様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

令和　元 年　 ８ 月 ２７ 日現在

機関番号：１７１０２
研究種目：若手研究(A)
研究期間：2015～2017
課題番号：１５Ｈ０５３０６

研究課題名（和文）Mobile Appコードの自動進化の実現に向けたリポジトリマイニング基盤の開発

研究課題名（英文）Development and Evaluation of Mining Software Repositories Infrastructure Toward Automatic Bug Repair for Mobile Apps

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）　　15,300,000円

研究成果の概要（和文）：本研究では，クリーンで，かつグリーンなMobile App（スマートフォン向けに開発されたアプリケーション）社会の実現を目指して，Mobile Appコードの自動進化技術の開発を行った．オープンソース上の大量のMobile Appリポジトリの活用により，コード変更がクリーンで，かつグリーンな状態であるか（クラッシュを引き起こさず，電力を異常に消費しないか）を予測し，リスクが高い場合はコードを自動的に進化させる．主な研究成果は，(T1) Mobile App開発リポジトリマイニング基盤，(T2) バグ混入予測のモデリング技術，(T3) コードの自動進化技術とその実証評価の3つに分類される．

研究成果の学術的意義や社会的意義
本研究成果の学術的意義は，近年ソフトウェア工学分野でホットな分野であるMobile App，自動バグ修正，グリーンマイニングを統合的に研究し，成果としてまとめた点である．社会的意義としては，ソフトウェア開発で広く用いられているソフトウェアアーキテクチャの1つであるMVC（Model-View-Controller）アーキテクチャにおいて，Model部分とView部分の間にbundling presenter（イベントを蓄積し，一括してViewに送信する機能）を介することで，Mobile Appの消費電力を最大で30%ほど節約できることを明らかにした点がある．

研究成果の概要（英文）：We propose mining software repositories infrastructure toward automatic bug repair for mobiles apps. By using MSR (Mining Software Repositories) techniques, we predict whether or not a software change keeps clean and green (i.e., the software change does not introduce crash nor abnormal energy consumption). We also automatically generate software patches if the software change is predicted as risky. The main contributions of this research project provide (T1) mining infrastructure for mobile apps, (T2) modelling techniques for risk of software changes, and (T3) techniques for automatic bug fixing (and the experimental results of the techniques).

研究分野：ソフトウェア工学，実証的ソフトウェア工学，マイニングソフトウェアリポジトリ

キーワード：モバイルアプリ　ソフトウェア信頼性　バグ予測　オープンソースソフトウェア　グリーンマイニング　自動バグ修正

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９、ＣＫ－１９（共通）

１．研究開始当初の背景

(1)　昨今，携帯端末は世界中で爆発的に普及してきており，例えば，Android OS が搭載されたスマートフォンは，2013 年 9 月の時点で，世界で 2 億人以上ものユーザによって利用されている．そのため，モバイルアプリ（Mobile App: スマートフォン向けに開発されたアプリケーションソフトウェア[1]）のたった 1 つのソフトウェアバグが波及する度合いは極めて大きい．

(2)　しかしながら，Mobile App は通常のソフトウェアよりも頻繁にクラッシュ（予期せぬ停止状態）を引き起こしているのが現状である．その理由の一つは，Mobile App では，組み込み機器を意識した実装（GPS の信号強度の急激な低下に対するエラー処理等）が求められ，従来よりも実装が困難であるためである．さらに Mobile App は，持ち運びが前提のデバイス上で動作するため，消費電力は開発者／ユーザ双方にとって目下の関心事である．それにもかかわらず，不適切なライブラリ利用やハードウェアとの相性によって，新バージョンのリリース毎に電力の異常消費がたびたび報告されている[2]．世界に 6 億人以上のユーザを有するスマートフォンにおける今日のニーズとして，クリーンで，かつ　グリーンな（クラッシュを引き起こさず，電力を異常に消費しない）Mobile App を安定的に提供することが求められる．

