

# ソースコードメトリクスを用いた 課題管理システムに報告される 技術的負債の特徴付け

---

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

# 高度情報化社会における課題

## 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
- システムの**継続的なバリューアップ**を可能にする

## 現状の課題

- システム維持にコストがかかり新規開発に取り組むことが困難
  - 開発予算の約8割がシステム維持費として浪費されている<sup>[1]</sup>

コスト増加の原因の一つとして  
**技術的負債**への関心が高まっている

[1] 経済産業省, DXレポート ～ITシステム「2025年の崖」の克服とDXの本格的な展開～, デジタルトランスフォーメーションに向けた研究会, 2018年9月7日

# 技術的負債：Technical Debt [2]

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

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
- 既存の技術的負債による新たな技術的負債の導入

開発資源（人的，金銭的，時間的）は有限

技術的負債が増大しないよう抑制する必要がある

[2] 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, AybuKe 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.

# 技術的負債検出に関する研究

## 技術的負債の検出

- 静的コード解析による検出<sup>[7]</sup>
- 依存関係解析による検出<sup>[8]</sup>
- 機械学習による検出<sup>[9]</sup>

## 課題

- 既存手法でカバーできない技術的負債の存在
  - 特定の事象にのみ着目 (God Class, Long Method など)
  - 開発者が自ら報告する技術的負債が確認されている

[7] Nico Zazworka, Antonio Vetr`o, Clemente Izurieta, Sunny Wong, Yuanfang Cai, Carolyn Seaman, and Forrest Shull. Comparing four Approaches for Technical Debt Identification. Software Quality Journal, 22:1–24, 2013.

[8] John Brondum and Liming Zhu. Visualising Architectural Dependencies. In Proceedings of the 3rd International Workshop on Managing Technical Debt, MTD, pages 7–14, 2012.

[9] Daniel Cruz, Amanda Santana, and Eduardo Figueiredo. Detecting Bad Smells with Machine Learning Algorithms: An Empirical Study. In Proceedings of the 3rd International Conference on Technical Debt, TechDebt, pages 31–40, 2020.

[10] Emerson Murphy-Hill, Chris Parnin, and Andrew P. Black. How We Refactor, and How We Know It. IEEE Transactions on Software Engineering, 38(1):5–18, 2012.

# Self-Admitted Technical Debt

## 開発者によって技術的負債が明示的に管理される事象

- SATD-Issue<sup>[11]</sup>
    - 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

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>

[11] 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.

