

課題解決コミットに含まれる 変更情報を用いた技術的負債の検出

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

DXとシステム保守による障壁

DX : Digital Transformation

- 新たな技術の登場や顧客のニーズに合わせて容易に進化できるシステムを開発すること
 - 世界的にあらゆる物事のデジタル化が推し進められており、情報システムの競争力を確保する必要性がある
- 継続的なシステムの価値創出（バリューアップ）が重要

システム保守によるDX推進の障壁

- 開発予算の大部分が保守コストとして浪費
 - 日本では約8割、米国では約6割が保守コストに費やされている
 - そのため、継続的なシステムのバリューアップに注力できない
- DXの障壁の一つである技術的負債の存在が注目されている**

技術的負債：Technical Debt^[1]

時間的制約などによって導入されるソフトウェアの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

- スパゲッティコード：処理が絡み合う複雑な構造を持つコード
- ゴッドクラス：あらゆる機能が詰め込まれたクラス

プログラム難読化や複雑化による保守コスト増加^{[2][3]}

- 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）

→ 早期にTDを発見し取り除くことが保守コストの低減に直結

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

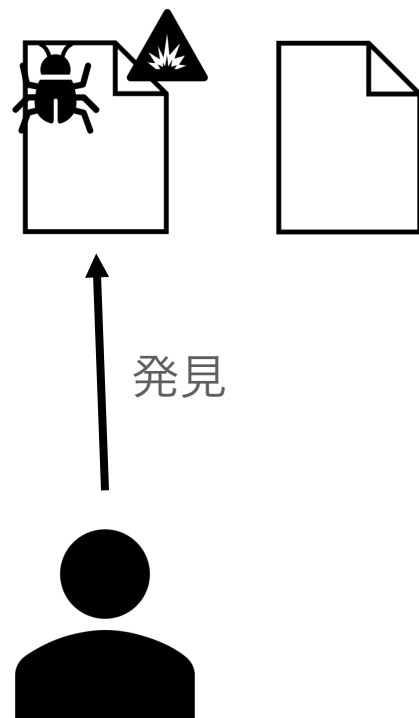
[2] Terese Besker, Antonio Martini, and Jan Bosch. The pricey bill of technical debt: When and by whom will it be paid? In Proceedings of the 33rd International Conference on Software Maintenance and Evolution, ICSME, pages 13-23, 2017.

[3] 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.

