様式Ｆ－１９

科学研究費助成事業（学術研究助成基金助成金）研究成果報告書

平成 25 年　6 月 3 日現在

機関番号：14401
研究種目：挑戦的萌芽研究
研究期間：2011～2012
課題番号：23650014
研究課題名（和文）　ソフトウェアプロダクトに対する印象の計測

研究課題名（英文）　On a measurement of software impression

研究代表者
楠本　真二（KUSUMOTO SHINJI）
大阪大学・大学院情報科学研究科・教授
研究者番号：30234438

研究成果の概要（和文）：
本研究では，ソフトウェアプロダクトに対する印象の計測を目的とし，その最初の段階として，ソフトウェア開発の中で最近注目を集めているソフトウェア保守におけるソフトウェアの印象の計測手法について検討した．具体的には，ソースコード上で人間が保守に悪影響を与えると思われる箇所（悪い印象を持った箇所）や複雑であると思われる箇所の特定を，ソースコードの特徴を評価するメトリクスを用いて行う手法を提案し，その有効性の評価を行った．

研究成果の概要（英文）：
This research examined the measurement method for software product impression. As the first step, we target the impression of software products used in software maintenance processes. Here, we tried to grasp the parts of programs that software developers have a significant negative impact on software maintenance by using several software metrics measured from them. The results of empirical evaluation show the usefulness of the proposed approach.

交付決定額

（金額単位：円）

	直接経費	間接経費	合　計
交付決定額	3,770,000	870,000	4,640,000

研究分野：総合領域
科研費の分科・細目：情報学・ソフトウェア
キーワード：ソフトウェア，メトリクス，ソフトウェア保守，複雑度，コードクローン

１．研究開始当初の背景

人間は，自分が感心を持つすべての対象に対して印象を感じ，その印象によって対象に対する行動(アクション)を決めることが少なくない．消費者である人は自分が感じている印象によって，製品やサービスを選択することが多い．印象を工学的に扱うための学問として印象工学というものが提案されている．ソフトウェアの分野においても印象は重要である．例えば，ユーザがある用途にソフトウェアパッケージを購入する場合，そのソフトウェアの機能や品質，あるいは，実際に使用したユーザの意見等を参考にして，最も自分に印象の良いものを選ぶ場合も多い．更に，実際のソフトウェア開発プロセスにおいても，ある作業工程の入力となるプロダクトの印象は，その作業の担当者の生産性や効率に影響を与える(図 1)．しかし，ソフトウェアを対象とした印象の計測，分析等の手法の研究は行われていない．

２．研究の目的

本研究では，ソフトウェアプロダクトに対する印象の計測を目的とし，その最初の段階として，ソフトウェア開発の中で最近注目を集めているソフトウェア保守におけるソフ

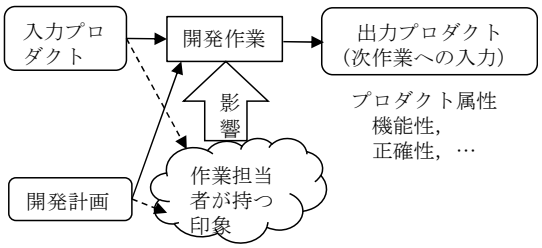

図１　ソフトウェア開発における印象

トウェアの印象の計測手法について検討する．

３．研究の方法

ソフトウェア保守は，通常，変更要請，変更の可否検討，保守のための分析，修正・テスト，移行の順番に行われる．この中で最も重要な作業は，保守対象ソースコードの理解であると言われている．保守対象ソースコードの内容が理解できて初めて，具体的な見積もりや作業が開始できる．理解がしづらいというのは，ソースコードに対する悪い印象を表すことになる．

そこで，本研究では，プロダクトとしては，ソースコードを，印象としてはソフトウェア保守における悪い印象を対象とする．また，ソフトウェアに対する悪い印象を評価するために，ソースコードから計測可能なメトリクスを利用する．

４．研究成果

(1) コードクローンに基づく評価

①概要

複数の被験者に対して，同一ソースコードを提示し，保守に悪影響を与えると思われる箇所（悪い印象を持った箇所）を指定してもらった．結果として，同じ箇所であっても，被験者によって印象が良いと判断される部分と悪いと判断される部分が多く見られた．その結果に基づいて，個々の被験者が悪い印象を持った箇所の情報に基づいて，別のソースコード上で同じ特徴を持つ箇所を提示するツールの試作を行った．

