

コードクローン中のコードスメルに関する調査研究

本田 澄 奥野 真伍 吉田 則裕

コードクローンとは、ソースコード中に存在する一致するまたは類似したコード片である。また、コードスメルとはソフトウェアに問題を引き起こす可能性があるコード片のことである。本研究では何らかのコードスメルがあるコード片がコードクローンによって増加することを問題と考え、コードクローン中にあるコードスメルについて調査研究を行う。その方法としてソフトウェアの開発履歴も含めてコードクローンの分析が行える CloneTracker を用いて対象とするソフトウェアのコードクローンを検出する。加えて、ソフトウェアの静的解析ツールである SonarQube を利用してコードスメルを検出する。CloneTracker が検出したコードクローンと SonarQube が検出したコードスメルを利用してコードクローン中のコードスメルを抽出しその特徴を分析する。対象とするソフトウェアは GitHub にて公開されているオープンソースソフトウェアである。

1 はじめに

コードクローンとは、ソースコード中に存在する一致するまたは類似したコード片である。また、コードスメルとは、ソフトウェアに問題を引き起こす可能性があるコード片のことである。本研究では何らかのコードスメルがあるコード片がコードクローンによって増加することを問題と考え、コードクローン中にあるコードスメルについて調査する。

本研究では、オープンソースソフトウェア (OSS) を対象として、コードクローンが存在するかを調べ、存在する場合にコードクローン中にコードスメルが存在するかを調査する。また、コードクローン中のコードスメルについてどのようなものがあるのかを分析する。調査では、CloneTracker^{†1} [3] を利用してコードクローンを分析し、SonarQube^{†2} を利用してコードスメルを分析する。

本研究の研究課題を以下に定義する。

RQ1: コードクローン中にコードスメルが存在するか

RQ2: コードクローン中に存在するコードスメルにはどのような特徴があるのか

これらの研究課題に対して、調査対象として GitHub にて公開されている Apache Software Foundation に関連するプロジェクト^{†3} を対象とした。また、分析を単純化するために、Java 言語で記述され、pom ファイルが含まれているプロジェクトのみを対象とする。

2 関連研究

コードスメルに関して、品質に与える影響や保守に与える影響を分析し評価しているものがある [1] [2]。本研究ではコードクローン中に存在するコードスメルを対象とした調査であり、品質に与える影響や保守に与える影響の分析に役立つと考えられる。

3 評価と結果

評価方法として、GitHub にて公開されているプロジェクトを Clone し、CloneTracker を利用してコードクローンであるコードを特定する。次に、Sonar-

Kiyoshi Honda, Shingo Okuno, 大阪工業大学, Osaka Institute of Technology .

Norihiro Yoshida, 立命館大学, Ritsumeikan University.

†1 <https://clonetracker.com/>

†2 <https://www.sonarsource.com/>

†3 <https://github.com/apache>

表 1 各プロジェクトのコードスメル中のコードスメルの数

	コード スメル	コード クローン	コードクローン中 のコードスメル
BCEL	1867	6	5
BeanUtils	403	12	5
Cli	183	1	0
Codec	1275	4	2
Collections	2338	10	0
Compress	3561	21	10
Configuration	2902	1	0
DBCP	1389	3	0
Digester	366	3	2
JEXL	360	6	6
Math	1301	74	14
Net	1023	6	5
SCXML	536	1	0
Text	2407	37	14

Qube を利用して Clone したプロジェクトを対象としてソースコードの静的解析を行い、コードスメルとされるコード片を特定する。本研究では SonarQube で定義されたコードスメルを対象として分析する。CloneTracker で特定したコードクローンであるコード中に SonarQube で特定したコードスメルがあるかを目視で調査する。

評価対象としたプロジェクトは GitHub にて公開されている Apache Software Foundation に関連する 14 プロジェクト (BCEL, BeanUtils, Cli, Codec, Collections, Compress, Configuration, DBCP, Digester, JEXL, Math, Net, SCXML, Text) である。

3.1 結果

表 1 に SonarQube で解析したコードスメルの数と、CloneTracker で解析したコードクローンの数と、二つの結果から特定したコードクローン中のコードスメルの数を示す。表 2 にコードクローン中に存在するコードスメルの種類とその数を示す。数は評価対象としたプロジェクトの全てを対象として数えたものである。

3.2 考察

RQ1: コードクローン中にコードスメルが存在するか

表 1 から 14 プロジェクト中 9 プロジェクトにコードクローン中にコードスメルが存在した。

表 2 コードクローン中のコードスメルの種類と数

コードスメルの種類	数
複数繰り返されるリテラルの指摘	9
認知複雑度に関する指摘	8
ループ内の break,continue を減らす指摘	8
ループ内のループカウンタに関する指摘	8
TODO コメントを行う指摘	6
反対の演算子「!=」を使用する指摘	4
「if-then-else」を単一の return に置き換える指摘	4
常に false になる式を削除する指摘	3
System.out,System.err をログに変える指摘	2
Null の代わりに空の配列を返す指摘	2
switch 文に default ケースを追加する指摘	2
コメントアウトしている行を削除する指摘	2
エラー出なく例外をキャッチする指摘	2
代入式内の代入式を分ける指摘	2
ローカル変数への無駄な代入を削除する指摘	1

RQ2: コードクローン中に存在するコードスメルにはどのような特徴があるのか

表 2 からコードクローン中にコードスメルが存在した 9 プロジェクト全体では、「複数繰り返されるリテラルの指摘」が最も多い種類であることがわかった。

4 まとめと展望

本研究ではコードクローン中のコードスメルについて調査し、14 プロジェクト中 9 プロジェクトに存在すること、その中で最も多く存在するコードスメルが「複数繰り返されるリテラルの指摘」であることを明らかとした。今後は対象を増やしプロジェクトの種類とコードスメルの種類について分析を行う。

謝辞 本研究は JSPS 科研費 JP19K20242, JP24K02923, JP20K11745 の助成を受けたものです。

参考文献

- [1] Sjøberg, D. I., Yamashita, A., Anda, B. C., Mockus, A., and Dybå, T.: Quantifying the Effect of Code Smells on Maintenance Effort, *IEEE Transactions on Software Engineering*, Vol. 39, No. 8(2013), pp. 1144–1156.
- [2] Yamashita, A. and Moonen, L.: Do developers care about code smells? An exploratory survey, *2013 20th Working Conference on Reverse Engineering (WCRE)*, 2013, pp. 242–251.
- [3] 三木聡, 大歳始, 浅原明広, 大澤俊晴, 千葉滋: 実務で使われるコードクローン検出・追跡システムをめざして, *日本ソフトウェア科学会第 40 回大会講演論文集*, 2023, pp. 1–8.