メトリクスと閾値による保守性・再利用性評価式の作成・更新プロセス

津田直彦[†1]　高田正樹[†1]　鷲崎弘宜[†1]
深澤良彰[†1]　杉村俊輔[†2]
保田裕一朗[†2]　二上将直[†2]

本研究では，エキスパートによる人手評価を複数の観点に分割し，各観点によるソースコード評価を自動化していくためのプロセスを提案する．特に，最適な評価式を構築するための二つの手法の使い分けについて議論を深めたい．

Metrics and Thresholds: Process to Create and Update Evaluation Formulas of Maintainability and Reusability

Naohiko Tsuda ,[†1] Masaki Takada ,[†1] Hironori Washizaki ,[†1]
Yoshiaki Fukazawa ,[†1] Shunsuke Sugimura ,[†2]
Yuichiro Yasuda [†2] and Masanao Futakami [†2]

1. 提案プロセスと事例

現在我々は建機メーカーと共同で，ソースコードを適切に自動評価するための評価式作成プロセスを研究している．そのプロセスでは下記の手順を繰り返すことにより，GQM モデルと評価式を順次改善し，最終的に適切な評価モデルを得ることを目指している．

- Step1: GQM モデルの作成・更新
- Step2: 評価式の作成・更新
- Step3: 評価式の最適化
- Step4: 評価式の分析

Step1 で取り扱う Goal-Question-Metric 法（GQM 法）[1] はソフトウェアの品質をメトリクスを用いて取り扱う手法であり，図 1 のような三階層モデルを作成する．GQM モデルによって品質評価をする際の観点が細分化され，問題の原因追跡が行いやすくなり，またメトリクスの利用目的も明文化される．当該プロジェクトの開発熟練者などのドメイン知識を有する人物らを交えた合議を通して GQM モデルを作成するのが望ましい．

†1 早稲田大学 基幹理工学研究科
Dept. Computer Science and Engg., Waseda University
†2 株式会社小松製作所
Komatsu Ltd., Development Division, ICT Development Center

図 1　GQM モデルの例

表 1 と表 2 に我々が「保守性・再利用性が高い」ことをゴールとして作成した GQM モデルの Q と M の抜粋を記した．抜粋した Q と M はケーススタディにおいて自動評価をエキスパート評価に近似させられたものであり，コードサイズと関数の複雑度と関係している．GQM モデル作成時には既存のソフトウェア品質診断ツールである Adqua[2] を参考とした．

表 1　保守性・再利用性の低いファイルを検出するための Q

Q	Description	Related Metrics
Q02	責務量（ステートメント数や関数数）は適切か	MFl019, MFl477
Q05	関数の処理が複雑すぎないか	MFl004K, MFl005K, MFl006K, MFl007K

次節では Step2 と Step3 について述べる．

2. 評価式の作成と最適化

Step3 の評価式の最適化では機械学習を用いる．システム全体のうちの一部のファイルに対してエキスパートが人手で評価を与え，それを教師ラベルとして

表 2　保守性・再利用性の低いファイルを検出するためのファイルメトリクス

Metrics	Description
MFl019	関数の数
MFl477	実行コード行数
MFl004K	各関数の制御フローネストのファイル内平均値
MFl005K	各関数の制御フローネストのファイル内最大値
MFl006K	各関数のサイクロマチック複雑度のファイル内平均値
MFl007K	各関数のサイクロマチック複雑度のファイル内最大値

学習させる．これにより，メトリクスの値を元にして行うファイルの自動評価を最適化（エキスパート評価への近似）させることができる．最適化の目標関数としては，Cohen のカッパ係数や F1 値といった二分類問題の一致度指標を用いる．尚，本研究で取り扱う人手評価と自動評価は「許容可能」「許容不可」のいずれかを下すものとしている．

Step1 で作成した GQM モデルには保守性・再利用性などの品質特性を評価する際の細分化された観点が Q として現れている．エキスパートに評価をしてもらう際には，各 Q の説明文を提示してチェックリストベースレビューをしてもらう．例えば人手評価するファイルが 5 個，Q が 10 個であれば，各ファイルに対して 10 個の評価，全部で 50 個の評価をしてもらう．

2.1　評価式の作成と最適化を自動化する場合

Step2 と Step3 を同時に行える機械学習手法として分類木（決定木）がある．例えば，学習データとして Q5 のエキスパート評価と Q5 に関するメトリクス値を用意すれば Q5 に関する分類木を構築することができる．分類木は以下のような形式の評価式を出力する．

- if (MFl007K <= 5): OK
- else:
- :...if (MFl006K > 3.333333): NG
- 　　else:
- 　　:...if (MFl006K <= 3.090909): NG
- 　　　else: OK

分類木のアルゴリズムはいくつか提案されているが，C5.0 というアルゴリズムでは二分木だけでなく三分木も扱える．

2.2　評価式をあらかじめ用意する場合の最適化

分類木は評価式の作成まで自動化してくれるが，学習データに偏りがある場合には不自然な形の評価式を出力してしまうことが懸念される．そこで，評価式の形はあらかじめ開発者が人手で決めておき，その式で使うべき閾値の組み合わせを最適化することを我々は考えた．また，この方法では式に組み込んだメトリクスの閾値が漏れなく得られる．

2.2.1　正常判定と異常判定による評価式の最適化

本研究では正常判定と異常判定による評価式の最適化を実現した．尚，本評価式では小さく抑えることが望ましいメトリクスの取り扱いを想定している．

正常判定とは，下記のようないずれかのメトリクスが閾値 T_{ORi} 以下であれば TRUE（許容）となる判定である．

$$(MFl019 <= T_{OR1})||(MFl477 <= T_{OR2}) \quad (1)$$

一方で異常判定とは，下記のようないずれかのメトリクスが閾値 T_{ANDi} を越えると FALSE（許容不可）となる判定である．

$$(MFl019 <= T_{AND1})\&\&(MFl477 <= T_{AND2}) \quad (2)$$

ある評価項目 Qi に関係するメトリクス全てを小さく抑えることが現実的には難しい場合，いずれかのメトリックが十分抑えられていればそれで許容したい．しかし，別のメトリクスが大きくなりすぎているならば，やはりそれは許容すべきではない．このような開発者の意図を組み込んだ評価式は正常判定と異常判定の連言で表せる．

この評価式で用いる閾値の組み合わせの最適化は焼きなまし法や遺伝的アルゴリズムで組み合わせを探索することで達成できる．

3. おわりに

Step4 は現在研究過程にあるが，Q1 や Q3 といった自動評価が上手く行かなかったものの原因のパターン分類や調査方法の一般化を行いたい．

加えて，Step3 では異なる二つの手法を取り上げたが，その使い分け方や長短についての議論を深めていきたい．今回の事例で我々は建機メーカーが開発した中型パワーショベルを制御するためのシステムを対象として Step3 を実施した．実施対象はそのシステムが含む C++の 23 個のサブシステム（C++ファイル 481 個）のうち，重要性と保守の要望が高い 3 つのサブシステムが含む 132 個のファイルとした．そして，Q2 と Q5 が我々の提案手法と分類木によって自動評価のエキスパート評価への近似が可能であることを確認した．

参　考　文　献

1) Basili, V.R. and et al.: A Methodology for Collecting Valid Software Engineering Data, *IEEE Trans. Softw. Eng.*, Vol.10, No.6, pp.728–738 (1984).
2) Washizaki, H. and et al.: A Framework for Measuring and Evaluating Program Source Code Quality, *Proc. of the 8th Int. Conf. on PROFES*, PROFES'07, Berlin, Heidelberg, Springer-Verlag, pp.284–299 (2007).