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　 ５ 月 ２８ 日現在

機関番号：１６３０１
研究種目：基盤研究(C)（一般）
研究期間：2016～2018
課題番号：１６Ｋ００００９

研究課題名（和文）開発者間で個人差の出やすい特徴がソースコード品質へ及ぼす影響に関する実証的研究

研究課題名（英文）Empirical Study of Developers' Variation Impacts on Source Code Quality

研究代表者
阿萬　裕久（Aman, Hirohisa）

愛媛大学・総合情報メディアセンター・准教授

研究者番号：５０３３３５１３

交付決定額（研究期間全体）：（直接経費）　　3,400,000円

研究成果の概要（和文）：本研究では，開発者の個人差が影響すると考えられる次の観点についてデータ分析を行った：（1）変数等の名前，（2）コメント文，（3）コーディングスタイル，（4）プロジェクトへの貢献度．特に，変数の名前には個人差はあるものの，一定の一般的傾向はあり，ローカル変数に長過ぎる名前を付けたり複合語の名前を付けたりするのは望ましくなく，品質の低下を招く要因の一つであることが確認された．コメント文については，適切に情報を付加できているかどうかによってその価値が大きく異なることを確認でき，変数名とともに品質評価への新たな切り口としてその効果の高さが期待される結果となった．
全部で29編の査読付き論文を発表できた．

研究成果の学術的意義や社会的意義
本研究では，変数名やコメントといった開発者間で個人差が出やすい特徴にあえて注目している．これらの特徴は，これまでの研究でほとんど対象外とされていたが，本研究の成果から，これらは決して無駄な情報ではなく品質管理を行う上で有用な情報にもなりうることが分かった．
まだ明確な基準を公開できるほど研究を成熟させることはできていないが，データを適切に蓄積してフィードバックしていくことで品質の低下が懸念されるケースをいち早く見つけ出すことができると考えられる．それゆえ開発の途中で早めに他者による確認を施すといった予防措置をとりやすく，低コストで実用性も高く，現場にも展開しやすいと考えられる．

研究成果の概要（英文）：This study conducted various data analysis focusing on the individual difference from the following points of view: (1) the name of a variable, (2) comment statements, (3) the coding style, and (4) the contribution to the project.
Although the variable naming has a diversity, we found the common trend that the names of local variables should be short, i.e., local variables with long names or compound names tend to appear in inadequate quality programs. Through an empirical analysis of comments, we proposed a novel method for evaluating the value of the comment and proved that the effect of comments on the quality of the program is dependent on how rich information the comments provide. Together with the above other points of view, the results of our study showed that the individual difference in software development is a worthy point to be further studied.
We successfully published 29 referred papers in total.

研究分野：ソフトウェア工学

キーワード：ローカル変数　識別子名　コメント文　バグレポート　開発者特性　個人差　ソフトウェア品質

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９、ＣＫ－１９（共通）

１．研究開始当初の背景

(1) プログラミングではソースプログラムをコンパクトで分かりやすいものに仕上げることが望ましい．一般に，プログラムの規模が大きくなるほど，あるいは内容が複雑になるほど，そこに欠陥（バグ）が潜在している疑いは高い傾向にある．それゆえ，ソースプログラムを簡潔で理解しやすい状態に保つことは,品質の低下を未然に防ぐ上で極めて重要である.従来より，ソースプログラムの品質を定量的に把握するため，品質特性に対する定量的尺度（ソフトウェアメトリクス）が産学両方の立場から幅広く研究・活用されてきている．例えば，プログラムにおける行数や条件分岐命令の個数といった尺度により，プログラムの規模や複雑さを定量化し，そこに統計モデルや機械学習モデルを応用することでバグの潜在が疑わしいプログラムをテストに先立って絞り込むといったことが行われてきている．近年ではこれらに加え，プログラムの変更履歴情報を使った研究も盛んに行われるようになってきている．

(2) これまでに研究・活用されてきているソフトウェアメトリクスの多くは，プログラムの構造的な側面や変更履歴を対象としたものであり，開発者に依存した要素は主要な対象になっていなかった．特に，変数にどのような名前を付けるのか，プログラムにコメント（注釈）を書くのか，コメントを書くとすればどのような内容にするのかといった項目は，開発者にとって身近なものであるが，プログラムの実行には一切影響を及ぼさないという理由から，これまでの定量的研究では主対象から外されていた．
しかし近年になって，応募者らは変数名やコメントといった要素に着目することにも価値があるという傾向を定量的に示すことに成功していた．例えば，長い名前の変数が登場するモジュールの方がバグ潜在率は高い傾向にあり，さらにはコメント記述の有無でもバグ潜在率の差を確認できていた．

