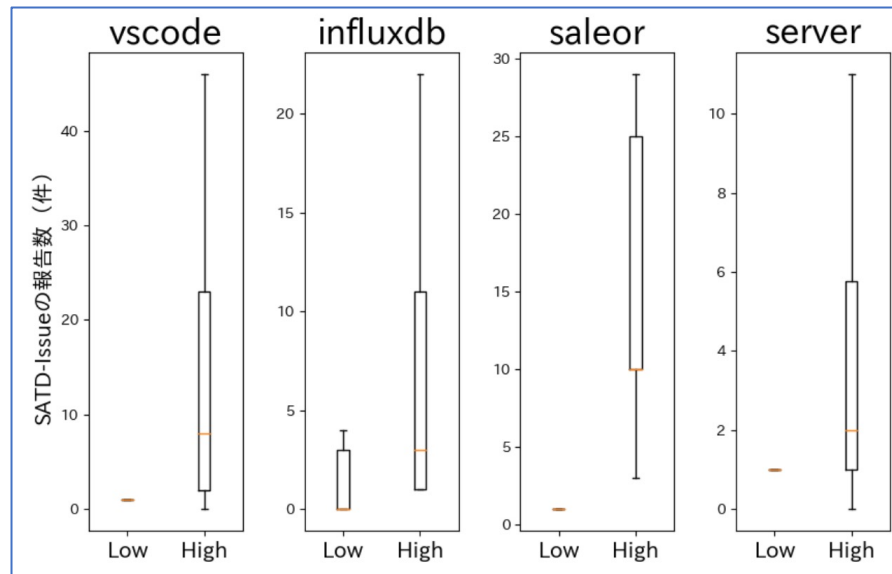
リサーチクエスション

- RQ3: 開発経験とSATD-Issueの解決に要する時間との関係
 - 開発経験はSATD-Issue の解決に要する時間に影響を与えるか？
- RQ4: SATD-Issueの種類と解決時間との関係
 - SATD-Issue の種類はSATD-Issue の解決時間に影響を与えるか？

RQ結果

- RQ1: 開発経験と報告・解決されるSATD-Issueの数との関係

SATD-Issueの報告数

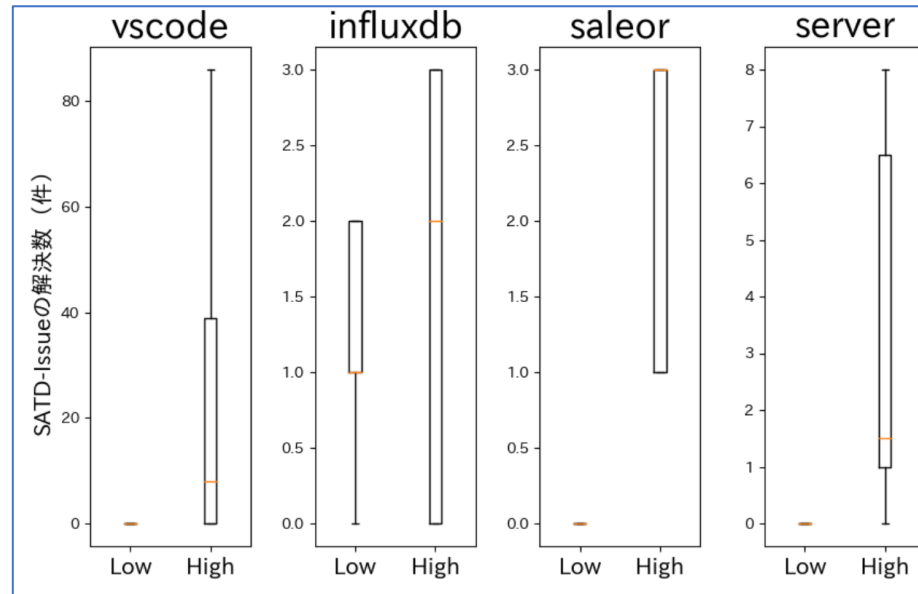


開発経験が豊富な開発者は、多くのSATD-Issueの報告を行う

RQ結果

- RQ1: 開発経験と報告・解決されるSATD-Issueの数との関係

SATD-Issueの解決数



開発経験が豊富な開発者は、多くのSATD-Issueの解決を行う

RQ結果

• RQ2: 開発経験と報告・解決される SATD-Issue の種類との関係

SATD-Issueの種類ごとに報告される件数 (vscode)

種類	Low	High
Requirements	2 (4.0%)	5 (0.6%)
Design	39 (78.0%)	717 (84.3%)
Test	1 (2.0%)	59 (6.9%)
UI	0 (0%)	19 (2.2%)
Performance	1 (2.0%)	17 (2.0%)
Security	0 (0%)	0 (0%)
Documentation	1 (2.0%)	3 (0.4%)
Infrastructure	5 (10.0%)	10 (1.2%)
Build	1 (2.0%)	6 (0.7%)
Code Style	0 (0%)	15 (1.8%)
合計	50 (100%)	851 (100%)

開発経験に関わらず報告されるSATD-Issueは”Design”に分類されるTDが多かった
開発経験が豊富な開発者が報告するSATD-Issueが多かった

RQ結果

• RQ2: 開発経験と報告・解決される SATD-Issue の種類との関係

SATD-Issueの種類ごとに解決される件数 (vscode)

