修 士 論 文 概 要 書

2010 年　02 月提出

専攻名 （専門分野）	情報理工学専攻	氏 名	坂本　一憲	指　導 教　員	鷲崎 弘宜 准教授 印
研究指導名	高信頼ソフトウェア工学研究	学籍番号	CD 5109B039-2		
研　究 題　目	柔軟かつ複数プログラミング言語対応のテストカバレッジ測定環境とテストの品質向上に関する研究				

1. はじめに

テストカバレッジ(以下，カバレッジ)とは，プログラムのソースコードがテストされた割合を示す．

カバレッジには，ステートメントを対象とする命令網羅(C0)，条件分岐を対象とする分岐網羅(C1)など様々な種類があり，テストの目的と用途に応じて適切に選択する．

近年，プログラミング言語の多様化，多言語のソフトウェア開発の普及に伴い，既存のカバレッジ測定ツール(以下，ツール)では様々な問題が生じている．

また，テストコードの実装手法の発展に伴い，テストコードが複雑化している．テストコードもソースコードの一部であるため，製品同様に高い品質が求められている．

本研究では，複数のプログラミング言語に対応したカバレッジ測定フレームワーク，題してOpenCodeCoverage Framework(OCCF)を提案する．OCCFは対応言語と測定基準の追加，統一的で柔軟な測定，テストコードの再構成を支援するカバレッジ測定環境を提供する．

図1で，OCCFが実現する簡略化のコンセプトを示す．既存のツールでは，言語とカバレッジが多対多の対応関係になっているが，OCCFでは，OCCFと言語，OCCFとカバレッジを一対多の対応関係で結ぶ．これによって，共通化を促し実装コストの低減を実現する．

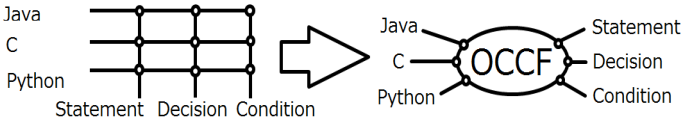

図1. OCCFのコンセプト

2. 既存のカバレッジ測定ツールの問題点

2.1. 高い開発コストと保守コスト

既存ツールが対応する言語は限られており，レガシーな言語のサポートが不足している．その上，既存ツールのモジュール化が不十分で，新たに言語の対応を増やすために必要なコストが莫大である．また，Java言語の1.4から5.0，Python言語の2から3へのアップグレードなど，言語仕様の変化に伴う修正も同様に莫大なコストがかかる．

2.2. 統一性を欠いた測定

既存ツールは，各言語に特化したものが多く，複数言語によるソフトウェア開発では利用が難しい．例えば，Java言語とPython言語を用いてサーバークライアントモデルのソフトウェアを開発する場合，Java用のCobertura [1]とPython用のStatement Coverage for Python(SCP) [2]を組み合わせる必要がある．しかし，後者は命令網羅のみで，結合テストにおいて分岐網羅を測定できない．

2.3. 柔軟性を欠いた測定

既存ツールは，カバレッジの測定対象や範囲を指定した柔軟な測定ができない．例えば，特定のライブラリ呼び出しのみをテストする場合，その部分のみのカバレッジを算出できない．また，測定基準を容易に追加できないため，プロジェクトに合わせた柔軟な測定が困難である．

2.4. 不完全な測定

既存ツールは，測定対象に漏れがある場合がある．

例えば，Coberturaでは，実行可能なバイナリを参照して測定するため，コンパイラによって削除されたデッドコードを測定対象に含むことができない．

3. OCCFの概要

OCCFでは，まず，抽象構文木を介して，ソースコードにカバレッジ測定用コードを埋め込む．次に，得られたコードを任意の処理系で実行する．最後に，実行時に得られた情報をもとに，測定結果を表示する．

測定用コードは，副作用がなくソースコードの意味を変化させない．測定用コードは，測定対象が実行されるたびに，その情報をファイルや共有メモリに出力する．その情報からカバレッジを解析して出力する．