(3) 一般に，変数名の付け方やコメントの書き方は開発者間で個人差の出やすいところであるが，そのような個人差が結果的にプログラムの品質へ影響を及ぼしている部分もあると考えられる．現状では，コーディング規約といったガイドラインはあるが，変数名やコメントについては “良くない書き方” という現場での経験的かつ漠然とした概念しか存在していない．このように，“開発者個人が比較的自由に決めることのできる（個人差の出やすい）要素”が品質に及ぼす影響については未だ不明なところが多く，定量的なデータ分析の観点から解明することは注目に値する新たな研究分野となっていた．

＜引用文献＞

①阿萬裕久，野中誠，水野修，ソフトウェアメトリクスとデータ分析の基礎，コンピュータソフトウェア，Vol.28，No.3，2011，pp.12－28.
②門田暁人，伊原彰紀，松本健一，ソフトウェアリポジトリマイニング，コンピュータソフトウェア， Vol.30，No.2，2013，pp.52－65.
③阿萬裕久，オープンソースソフトウェアにおけるコメント記述およびコメントアウトとフォールト潜在との関係に関する定量分析，情報処理学会論文誌，Vol.53，No.2，2012，pp.612－621.
④Hirohisa Aman，Sousuke Amasaki，Takashi Sasaki and M. Kawahara，Empirical Analysis of Change-Proneness in Methods Having Local Variables with Long Names and Comments，Proc. 9th Int'l Symp. Empirical Softw. Eng. & Measurement (ESEM2015)，pp.50－53，2015.
⑤Hirohisa Aman，Sousuke Amasaki，Takashi Sasaki and Minoru Kawahara，Lines of Comments as a Noteworthy Metric for Analyzing Fault-Proneness in Methods，IEICE Trans. Inf. & Syst.，Vol.E98-D，No.12，pp.2218－2228，2015.

２．研究の目的

(1) 本研究の目的は，ソースプログラム中の変数名やコメントのようにプログラミングにおいて開発者間で個人差の出やすい要素が品質に及ぼす影響を統計的なデータ解析の立場から明らかにし，明確な指標を確立することである．一般にプログラムの基本的な書き方についてはさまざまなコーディング規約が基準として知られているが，コメントの書き方や変数名の付け方のように開発者間で個人差が出やすい要素については漠然としたルールしか存在していないのが実状である．そういった個人差が品質のばらつきを生み出していることが懸念されるが，現場での感覚的・経験的な議論に留まっている．本研究では実データ解析の立場から実務者にとって有益な知見を見出すことを目指す．

(2) さらに本研究では，さまざまな個人差が混在する中で，どのような体制（分担）でソフトウェア開発が行われるのが適切なのかにも着目する．具体的には，一つのソースファイルを開発･保守していく中で,特定の１人の開発者だけがそのソースファイルに携わる場合があれば，複数の開発者が協力してそのソースファイルの保守を行っていく場合もあると考えられる．後者の場合，本研究で着目している個人差が何らかの影響を及ぼしている可能性があると思われ

ることから，そのような体制の違いについても定量的なデータ解析の観点から研究を行う．

３．研究の方法

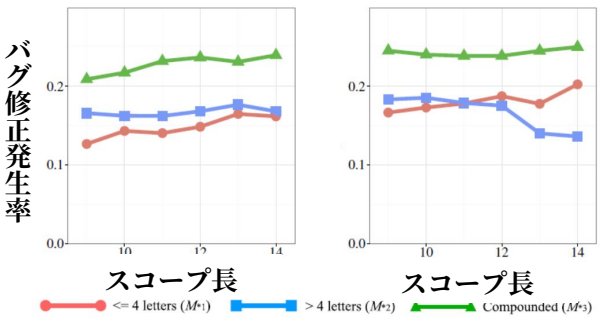
(1) まずは分析対象のソースコードをインターネット上のオープンソースプロジェクトから取得する．研究の一般性を高めるためには，さまざまな分野のプロジェクトからソースコードを集めて分析することが必要不可欠である．近年ではソースコードのバージョン管理システムとして Git が広く採用されており，ソースコード並びにそれらの改変履歴を自動的かつ容易に取得できるようになっている．しかしながら，旧バージョンも含めて全てのソースコードに対して静的解析を行い，その上で本研究で着目する特徴データ（変数名等）を取得するには多大な時間がかかる．そのため，時間はかかるが研究分担者と協力しながら，可能な限り多くのプロジェクトからデータを収集していく．そして，収集データをデータベース化して共同利用可能な形に整理する．その上で，これまでに得られている知見にどこまで一般性があるか，データを蓄積しながら検討を行っていく．