(3)　その解決のため，Mobile App コードへの静的解析によるクラッシュの検出方法や，電力消費に対してはさらに，Mobile App プロジェクトの開発履歴（リポジトリ）にデータマイニング手法を適用することで，電力の異常消費の原因となるコードを発見しようとする試み（グリーンマイニング）という概念も提唱されている[3]．各コード変更の特徴と電力の異常消費の関係性や，その問題の解決方法といった「過去のノウハウ」を開発データの中からうまく発見しようという試みである．

２．研究の目的

(1)　本研究では，クリーンで，かつグリーンな Mobile App 社会実現のために，Mobile App 開発リポジトリマイニング基盤の開発と，その成果を用いた Mobile App コードの自動進化技術を提案する．

(2)　具体的には，大量の Mobile App 開発プロジェクトの開発履歴を取得し活用する方法，及び，実機上で Mobile App の全バージョンを自動的に動作させ，電力消費に関するデータを自動収集する方法を明らかにする［T1］．また，Mobile App 向けの予測モデル技術の開発に加えて［T2］，バグ修正に関する修正候補の自動生成・自動修復方法［T3: Mobile App コードの自動進化方法］を明らかにする．

３．研究の方法

(1)　クリーンで，かつグリーンな Mobile App 社会の実現に向け，リポジトリマイニング基盤の開発と，その成果を用いた Mobile App コードの自動進化技術を提案する．研究の方法として，各サブタスクは，互いにその成果を利用し合うので，サイクリックに開発を行った．

(2)　［T1］Mobile App 開発リポジトリマイニング基盤の開発：実現のために，3 つのサブタスクを設定し，研究を行った．まず，多種・多様な形態で公開されている Mobile App 開発のリポジトリを網羅的に取得するための収集機構を開発する［T1-A］．次に，それらリポジトリから，バグ予測に有用なコード変更の特徴量を開発し計測する［T1-B］．最後に，本基盤にスマートフォンを物理的に接続することで，リポジトリの各バージョンを実機上で自動的にビルド・実行し，消費電力を計測・蓄積する機構（グリーンデータ計測機構）を開発する［T1-C］．

(3)　［T2］コード変更に対するバグ予測モデルの開発：開発者の Mobile App コード修正時に，その修正がクラッシュを引き起こさないか（クリーンであるか否か），さらに，電力消費に関するバグを引き起こさないか（グリーンであるか否か）を，予測するモデルを開発する．Mobile App のコード変更において高精度の予測を実現するための特徴量，及び，モデリング技術を明らかにする．

(4)　［T3］Mobile App コードの自動進化手法の開発：修正コードを自動生成して，開発者にフィードバックするための技術（Mobile App コードの自動進化）を開発する．蓄積した過去の実データから，クラッシュや消費電力バグに対する修正パターンを発見する．さらに，修正パターンをそのまま現在のコードに適用してもビルドが成功する可能性が低いので，うまく変換して適用できる修正コードを生成する方法を明らかにする．

４．研究成果

(1) [T1] Mobile App 開発リポジトリマイニング基盤の開発：大きく 2 つの研究成果が挙げられる．1 つ目は，[T1-A, T1-B]に関連して，ソフトウェア工学研究で幅広く利用されている SZZ アルゴリズムを実装した G-rex を開発した点である．2 つ目は，[T-3]に関連して，Green-miner2 を実装し運用している点である．詳細は次の通りである．

(2) G-rex：バージョン管理システムの git を対象に，開発履歴を解析し，バグ修正を引き起こしたコード変更を特定する．本ツールを活用することで，例えば，オープンソースソフトウェア開発プロジェクトである Qt や OpenStack の git リポジトリに蓄積された計 37,000 以上のコミットを対象にし，約 3,600 のバグ混入の可能性が高いコミットを特定できた．さらに，本ツールを活用し，4 つの大カテゴリに分類される計 17 種類のメトリクス（例：修正行数や前回からの変更日時）を収集し，どのメトリクスがバグ混入のリスクを上げるかに関する調査も行うことができた．本研究成果の詳細は，本報告書「５．主な発表論文等」の雑誌論文②や学会発表②等で報告されている．