3.1. OCCFの構成

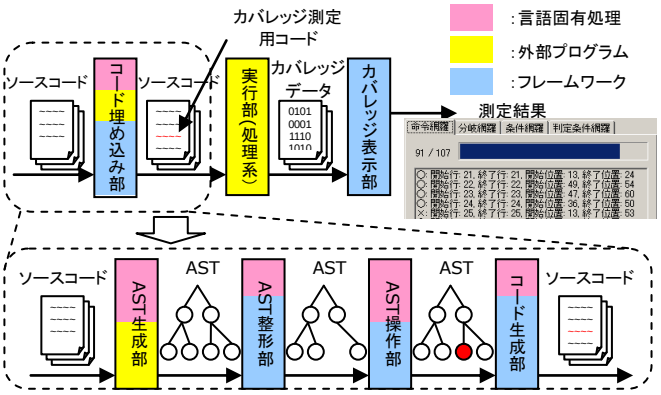

図 2. OCCF の構成

3.2. コード埋め込み部（四つの構成要素）

AST生成部：ソースコードを入力してASTを生成する．この処理は，各言語に強く依存するため，言語に特化した実装が必要である．しかし，構文解析器を自動生成するコンパイラコンパイラや，ライブラリを利用することで，開発コストを低減できる．

AST整形部：冗長要素の削除や，補助要素の追加などを行う．例えば，省略形のif文に対して測定用コードを正常に埋め込めるように変形する．

AST 操作部：AST 上で測定用コードの埋め込み先ノードの列挙，測定用コードに対応するノードの生成，ノードの挿入や置換処理を行う．例えば，図 3 のように，命令網羅の測定で，各ステートメントの要素を列挙して，その直前に測定用コードの要素を挿入する．

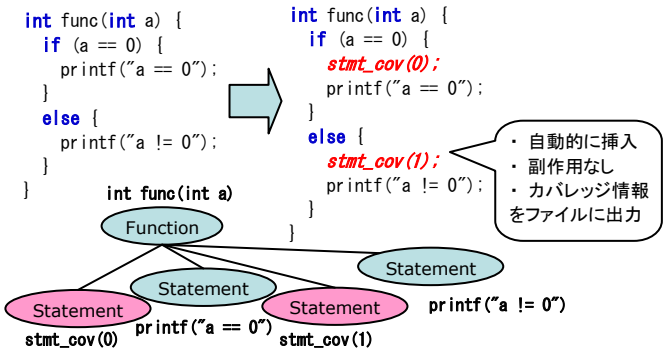

図3. 測定用コードの埋め込み例

AST操作部はデザインパターンであるCommandパターンによって設計されているため，例えば，列挙操作を組み合わせることで，複雑な列挙操作を簡単に実現できる．

コード生成部：ASTを入力してソースコードを生成する．AST生成部で各ノードにソースコードのトークンを記憶させておくことで，生成部の処理を大幅に簡略化できる．OCCFは大部分の処理を共通処理として提供するため，ユーザーは特定の非終端記号に対する処理のみ実装すればよい．

3.3. コード実行部

埋め込み済みコードを実行して，測定に必要な情報を収集する．ユーザーは任意の処理系を用いて，得られたソースコードでテストを行えば良い．

3.4. 測定結果表示部

収集した情報から測定結果の表示を行う．OCCFは，測定結果表示部の完全な実装を提供する．ユーザーが独自に改良することも可能である．

4. OCCFの実装

OCCFは.NET Framework 3.5上で動作し，Managed Extensibility FrameworkとDynamic Language Runtimeを搭載する．そのため，.NET言語で実装されたアセンブリか対応するスクリプト言語を配置することで，言語固有の処理の追加や機能拡張が容易に可能である．

本研究では，OCCFを用いて，Java，Python，Cの3言語に，命令網羅，分岐網羅，条件網羅，分岐／条件網羅の4種類のカバレッジで対応するツールを実装した．さらに，C#, JavaScript, Ruby, Lua言語用のASTとコード生成部を実装し，同様に測定できることを確認した．

5. 評価

OCCFを利用した実装例と，既存ツールで比較する．

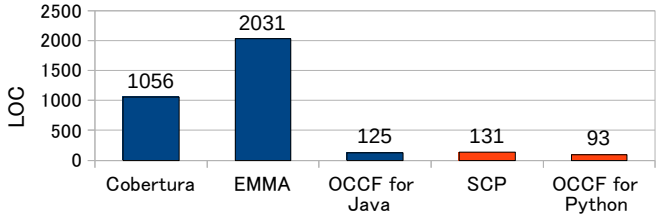

図4. 埋め込み処理の言語固有な実装のコード行数

表 1. 新カバレッジの実装実験

実装内容(使用したツール)	達成人数	平均時間
新カバレッジ追加(OCCF)	4 人	13.5 分
Python3 への対応(OCCF)	4 人	47.5 分
新カバレッジ追加(SCP)	0 人	－
Python3 への対応(SCP)	0 人	－