(2) 変数名等の“開発者間で個人差の出やすい”特徴とプログラム品質との関係について原因を解明していくため，可能な限りさまざまな視点から特徴を定量化していく．例えば，変数の名前に使われている文字数や単語数，その変数の有効範囲（スコープ）の長さ，コメント文の位置や登場する文の長さ等を定量化し，数理モデルによる解析を行っていく．これまでにコーディング規約違反という視点からプログラムの書かれ方（個人差）が品質に及ぼす影響について検討した研究はあるが，具体的な違反内容や特徴との関係については触れられていない．本研究で着目する特徴データを取得・分析していくことで新たな知見が得られるのではないかと考えられる．その他にも，バージョンアップに伴う特徴データの変化についても数理モデル化して分析し，どういった場合にソースコードの特徴が変化して品質に影響を及ぼしているかを解析していく．

(3) 個々のプログラムを対象とした研究のみならず，開発者を対象とした研究も並行して進める．具体的に，プロジェクトで登場する開発者を名前とメールアドレスを使って識別し，どの開発者がどのソースファイルに携わっているかやプロジェクト内でどれほど多くの貢献を行っているかについて定量データを収集する．そして，収集データの統計解析を通じて，プロジェクトの評価やソースファイルの開発・保守のされ方の評価を行っていく．

４．研究成果

(1) ローカル変数等の名前：従来から経験的には“ローカル変数の名前は短くてよい・短くすべき”と言われていたが，本研究での定量的なデータ分析を通じて，このことを実証することができた．まず，ローカル変数の名前が長いプログラムの方が，結果的にバグが潜在している可能性が高いという傾向を確認することができた（図１）．図１では名前が長く，なおかつ複合語になっている場合（緑色の折れ線）の方がバグ修正が起こりやすいという傾向を示している．横軸は変数のスコープ長であり，スコープが広い方がややバグ修正が起こりやすい傾向も示している．なお，この傾向にはコメントが有るか無いかも影響しており，全体的にコメントが書かれている方がバグ修正が起こりやすかったが，スコープが広くなると逆にコメントの存在によって品質が上がっている（バグ率が低くなる）傾向も見られた．



(a)コメント無し　　(b)コメント有り

その後，この成果を発展させ，変数の名前の長さ，スコープの長さ及び変数の型に応じて変数名の異常度をマハラノビス距離の概念を使って定量化する手法を考案した．国際会議で発表の後，現在，国際ジャーナルへ投稿中である．本件に関連する研究成果として，査読付き論文を７編発表できた．そのうちの １編については，指導学生を筆頭著者として発表を行い，IEEE Computer Society Japan Chapter FOSE Young Researcher Award を受賞した．

(2) コメント文の評価：コメント文は開発者がプログラム中に自由に記述できるものであるだけに，その個人差は大きい．それゆえ，コメントの多寡の評価基準も人によって異なることから，各開発者の平均と分散を考慮してコメントの量を（一種の偏差値として）評価する手法を提案した．そして，評価実験を通じて，コメントが多すぎる場合はバグ潜在の疑惑度を通常よりも高めに設定することでより高精度でバグ予測が行えることを学術雑誌で発表した．また，コメント文は自然言語で書かれることが一般的であるため，本研究では自然言語処理技術を利

