

開発者が管理する技術的負債 に関する課題票の内容理解

オープンソースソフトウェア工学研究室

学籍番号：60219001

氏名：木村 祐太

技術的負債：Technical Debt^[1]

時間的制約によって導入される低品質な設計や実装

- 例) コードスメル：深刻な問題を抱えているコードの特徴

保守コスト増加を引き起こす^{[3][4][5][6]}

- 開発の約23～35%が余分な作業
- 既存のTDによる新たなTDの導入（TDの蓄積）

開発資源（人的，金銭的，時間的）は有限
TDの蓄積量を管理する必要がある

[1] Ward Cunningham. The wycash portfolio management system. SIGPLAN OOPS Mess., 4(2):29–30, 1992.

[3] Carolyn Seaman, Yuepu Guo, Nico Zazworka, Forrest Shull, Clemente Izurieta, Yuanfang Cai, and Antonio Vetr`o. Using technical debt data in decision making: Potential decision approaches. In Proceedings of the 3rd International Workshop on Managing Technical Debt, MTD, pages 45–48, 2012.

[4] Edith Tom, Ayb uKe Aurum, and Richard Vidgen. An exploration of technical debt. 86(6):1498–1516, 2013.

[5] Terese Besker, Antonio Martini, and Jan Bosch. The pricey bill of technical debt: When and by whom will it be paid? In Proceedings of the 33rd International Conference on Software Maintenance and Evolution, ICSME, pages 13–23, 2017.

[6] Terese Besker, Antonio Martini, and Jan Bosch. Technical debt cripples software developer productivity: A longitudinal study on developers 'daily software development work. In Proceedings of the 1st International Conference on Technical Debt, TechDebt, pages 105–114, 2018.

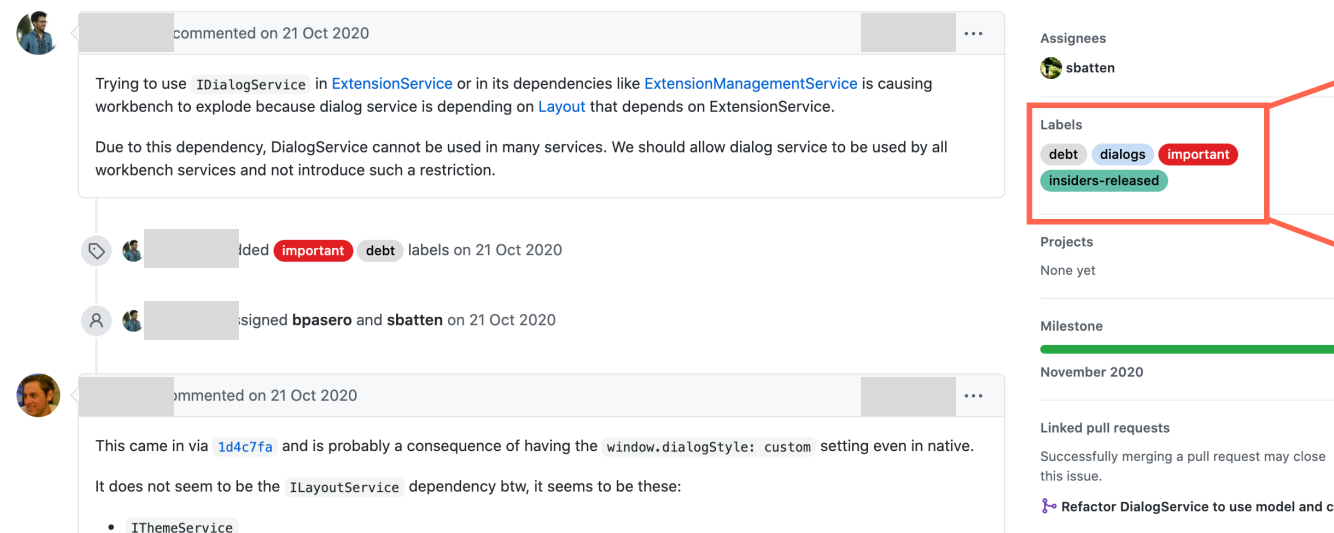
Self-Admitted Technical Debt

開発者によってTDが明示的に管理される事象

- SATD-Issue^[7]
 - 課題票 (Issue) にTDラベルを割り当てて明示されるSATD
- ※ Issue : 不具合, 機能改善, 機能要求など様々な課題が管理される