# 着眼点と目的

---

## 着眼点

- SATD-Issueを用いて、開発者自らが扱う技術的負債を理解し  
検出技術支援へ活用

## 目的

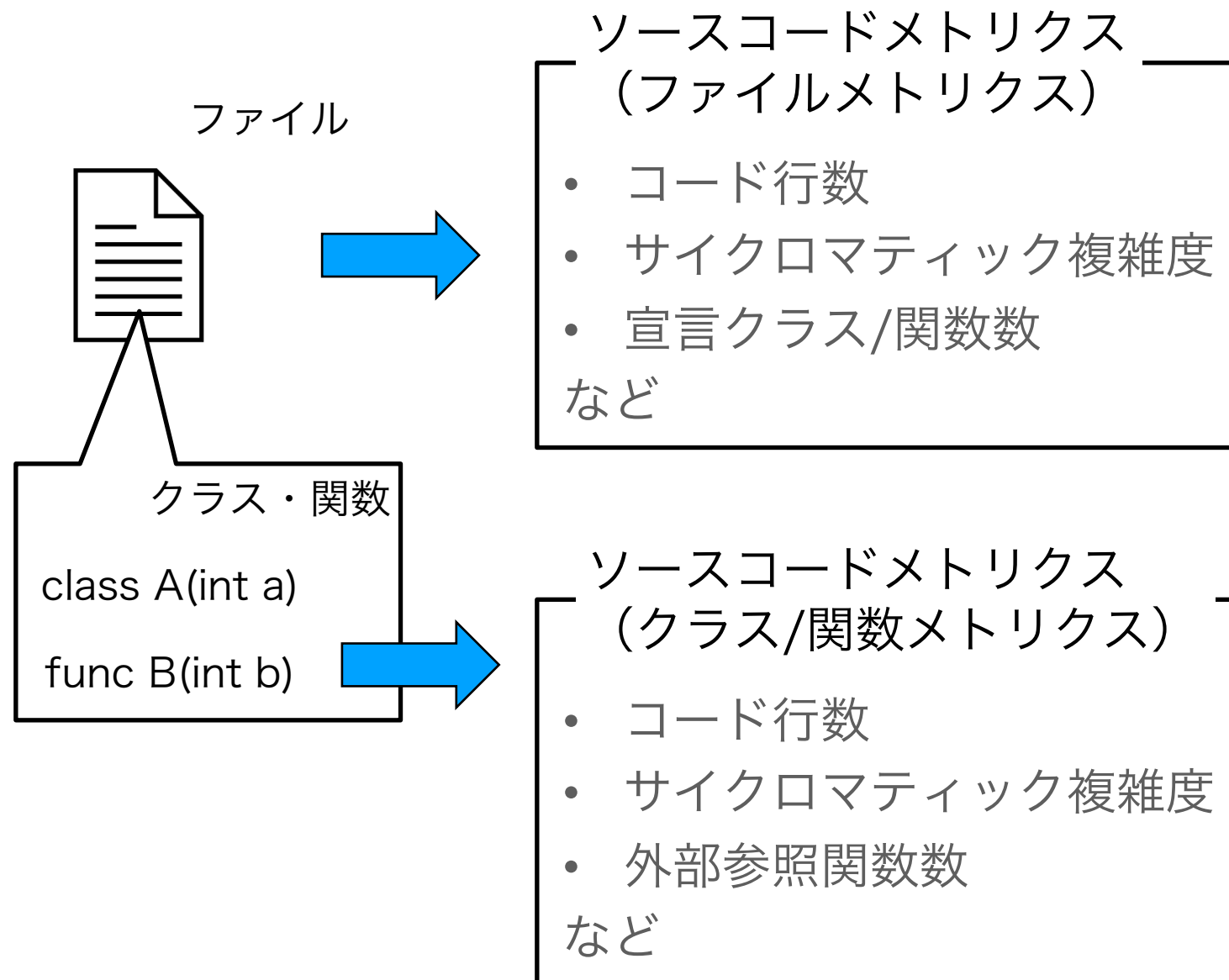
- 開発者自ら発見し取り除く技術的負債の調査に基づいた技術的  
検出手法の提案