用したコメントの解析も行った．その結果，どういった単語が使われやすい傾向にあるかが分かった．具体的には，Java メソッドの内部に書かれるコメントの場合，一種の禁止事項（～してはいけない）や推奨事項（～すべき）に該当する単語が比較的多く使われる傾向にあることが分かった．この結果から，メソッド内にコメントが必要とされる場合，ソースコードだけでは気付きにくい制約情報がコメントとして書かれることが比較的多く，そのような制約の存在がプログラムをより複雑にしてしまい，ひいては（これまでの研究で示されているように）コメント有りのプログラムの方が結果的にバグ修正は起こりやすいという傾向につながっているのではないかと推察される．この視点についてより定量的にとらえるため，Doc2Vec を使ってプログラムをベクトル化し，コメントの削除によってどれほどベクトルが変化するかによってコメントが提供している情報の量を定量化する手法の提案も国際会議で行った．本件に関連する研究成果として，査読付き論文を 4 編発表できた．

(3) プログラムの書き方：プログラムの書き方（コーディングスタイル）もまた，個人差が出やすい特徴の一つである．そこで本研究では，静的解析ツールを活用し，そこで出力される警告（コーディング規約違反）の動向を調べるという研究も行った．具体的には，一般に広く知られている静的解析ツールでは，多数の警告が出力される一方，それらの多くが実際には無視されているという傾向をデータ解析を通じて定量的に示すことができた．そして，どういった警告は重要視されてすぐに修正されるのかについても分析を行った．結果として，プロジェクトによって傾向が変わりやすく，しかも個人を区別して分析を行うとさらに多様性が増す結果となった．それゆえ，いくつかの関連研究でも指摘されているように，適切なカスタマイズの仕組みが重要であるといえる．本研究で提案している手法はツールユーザに負担をかけることなく，リポジトリからデータを収集して各警告の重要性を自動評価する仕組みになっており，最新の研究成果は 2019 年 5 月末に開催される国際会議で発表する予定である．
本件に関連する研究成果として，査読付き論文を 3 編発表できた．そのうちの 1 編は，指導学生を筆頭著者として国際会議で発表し，Best Student Presenter Award を受賞した．

(4) 開発者の貢献：各開発者がソースファイルの開発・保守にどのように貢献しているか，という観点での分析も行った．これは，プログラミングには一定の個人差があるのが自然であり，その上でどういった体制で開発・保守が行われるのが望ましいのか？という疑問を出発点として始めた研究である．結果として，プロジェクト内での開発には“パレートの法則”が概ね成立しており，比較的少数の開発者たちの貢献が大部分を占めることが多く，逆にこの比率が低い場合，そのプロジェクトの成果物にはバグが比較的多いという傾向が見られた．つまり，適度にプロジェクト全体を管理できる人たちがいないとプロジェクトの健全な発展は難しいという傾向にあると推察される．次に，ソースファイル単位で分析を行ったところ，１人で開発を行っていたソースファイルに別人（２人目）が参画した直後は一時的に品質が低下していることが比較的多いことが分かった．実際，コード修正からバグ修正が起こるまでの時間間隔を生存時間分析を通じて比較したところ，新たな人物がコードに手を加えた場合がバグ修正発生までの時間間隔は最も短いことが分かった．
本件に関連する研究成果として，査読付き論文を 3 編発表できた．そのうちの 1 編は，指導学生を筆頭著者として国際会議で発表し，Best Student Paper Award を受賞した．

５．主な発表論文等

〔雑誌論文〕（計　6　件）

① 鈴木 翔，阿萬 裕久，川原 稔，決定木を利用した Java メソッドの名前と実装の間の適合性判定モデルとその評価，コンピュータソフトウェア，vol.35，no.4，2018，pp.115－121，査読有
DOI: 10.11309/jssst.35.115

② Akito Sunouchi, Hirohisa Aman and Minoru Kawahara, A Quantitative Analysis on Relationship between an Early-Closed Bug and Its Amount of Clues: A Case Study of Apache Ant, IEICE Transactions on Information and Systems, vol.E101-D, no.10, 2018, pp.2523－2525, 査読有
DOI: 10.1587/transinf.2018EDL8094

③ Kazuki Yamauchi, Hirohisa Aman, Sousuke Amasaki, Tomoyuki Yokogawa and Minoru Kawahara, An Entropy-Based Metric of Developer Contribution in Open Source Development and Its Application to Fault-Prone Program Analysis, International Journal of Networked and Distributed Computing, vol.6, no.3, 2018, pp.118－132, 査読有
DOI: 10.2991/ijndc.2018.6.3.1

