オブジェクト指向ソフトウェアメトリクスの一考察

2C－3

白壁広光　山田宏之

愛媛大学 工学部

1. はじめに

オブジェクト指向技術の普及に伴いオブジェクト指向ソフトウェアの定量化に関する研究が多くなされている[1]-[3]．本研究では，オブジェクト指向ソフトウェアの変更時に，その指標となるメトリクスについて考察し，具体的なソフトウェアに対して測定を行い，その結果を示す．

2. ソフトウェアの変更

オブジェクト指向ソフトウェアは，データ抽象化，多相性，継承といった特徴により，変更に強いといわれてきた．これはデータ抽象化によりインターフェースが変更されなければ，他のオブジェクトに影響を与えないため，多相性等により現存のコードを変更せずに，新しいデータに対する処理を容易に追加できるためである．しかしながら，同一オブジェクト内で考えた場合，ある部分の僅かな変更が他の部分に大きな影響を与えることは十分に有り得る．また，オーバーライディングにより，あるメソッドの変更がサブクラスに問題を引き起こすことは多々考えられる．このようにオブジェクト指向ソフトウェアにおいても変更は容易にできない．

そこで，ソフトウェアの変更時に特にメソッドに注目し，変更の容易さや変更にかかるコストを知るための手がかりとなるメトリクスとして，メソッドの複雑度と束縛度を提案する．

・メソッドの複雑度
　メソッドの複雑さを示す．複雑さの評価方法の研究はいくつかなされている[3]．ここでは，プログラムの変更という視点で，複雑さの評価を行う．

・メソッドの束縛度
　メソッドの変更時に，そのメソッドが呼び出すメッセージや，継承によるメソッドのオーバーライドなどにより，幾らかの束縛を受ける．そこで，あるメソッドとその他のメソッドとの関連について示す指標について考察する．

A Study on Object Oriented Software Metrics
Hiromitsu Shirakabe and Hiroyuki Yamada
Ehime University, Faculty of Engineering
3, Bunkyo-cho, Matsuyama, Ehime 790-8577 JAPAN

3. メソッドの定量化

複雑さ・束縛度を算出するために，次のような要素を設定した．

a:計算式（代入）
　代入文の数
b:ループ
　for，while 文等のループステートメントの数
c:分岐
　if，switch 文等の条件分岐ステートメントの数
d:一時変数
　メソッド内で定義される一時変数の数
e:パラメータ付きメッセージ（外）
　他のクラスで定義されているメソッドで仮引数を持つものの数
f:パラメータ付きメッセージ（内）
　同じクラス内で定義されているメソッドで仮引数を持つものの数
g:パラメータなしメッセージ（外）
　他のクラスで定義されているメソッドで仮引数を持たないものの数
h:パラメータなしメッセージ（内）
　同じクラス内で定義されているメソッドで仮引数を持たないものの数
i:オーバーライド（上）
　上位クラスで定義されたメソッドをオーバーライドしている場合，クラス階層上で最初にそのメソッドが定義されてから，自分のクラスに至るまでにオーバーライドされている回数
j:オーバーライド（下）
　あるクラスで定義またはオーバーライドされたメソッドが，そのすべての下位クラスにおいてオーバーライドされている数

4. 算出方法

3 章で述べた要素のうち，複雑度または束縛度に関連するものを取り挙げ，以下に示す重み付けを行い，その結果の総和によりそれぞれの値を算出する．

・メソッドの複雑度
　a:b:c:d:e:f:g:h ＝ 0.5:1:1:0.5:3:2:1.5:1
　という比率で各要素の重み付けを行う．

・内訳
　計算式（代入）と一時変数はループや分岐文に

比べると簡潔な構造のため複雑さは小さいと考えた．また，メッセージ呼び出しに関しては，プログラムの変更時に，そのメッセージの動作内容を理解する必要があると考え，複雑さは大きいとした．この時，同じクラス内で定義されているメソッドを参照する場合より他のクラスのメソッドを参照するほうが，また，引数を持たないメッセージより，持つメッセージのほうがそれぞれ複雑であるとした．