(3) Green-miner2：Android スマートフォン上の Mobile App を自動的に動作（操作を行い）させ，各操作に対する消費電力量を計測するフレームワークである．ハードウェア・ソフトウェア構成は，ASUS ZenFone 2，Android 5.0.2，INA159，Arduino Uno microcontroller，Raspberry Pi model B computer である．各操作はテストコードとして与えられる．本成果によって，1 つのベンチマーク，及び，4 つの実 Mobile App から，消費電力に関するデータを取得し，ソースコードと関連付けることができた．本研究成果の詳細は，本報告書「５．主な発表論文等」の雑誌論文①等で報告されている．

(4) [T2] コード変更に対するバグ予測モデルの開発：大きく 2 つの研究成果が挙げられる．1 つ目は特徴量に関するものであり，2 つ目はモデリング技術についてである．詳細は次の通りである．

(5) 特徴量の開発と実証評価（クラッシュに関する特徴量と実証評価）： オープンソースとして公開されている Mobile App のプロジェクトを用いて，定義したメトリクス（例：コミットログ中に記載されたソースコードの変更内容）が Mobile App のクラッシュとどの程度関連しているかを実証的に評価した．その結果，本メトリクスを用いることで，単純に予測する場合と比べて，AUC (Area Under the Receiver Operating Characteristic Curve)が約 28%改善できることがわかった．その他に，先行研究のアルゴリズムを改良・実装することで，クラッシュレポートからソースコード内の欠陥箇所を予測するシステムを開発した．本研究成果の詳細は，本報告書「５．主な発表論文等」の雑誌論文④や学会発表④等で報告されている．

(6) 特徴量の開発と実証評価（Mobile 開発の実応用ドメインへの適用）： Android ソフトウェア開発（企業における実開発プロジェクト）におけるコードレビューメトリクスの効果を実証的に評価した．開発プロジェクト・プロセスの特性を考慮することで，コードレビューデータ（例：ソースコードの変更を行った開発者自身がテストを実施しているか否か）を Mobile 開発に役立てられる可能性を示した．本研究成果の詳細は，本報告書「５．主な発表論文等」の雑誌論文⑤や学会発表⑥等で報告されている．なお，本研究成果の一部は，IPSJ/SIGSE 卓越研究賞 2016 を受賞した．

(7) モデリング技術の改良と実証評価（他プロジェクトデータ利用のガイドライン）：バグ予測混入予測モデルの開発では，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・判別を行う方法の実証的実験を行った．その結果，モデル構築に関する 3 つの利用ガイドラインを示すことができた．本ガイドラインを利用することでより精度の高いモデル構築が期待できる．本研究成果の詳細の一部は，本報告書「５．主な発表論文等」の雑誌論文③等で報告されている．

(8) モデリング技術の改良と実証評価（コンセプトドリフトの評価）：モデリング技術の改善のため，モデル構築に用いるデータの分割方法を調査した（例えば，当該プロジェクトの全コミット履歴を用いるのがよいのか？それとも３／６ヶ月前までのコミット履歴に絞るのがよいのか？）．実験結果の１つとして，(a) 一年以上前のデータを用いたモデルでは，予測精度が大幅に低下すること，(b) ６ヶ月分のコミット履歴を用いた場合，予測結果が（３ヶ月分のコミット履歴よりも）安定することがわかった．本研究成果の詳細は，本報告書「５．主な発表論文等」の学会発表②の元となった雑誌論文等で報告されている．なお，本研究成果の一部は，

IPSJ/SIGSE 卓越研究賞 2017 を受賞した.

