3L-5

DAO パターンを用いる Web アプリケーションのための単体テストケース補完ツール

鎌田 高如† 樋口 昌宏‡
近畿大学 大学院総合理工学研究科

1 はじめに

近年では，業務処理システムの多くが Web アプリケーションとして開発されている．また，ソフトウェアの品質を保証する上で，単体テストの重要性が増してきている．単体テストのテストスイートに求められる要件として，試験対象のモジュールの機能を確認すると共に再利用性，保守性の観点からプログラムコードの高網羅率が挙げられる．V 字モデルでは，詳細設計工程で作成したモジュールの仕様書を基にテストケースを抽出し，それらをまとめたテストスイートを作成する．更に網羅率を高めるためにプログラムコードを分析し，手作業によりテストケースを追加する．そのような手作業が煩雑であるため，単体テスト自体が軽視されがちなのが現状である．このため，単体テストのテストスイート生成の自動化に関する研究が行われている [1] [2]．本研究では，Web アプリケーションにおける，データベースアクセスモジュールに特化した高網羅率の単体テスト自動化について検討する．

2 Web アプリケーションの単体テスト

2.1 Web アプリケーションについて

図 1 に，MVC(Model,View,Controller) モデルと呼ばれる典型的な Web アプリケーションの構成を示す．まず，クライアントからの http Request を Controller 層が受け取る．Controller 層は，http Request に基づいて Model 層に処理内容を伝える．Model 層は，必要に応じてデータベースアクセスを行い，処理完了を Controller 層に伝える．その後，Controller 層は View 層を起動する．View 層は，Model 層の処理結果を参照し，クライアントに http Response を返す．

2.2 データベースアクセスについて

図 2 に Model 層の構成図を示す．Model 層は複数の Service クラスと DAO クラス，VO クラスから構成される．Service クラスは，Model 層の主体となるクラスで，DAO クラスとやり取りをしながら処理を進める．DAO クラスは，データベースに接続し，データベースへの追加，変更，参照，削除を行う． データベースアクセスに際しては，データベースのテーブルの属性に対応した Getter/Setter を持つ VO クラスを DAO クラスが利用する．

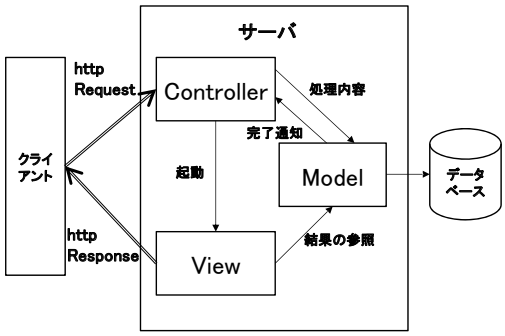

図 1 Web アプリケーション

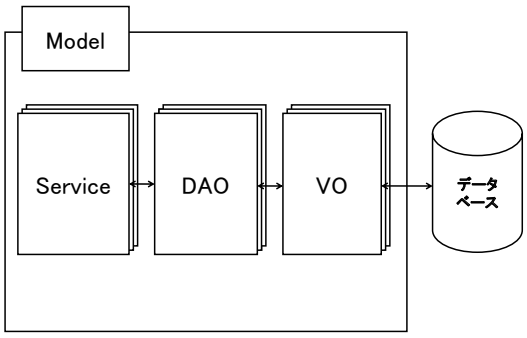

図 2 Model 層の構成

2.3 単体テストと網羅率

単体テストは，ソフトウェアのモジュール単位に個々の動作を確認するテストである．機能確認テストに加え，網羅率が重要となる．高網羅率を実現することが，再利用性，保守性の向上に繋がる．テストで実行した文の割合を文網羅率，実行した分岐の割合を分岐網羅率，条件判定に用いられる論理式の中で，真偽双方を持つ割合を条件網羅率と言う．

本研究では，文網羅率の向上を扱う．

2.4 テストファースト

従来の開発手法では，実装後にテストスイートを作成し，単体テストを行っている．これに対し，テストファースト開発では，詳細設計後，実装前にテストスイートを作成し，テストを実行しながら実装を進める．単体テストに合格するまで繰り返しながら実装の見直しを行っていく開発手法である．

2.5 dbUnit

本研究は, プログラミング言語は Java, データベースは MySQL，統合開発環境の eclipse に dbUnit をプラグインして使用することを前提とする．データベースアクセスのテストで，テスト結果の確認を目視で行うには，データ項目数やテーブル数が多数になると限界があり，確認ミスの可能性が高くなる．dbUnit は，データベースの各テーブルに初期値として用意する属性値を.xml ファイル形式によりテスト

A Unit Test-case Supplement Tool for Web-applications Using the DAO Pattern

† Takayuki Kamada: Interdisciplinary Graduate School of Science and Engineering, Kinki University
‡ Masahiro Higuchi: Interdisciplinary Graduate School of Science and Engineering, Kinki University

Copyright ©2012 Information Processing Society of Japan.
All Rights Reserved.