・メソッドの束縛度

z = e+f+g+h :メッセージの総数

z:i:j = 1:0.5:4

という比率で各要素の重み付けを行う．

・内訳

メッセージは全て，他のメソッドへの依存を表す．メソッド毎に束縛度を考えるため，ここでは他のメソッドへの依存の強さは等しいとし，メソッド内で呼び出されるすべてのメッセージに同じ重み付けをした．また，メソッドの変更時に，上位クラスのメソッドがオーバーライドされることにより強い影響受けることは少ないと考えられるため，束縛度は小さいとした．逆に下位クラスでそのメソッドがオーバーライドされる場合，下位のメソッドが強い束縛を受ける場合があるため，束縛度は大きいとした．

5. 測定結果

Java 言語で書かれたあるソフトウェアにおいて，変更前 a と，変更後 b のデータを測定した．

図 1 は縦軸に束縛度，横軸に複雑度としたときの a のメソッドの散布図である．右上の点ほど変更する場合には困難でコストのかかるメソッドであることを示す．

a と b の束縛度，複雑度を比較した場合，どちらも平均値は減少していることが分かる．また，メソッドのステートメント数の平均も減少しているため，一つ一つのメソッドの作業量は減少し，簡潔な構造となっていく傾向が伺える．

6. むすび

本稿ではソフトウェアの変更を支援するために，メソッドの複雑度，束縛度について考察を行った．今回行った重み付けのパラメータ比は一例である．今後いくつかのソフトウェアを測定しながら，適切なパラメータ比について検討していきたい．また，メソッドに関するこの 2 つの値をクラスおよびソフトウェア全体の評価へと反映させたい．

表 1 測定対象ソフトウェア

	a	b
クラス総数	90	147
メソッド総数	1012	1498

表 2 メソッドの複雑度

複雑度	a	b
平均	13.18	11.50
中央値	5	4.5
最頻値	1	0
最小	0	0
最大	155	156.5
合計	13337	17225

表 3 メソッドの束縛度

自由度	a	b
平均	6.22	5.45
中央値	3	2.5
最頻値	1	1
最小	0	0
最大	93	94
合計	6292	8166.5

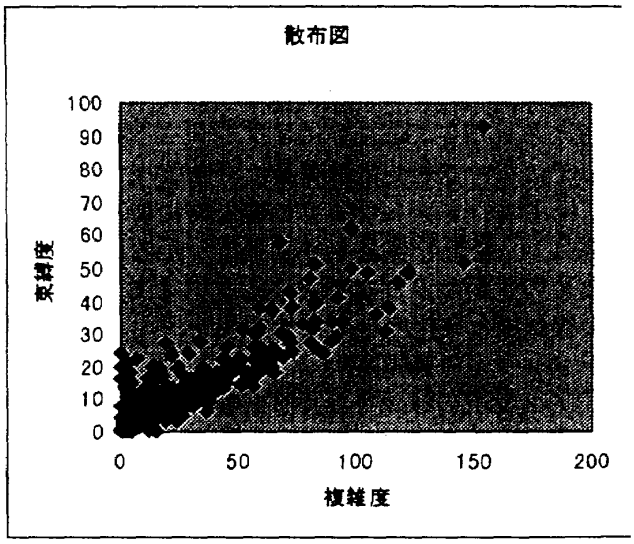

図 1 散布図

参考文献

[1] S. R Chidamber 他, "A Metrics Suite for Object Oriented Design," IEEE Trans. SE., 20, 6, pp. 476-493, 1994.

[2] マークス・ロレンツ 他, "オブジェクト指向ソフトウェアメトリクス," プレンティスホール, p. 163, 1995.

[3] 金恩美 他, "C++プログラムに対する複雑さメトリクスの提案と大学環境での実験的評価," 信学論, J79-DI, 10, pp. 729-737, 1996.