

**科学研究費助成事業 研究成果報告書**

平成 27 年 5 月 28 日現在

機関番号：17102

研究種目：若手研究(A)

研究期間：2012～2014

課題番号：24680003

研究課題名(和文)リポジトリ活用型Just-In-Timeソフトウェア品質モデルの開発と評価

研究課題名(英文)Development and Evaluation of Just-In-Time Software Quality Models by Mining Software Repositories

研究代表者

亀井 靖高 (Yasutaka, Kamei)

九州大学・システム情報科学研究科(研究院・准教授)

研究者番号：10610222

交付決定額(研究期間全体)：(直接経費) 16,100,000円

研究成果の概要(和文)：本研究では、ソフトウェア変更に対する品質モデル(Just-In-Timeソフトウェア品質モデル)を提案する。JITソフトウェア品質モデルは、ソフトウェアリポジトリマイニング技術を活用することで、現場の開発者が予測モデルを利用したいと計画したときにすぐに利用できる仕組みを提供する。予測結果をリアルタイムに開発者にフィードバックすることで、本提案は開発者の記憶がフレッシュなうちにバグの修正に取り組むことを可能にする。

研究成果の概要(英文)：We propose a Just-In-Time (JIT) software quality model, which provides an environment for predicting whether or not a change induces a defect. By using MSR (Mining Software Repository) techniques, the JIT software quality model does not require a large amount of historical data in order to train a model that will perform well. It allows practitioners in a software development project to use the model immediately after they decide to use it. The prediction can be performed at the time when the change is submitted. Such immediate feedback ensures that the design decisions are still fresh in the minds of practitioners.

研究分野：ソフトウェア工学

キーワード：Just-In-Time ソフトウェア信頼性 バグ予測 リポジトリマイニング オープンソースソフトウェア

1. 研究開始当初の背景

- (1) 情報システムは社会生活の基盤となっており、システム故障による社会的影響は甚大である。その一方で、開発リソースは限られており、これまでになく短期間で開発することも同時に要求されている。高信頼性を目指すほど開発コストが飛躍的に増大するため、今日の社会ニーズとして、信頼性の確保と開発の効率化を両立することが求められている。
- (2) その一手段は、ソフトウェア中の各モジュールの信頼性(バグの有無)を予測し、バグを含むと判断されたモジュールにテスト工数を重点的に割り当て、信頼性の確保と無駄なテスト工数の削減を両立することである。そのため、モジュールから計測された多数の特性値(プログラム行数、複雑度、変更行数など)を入力として、モジュールのバグの有無を判別するモデル(以降、バグ予測モデル)[1]が国内外の研究者によって多数提案、および、評価されてきた[2][3]。しかしながら、バグ予測モデルは一部の開発現場における適用事例[4]が報告されているものの、現時点において広くは普及していない。

2. 研究の目的と現状の問題点

- (1) 本研究の目的は、バグ予測において現状の課題とされるモデル構築に関する問題、及び、モデル利用に関する問題を解決することである。今までの研究活動、および、企業との共同研究を通じて得られた現状の問題点は以下の集約される。
- (2) モデル構築に関する問題: 従来手法には、「十分な開発履歴(学習データ)がないと予測モデルを構築できない」という運用上の大きな制約/課題がある。一般に、学習データは過去プロジェクトから計測するものであり、テスト、リリースを経て、バグが報告された後にどのコードに混入していたのかを事後分析されることで得られる。そのため、データの計測を計画してから実際に取得できるまでに長い期間と多くの工数を要する。
- (3) モデル利用に関する問題: 従来手法による予測の対象は、ファイル(もしくは1つ以上のファイルの集合であるパッケージ)であることが多い。予測モデルによってファイル中にバグがあると予測されたとしても、開発者はどの部分を重点的にテストしてよいかの判断が困難である。さらに、従来手法は、実装工程完了後にモジュールがテストされることを前提として提案されている。テスト工程で予測モデルがバグを検出したとしても、開発者はそのモジュールのどの部分にバグがあるのか、そのモジュールをどう実装したのか、を思い出すのに時間がかかりすぎるという問題がある。

