OCLとの比較を用いた形式仕様記述言語JMLの品質評価メトリクスの提案

吉岡一樹[†1] 岡野浩三[†1] 楠本真二[†1]

Design by Contract の考え方に基づき，Java に対して契約を付加する JML という言語がある．本研究では，UML に対する注釈を行う言語 OCL との比較を用いた，JML によって記述された契約に対する品質評価メトリクスを提案する．

A Proposal of a Metrics to evaluate a Quality of JML which is a Formal Specification Language by Comparision with OCL

Kazuki Yoshioka,[†1] Kozo Okano[†1] and Shinji Kusumoto [†1]

JML is defined as a language to annotate contracts to Java based on the notion of Design by Contract. In this paper, we propose a metrics to evaluate the quality of contract described by JML by comparing with OCL that annotates UML.

1. はじめに

Design by Contract（以下 DbC）[1] はソースコードや設計書に対して満たすべき契約を記述する手法である．契約とはメソッドやクラスが満たすべき条件のことで，設計書や実装に仕様を記述して設計の安全性を高める手法のことである．設計書レベルでの契約として，UML 記述に対して契約を記述する OCL[3]，実装レベルでは Java に対して契約を記述する JML (Java Modeling Language)[2] などの言語が存在する．

2. 背景

2.1 OCL

OCL(Object Constraint Language)[3] は UML モデルに対し，さらに詳細に性質記述を行うために設計された言語である，OCL を用いることにより，実装時にモデルがどのように開発されるべきかという詳細な情報を記述することができる．これにより，メソッドの事前条件，事後条件，クラスの不変条件などを記述できる．

2.2 JML

JML(Java Modeling Language)[2] とは，Java で記述されたソースコードに対して契約を付加することのできる言語で，コメントの形で書かれるためソースコード本体へは影響を及ぼさないという利点がある．OCL と同様に，契約として事前条件，事後条件，不変条件などを記述できる．JML にはランタイムアサーションチェッカや，実装の正しさをメソッド単位で検証できる ESC/Java2[4] など，多くのツールがサポートされている．

2.3 変数カバレッジ

武藤ら[5] により JML の品質評価メトリクスである変数カバレッジが提案されている．このメトリクスは実装中に出現する変数のうち，JML で制約が記述されている変数の割合を計測したもので，チェックすべき項目である変数に十分な形式仕様記述があるかを測っている．

このメトリクスにより，どのメソッドやクラスにおいて十分な仕様が記述されていないかが把握可能になる．

3. 提案手法

既存手法である変数カバレッジでは，実装との比較によって形式仕様記述の品質を評価している．

しかし，形式仕様記述は通常実装よりも早い段階で記述されるものである．そのため，実装が進まなければ品質を評価することができず，形式仕様記述を書き上げた時点では品質を測ることはできない．つまり，形式仕様記述に問題があったとしても，既存手法では実装が行われなければその問題を把握できず，形式仕様記述を書いているときにその品質を確かめることが

†1 大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University

できない．

そこで我々は上位仕様である OCL との比較をすることで JML の品質を測る手法を提案する．OCL は通常 JML よりも先に記述されるため，本手法により JML の品質を常に確認しながら記述していくことができる．

ただし，JML にしかない記述を用いている，OCL の記述を見た上で JML にどう記述するかは開発者によって異なるなど，単純に比較することでは形式仕様記述の品質は測れない．

この問題に対応するため我々は複数の観点から品質を評価する．

3.1 文

それぞれのメソッドの事前条件，事後条件，および属性の不変条件について，契約として記述されている文の数を比較する．品質を測る式として以下の式を提案する．

$$\frac{Exp_{OCL}}{Exp_{JML}} \tag{1}$$

Exp_{OCL} は OCL における式の出現回数，Exp_{JML} は JML における式の出現回数である．JML に十分な数の式が記述されていれば高い値を，少ない数しか記述されていなければ低い値を示すとかんがえられる．

3.2 変　　数

既存研究では変数が出現したかどうかに着目していたが，本研究では変数の出現回数の比較によって品質を計測する．OCL との比較において，変数が出現したかどうかだけを計測するのでは複数の式に同じ変数が出現した時に適切な結果にならない．そのため，変数の出現回数を比較することで，同等の量の形式仕様が記述されているかを確認する．すなわち，OCL における変数の出現回数を Val_{OCL}，JML における変数の出現回数を Val_{JML} とするとき，品質は

$$\frac{Val_{OCL}}{Val_{JML}} \tag{2}$$

で表せる．この値が高いほど JML の品質が高く，低いほど品質が低いと考えられる．図 1，2 にそれぞれ OCL，JML の例を示す．実際に変数 val の記述について，まず事前条件について考える．上記の品質を示す式の値は 1 となり，十分に高いことがわかる．しかし，事後条件においては，上記の式の値は 0.5 となり JML 記述は充分でないことがわかる．

```
context Account::withdraw(val:Integer)
    : Integer
pre : balance - val > 0
pre : val > 0
post: \result = balance@pre - val
post: balance = balance@pre - val
```

図 1　OCL による制約の記述例
Fig. 1　a example of contract described by OCL

```
public class Account{
   int balance;
   /*@
       requires balance - val > 0;
       requires val > 0;
       ensures \result == \old(balance) - val;
   @*/
   public int withdraw(int val){
      balance -= val;
      return balance;
   }
}
```

図 2　JML による制約の記述例
Fig. 2　a example of contract described by JML

4. 今後の予定

現在提案しているメトリクスでは，比較している式が意味的に同じであるかを計測できない．そのため，今後は出現する変数や式が同じ意味を持つかどうかで比較したい．

5. ま と め

OCL との比較を用いることで JML の品質を計測する手法を提案した．今後は本手法を実装し，実際の JML 記述の付加されたソフトウェアに対して適用したい．

参 考 文 献

1) Meyer, B.: *Object-oriented software construction (2nd ed.)*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1997).
2) The Java Modeling Language, `http://www.eecs.ucf.edu/\~leavens/JML/`.
3) Group, O. M.: Ocl 2.0 specification, 2006, `http://www.omg.org/cgibin/apps/doc?formal/06-05-01.pdf`.
4) ESC/Java2, `http://secure.ucd.ie/products/opensource/ESCJava2/`.
5) Muto, Y.: Variable Coverage: A Metric to Evaluate the Exhaustiveness for Program Specifications Based on DbC, Master thesis, Osaka University, 2012.