TD発見における課題と支援

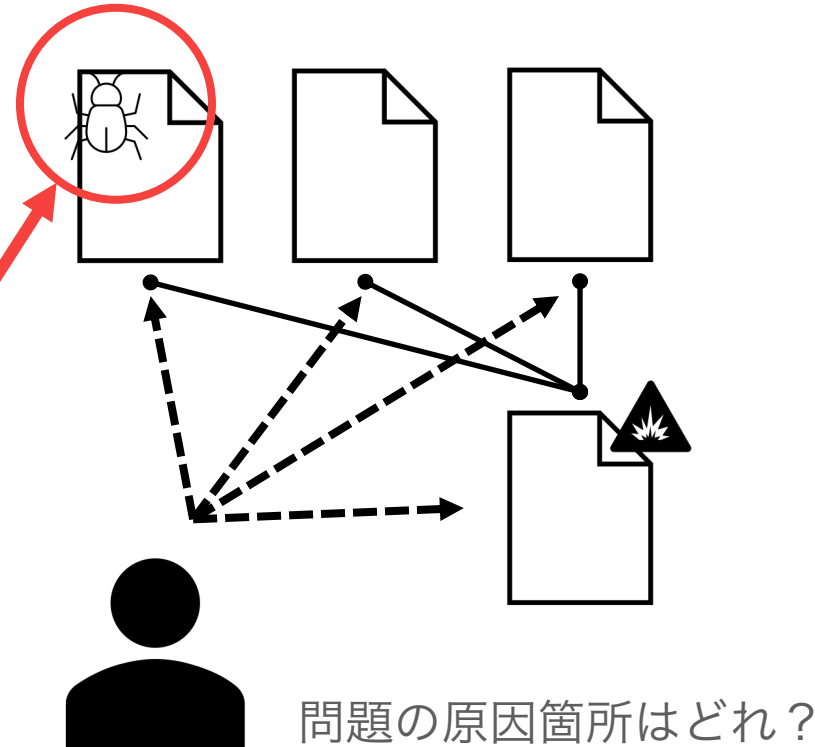
直接的に問題が顕在化しないために発見が困難

バグや欠陥は混入と
問題の発生時期がほぼ同じ
エラーが出る



検出ツール

TDの混入と
問題の発生時期が異なる
エラーが出ない



凡例

- ▲ 問題の発生
- 🐛 バグや欠陥
- 🐛 TD
- 📄 ソースファイル
- 👤 開発者
- 依存関係

TD発見支援のために検出ツールが提案・開発されている^{[4][5]}

[4] SonarQube, <https://www.sonarqube.org/>

[5] FindBugs, <https://findbugs.sourceforge.net/>

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^[6]
 - 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
- ツールを使わずに人手によって報告される

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

commented on 21 Oct 2020

Trying to use `IDialogService` in `ExtensionService` or in its dependencies like `ExtensionManagementService` is causing workbench to explode because dialog service is depending on `Layout` that depends on `ExtensionService`.

Due to this dependency, `DialogService` cannot be used in many services. We should allow dialog service to be used by all workbench services and not introduce such a restriction.

Added `important` `debt` labels on 21 Oct 2020

Assigned `bpasero` and `sbatten` on 21 Oct 2020

commented on 21 Oct 2020

This came in via `1d4c7fa` and is probably a consequence of having the `window.dialogStyle: custom` setting even in native. It does not seem to be the `ILayoutService` dependency btw, it seems to be these:

- `IThemeService`

Assignees
`sbatten`

Labels
`debt` `dialogs` `important` `insiders-released`

Projects
None yet

Milestone
November 2020

Linked pull requests
Successfully merging a pull request may close this issue.
[Refactor DialogService to use model and c...](#)



“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>

[6] 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検出の新たな課題

SATD-Issueで扱われるTDを検出できる手法の構築

- SATD-Issueはツールを使わずに報告される^[?]ため、ツールでカバーできない可能性が高い

手法構築の必要性

- SATD-Issueで扱われるTDの内容を理解し、その内容と合致するコードなどを探索することは労力が大きい
 - 開発されるシステムの規模が大きいほど労力が大きくなる
 - 完全に同じ状態である可能性が少ない
- 特に企業開発の場合、初期コストの敷居の高さからTDの理解や探索がおこなわれない可能性がある

