3M-5

テスト貢献度に基づくゲーミフィケーションを用いた教育用テストツールの提案

髙澤　亮平†　　坂本　一憲†　　鷲崎　弘宜†　　深澤　良彰†

早稲田大学 基幹理工学研究科 情報理工学専攻†

1. はじめに

カバレッジとは，製品コードがどの程度テストされているかを示すメトリクスである．著名なツールとしては，カバレッジ可視化ツールのCobertura[1]がある．Coberturaによる可視化の例を図 1 に示す．図 1 において，背景が赤色の行はテストされていない箇所であり，テスト済の箇所と明確に区別することが可能となる．

```
110 128          else if ( nav.isElement( first ) )
111              {
112 100              return nav.getElementQName( first );
113              }
114 28           else if ( nav.isAttribute( first ) )
115              {
116 0                return nav.getAttributeQName( first );
117              }
118 28           else if ( nav.isProcessingInstruction( first ) )
119              {
120 0                return nav.getProcessingInstructionTarget( first );
121              }
122 28           else if ( nav.isNamespace( first ) )
123              {
124 0                return nav.getNamespacePrefix( first );
125              }
126 28           else if ( nav.isDocument( first ) )
127              {
128 28                return "";
129              }
130 0              else if ( nav.isComment( first ) )
131              {
132 0                return "";
133              }
134 0              else if ( nav.isText( first ) )
135              {
136 0                return "";
137              }
```

図 1 Coberturaによる可視化の例

カバレッジを可視化することによるテスト支援ツールは数多く存在するものの，可視化によってテスター個人に対するフィードバックを行うものは少ない．

・誰がどの部分をテストしたのか
・自分はどれだけテストに貢献しているのか

上述の事柄に関して把握することは現状では困難であり，テスターのモチベーション低下，あるいはテストを放棄する一因になっているのではないかと考えられる．そこで，我々はゲーミフィケーションを利用し，テスターのモチベーション向上を図る．

ゲーミフィケーションとは，ゲームを構成する要素をゲーム以外のものに応用し，利用者のモチベーション向上を図る手法[2]であり，近年では様々な分野での利用が進んでいる．ソフトウェア開発への適用例は次章で述べる．

2. 関連研究

ソフトウェア開発にゲーミフィケーションを利用する研究として，バージョン管理システムである Git へのコミット回数を利用し，開発を促進させる研究[3]が存在する．また，ソフトウェアテストに対して利用した研究としては，機械的な作成が困難なテストケースの作成をパズル化し，人間が解くことでカバレッジの向上を図る Puzzle-Based Testing[4]が存在する．

開発支援ツールとしては，開発の内容に応じて，図 2 のようにリアルタイムで称号が得られる Visual Studio Achievements[5]が存在する．現状では，カバレッジをゲーミフィケーションに利用する研究は存在しなかった．

図 2 Visual Studio Achievements

3. 提案手法

我々は，テスト対象の行単位のカバレッジを測定し，その結果を基に各テスターの貢献度を算出する手法を提案する．本手法の全体の構成を図 3 に示す．本手法のプロセスは以下の通りである．1. 対象のカバレッジを測定，2. テスターの特定，3. 貢献度の数値化の順に行う．

3.1 カバレッジ測定

Open Code Coverage Framework[6]を用い，対象となる製品コードの行単位のカバレッジを測定する．結果として，各テストメソッドの名前と，どのファイルの何行目をテストしたのかを対応づけたものが得られる．

Suggestion of contribution-based gamified testing tool for education
† Ryohei Takasawa, Kazunori Sakamoto, Hironori Washizaki, Yoshiaki Fukazawa, Waseda University.

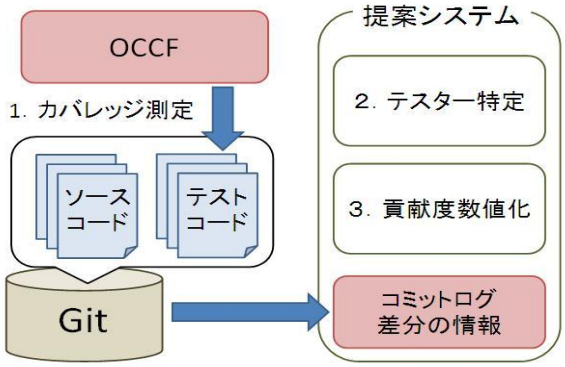

図 3 提案手法の構成

3.2 テスターの特定

各テストメソッドの記述者を特定することにより，製品コードの各行が誰によってテストされたかを特定することができる．本手法では，テストメソッドを記述し，リポジトリにコミットした人の名前をテスターとして判別する．利用する Git リポジトリの内容を JGit[7]を使って参照し，テストメソッドの名前とテスターを対応づける．

