

後方互換性の損失に伴うテストコード変更内容の分類

前川 大樹[†]
和歌山大学[†]

伊原 彰紀[‡]
和歌山大学[‡]

大森 楓己[§]
和歌山大学[§]

才木 一也[¶]
和歌山大学[¶]

1 はじめに

ソフトウェア開発では、開発効率向上のためにライブラリと呼ばれるプログラムを再利用することが多い。新機能追加、バグ修正のためにライブラリの新バージョンがリリースされた場合、ライブラリ利用者は適宜ライブラリのバージョン更新を余儀なくされる。ライブラリの更新には後方互換性を損失する変更（破壊的変更）を含むことがあり、破壊的変更は利用者のプログラムの実行エラーにつながる。そのため、ライブラリ開発者は、適切なバージョン名によって破壊的変更の有無を利用者に伝える。しかし、ライブラリ開発者が破壊的変更を含むことを見逃し、意図せず利用者に被害を与えることも少なくない。

従来研究 [1] では、破壊的変更を含むライブラリの更新は、プログラムの更新と合わせてテストコードも修正すると考え、テストコードの変更有無による破壊的変更の検出手法を提案した。しかし、テストコードの変更には、テストの誤り修正や実行手順の修正など、ライブラリ変更とは無関係にテストコードを変更することも考えられ、そのような変更は破壊的変更を誤検出する。本研究では、従来手法の誤検出を減らすために、テストコード変更内容の分類に取り組み、破壊的変更検出に寄与するテスト変更

を明らかにする。

2 分析

2.1 分析手法

本研究では、Mujahid らが npm から収集したライブラリ [2] の中で、人気度を示す npm スコア上位 500 件を対象とする。また、各バージョンのテスト実行時の成功率が 100% であるライブラリのバージョン更新 2,111 組から、破壊的変更が含まれるバージョンと含まれないバージョンを 50 組ずつランダムに抽出し計 100 組のテスト変更内容を目視により分析する。本研究では、ライブラリ利用者のプログラムがライブラリの新バージョンの利用において、テストの実行結果が失敗であった場合に、破壊的変更が加えられたと判断する。

テストプログラムは複数のテストケースからなるテストスイートが組み合わさって構成されており、個々のテストケースに前提条件、期待する結果などが記述されている。著者らはテストプログラムの変更内容を目視で確認し、変更理由に基づきテストの変更を分類した。その後、それぞれの変更の発生回数を集計した。

2.2 結果

ライブラリテストの変更内容を分析した結果、テスト変更内容を 19 種類に分類できた。具体的な種類は紙面の都合上省略し、一部を表 1 に示す。表には、破壊的変更の有無に影響を与えるテスト変更パターンを探索するために、テスト変更内容 19 種類を説明変数、破壊的変更の有無を目的変数とした決定木分析により、特に破壊的変更有無に影響が高いとわかった 3 種

A classification of test code changes accompanied by a lack of backward compatibility

[†] Daiki Maekawa, Wakayama University

[‡] Akinori Ihara, Wakayama University

[§] Fuki Omori, Wakayama University

[¶] Kazuya Saiki, Wakayama University

表1: ライブラリテスト変更パターンの分類

テスト変更パターン	破壊的変更あり	破壊的変更不明	合計件数
等価な変数や関数に置換	11 件	1 件	12 件
テストケースの追加	57 件	28 件	85 件
期待する結果の変更	27 件	6 件	33 件

類を示す.

等価な変数や関数に置換: このテスト変更においては, 1 件を除いて破壊的変更を含むライブラリの更新であった. しかし, 等価な変数や関数に置換するテスト変更自体は, 破壊的変更に伴う変更ではないが, 開発者が同時に修正することで破壊的変更に関係するテスト変更として検出する. 今後は, 破壊的変更とは関係ないと考えられるテスト変更は無視することで, 再分析する必要があると考える.

テストケースの追加: テストケースの追加は, 破壊的変更に伴いテストケースを追加することも多いが, 破壊的変更に関係なくテストケースを追加することも少なくない. 破壊的変更に伴うテストケースの追加は, テスト対象の関数の挙動変更に対応するテストケースの追加を確認した. 一方で破壊的変更を確認できなかったライブラリ更新では, テスト品質向上のためライブラリの更新内容とは関係なくテストケースを追加していた. ライブラリ更新前後でテストケースを記述する関数の追加有無の検出により, 破壊的変更を高い精度で確認できると考える.

期待する結果の変更: 期待する結果の変更事例として, equals 文や assert 文など, アサーション関数の引数の変更がある. テスト対象の関数の出力内容が変更されたことで, 変更前の形式で出力を検証していたテストにも変更が加えられたと考えられる. 期待する結果の変更は, 関数の出力内容の変更という破壊的変更によるテスト変更であるため, 変更前後の AST を比較することによって引数の変更を捉え, 破壊的変更の検出ができると考える.

3 おわりに

本研究は, テスト変更の内容分類に取り組み, テスト変更パターンと破壊的変更有無の関係性を決定木により分析した. 分析の結果, ライブラリテストの変更パターンとして, テストケースの追加, 期待する結果の変更が含まれると, 破壊的変更が含まれる可能性が高いとわかった. 今後は, 破壊的変更とは関係ないと考えられるテスト変更を除外し, テスト変更パターンの自動分類に取り組むとともに, テスト変更内容に基づく破壊的変更の検出手法の精度向上を目指す.

謝辞

本研究は和歌山大学「萌芽的個別研究支援」, および柏森情報科学振興財団より助成を受けたものです.

参考文献

- [1] 松田和輝, 伊原彰紀, 才木一也. ライブラリのテストケース変更に基づく後方互換性の実証的分析. 第 28 回ソフトウェア工学の基礎ワークショップ論文集, pp. 139-144, 2021.
- [2] S. Mujahid, R. Abdalkareem, E. Shihab, and S. McIntosh. “Using others’ tests to avoid breaking updates,” Proc of the International Conference on Mining Software Repositories (MSR), pp.466-476, 2019.