3. 研究の方法

- (1) 本研究では、バグ予測における上記の課題を解決し、データの蓄積を必要とせず、予測結果をリアルタイムに開発者にフィードバックする品質モデル「Just-In-Time ソフトウェア品質モデル」を提案する。
- (2) モデル構築に関する問題に対する解決方法: オープンソースソフトウェアなどに公開されているデータや、他プロジェクトのデータを利用してモデル構築・判別を行うことを試みる。他プロジェクトを用いても性能の良いモデル構築に寄与するであろう仮説(例えば、開発のコンテキストが類似するプロジェクトを用いると、性能のよいモデルを構築することができる)を立て、それらの仮説を実データを用いて実験的に検証する。
- (3) モデル利用に関する問題に対する解決方法: ソフトウェアの変更(版管理システムのコミット)に対してバグが含まれているか否かを予測する。そして、その予測結果を開発者にリアルタイムにフィードバックする。そのために、ソフトウェアの変更に対する予測モデル構築のための特徴量や、モデリング技術を開発する。
- (4) 従来のバグ予測と異なり、本提案は他のOSS 開発データを活用することで、事前のデータ計測を必要とせずバグ予測を実行できる点が本研究の大きな特色である。データ計測に要する期間と工数の削減はもちろんのこと、複数プロジェクトのデータを用いることで従来よりも精度の高い予測が期待できる。まあ、開発者が当該モジュールを変更する際にバグの有無を予測するため、開発者の記憶がフレッシュなうちにフィードバックを与えることができる点も大きな特色である。

4. 研究成果

- (1) **Just-In-Time** バグ予測モデルの実証的評価: ソフトウェアの変更から計測可能な14種類の特徴量(変更行数や変更者の経験量)を定義し、予測モデルを構築した。評価実験では、6つのオープンソース(Bugzilla, Columba, Eclipse JDT, Eclipse Platform, Mozilla, PostgreSQL), 及び、5つの企業プロジェクトから計測したデータセットを対象に、ロジスティック回帰分析を用いた。その結果、68%の精度(Accuracy), 及び、64%の再現率(Recall)で予測できることがわかった。さらに、実際にレビューに費やすことが可能な工数を考慮した上で再評価した結果、全レビュー工数の20%を用いるだけで、35%のバグ混入変更を特定できることがわかった。本研究成果の詳細は、本報告書「5.

発表論文等」の[雑誌論文]の④等で報告されている。なお、本成果はソフトウェア変更に対する予測に着目したところが大変ユニークである。アプローチのユニークが評価され、IEEE Computer Society Japan Chapter Young Author Award が授与されるに至った。

- (2) **High Impact バグ:JIT 品質モデル構築**の際、品質 (バグ) の重みを均一に扱うのではなく、重み付けするための方法の開発を行った。例えば、修正にかかる労力 (同時修正ファイル数, 同時修正ディレクトリ数, 変更行数) が大きい欠陥は、重みの大きい欠陥と定め、労力の大きさに基づき欠陥の重み付けを行った。その他には、修正完了の後に再度修正されるバグを **Re-Open** バグと定義し、通常のバグよりも重みの大きいバグとした。これらの方法は労力の重み付けを自動的に行うことが可能であり、大規模データにも適用できる点がメリットである。

本研究成果の詳細は、本報告書「5. 発表論文等」の[雑誌論文]の①や⑤等で報告されている。

- (3) **Cross Project** データの学習データとしての活用方法:オープンソースソフトウェアなどに公開されているデータや、他プロジェクトのデータを利用してモデル構築・判別を行うことを試みる。試行した実験として、例えば、よくモデリング方法として用いられるロジスティック回帰分析と、ランダムフォレストのどちらが **Just-In-Time** バグ予測で有効であるかがあげられる。その他に、説明変数である特徴量の類似度を数値化し、最も特徴量の傾向が類似している他プロジェクトを学習データ (モデル構築用のデータ) として用いることが有効であるか否かがあげられる。現時点で得られた最も有効な方法の1つは、蓄積された全プロジェクトの過去データを一つのデータとしてみなして結合し、モデルを構築する方法である。

本研究成果の詳細は、本報告書「5. 発表論文等」の[学会発表]の①等で報告されている。

- (4) **プロトタイプシステムの作成:リポジトリマイニングを効率的に行うためのDSL 言語の開発**、及び、**Just-In-Time** ソフトウェア品質モデルを実施するためのツールのプロトタイプシステムを **Ruby** 言語によって実装した。