3.3 貢献度の数値化

3.1 と 3.2 の結果と組み合わせることで，テスターと製品コードのテスト箇所の対応を特定することが可能となる．これをもとに，各テスターの貢献度の数値化を行う．

貢献度を数値化する上で，製品コードに対する貢献が大きいと判断出来る場合を 2 つ定義した．

1 つ目は，テストケースを新たに作成した場合である．テストケースの数は多ければ多いほど良いという観点から，個人がテストしたのべ行数に比例して点数を与える．これによって，テスターによるテスト記述の促進に繋がると考えられる．

2 つ目は，単一のテスターのみがテストした行が存在する場合である．この場合，製品コード全体で見た時に価値が高いと考えられるため，ボーナス点を与える．また，このような状況が発生した場合，部分的に製品コードへの理解が共有出来ていないということであり，解消されることが望ましい．ボーナス点の存在により，他者による当該箇所へのテスト記述の促進にも繋がると考えられる．

3.4 ゲーム要素としての貢献度の利用

前述の通り，貢献度を数値化することで，テスト工程へのモチベーションの向上，あるいはテスターが行う行動を強化することに繋がると考えられる．

また，数値化によって特定の行動を推奨することができ，効率的なテスト記述を習得させる教育的な利用法も考えられる．

4. 適用例

手法の適用例として，統合開発環境での利用イメージを図 4 に示す．テスト記述と平行して貢献度を更新していくことで，テスターへのフィードバックと支援を図る．

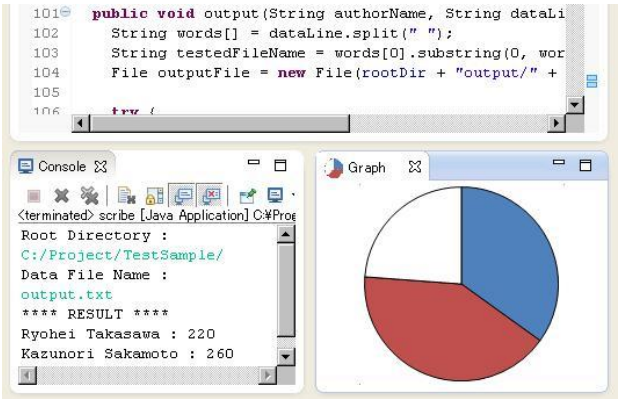

図 4 統合開発環境における利用イメージ

5. まとめと展望

我々は，テスターがテストしたコードの行数をもとに，貢献度の算出を行った．また，算出した貢献度の数値をゲーミフィケーションに利用し，テスト記述を促進させる手法を提案した．

今後の展望として，提案手法をより実用的なツールにすることや，貢献度の数値化をより適切なものに改良することを行いたい．また，被験者実験を行い，モチベーションに与える影響についても深く考察したい．

参考文献

[1] Mark Doliner, at el, Cobertura, http://cobertura.sourceforge.net/
[2] Sebastian Deterding, at el, “From game design elements to gamefulness: defining ″gamification″,” (MindTrek 2010), pp.9-15.
[3]Leif Singer and Kurt Schneider, “It was a Bit of a Race: Gamification of Version Control,” (GAS 2012), pp.5-8.
[4] Ning Chen and Sunghun Kim, “Puzzle-based automatic testing: bringing humans into the loop by solving puzzles,” (ASE 2012), pp.140-149
[5] Microsoft, Visual Studio Achievements, https://channel9.msdn.com/achievements/visualstudio
[6] Kazunori Sakamoto, at el. Open Code Coverage Framework: A framework for consistent, flexible and complete measurement of test coverage supporting multiple programming languages.” IEICE Transactions, vol.94-D, no.12, pp.2418-2430, 2011.
[7] Chris Aniszczyk, at el, JGit, http://www.jgit.org/