### <STEP>

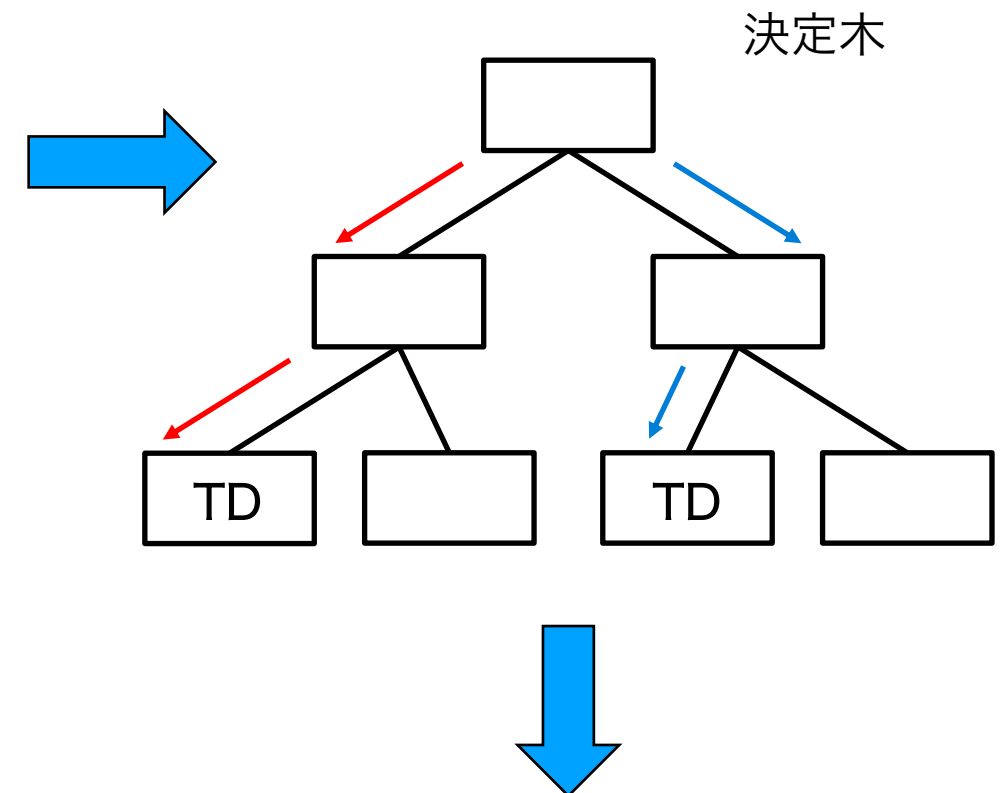
1. ソースコードメトリクスを用いて技術的負債のパターンを分析
2. ソースコードを入力とした深層学習モデルでの検出