博士後期課程の研究計画

目標

- 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

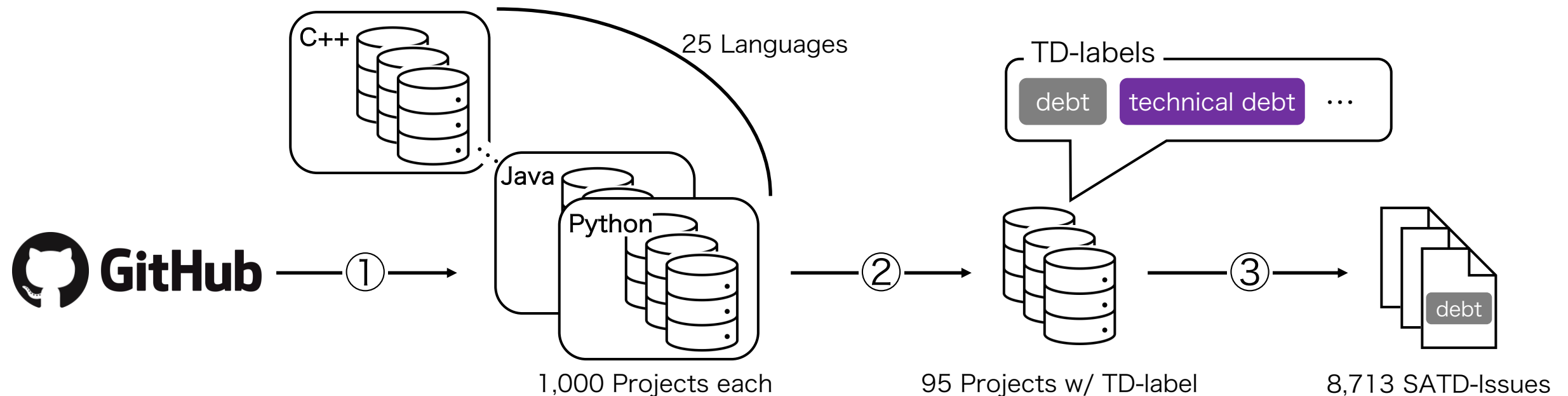
研究課題

1. SATD-Issueで扱われる内容は？
 - (1-1) SATD-Issueのカテゴライズ（ラベル付け）
 - (1-2) SATD-Issueの解決でおこなわれる変更の分析
2. 1.で明らかにした特徴をもとにTD検出ができるか？

データセット

* GitHubで人気がある
25のプログラミング言語
(補足資料記載)

8,713件の解決済みSATD-Issueを収集



1. GitHubにホスティングされているプロジェクトを収集

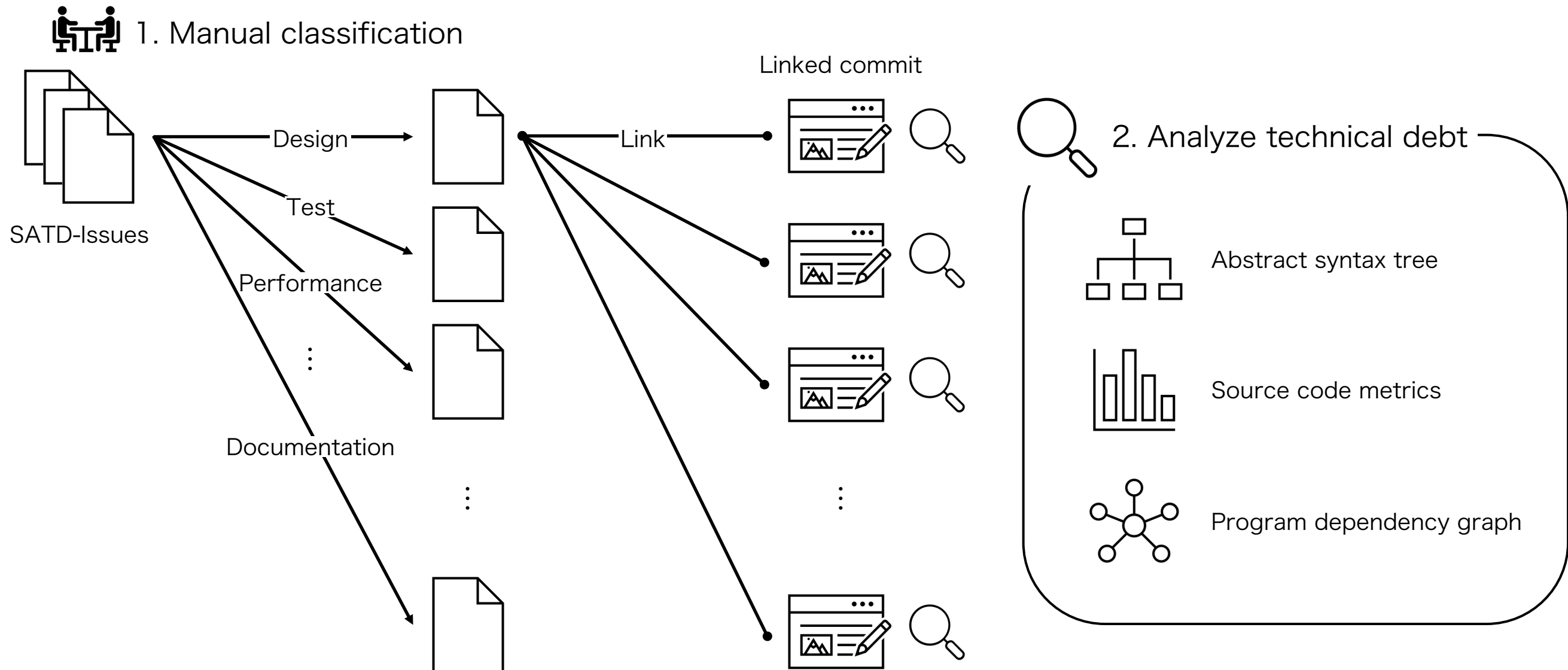
- GitHub GraphQL APIを利用して、使用している開発言語*別にスター数上位1,000プロジェクトを収集