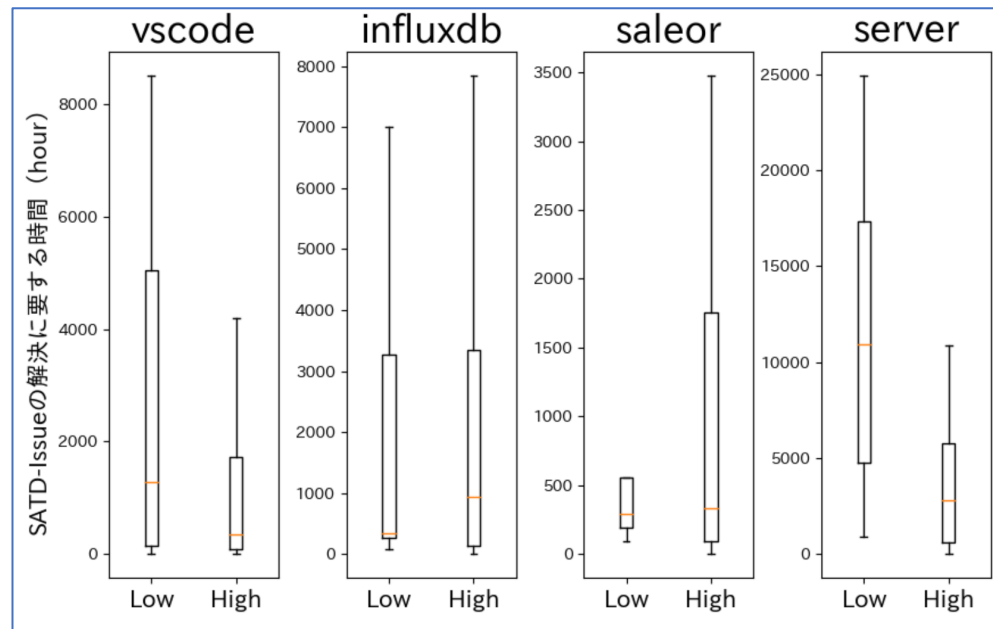
種類	Low	High
Requirements	0 (0%)	7 (0.7%)
Design	9 (60.0%)	799 (83.4%)
Test	0 (0%)	64 (6.7%)
UI	4 (26.7%)	18 (1.9%)
Performance	0 (0%)	24 (2.5%)
Security	0 (0%)	0 (0%)
Documentation	1 (6.7%)	4 (0.4%)
Infrastructure	1 (6.7%)	16 (1.7%)
Build	0 (0%)	10 (1.0%)
Code Style	0 (0%)	16 (1.7%)
合計	15 (100%)	958 (100%)

開発経験に関わらず報告されるSATD-Issueは”Design”に分類されるTDが多かった
開発経験が乏しい開発者が解決するSATD-Issueはごく僅かであった

RQ結果

- RQ3: 開発経験と SATD-Issue の解決に要する時間との関係

SATD-Issueの解決に要する時間



各プロジェクト毎に解決する時間に一貫性が見られなかった

RQ結果

• RQ4: SATD-Issue の種類と解決時間との関係

SATD-Issueの種類ごとの解決時間 (vscode)

種類	Low	High
Requirements	11991.0	715.0
Design	615.0	334.0
Test	2524.0	291.0
UI	-	1079.0
Performance	15242.0	334.0
Security	-	-
Documentation	23.0	1944.0
Infrastructure	3992.0	300.5
Build	4118.0	1891.0
Code Style	-	297.0

SATD-Issueの種類ごとに解決に要する時間は大きく異なっていた
開発経験の違いによるSATD-Issueの解決時間も種類ごとに大きく異なっていたが、
ほとんどの場合開発経験が豊富な開発者の方が解決時間が短かった

考察

- **vscode**以外のプロジェクトに参加している開発者数が少ないこと
- 開発経験が乏しい開発者が解決しているSATD-Issueがごく僅か



データに偏りが発生

開発経験が豊富な開発者は、解決に時間を要するSATD-Issueに携わっている可能性がある

SATD-Issueの追加調査結果

開発経験が豊富／乏しい開発者が報告するSATD-Issueの内容を調査

上位ではどのように変更を行うのか，なぜ変更を行うのかが明確にされているものが多数

下位では簡素な内容が報告されていることが多数

開発経験の違いによって解決されるSATD-Issueの内容が異なるため，開発経験が豊富な開発者でも解決に時間を要する場合がありますと考えられる

まとめ

目的

- タスク割り当てを用いたSATD-Issueの優先順位付け手法の構築へ向けた知見を得ること

結果

- 報告数，解決数ともに“Design”に分類されるTDが多く，開発経験が乏しい開発者はほとんどSATD-Issueを解決できない
- SATD-Issueの種類が異なると解決時間が異なり，開発経験が豊富な開発者の解決時間が開発経験が乏しい開発者の解決時間よりほとんど短い
- 開発経験の違いによってSATD-Issueの内容に違いが生じる

補足資料

TDの種類

種類	説明
Requirement	実装上の問題を引き起こす要求仕様上の技術的負債
Design	メソッドやアーキテクチャで使用される技術的なショートカット
Test	テストの不足やテスト用の既存コードに回避策がある技術的負債
UI	ユーザーインターフェイスに関連する技術的負債
Performance	実行時間やメモリ使用率などのシステム性能に影響する技術的負債
Security	システムのデータやユーザのアクセス権限などのショートカット
Documentation	不十分, または古いドキュメント
Infrastructure	外部のツールや陳腐化した技術, 非推奨な API による技術的負債
Build	ビルドを複雑にしたり実行を遅くする技術的負債
Code Style	コーディング規約違反