# パターン分析の手順

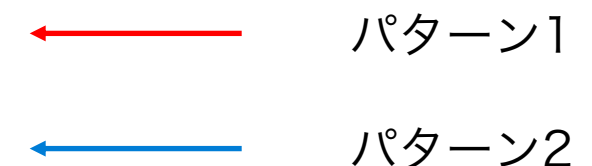
## 1. ソースコードメトリクス計測



## 2. 決定木の学習



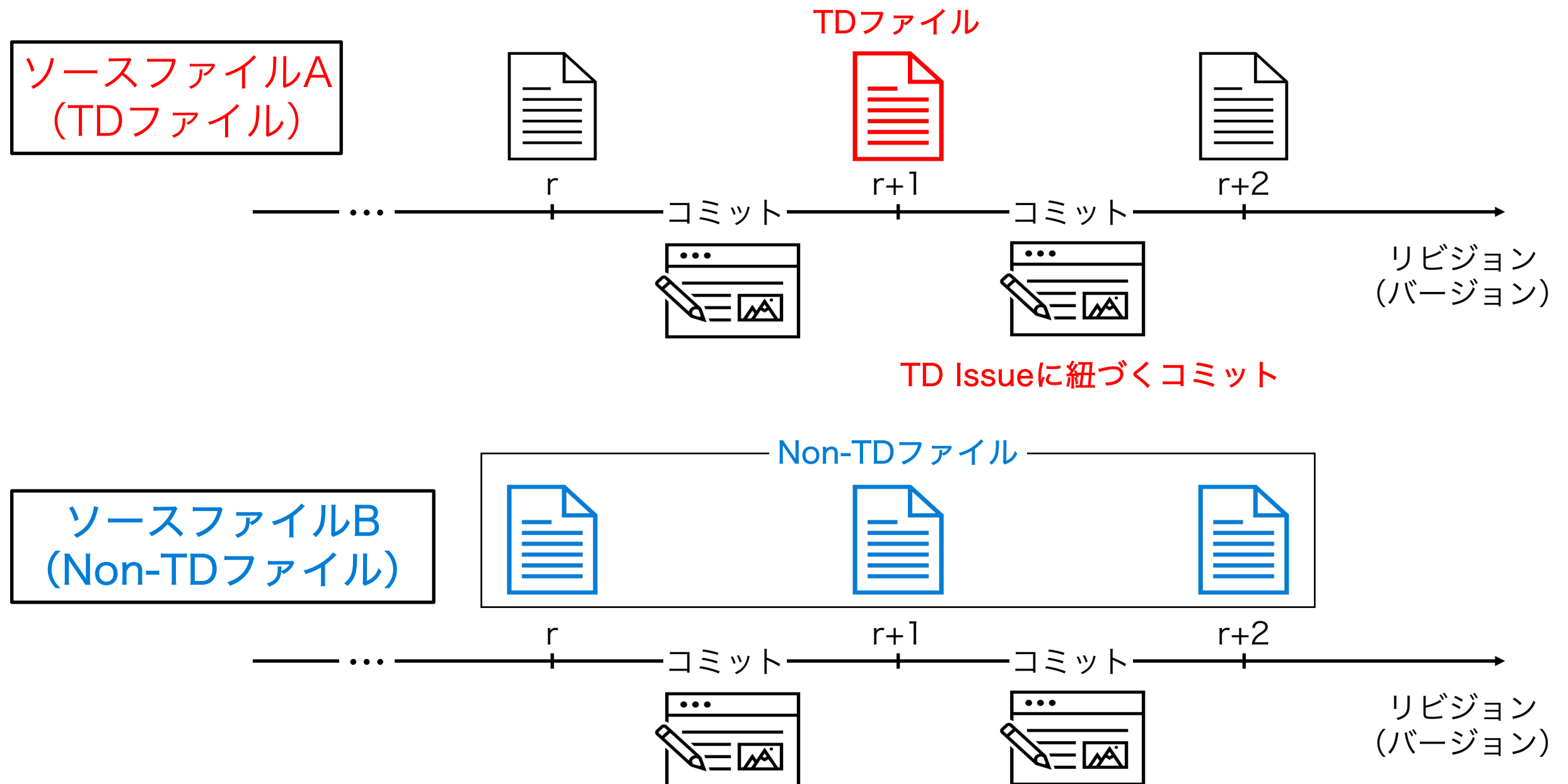
## 3. パターンの抽出と解釈



# 分析対象の定義

## 技術的負債を含むファイル (TDファイル)

- 技術的負債の除去が行われる前のバージョンのファイル



# データセット

## 654リポジトリを対象に分析を実施

- Java：208リポジトリ，Python：446リポジトリ
  - 既存研究のデータセット<sup>[?]</sup>をフィルタリングし利用

## フィルタリング

- 公開されているリポジトリであること
- 過去に技術的負債の除去が行われていること
- Java, Pythonプロジェクトであること
  - (Java) 既存のメトリクスベースでの検出はJavaを対象としており，それらの結果と比較するために利用
  - (Python) Javaの結果と比較するために，異なる言語パラダイムかつ人気のある言語を利用

[?] Technical Debt Classification in Issue Trackers using Natural Language Processing based on Transformers, International Conference on Technical Debt (TechDebt `23)