本研究成果の詳細は、本報告書「5. 発表論文等」の[学会発表]の③、④、⑤等で報告されている。

## 参考文献

- [1] John Munson, and Taghi M. Khoshgoftaar, ``The Detection of Fault-prone Programs,’’ IEEE Transactions on Software Engineering, Vol.18, No.5, pp.423-433, 1992.
- [2] Tibor Gyimothy, Rudolf. Ferenc, and Istvan. Siket, ``Empirical Validation of Object-Oriented Metrics on Open Source Software for Fault Prediction,’’ IEEE Transactions on Software Engineering, Vol.31, No.10, pp.897-910, 2005.
- [3] A.Gunes Koru, Dongsong Zhang, Khaled El Emam, and Hongfang Liu, ``An Investigation into the Functional Form of the Size-Defect Relationship for Software Modules,’’ IEEE Transactions on Software Engineering, Vol.35, No.2, pp.293-304, 2009.
- [4] Marco D’Ambros, Michele Lanza, and Romain Robbes, ``An Extensive Comparison of Bug Prediction Approaches,’’ In Proceedings of International Working Conference on Mining Software Repositories, pp.31-41, 2010.
- [5] Nachiappan Nagappan, Andreas Zeller, Thomas Zimmermann, Kim Herzig, and Brendan Murphy, ``Change Bursts as Defect Predictors,’’ In Proceedings of International Symposium on Software Reliability Engineering, pp.309-318, 2010.

## 5. 主な発表論文等

(研究代表者、研究分担者及び連携研究者には下線)

[雑誌論文] (計8件、全て査読有り)

- ① Ayse Tosun Misirli, Emad Shihab and Yasutaka Kamei, ``Studying High Impact Fix-Inducing Changes,’’ Journal of Empirical Software Engineering. (To appear)
- ② Shane Mcintosh, Yasutaka Kamei, Bram Adams and Ahmed E. Hassan, ``An Empirical Study of the Impact of Modern Code Review Practices on Software Quality,’’ Journal of Empirical Software Engineering. (To appear).
- ③ 小林 寛武, 戸田 航史, 亀井 靖高, 門田 暁人, 峯 恒憲, 鶴林 尚靖, ``11種類の fault 密度予測モデルの実証的評価,’’ 電子情報通信学会論文誌, Vol. J96-D, No.8, pp.1892-1902, August 2013.
- ④ Yasutaka Kamei, Emad Shihab, Bram Adams, Ahmed E. Hassan, Audris Mockus,

Anand Sinha, and Naoyasu Ubayashi, ``A Large-Scale Empirical Study of Just-In-Time Quality Assurance,’’ IEEE Transactions on Software Engineering, Vol. 39, No. 6, pp. 757-773, June, 2013.

- ⑤ Emad Shihab, Akinori Ihara, Yasutaka Kamei, Walid M. Ibrahim, Masao Ohira, Bram Adams, Ahmed E. Hassan, and Ken-ichi Matsumoto, ``Studying Re-opened Bugs in Open Source Software,’’ Journal of Empirical Software Engineering, Vol. 18, No. 5, pp. 1005-1042, October 2013.

〔学会発表〕(計 22 件、内査読有り 10 件、招待講演 1 件) 筆頭著者が発表者。

- ① Takafumi Fukushima, Yasutaka Kamei, Shane McIntosh, Kazuhiro Yamashita and Naoyasu Ubayashi, ``An Empirical Study of Just-In-Time Defect Prediction Using Cross-Project Models,’’ In Proceedings of International Working Conference on Mining Software Repositories (MSR 2014), pp. 172-181, June 2014. (Hyderabad, India).
- ② Shane Mcintosh, Yasutaka Kamei, Bram Adams and Ahmed E. Hassan, ``The Impact of Code Review Coverage and Code Review Participation on Software Quality: A Case Study of the Qt, VTK, and ITK Projects,’’ In Proceedings of International Working Conference on Mining Software Repositories (MSR 2014), pp. 192-201, June 2014. (Hyderabad, India).
- ③ 田中 秀太郎, 山下一寛, 亀井 靖高, 鶴林 尚靖, ``変更レベルに着目したバグ予測支援ツールの設計と実装’’ 電子情報通信学会技術報告, ソフトウェアサイエンス研究会, pp. 113-118, January 2014. (豊田中央研究所, 愛知).
- ④ 大坂 陽, 山下一寛, 亀井 靖高, 鶴林 尚靖, ``リポジトリマイニングに対する Hadoop の導入に向けた性能評価,’’ SES2013, pp. 1-8, September 2013. (東洋大学, 東京).
- ⑤ Hiroki Nakamura, Rina Nagano, Kenji Hisazumi, Yasutaka Kamei, Naoyasu Ubayashi and Akira Fukuda, ``QORAL : External Domain-Specific Language for Mining Software Repositories,’’ In Proceedings of International Workshop on Empirical Software Engineering in Practice (IWESEP2012), pp. 23-29, October 2012. (Osaka, Japan)

〔図書〕(計 0 件)

〔産業財産権〕

○出願状況 (計 0 件)

○取得状況 (計 0 件)

〔その他〕

ホームページ等

6. 研究組織

(1) 研究代表者

亀井 靖高 (KAMEI, Yasutaka)

九州大学・大学院システム情報科学研究  
院・准教授

研究者番号 : 10610222