Cannot use IDialogService in some services due to custom dialog service dependencies
#109056

 Closed sandy081 opened this issue on 21 Oct 2020 · 13 comments



“debt”というTDラベルが
割り当てられている

引用 : <https://github.com/microsoft/vscode/issues/109056>

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

TD管理支援の問題点

SATD-Issueを扱っているプロジェクトが増加

TDの検出

SonarQubeやSpotBugsといった静的コード解析ツールによる検出

- ツールに用意されている検出ルールでは発見できないTDが存在 [?]

TDの返済（除去）

IDEに実装されている自動リファクタリング*による変更

- 保守性を向上するための開発者が期待する変更結果が得られない [?]

リファクタリング*：コードの機能を変更するのではなく、実装方法を変更しコードの品質を向上させる方法

機械的に検出や返済ができないため、
開発者はTDを意識的に管理する必要がある

博士後期課程での研究計画

開発者が管理するTDを理解し, TD管理を容易にするためのTD検出手法を構築する

(1年次・2年次)

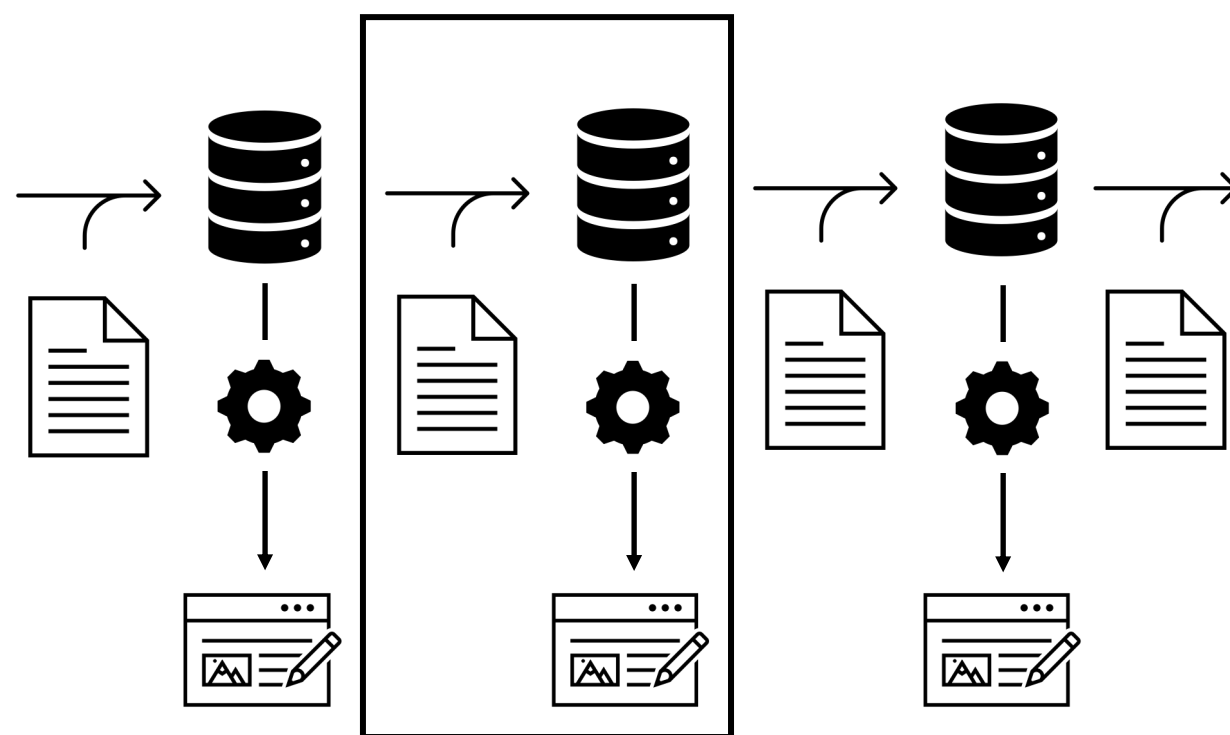
- TDの問題と対応するソースコードの特徴の調査
 - SATD-Issueに含まれる報告内容の分析
 - SATD-Issueに紐づく変更内容の分析