②悪い印象を持った箇所

ここでは，開発者が悪い印象を持つ箇所として，ソースコード上のコードクローンに着目した．コードクローンとは，ソースコード中の同一，あるいは類似するコード片のことを指す．一般的にコードクローンの存在はソフトウェアの保守に悪影響を与えると考えられており，これまでにコードクローンの検出技術，及びその除去技術が盛んに研究されている．しかし，コードクローンの中には有用なものもあるという指摘があり，開発者によってその判断が分かれる．つまり，あるコードクローンが有用か否かを判別する基準は，個々の開発者によって，あるいはコードクローンの検出目的によって異なる．したがって，どのようなコードクローンがソフトウェア保守に有用かを一般化することは難しい．

③提案手法

開発者がソフトウェア保守に悪いという印象を持つコードクローンを評価するために，機械学習を用いたコードクローンの自動フィルタリング手法による特定方法を提案した．提案する手法では，まず利用者に一部のコードクローンを“有用”（すなわち，悪影響を及ぼすもの）または“有用でない”(すなわち，悪影響を及ぼさない）に分類してもらう．その後，それらを学習データとして，残りのコードクローンを“有用” または“有用でない” に自動的に分類し，利用者に提示する．提案手法を用いることで，利用者は一部のコードクローンに対して自身が有用と思うか否かを入力するのみで，膨大な検出結果から自身にとって有用なコードクローンのみを抽出することが可能となる．これにより，有用なコードクローンの特定に要するコストを低減させることができる．さらに，利用者ごとに学習データを作成するため，利用者や利用状況による有用か否かの判別基準の違いにも対応できる．

提案手法の概要を図２に示す．

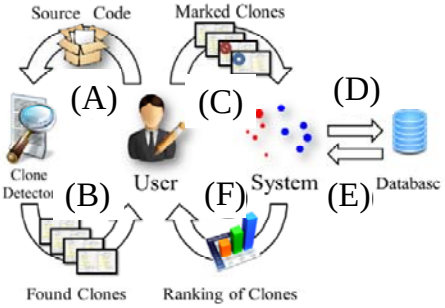

図２　フィルタリング処理の流れ

提案手法が入力として必要とするものは以下の通りである．

・対象ソフトウェアのソースコード

・対象ソフトウェア中に存在する全クローンセットの位置情報

・学習データ

ここでクローンセットとは，互いにコードクローン関係にあるコード片の集合を指す．また学習データとは，対象ソフトウェアから検出されたクローンセットの一部であり，利用者によって“有用” あるいは“有用でない” のいずれかの分類が付与されている．提案手法は与えられた入力情報をもとに，利用者によって分類されていないすべてのクローンセットについて，それらが“有用” か“有用

でない” かを自動的に判別し，提示する．
以下，図２の(A)～(F)について説明する．
(A) 利用者が対象システムのソースコードを検出ツールに入力する．
(B) 検出ツールは入力されたソースコードからクローンセットを検出する．
(C) 検出ツールにより検出されたクローンセットの中のいくつかを，利用者がその有用性に応じて“有用” か“有用でない”かに分類し，登録する．
(D) 提案システムは利用者によって分類されたクローンセット(学習データ) をデータベースに格納する．
(E) 提案システムは学習データを分析し，機械学習を用いてそれらの特徴を抽出する．
(F) 提案システムは(E)で抽出した特徴をもとに，学習データに含まれていないクローンセットすべてに対し，それらの有用性を判別する．

④評価結果
提案手法の予測精度を評価するため，4 つのオープンソースソフトウェアを対象とした評価実験を行った．具体的には，先ず，被験者によって分類されたクローンセットのうち，一定数をランダムに抽出し，学習データとしてシステムに登録する．次に，それぞれの被験者について，学習データとして登録されていないクローンセットの分類をシステムに予測させる．更に，システムの予測結果と，実際に被験者が分類を行った結果がどの程度一致しているのかを調査する．上記の作業を 256 回繰り返し，予測精度の平均値を算出する．
結果として，提案手法の予測精度は概ね 70%を超え，被験者やプロジェクトによっては 90%以上の精度で予測ができることを確認した．さらに，あるコードクローンが“有用”か否かの判断基準が利用者によって異なるということを確認した．

(2)複雑度に基づく評価
①概要
保守のしやすさ（理解しやすさ）を評価するための手法として，ソースコードの複雑度メトリクスを用いる手法がある．しかし，必ずしも開発者の主観的な評価（印象）と一致するとは限らないという指摘がある．本研究では，メトリクスによる評価を改善することで，主観的評価と合致した結果が得られるかどうかを評価した．
②複雑度メトリクス
ソースコードの複雑度を表すメトリクスの中で最も有名なものの一つとして，McCabe のサイクロマチック数がある．サイクロマチック数は，ソースコードを有向グラフとして見た時に含まれる閉路の数を評価するものであり，直観的には分岐数に１を加えた値になる．分岐が多いほど理解するのが複雑になるという解釈である．しかし，サイクロマチック数とソースコードの可読性の相関が低いという報告もある．一つの理由として，switch 文中に含まれる case 文や単純な if 文の連続というのは，高いサイクロマチック数を示すが，それは開発者にとってはあまり複雑であるという評価にはならない．