# 今後の予定

---

## 分析の実施

- ソースコードメトリクスでのパターン分析
  - プロダクトコードとテストコード
  - 変更の種類（追加, 削除, 両方） など

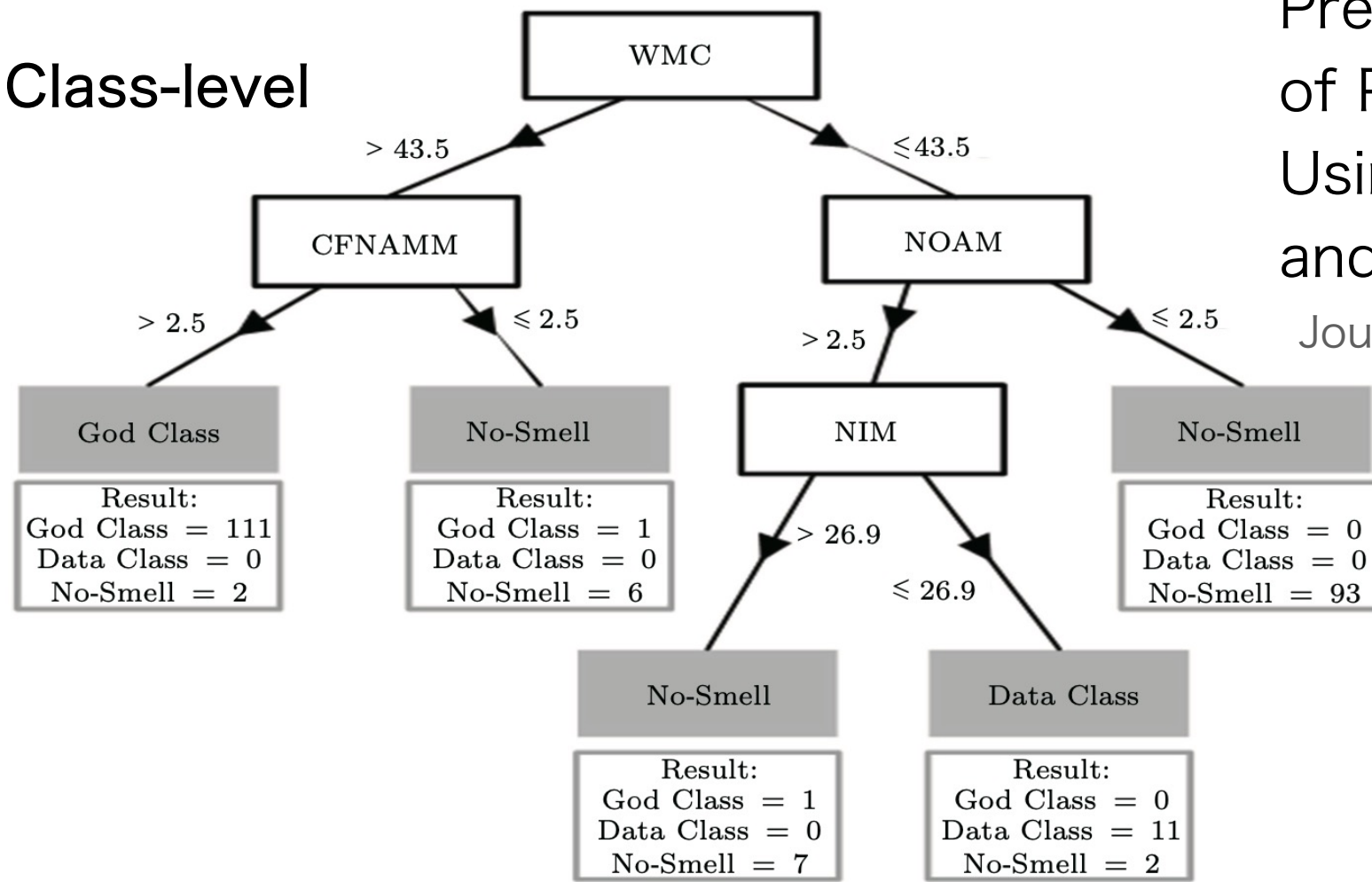
## その他

- コード以外の要素の分析
  - 技術的負債を混入される開発者と除去する開発者
  - 技術的負債が除去されるまでの時間
  - 技術的負債を構成するコミットの数 など