2. TDラベルを保有するプロジェクトを抽出

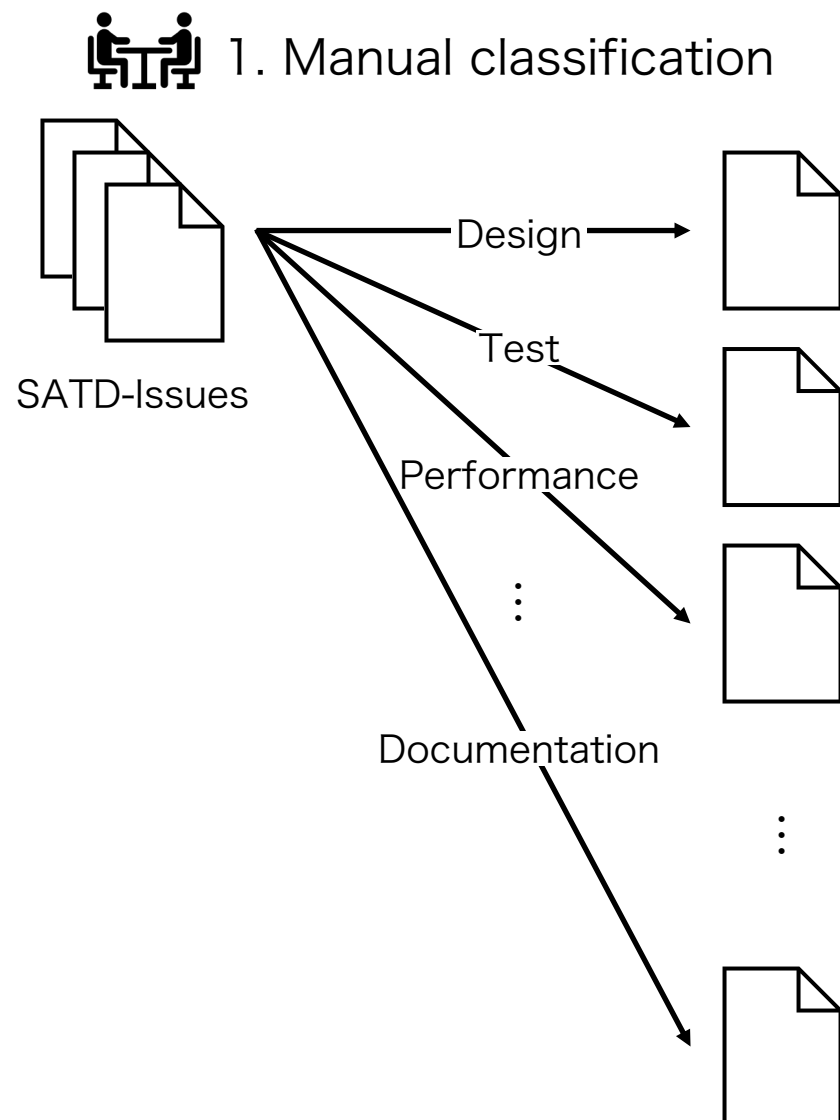
- ラベル名に“debt”を含むラベルを保有
- 解決済みSATD-Issueが10件以上保有

