計測に基づくソフトウェア品質管理

楠　本　真　二*

Software Quality Management based on Measurement

Key Words：Software Metrics, Testing, Review, Object-Oriented Development

1．は　じ　め　に

近年，コンピュータシステムの応用分野は拡大し，システムに対する要求も多様化してきている．このような状況に柔軟に対応するため，コンピュータシステムにおけるソフトウェアの役割はますます重要になってきている．それに伴い，ソフトウェアの品質が，コンピュータシステム全体の品質，あるいは，開発の生産性に大きな影響を与える場合が少なくない，ソフトウェア品質，生産性の向上は，今日のソフトウェア工学における主要な目標として位置付けられている．

著者はソフトウェア工学における研究課題の中で，次に述べるようなテーマについて研究を行っている．

(1) ソフトウェア開発プロセスの改善

(2) ソフトウェア規模の見積り

(3) 開発資源の効果的な割当

以下では，(3)の開発資源の効果的な割当に関する研究テーマ[3]について概要を述べる．

2．開発資源の効果的な割当（メトリクスを用いたfault-proneクラスの予測）

ソフトウェアのレビューやテストは，出荷するソフトウェアに含まれるフォールトを発見し除去するために必要な作業である．それらに費やされる労力は膨大なものであるため，フォールトが含まれると予測されるモジュールを予め特定することにより，レビューやテストの労力をそのモジュールに集中させることが効果的である．

2.1　目的

本研究では，設計の初期段階において，オブジェクト指向ソフトウェア向けのいくつかの複雑度メトリクスを用いて，クラスにフォールトが発生するかどうか(fault-proneかどうか)を予測する手法を提案する．提案する手法では，オブジェクト指向開発法(OMT[4])に基づく分析／設計／実装フェーズに4つのチェックポイントを設ける．各チェックポイントにおいて，入手可能なプロダクトに関する情報(設計仕様書やソースコード)を用いて，複雑度メトリクスのうち適用可能なものを適用する．次に，多変量ロジスティック回帰分析[5]を用いて，fault-proneなクラスを予測する．

2.2　使用するメトリクス

本研究で用いたメトリクスは次のものである[1]．

(1) WMC(クラスの重み付きメソッド数)：計測対象クラスに含まれるメソッド数．

(2) DIT(継承木における深さ)：計測対象クラスの継承木中での位置．

(3) NOC(子クラスの数)：計測対象クラスの子クラスの数．

(4) CBO(クラス間の結合)：計測対象クラスの他クラスとの結合数．更に，結合対象クラスの種類によって以下の2種類に分類する[2]．

- CBON：新規開発クラスとの結合数．
- CBOR：再利用クラスとの結合数．

(5) RFC(クラスの反応)：メッセージに反応して潜在的に実行される計測対象クラスのメソッド数．

(6) LCOM(メソッドの凝集の欠如)：計測対象クラスに含まれるメソッド間でのインスタンス変

*Shinji KUSUMOTO
1965年9月24日生
1990年大阪大学大学院基礎工学研究科博士前期課程修了
現在，大阪大学大学院・基礎工学研究科・情報数理系，講師，博士(工学)，情報工学
TEL　06−6850−6573
FAX　06−6850−6574
E-Mail kusumoto@ics.es.osaka-u.ac.jp

表1 チェックポイントと適用可能なメトリクス

チェックポイント	増加する情報	適用可能なメトリクス
(CP1)	クラス間関係，クラスの属性	NIV, CBON
(CP2)	クラスの継承構造，メソッド，再利用されるライブラリ	NIV, CBON, CBOR, CBO, NIM, DIT, NOC
(CP3)	メソッドのアルゴリズム	NIV, CBON, CBOR, CBO, NIM, DIT, NOC, RFC, LCOM
(CP4)	ソースコード	NIV, CBON, CBOR, CBO, NIM, DIT, NOC, RFC, LCOM, LOC

数の共有の度合い．

(7) NIV(インスタンス変数の数)：計測対象クラスのインスタンス変数の数．

(8) LOC(クラスの行数)：

2.3 チェックポイントと適用可能メトリクス

オブジェクト指向開発における，分析/設計/実装フェーズに以下の4つのチェックポイントを設ける．(CP1)クラス間の参照関係とクラスの属性が決定されている．(CP2)クラスの導出関係とクラスのメソッドが決定されている．(CP3)各メソッドのアルゴリズムとメソッド間の呼び出し関係が決定されている．(CP4)ソースコードが実装されている．