③提案手法
そこで，本研究ではソースコード中の繰り返し構造を簡略化してサイクロマチック数を計測する手法を提案する．具体的には，ソースコード中の繰り返し部分を抽出し，それらを折り返した上で，サイクロマチック数を計測する．提案手法の概要を以下に示す．
(A) 計測対象ソースコードを AST(Abstract Syntax Tree)に変換する．
(B) AST 上の繰り返し構造を折りたたむ．AST 中の兄弟ノードは，ソースコード上の出現順による順序性を持っている．そこで，連続する兄弟ノードが類似していれば，それらの兄弟ノードからなる部分木の集合を繰り返し構造とみなす．
(C) 折りたたまれた AST を元に，簡略化されたソースコードを出力する．
(D) (C)のソースコードに対してサイクロマチック数を計算する．

④評価実験
提案手法を実装し，評価実験を行った．実装は Java で記述されたソースコードのみが対象である．本実験では，UCI source code data sets に含まれる約 13000 のソフトウェアである．これらのソフトウェアに含まれるすべてのメソッドに対して，提案手法を適用した場合と適用しなかった場合の，サイクロマチック数を計測した．更に，提案手法によって計測されるメトリクスは，人の印象に一致しているかどうかを評価した．
実験の結果，従来サイクロマチック数の数値が大きかった多くのメソッドの数値が削減された．また，ほとんどの被験者が，提案手法によって計測されたサイクロマチック数に基づく結果の方が，印象による結果に一致するという回答を得た．

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　件）
① 楊嘉晨，堀田圭佑，肥後芳樹，井垣宏，楠本真二，″機械学習を用いた類似度に基づく有用なコードクローンの自動特定手法，″ 情報処理学会論文誌，54，2，pp.

807-819(2013).

②肥後芳樹, 植田泰士, 西野稔, 楠本真二：“プログラム依存グラフを用いた増分的なコードクローン検出”, 情報処理学会論文誌, 53(2), pp.601-611 (2012).

〔学会発表〕(計　件)
①佐々木唯, 石原知也, 堀田圭佑, 畑秀明, 肥後芳樹, 井垣宏, 楠本真二, :″プログラム構造の簡略化によるメトリクス計測方法の改善,″ 電子情報通信学会ソフトウェアサイエンス研究会, 2012 年 7 月 27 日.

②Keisuke Hotta, Yoshiki Higo, and Shinji Kusumoto: ″Identifying, Tailoring, and Suggesting Form Template Method Refactoring Opportunities with Program Dependence Graph,″ presented in the 16th European Conference on Software Maintenance and Reengineering (CSMR2012), pp. 53-62, Hungary, March 28, 2012.

③Yoshiki Higo, Akira Saitoh, Goro Yamada, Tatsuya Miyake, Shinji Kusumoto, and Katsuro Inoue: ″A Pluggable Tool for Measuring Software Metrics from Source Code,″ presented in the Joint Conference of the 21th International Workshop on Software Measurement and the 6th International Conference on Software Process and Product Measurement (IWSM/MENSURA2011), pp.3-12, November 3, 2011.

④石原知也, 堀田圭佑, 肥後芳樹, 井垣宏, 楠本真二, ″大規模ソフトウェア群に対するメソッド単位のコードクローン検出,″ 電子情報通信学会ソフトウェアサイエンス研究会, 2012 年 3 月 13 日.

⑤村上寛明, 堀田圭佑, 肥後芳樹, 井垣宏, 楠本真二, ″ソースコード中の繰り返し部分に着目したコードクローン検出手法の提案,″ 電子情報通信学会ソフトウェアサイエンス研究会, 2012 年 3 月 13 日.

〔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
楠本　真二（KUSUMOTO SHINJI　）
大阪大学・大学院情報科学研究科・教授
研究者番号：30234438

(2)研究分担者
岡野　浩三（OKANO KOZO）
大阪大学・大学院情報科学研究科・准教授
研究者番号：70252632

(3)連携研究者
肥後　芳樹（HIGO YOSHIKI）
大阪大学・大学院情報科学研究科・助教
研究者番号：70452414