3. 解決済みSATD-Issueを抽出

SATD-Issueで扱われる内容は？



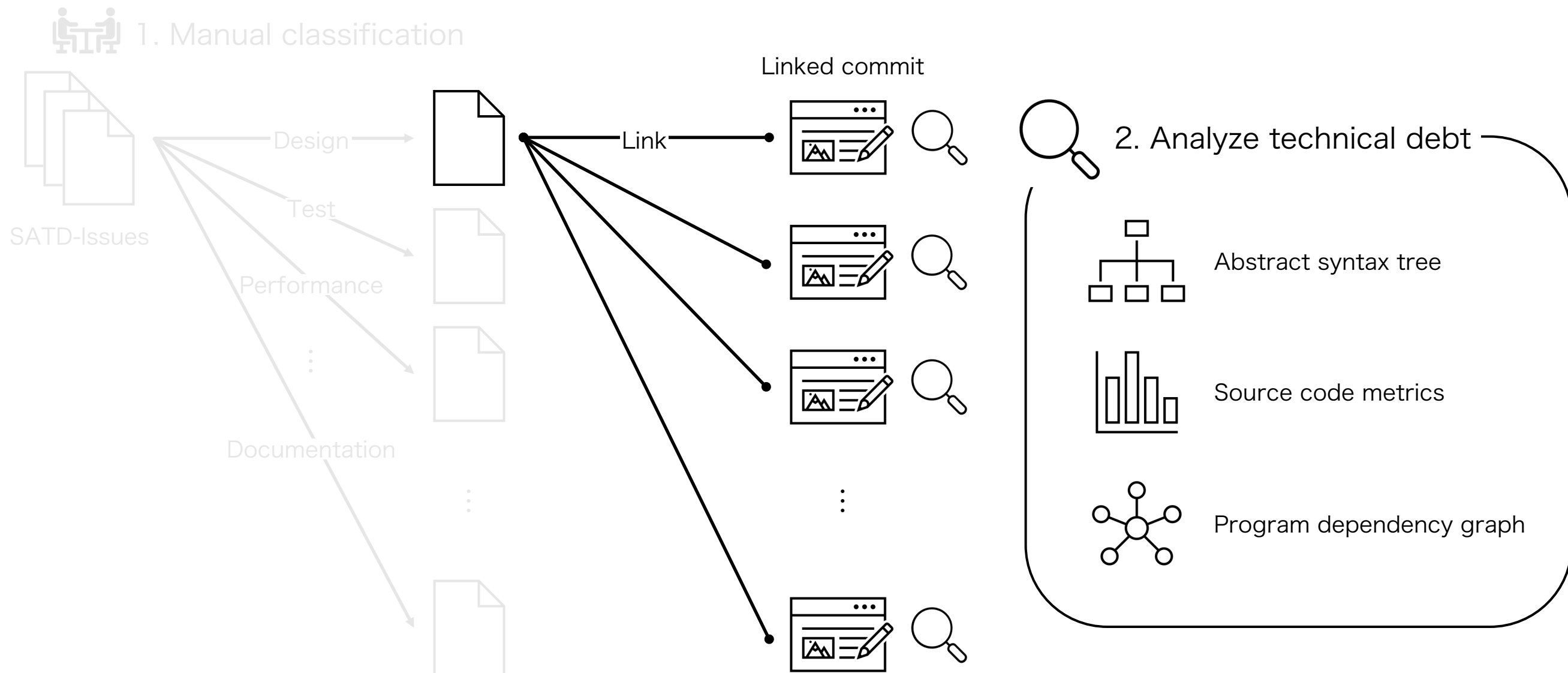
SATD-Issueで扱われる内容は？



分類に利用する10カテゴリ[6]