各チェックポイントで適用可能なメトリクスを表1にまとめる．

2.4 多変量ロジスティック回帰分析

メトリクスのfault-prone予測性能(対象のクラスにフォールトが発生するかを予測する性能)を評価するためには，まず，クラスのメトリクス値を入力とし，真偽値(予測)を出力する関数を作る必要がある．ここでは多変量ロジスティック回帰モデルを用いる[5]．このモデルでは以下の関係式を用いる．

$$P(X_1, \ldots, X_n)=1/(1+\exp(-(C_0+C_1 \cdot X_1+\ldots+C_n \cdot X_n)))$$

ここで，Pは与えられたクラスにフォールトが見つかる確率であり，X_jはクラスのメトリクス値である．もし与えられたメトリクス値がPを0.5以上にするなら，クラスはフォールトを持つ(fault-proneである)と予測する．

係数C_iを決定する際には，最尤度(maximum-likelihood)基準が用いられる．すなわち，係数は観測された結果をもっともよく反映するような値が選ばれる．

2.5 実験的評価

ある企業の新人研修で行われたC++プログラム開発演習から141個のクラス，80個のフォールトに関するデータを収集し，提案した手法を適用した．

表2はチェックポイントCP1で収集されたデータを多変量ロジスティック回帰分析することで得られた予測結果を表している．例えば，112個のクラスがフォールトを持たないと予測され，実際にフォールトが発見されなかった，ということが読み取れる．同様のことを他のチェックポイントでも行った．

ここで，予測式の精度を評価するため，3つの指標を導入する．(1)正確性：正しくフォールトがあると予測されたクラスの割合(%)．(2)完全性：フォールトがあるクラスが検出された割合(%)．(3)フォールト数に関する完全性：フォールトがあると予測されたクラスで実際に検出されたフォールトの割合(%)．

これらの基準は以下のように定義される．

$$\text{正確性}=C_{PFAF}/(C_{PFAF}+C_{PFAN})$$
$$\text{完全性}=C_{PFAF}/(C_{PFAF}+C_{PNAF})$$
$$\text{フォールト数に対する完全性}=E_{PFAF}/(E_{PFAF}+E_{PNAF})$$

ここで，C_{PFAF}はフォールトがあると予測され実際にフォールトがあったクラスの数，C_{PFAN}はフォールトがあると予測されたが実際にはフォールトがなかったクラスの数，C_{PNAF}はフォールトがないと予測されたが実際にはフォールトがあったクラスの数，E_iは対応するC_iのクラスで発見されたフォールトの数である．

チェックポイントCP1からCP4でのfault-prone予測精度を表3に示す．後期のチェックポイントほど，より正しく予測を行える．CP4は開発プロセスの最終フェーズであり，本実験における予測精度の上限である．

CP1においては，完全性は低く(33%)，正確性は高い(82%)．よって，フォールトが発生しそうなクラスを重点的にレビュー，テストするクラスの候補と指定できる．CP2では，C&Kメトリクスのメソッ

表2 CP1におけるフォールト予測

予測		フォールト無	フォールト有
実測	フォールト無	112	2
	フォールト有	18(43)	9(37)

括弧の外の数字はクラスの数．括弧内の数字はクラスで発見されたフォールトの数

表3 フォールト予測の精度

チェックポイント	CP1	CP2	CP3	CP4
正確性(%)	82	76	86	86
完全性(%)	33	59	67	70
フォールト数に関する完全性(%)	46	75	78	83

ドのアルゴリズムに関するものは用いられていないにも関わらず，予測精度の上限と比較して，かなりよい予測精度となっている．この結果は，設計フェーズにおいて，アルゴリズムが決定していない段階で，設計仕様書からフォールトの発生を予測する可能性を示唆している．

CP3での予測はCP2での予測に比べて，予測精度がそれほど向上していない．我々は，CP3における予測精度は，より詳細なレベルでメソッドの複雑さを計測できるメトリクスを用いることで改善できると考えている．

3．ま と め

本稿では，オブジェクト指向複雑度メトリクスを用いて，設計フェーズの早期にフォールトの発生を予測する方法を紹介した．今後の課題として，今回利用したメトリクス以外のメトリクスを用いることによる提案手法の拡張や設計仕様書に対するメトリクス計測ツールの開発が考えられる．

参 考 文 献

1) S. R. Chidamber and C. F. Kemerer. "A. metrics suite for object-oriented design", IEEE Trans. on Software Eng., Vol.20, No.6, pp.476-493 (1994).
2) T. Kamiya, S. Kusumoto, K. Inoue and Y. Mohri : "Empirical evaluation of reuse sensitiveness of complexity metrics", Information and Software Technology, 41, pp.297-305 (1999).
3) T. Kamiya, S. Kusumoto, K. Inoue : "Prediction of fault-proness at early phase in object-oriented development", The Second IEEE International Symposium on Object-Oriented Real-time Distributed Computing, pp.253-258 (1999).
4) J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy and W. Lorensen : Object-Oriented Modeling and Design, Prentice Hall (1991).
5) SPSS Base 8.0 Applications Guide/Professional Statistics, SPSS (1998).