(9) モデリング技術の改良と実証評価（深層学習の応用）：ソースコードの変更に対する予測モデルに深層学習を拡張・適用した．具体的には，従来研究におけるメトリクスを用いたモデリング技術では積極的に活用されてこなかったコミットメッセージやソースコード自体を特徴量計測のために併用した．End-to-End の深層学習フレームワークを用いて計測からモデル化までを行った．2 つのオープンソースソフトウェア開発プロジェクトから計測したデータに適用した結果，従来の予測手法よりも精度を約 10 ポイント向上することができた．本研究成果の詳細は，本報告書「５．主な発表論文等」の雑誌論文②や学会発表③等で報告されている．

(10) [T3] Mobile App コードの自動進化手法の開発：大きく 2 つの研究成果が挙げられる．1 つ目は，過去の修正履歴の傾向を自動修正結果に反映することができる自動バグ修正技術 Prophet の実証的評価を行った点である．もう一つは，消費電力に影響を及ぼすソフトウェア設計パターンの一部を特定し，解決方法を示した点である．詳細は次の通りである．

(11) 自動バグ修正手法の実証的評価：修正履歴の傾向を反映した自動バグ修正技術 Prophet を本研究課題に適用し（実行環境を Amazon AWS 上で実現し，クラウド基盤の構築），自動バグ修正の性能を評価した．実証評価には，学習データとして 8 プロジェクト，777 件の修正履歴を用い，修正対象として 4 プロジェクト，85 件のバグを用いた．得られた知見の 1 つは，Prophet を適用する際，同一のプロジェクトから履歴を収集することで，無意味な修正パッチの生成数を減らせることを明らかにした点である．本研究成果の詳細は，本報告書「５．主な発表論文等」の学会発表①等で報告されている．

(12) 消費電力に影響を及ぼすソフトウェア設計パターン：広く用いられているソフトウェアアーキテクチャの 1 つである MVC アーキテクチャ（Model-View-Controller）を対象に，Mobile App における消費電力の調査を行った．Green-miner2 を用いた実証実験の結果，Model 部分と View 部分の間に bundling presenter（イベントを蓄積し，一括して View に送信する機能）を介することで，最大で 30%の消費電力を節約できることを明らかにした．本研究成果の詳細の一部は，本報告書「５．主な発表論文等」の雑誌論文①等で報告されている．

(13) 上記の成果についてはそれぞれ論文としてまとめ，国内外の論文誌や研究集会での口頭発表を行った．今後はこれらの研究成果を統合，及び，高度化させ，当該研究分野をより発展させていく予定である．

参考文献

[1] Roberto Minelli and Michele Lanza, “Software Analytics for Mobile Applications--Insights & Lessons Learned,” CSMR, pp.144-153, 2013.
[2] Hammad Khalid, Emad Shihab, Meiyappan Nagappan, Ahmed E. Hassan, “What Do Mobile App Users Complain About? A Study on Free iOS Apps,” IEEE Software, 2013.
[3] Abram Hindle, “Green Mining: Investigating Power Consumption across Versions,” ICSE, pp.1301-1304, 2012.

５．主な発表論文等（研究代表者には下線）

〔雑誌論文〕（総件数・計 17 件，全て査読あり）

① Shaiful Chowdhury, Abram Hindle, Rick Kazman, Takumi Shuto, Ken Matsui, and Yasutaka Kamei, “GreenBundle: An Empirical Study on the Energy Impact of Bundled Processing,” In Proceedings of International Conference on Software Engineering (ICSE 2019), (To appear).
[PDF] http://posl.ait.kyushu-u.ac.jp/~kamei/publications/Chowdhury_ICSE2019.pdf

② Thong Hoang, Hoa Khanh Dam, Yasutaka Kamei, David Lo, and Naoyasu Ubayashi, “DeepJIT: An End-To-End Deep Learning Framework for Just-In-Time Defect Prediction,” In Proceedings of International Conference on Mining Software Repositories (MSR 2019), (To appear).
[PDF] http://posl.ait.kyushu-u.ac.jp/~kamei/publications/Thong_MSR2019.pdf