カテゴリ	説明
設計 (Design)	メソッドやアーキテクチャに関する問題
UI	ユーザインターフェースの改善に関する問題
テスト (Test)	テストケースの不足やテスト方法に関する問題
性能 (Performance)	実行時間やメモリ効率などに関する問題
インフラストラクチャ (Infrastructure)	サードパーティ製品やAPIの非推奨化に関する問題
ドキュメント (Documentation)	正しくない記述や記述の不足に関する問題
コードスタイル (Code Style)	コードスタイルの違反に関する問題
ビルド (Build)	ビルドの複雑化などを招くスクリプトに関する問題
セキュリティ (Security)	データの保全やユーザの権限に関する問題
要求 (Requirement)	実装ミスなどを引き起こす要求仕様に関する問題

SATD-Issueで扱われる内容は？



SATD-Issueに紐づくコミットの変更情報を用いて、TDとなるコードのパターンなどを分析

まとめ

目標

- 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

研究課題

1. SATD-Issueで扱われる内容は？

(1-1) SATD-Issueのカテゴリライズ（ラベル付け） ← 現在取り組んでいること

(1-2) SATD-Issueの解決でおこなわれる変更の分析

2. 1.で明らかにした特徴をもとにTD検出ができるか？

今後の予定

- SATD-Issueのカテゴリライズの継続
- カテゴリライズに基づくSATD-Issueの変更内容の分析

研究業績

論文誌

- 木村 祐太, 大平 雅雄, “技術的負債に関連する課題票分類手法の構築”, 情報処理学会, 2023, 64巻, 1月号

口頭発表

- 木村 祐太, 大平 雅雄, “技術的負債に関連する課題票分類手法の構築”, ソフトウェア工学研究会, 2022, 3月
- 田口 舞奈, 木村 祐太, 大平 雅雄, “技術的負債に関する課題票の分類モデルに単語分散表現が与える影響の分析”, ソフトウェア工学の基礎研究会, 2022, 11月

SATD-Issueの具体例（1）

例）設計のSATD-Issue

remove usage of EventSource in favour of Event & Emitter #1522

✓ Closed

ed this issue on 21 Dec 2015 · 3 comments



mented on 21 Dec 2015

Member ...

Event was a stepping stone towards the new event mechanism we now use. In favour for clarity and a compact codebase we should remove usages and implementation of EventSource et al. Staring with @

Assignees

Labels

コードベースの明確化・縮小化のためにあるモジュールの実装を取り除く

例：<https://github.com/microsoft/vscode/issues/1522>

- 不要な実装は開発製品に対する開発者の理解を低下させるため、取り除いておくことが望ましい

SATD-Issueの具体例 (2)

例) 性能のSATD-Issue

[folding] Code Folding in larger files slows down editor #3582

規模が大きいファイルの中でコード折り畳みはエディタの動作を遅くする



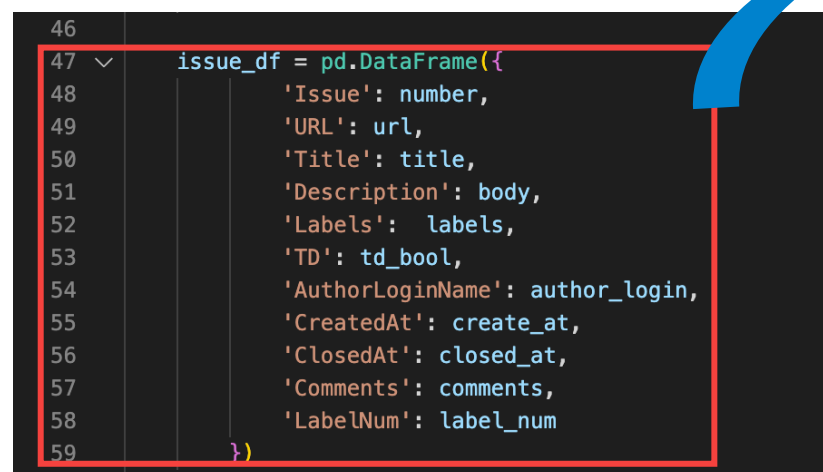
commented on 1 Mar 2016

...