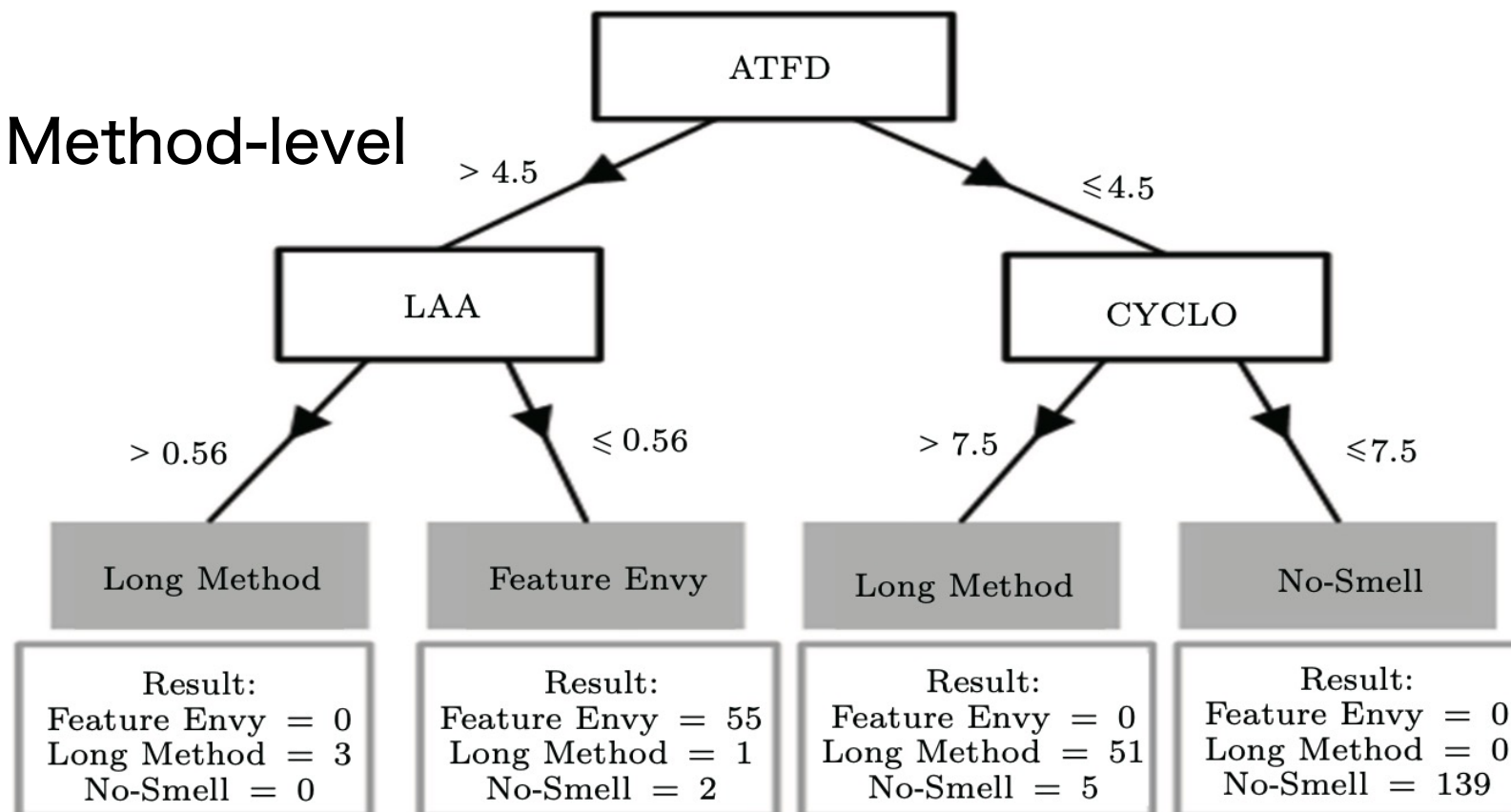
# 関連研究

---

## Class-level



## Method-level



Predicting Code Smells and Analysis  
of Predictions:  
Using Machine Learning Techniques  
and Software Metrics

Journal of Computer Science and Technology  
Volume 35, issue 6, 2020

## 対象コードスメル

### <Class-level>

- God Class
- Data Class

### <Method-Level>

- Long Method
- Feature Envy