(3年次)

- 1年次, 2年次での調査をもとにTD検出手法の構築

TD検知手法のイメージ

コミットがされたタイミングで手法を適用



エンティティ



ソフトウェア製品



変更

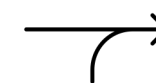


TD検出モデル



検知結果

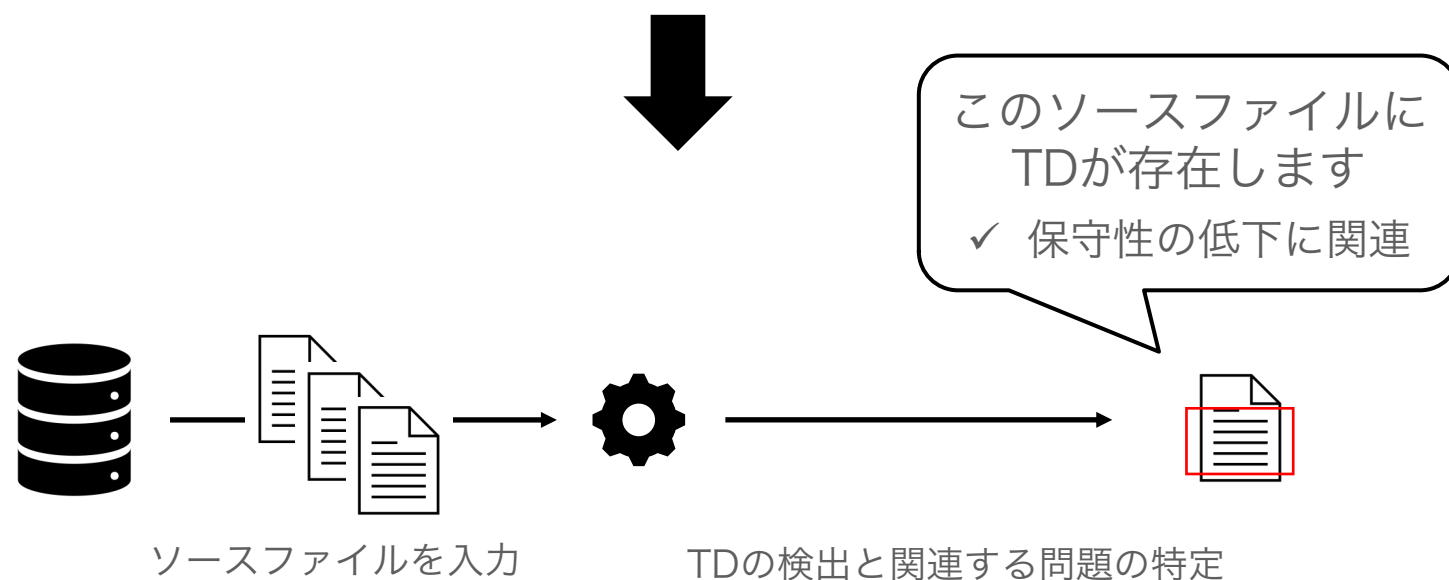
プロセス



変更の反映（コミット）



モデルの適用と結果の出力



1年次・2年次の研究計画

目的

- TD検知手法を構築するために，開発者が管理するTDの内容や特徴について理解する

方法

- SATD-Issueをいくつかの観点で分類（現在取り組んでいること）
 - 既存研究の分類カテゴリ
 - ソフトウェア製品品質
- 分類したカテゴリごとにSATD-Issueに紐づく変更内容などを分析
 - 特定の品質を低下させるコードの特徴など

データセット

| TDに関するラベルを保有するプロジェクトを抽出

- ラベル名に“debt”を含むもの

表を載せる

今後の予定

| 収集したSATD-Issueを目視で分類

- 各カテゴリの具体例を用意する