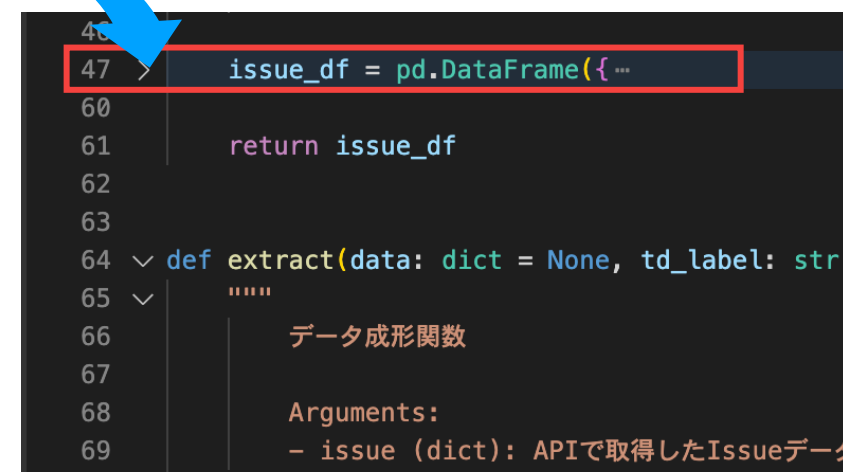
Assignees

例: <https://github.com/microsoft/vscode/issues/2511>

- コード折り畳み (Code Folding)
 - 括弧でまとめられたコードを1行に折り畳むエディタの機能



```
46
47  issue_df = pd.DataFrame({
48      'Issue': number,
49      'URL': url,
50      'Title': title,
51      'Description': body,
52      'Labels': labels,
53      'TD': td_bool,
54      'AuthorLoginName': author_login,
55      'CreatedAt': create_at,
56      'ClosedAt': closed_at,
57      'Comments': comments,
58      'LabelNum': label_num
59  })
```



```
46
47  > issue_df = pd.DataFrame({ ...
60
61      return issue_df
62
63
64  def extract(data: dict = None, td_label: str
65  ):
66      データ成形関数
67
68      Arguments:
69      - issue (dict): APIで取得したIssueデー
```

開発言語とデータセットの内訳

GitHubで人気の言語（プロジェクト数）

C (3)	C# (8)	C++ (7)	CoffeeScript (0)	CSS (0)
Dart (3)	DM (0)	Elixir (0)	Go (17)	Groovy (1)
HTML (0)	Java (3)	JavaScript (1)	Kotlin (5)	Objective-C (0)
Perl (1)	PHP (6)	PowerShell (0)	Python (7)	Ruby (3)
Rust (6)	Scala (6)	Shell (0)	Swift (3)	TypeScript (15)

Advanced options

From these owners

In these repositories

Created on the dates

Written in this language

☒ Any Language

Popular

C

C#

C++

CoffeeScript

CSS

Dart

DM

Elixir

Go

Groovy

HTML

Java

JavaScript

Kotlin

Objective-C

Perl

PHP

Repositories options

With this many stars

With this many forks

Of this size

Pushed to

With this license

Return repositories

- 検索のオプションで開発言語が指定できる
- 人気がある言語は“Popular”として扱われている

カテゴリの概要

カテゴリ	説明
設計	メソッドやアーキテクチャに関する問題
UI	ユーザインターフェースの改善に関する問題
テスト	テストケースの不足やテスト方法に関する問題
性能	実行時間やメモリ効率などに関する問題
インフラストラクチャ	サードパーティ製品やAPIの非推奨化に関する問題
ドキュメント	正しくない記述や記述の不足に関する問題
コードスタイル	コードスタイルの違反に関する問題
ビルド	ビルドの複雑化などを招くスクリプトに関する問題
セキュリティ	データの保全やユーザの権限に関する問題
要求	実装ミスなどを引き起こす要求仕様に関する問題