プロダクトコード変更に伴い共進化するテストコード特定手法の提案

南　智孝[†1]　伊原彰紀[†1]
坂口英司[†1]　松本健一[†1]

オープンソースソフトウェア (OSS) 開発において，プロダクトコードの変更と同時に，当該プロダクトコードを検証するテストコードの変更（共進化）ができていないことがある．本論文では，OSS 開発における継続的なテストコードの保守に向けて，過去の変更履歴（プロダクトコードとテストコードの共進化）に基づき変更すべきテストコードを特定する手法を提案する．

Toward identifying a Test Code Co-Evolved with a Product Code

Tomotaka Minami,[†1] Hideshi Sakaguchi,[†1] Akinori Ihara[†1]
and Ken-ichi matsumoto [†1]

In Open Source Software (OSS) development, developers sometimes miss changing the test code when the product code changes. This study proposed an approach to identify a test code that should be changed with a product code change (co-evolution between product code and test code) to maintain test codes continuously in OSS development.

1. はじめに

オープンソースソフトウェア (OSS) は，リリースしたソフトウェア（ソースコード）を無償で公開することで，開発者だけでなく，多くの利用者もプログラムを様々な目的で実行することができる．その結果，プロジェクトには日々多くの不具合が発見され，報告されている．Eric S. Raymond は，利用者から多くの不具合が報告される開発形態について “The Cathedral and the Bazaar” で「目玉の数さえ十分あれば、どんなバグも深刻ではない」と主張している．

その一方で，テスト計画の未作成など，組織的なテスト活動が実施されておらず[3]，リリースされた OSS に多くの欠陥が含まれている．より高品質な OSS のリリースに向けて，昨今では，テスト実行を自動化するフレームワーク JUnit やテストコードを容易に管理するシステム Maven などテストの実施を支援するシステムが普及しつつある．しかし，テスト実施に対する課題は未だ存在し，Zaidman らは，プロダクトコードの変更と同時に，当該プロダクトコードを検証するテストコードが変更（共進化）されていない課題を指摘している[2]．

本論文では，OSS 開発における継続的なテストコード保守の実現に向けて，プロダクトコード変更時に，同時に変更すべきテストコードを特定する手法を提案する．特に本論文では，従来研究で提案されている同時変更すべきプロダクトコードを推薦する技術（Co-change コード推薦技術）の適用を検討する．

2. 関連研究

近年，OSS 開発におけるテストを対象とした研究が行われており，多くの OSS プロジェクトでは，テストコードを十分に管理できていないことが明らかとなった．Takasawa ら[1] は，テストコードを公開している OSS プロジェクトを対象に， 各プロジェクトのテストコードを実行した．その結果，ビルドに成功したプロジェクトは 57.1%であり，OSS プロジェクトではテストのビルド成功率が低いことを明らかにした．

Zaidman ら[2] は，プロダクトコードとテストコードの共進化を可視化するツールを開発している．開発したツールにより，プロダクトコードとテストコードが必ずしも共進化していないことを指摘している．また，Zimmermann ら[4] はソースファイルにおける行レベルでの共進化について調査を行っているが，テストコードに着目していないという点で本研究と異なる．

†1 奈良先端科学技術大学院大学
Nara Institute of Science and Technology

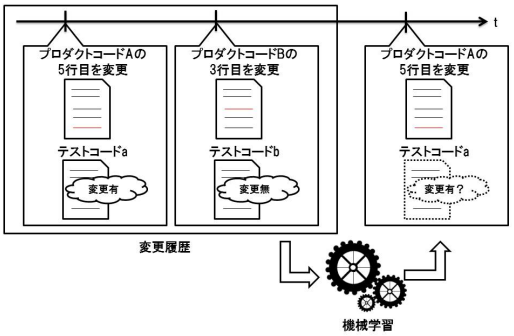

図 1 プロダクトコード変更履歴に基づく共進化するテストコードの特定手法

3. プロダクトコード変更に伴い共進化するテストコード特定手法の提案

本論文は，継続的なテストコードの保守に向けて，プロダクトコード変更に伴い共進化するテストコードを特定する手法の一つとして，過去の変更履歴（プロダクトコードとの共進化）に基づくアプローチを提案する．本研究を進めるにあたり,「過去に共進化したテストコードは将来も同様に共進化する」という仮説を立てる．

図 1 に，本論文で提案するアプローチの概要を示す．図 1 では,「プロダクトコード A の 5 行目が変更されると，テストコード a が変更される」,「プロダクトコード B の 3 行目が変更されても，テストコード b が変更されない」という変更履歴がある場合を考える．このとき，我々の仮説に基づくと，再びプロダクトコード A の 5 行目が変更された場合，テストコード a は変更されるという予測ができ，再びプロダクトコード B の 3 行目が変更された場合，テストコード b は変更されないという予測ができる．

本アプローチは，プロダクト間の同時変更を予測する研究で一般的に使用される Co-change ファイル特定技術の一つである．プロダクト間の Co-change ファイル特定技術の場合，同時変更を忘れた場合にソフトウェアのビルドに失敗することが多く,（コミット時期が異なったとしても）同時変更を忘れるということが少ない．一方で，プロダクトコードとテストコード間では，ソフトウェアのビルドは成功するが，プロダクトコードとテストコードの共進化が発生していないことも多く，過去のデータで予測結果を評価することは必ずしも妥当ではない．本研究では，プロダクトコードの特定の行が変更されたときに，本アプローチを用いて特定された共進化すべきテストコードが変更されなかった理由を調査し，共進化するテストコード特定技術の精度向上を目指す．

4. おわりに

本論文は，継続的なテストコードの保守に向けて，プロダクトコード変更に伴い共進化するテストコードを特定する手法の一つとして，過去の変更履歴（プロダクトコードとの共進化）に基づくアプローチを提案した．過去に同時変更したテストコードを特定する手法は，共進化するテストコードを特定する一つのアプローチであり，プロダクトコードとテストコードの静的解析等により，変更されるべきテストコードの特定を検討する．

謝辞 本研究の一部は，頭脳循環を加速する戦略的国際研究ネットワーク推進プログラム，および，文部科学省科学研究補助費（萌芽研究: 課題番号 25540026）による助成を受けた．

参考文献

1) Takasawa.R, Sakamoto.K, Ihara.A, Washizaki.H and Fukuzawa.Y: Do open source software projects conduct tests enough?, *Proc. of the 15th International Conference of Product Focused Software Development and Process Improvement (PROFES'14)*, pp.322–325 (2014).
2) Zaidman.A, Rompaey.V.B, Deursen.v.A and Demeyer.S: Studying the co-evolution of production and test code in open source and industrial developer test processes through repository mining, *Empirical Software Engineering*, Vol.16, pp.325–364 (2010).
3) Zhao.L and Elbaum.S: A Survey on Quality Related Activities in Open Source, *SIGSOFT Softw. Eng. Notes*, Vol. 25, No. 3, pp. 54–57 (2000).
4) Zimmermann.T, Kim.S, Zeller.A and Whitehead.J.E.J: Mining Version Archives for Co-changed Lines, *Proceedings of the 2006 International Workshop on Mining Software Repositories*, MSR '06, New York, NY, USA, ACM, pp.72–75 (2006).