図 4 のコード行数では，Java への対応において，OCCF の方が 90%程度行数を削減している．さらに，Python への対応でも，実装例の方が若干優れている．

表 1 では，Python2 の print 文のみを対象とした特殊なカバレッジの実装と，Python3 への対応修正を学生の被験者 4 人に対して実験した結果を示す．OCCF の場合は 2 時間以下で実装できたが，SCP のコードを修正した場合は 4 時間以上かけても出来なかった．

6. テストコードの再構成

OCCFの発展研究として，テストコードの再構成への応用を提案する．

6.1. 既存のテストコード実装手法の問題点

テスト駆動開発といったアジャイルソフトウェア開発における手法では，品質保証のためのテストのみならず，ソフトウェアの実装のためにテストコードが記述されることが多い．そういった手法によって記述されたテストコードの量は多く，品質保証のために記述したテストコードと合わせると，テストの保守や実施が困難になる．

6.2. テストコードの再構成

前述の問題点を解決するため，テストカバレッジに基づくテストコードの再構成パターンを提案する．これは，重複したテストコードをカバレッジに基づいて機械的に検出して削除するパターンである．

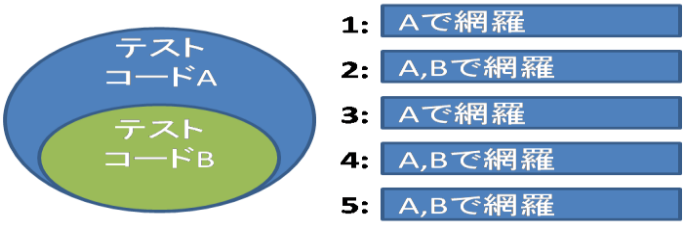

図5. テストコードの包含関係

OCCF を用いることで，テストコードの包含関係が得られる．図 5 が包含関係を示しており，横棒がカバレッジの測定対象であるコードの実行単位である．テストコード A にて 1～5 行の全個所が，B にて 2,4,5 行目の三か所が実行される．このように，テストコード A と，テストコード A と B の両方にて得られたカバレッジ結果が等しい場合，テストコード A は B を包含していると呼ぶ．

7. 関連研究

桐井ら[3]は，ソースコードに測定用コードを埋め込むことで，測定ツールを実装する手法を提案している．この手法では，4種類の言語に対応した例が挙げられているが，本研究とは異なり，新たに対応言語を増やす際の支援や，柔軟な測定はできない．

Hridesh Rajanら[4]は，アスペクト指向プログラミング言語を応用して，測定対象を柔軟に指定する手法を提案している．この手法では，1種類の言語に特化した例が挙げられているが，本研究とは異なり，複数言語に対応する統一的な測定はできない．

8. おわりに

本研究では，多言語対応のテストカバレッジ測定フレームワークOCCFを提案した．複数のプログラミング言語への統一的な対応，共通処理の再利用による開発コストの低減，機能拡張の支援による柔軟な測定，ソースコードへの埋め込みによる完全な測定を実現する．さらに，発展研究としてカバレッジに基づくテストコードの再構成を提案して，OCCFの有用性を比較実験から示した．

今後の課題としては，テストコードの再構成を支援するツールの実装と評価が挙げられる．

参考文献

[1] Cobertura, http://cobertura.sourceforge.net/
[2] Statement Coverage for Python, http://garethrees.org/2001/12/04/python-coverage/
[3] 桐井隆志，三好辰弥，岸上諭，大里立夫，曽根原勝，"ソースコード埋め込み型カバレッジツールについて"，情報処理学会第65回全国大会，2003.
[4] Hridesh Rajan and Kevin Sullivan, "Aspect Language Features for Concern Coverage Profiling", International Conference on Aspect-Oriented Software Development, 2005.

業績

第 8 回情報科学技術フォーラム（FIT2009）船井ベストペーパー賞, pp.103-112, 2009. 9.
3rd International Workshop on Software Patterns and Quality (SPAQu'09), pp.26-30, 2009. 10.