プログラム内で設定できる．テスト実行結果の確認は，期待する出力値を.xml ファイル形式で準備し，テスト実行後のテーブルの属性値と比較する．

3 DAO クラスのための単体テスト自動生成

仕様書よりテストファーストで作成したテストスイートで網羅できなかった未実行のプログラムコードである未網羅文を補完するためのテストケースを自動生成する．自動生成するテストケースは，テスト対象メソッドの引数と期待値である．

本研究では，DAO パターンによるデータベースアクセスに特化した未網羅文を網羅するためのテストケース補完ツールを考案する．

3.1 テストケース補完ツールの概要

図 3 にシステム構成を示す．テスト対象メソッドとテストスイートをインタプリタで実行し，実行レポートを出力する．実行レポートより未網羅パス(連続する未網羅文の列) を抽出し，未網羅パスとテストデータをテストケース自動生成器に通し，新たなテストケースをテストスイートに追加する．以上のサイクルを十分な網羅率が達成されるまで繰り返す．

3.2 例外処理のテストケース自動生成

DAO クラス内の各メソッドには，try_catch 文が存在し，catch ブロックを実行させる必要がある．未網羅パスの解析より catch ブロックが未網羅であると判明した時，これを網羅するテストケースを生成する．前提条件として，データベースドライバ名とその接続 URL を DAO クラス内のコンストラクタの引数に与えられているものとすると，CLASSNOTFOUNDEXCEPTION を発生させるためにはデータベースドライバ名を空列にし，SQLEXCEPTION を発生させるためには接続 URL を空列にすればよい．

3.3 構造的テストの自動生成

仕様書よりテストファーストで作成したテストスイートでの未網羅文を，プログラムコードからテストケースを作成し，網羅率向上を計る．

3.3.1 SELECT 文を含むテストケース

未網羅パス中の条件式に，データベースの属性値を取得するために用いる getInt() や getString() メソッドが含まれているかを確認する．含まれていれば，対応する SELECT 文を特定し，網羅するための値を.xml ファイルを参照して決定する．

3.3.2 INSERT 文を含むテストケース

未網羅パス中に INSERT 文が存在する時，SQL 文の VALUES 句で指定する変数の値がテーブルの属性値になる．そこで，対象の SQL 文より属性値を挿入するテーブル名を解析し，変数を基に条件式を遡りテスト対象メソッド実行後の期待値データを作成する．

4 実験

酒屋問題を例題として，Web アプリケーションを開発し，実験を行った．今回開発した Web アプリケーションは，Controller 層の Servlet 数は 6，

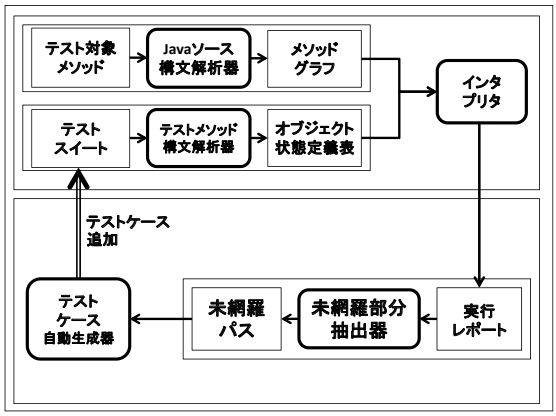

図 3　システム構成

View 層の jsp 数は 9，Model 層の Service クラス数が 5，DAO クラス数が 6，VO クラス数が 6 である．Service クラスは平均 50 行，DAO クラスは平均 200 行程度のプログラムである．

4.1 実験の概要

上記中の，DAO クラスの例外処理，SELECT 文や INSERT 文を含むメソッドに対し，提案手法のテストケース自動生成器を実装し，テストケースをテストスイートに追加する実験を行った．

4.2 実験結果

実験結果を表 1 に示す．例えば，makeInstruction() は仕様書よりテストファーストで作成したテストスイートでの文網羅率は 64% であった．それに対し，テストケース自動生成器により例外処理を追加することで 80% になった．更に，構造的テストのテストケースを追加し，100% にすることができた．

表 1　文網羅率測定結果

メソッド名	テスト ファースト	例外処 理追加	構造的 テスト追加
makeInstruction()	64%	80%	100%
checkCargo()	68%	88%	100%
registerCargo()	65%	80%	100%
registerInst()	69%	81%	100%
registerShort()	61%	77%	100%

5 結論

DAO パターンのデータベースアクセスモジュールに対して，仕様書から作成したテストスイートでの未網羅文を網羅するために，未網羅パスよりテスト対象メソッド内で利用されている例外処理および，SELECT 文や INSERT 文を含む SQL 文を特定することで，構造的テストのテストケースを自動生成する手法を提案した．

また，実験により網羅率の向上が確認でき，本研究で提案する手法の有用性を示すことができた．

参考文献

[1] Patrice Godefroid, et al., “ Automating Software Testing Using Program Analysis ”, IEEE Software, Vol.25, No.5, pp.30-37 (2008).

[2] 式見遼，小形真平，松浦佐江子，“ UML 要求仕様からのカバレッジに基づく機能テストのテストケース生成 ”，情報処理学会第 73 回全国大会，(2011).

Copyright ©2012 Information Processing Society of Japan.
All Rights Reserved.