③ Yasutaka Kamei, Takafumi Fukushima, Shane McIntosh, Kazuhiro Yamashita, Naoyasu Ubayashi and Ahmed E. Hassan, “Studying Just-In-Time Defect Prediction using Cross-Project Models,” Journal of Empirical Software Engineering, Vol.21, No.5, pp.2072-2106, October 2016. [DOI] https://doi.org/10.1007/s10664-015-9400-x

④ Xin Xia, Emad Shihab, Yasutaka Kamei, David Lo and Xinyu Wang, “Predicting Crashing Releases of Mobile Applications,” In Proceedings of International Symposium on

Empirical Software Engineering and Measurement (ESEM 2016), pp.29:1-29:10, September 2016. [DOI] https://doi.org/10.1145/2961111.2962606

⑤ Junji Shimagaki, Yasutaka Kamei, Shane Mcintosh, Ahmed E. Hassan and Naoyasu Ubayashi, "A Study of the Quality-Impacting Practices of Modern Code Review at Sony Mobile," In Proceedings of International Conference on Software Engineering (ICSE2016), Software Engineering in Practice (SEIP), pp.212-221, May 2016. [DOI] https://doi.org/10.1145/2889160.2889243

⑥ Yasutaka Kamei and Emad Shihab, "Defect Prediction: Accomplishments and Future Challenges," In Proceedings of Leaders of Tomorrow / Future of Software Engineering Track at International Conference on Software Analysis, Evolution, and Reengineering (SANER2016), Vol.2, pp.33-45, March 2016. [DOI] https://doi.org/10.1109/SANER.2016.56

〔学会発表〕(総件数・計 13 件, そのうち招待講演 3 件, 発表者は筆頭著者)

① 首藤 巧, 亀井 靖高, 鵜林 尚靖, 佐藤 亮介, "ソースコード修正履歴が自動バグ修正の結果に与える影響の分析," 情報処理学会研究報告, ソフトウェア工学研究会, 2019 年 3 月.

② Shane McIntosh and Yasutaka Kamei, "Are Fix-Inducing Changes a Moving Target? A Longitudinal Case Study of Just-In-Time Defect Prediction," IEEE Transactions on Software Engineering, Vol.44, No.5, pp.412-428, May 2018. [Selected for the journal first program of the ICSE 2018]

③ 松本 卓大, 山下 一寛, 亀井 靖高, 鵜林 尚靖, "Deep Learning のリポジトリマイニングへの適用に向けた初期研究," 情報処理学会研究報告, ソフトウェア工学研究会, 2016 年 7 月.

④ 小須田 光, 亀井 靖高, 鵜林 尚靖, "ユーザ障害情報によるソースコード欠陥箇所予測ツール," 情報処理学会研究報告, ソフトウェア工学研究会, 2015 年 12 月.

⑤ 松本 卓大, 亀井 靖高, Shane McIntosh, 鵜林 尚靖, "マルチプラットフォーム向けソフトウェアに関する特定 OS 向け欠陥修正コミットの分析," 情報処理学会, ソフトウェアエンジニアリングシンポジウム(SES 2015), pp.1-8, 2015 年 9 月.

⑥ 島垣 潤二, 亀井 靖高, 鵜林 尚靖, "商用 Android ソフトウェア開発環境におけるコードレビュー統計の実証的研究," 情報処理学会研究報告, ソフトウェア工学研究会, 2015 年 7 月.

〔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 (PDF 公開)
http://posl.ait.kyushu-u.ac.jp/~kamei/publications.html

６．研究組織
研究協力者 (国際共同研究先):
(1) Canada : Ahmed E. Hassan (Queen's University), Shane McIntosh (McGill University), Emad Shihab (Concordia University), Abram Hindle (University of Alberta)
(2) Singapore : David Lo (Singapore Management University)
(3) Thailand : Pattara Leelaprute (Kasetsart University)
(4) Australia : Hoa Khanh Dam (University of Wollongong)
(5) China : He Jiang and Xiaochen Li (Dalian University of Technology)