④ Aji Ery Burhandenny, Hirohisa Aman and Minoru Kawahara, An Evaluation of Coding Violation Focusing on Change History and Authorship of Source File, International Journal of Networked and Distributed Computing, vol.5, no.4, 2017, pp.211－220, 査読有
DOI: 10.2991/ijndc.2017.5.4.3

⑤ Aji Ery Burhandenny, Hirohisa Aman and Minoru Kawahara, Change-Prone Java Method Prediction by Focusing on Individual Differences in Comment Density, IEICE Transactions on Information and Systems, vol.E100-D, no.5, 2017, pp.1128－1131, 査読有
DOI: 10.1587/transinf.2016EDL8224

⑥ Yuto Miyake, Sousuke Amasaki, Hirohisa Aman and Tomoyuki Yokogawa, A Replicated Study on Relationship Between Code Quality and Method Comments, Applied Computing and Information Technology. Studies in Computational Intelligence, vol 695. Springer, 2017, pp.17－30, 査読有
DOI: 10.1007/978-3-319-51472-7_2

〔学会発表〕（計 23 件）

① Hirohisa Aman, Sousuke Amasaki, Tomoyuki Yokogawa and Minoru Kawahara, A Doc2Vec-Based Assessment of Comments and Its Application to Change-Prone Method Analysis, Proc. 25th Asia-Pacific Software Engineering Conference, 2018, pp.643－647, 査読有

② Keiichiro Tashima, Hirohisa Aman, Sousuke Amasaki, Tomoyuki Yokogawa and Minoru Kawahara, Fault-Prone Java Method Analysis Focusing on Pair of Local Variables with Confusing Names, Proc. 44th Euromicro Conference on Software Engineering and Advanced Applications, 2018, pp. 154－158, 査読有
DOI: 10.1109/SEAA.2018.00033

③ Hirohisa Aman, Sousuke Amasaki, Tomoyuki Yokogawa and Minoru Kawahara, A Survival Analysis of Source Files Modified by New Developers, M. Felderer, D. M. Fernández, B. Turhan, M. Kalinowski, F. Sarro, D. Winkler (Eds.) Product-Focused Software Process Improvement, Lecture Notes in Computer Science, vol. 10611, Springer, 2017, pp. 80－88, 査読有
DOI: 10.1007/978-3-319-69926-4_7

④ Hirohisa Aman, Sousuke Amasaki, Tomoyuki Yokogawa and Minoru Kawahara, Empirical Study of Abnormalities in Local Variables of Change-Prone Java Methods, Proc. IEEE 28th International Symposium on Software Reliability Engineering Workshops, 2017, pp. 214－221, 査読有
DOI: 10.1109/ISSREW.2017.37

⑤ Hirohisa Aman, Aji Ery Burhandenny, Sousuke Amasaki, Tomoyuki Yokogawa and Minoru Kawahara, A Health Index of Open Source Projects Focusing on Pareto Distribution of Developer's Contribution, Proc. IEEE 8th International Workshop on Empirical Software Engineering in Practice, 2017, pp. 29－34, 査読有
DOI: 10.1109/IWESEP.2017.14

⑥ Hirohisa Aman, Sousuke Amasaki, Tomoyuki Yokogawa and Minoru Kawahara, Local Variables with Compound Names and Comments as Signs of Fault-Prone Java Methods, Joint Proc. 4th International Workshop on Quantitative Approaches to Software Quality and 1st International Workshop on Technical Debt Analytics, 2016, pp. 4－11, 査読有
http://ceur-ws.org/Vol-1771/

〔その他〕
ホームページ等
http://se.cite.ehime-u.ac.jp/research/papers-j.html
http://se.cite.ehime-u.ac.jp/tool/index-j.html
http://se.cite.ehime-u.ac.jp/data/index-j.html

６．研究組織

(1)研究分担者

研究分担者氏名：天嵜　聡介

ローマ字氏名：(AMASAKI, Sousuke)

所属研究機関名：岡山県立大学

部局名：情報工学部

職名：助教

研究者番号（8 桁）：00434978

研究分担者氏名：横川　智教

ローマ字氏名：(YOKOGAWA, Tomoyuki)
所属研究機関名：岡山県立大学
部局名：情報工学部
職名：准教授
研究者番号（8 桁）：50382362