Vol.2013-SE-179 No.13
2013/3/11

設計文書とテストケースとを関連付けたテストケース管理手法に関する研究

西上明普[†1]　古川善吾[†2]　高木智彦[†3]　八重樫理人[†4]

近年，ソフトウェアは大規模化・複雑化し，利用方法が多様化する中で，仕様変更が頻繁に起こるようになった．設計の変更に伴うテストケースの修正が必要であり，人手での修正にはコストがかかる．本研究では，仕様変更で生じる設計変更情報を用いて，再利用可能なテストケースを抽出する手法を提案する．また，提案手法を用いてテストケース管理システムを試作し，開発プロジェクト例と利用者を想定したツールの使用評価を行う．

Study on test management method associated with the design documents and test cases

AKIHIRO NISHIGAMI [†1]　ZENGO FURUKAWA [†2]　TOMOHIKO TAKAGI [†3]
RIHITO YAEGASHI[†4]

In recent years, diverse usage of software becomes scale and complexity, and specification changes occur frequently.Design changes is a need to modify the test case, it is expensive to fix manually.We propose a method using information of design changes caused by changes in specifications, to extract the reusable test cases.Also, a prototype test case management system using the proposed method and I evaluated the use of tools that are intended to be an example of development projects.

1. 研究背景

近年，ソフトウェア開発は大規模化，複雑化が進み，工数の過半数はソフトウェアテストが占めている．要求仕様の変化は開発工数を肥大化させ，膨大な手戻り工数とコストが発生するなど，開発の恒常的な問題となっている．製品リリース後であっても，バージョンアップや脆弱性対応のために仕様変更を伴う場合があり，テスト工程でも運用管理の重要性が増している[1]．仕様変更でテストケースの修正が必要になった場合は，再利用することで手戻りコストを最小限に抑え，修正が必要な物とそうでないものを明らかにする事が必要である．これらの文書は，利用実績の多さからExcelで作成・管理される事が多いが，数の多さや文書の変化が多いことから，管理用のシステムを別途用意する事例が増えている．近年ではアジャイル開発やチケット駆動開発の登場により，Redmine等の開発工程管理ツールが増えたが，要件定義から単体テストまでの管理を前提とされており[2]，テスト工程を管理するWebシステムは数が少ないのが現状である．

2. 既存のテストケース管理システム

開発で生じる文書を管理するシステムとして，TestLink Communityが開発するTestLinkがある（図１）．これは，テスト工程で発生するドキュメントを管理する，Webベースのテスト管理システムである．McAfeeやYahoo!USAなどで採用実績があり，現在はVer.1.9.5が公開されている[3]（英語版）．csv形式での文書インポート機能や，pdfでのエクスポート機能が実装されている．TestLinkの特徴として，要求仕様に基づくテスト計画の作成及び実行が可能である．個々の要件に対して，テストケースを多対多でアサインすることができる．また，要件にアサインされたテストケースを元に，要件網羅率をカバレッジとして提供する機能がある．テストケースは，キーワードによるフィルタリング検索が可能で，１つのテスト計画に対して複数の検索キーワードを設定できる．これらの文書の操作には，実行者権限を細かく設定できる．　TestLinkは，日本語版の開発が遅れ気味であり，最終更新はver.1.8.5のAllInOneパッケージで停止している[4]．また，上述の要件仕様に基づいたテストを行うには，機能を有効化しなければならず，デフォルト状態ではキーワードの作成等も管理者以外は行えない設定となっている．要求仕様書の管理機能はテスト計画を立てる上での補助文書として扱う前提であり，仕様変更に伴う回帰テストをサポートすることは難しい．

3. 研究目的

本研究では，TestLinkを参考に，仕様変更に伴う回帰テストをサポートしたWebテスト管理手法を提案する．設計文書に変更が生じたら，修正が必要と考えられるテストケースを検出する．設計文書の内容から関連するテストケースを探すため，設計文書とテストケースそれぞれに，検索タグを文字列として付与する．関連があるテストケース全てに対応するため，テストケースの実行順序を解析して抽出する．関連の有無を調べるため，関連度を設定し，それぞれの設計文書とテストケースごとに計算する．

これをテストケース選択の閾値として用いて，設計文書と関連度が高いテストケースのみを抽出し，修正必要性の高いものから編集できるようにすることを目指す．

図1：テストケース管理システム TestLink

4. 提案手法

4.1 手法概要

本研究では，開発で生じる設計文書とテストケースを，文字列として解析処理し，情報を付与することにより，設計変更情報を定義する(図2)．設計文書とテストケースは，あらかじめ規定したフォーマットに従うものとする．文書のフォーマットを定めることにより，可搬性を向上し，入力の自動化を行いやすくする．　設計変更情報は，主に4つのデータで構成する．1つ目は，タグ文字列である．タグ文字列とは，設計情報を定義する単語で，設計文書，テストケースにそれぞれ最低5つ以上のタグ文字列を付与する．

2つ目は，テストケースの依存関係である．テストケースは，所定の実行順序でなければ，テスト結果を正しく確認できない場合がある．本研究では，この実行順序列を依存関係と呼び，テストケースの文書番号によって記述する．

3つ目は，関連性である．定義されたタグ文字列とテストケースの依存関係をもとに，テストケースが設計文書とどれだけ相関性があるかを，関連度として計算する．

4つ目は，アサイン情報である．アサインとは，設計文書に書かれた項目をテストするテストケースとして，関連づけることである．設計文書には，最低1つのテストケースをアサインする．

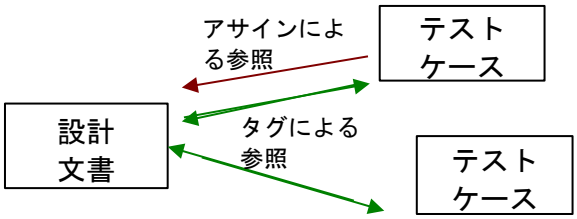

図2：設計変更情報による参照例

4.2 想定している適用範囲

本手法は，主に機能テストを対象とした手法である．テストケース作成者が，設計文書に従って，テストケースを作成するときに利用することを想定している．また，設計文書はあらかじめ作成されていることを前提としている．

4.3 手法詳細

4.3.1 設計情報としてのタグ付け

タグ文字列は，設計文書やテストケースの内容を表す単語を使用する．設計文書とテストケースへのタグ文字列の付与は，事前にどの用語を利用するのかを合議し，明確にする．タグ文字列は，後述する関連度や依存関係の計算で利用する．

タグ文字列には，名前空間の概念を導入する．同じタグであっても，用語が指す概念が別であるケースを考慮する．タグが同じで名前空間が別だった場合は，別文字列として区別する．タグ文字列は単語であり，最低5個以上のタグ文字列を付与する．個数上限は設けない．

4.3.2 テストケース依存関係の記述

テストに実行順序の記述が必要である場合，依存関係として，次にテストするテストケースの文書番号を記述する．例えば，テストケース1番をテストした後に2番または3番をテストする場合，1番に記述する依存先は2または3となり，2番と3番には何も書かない(図3)．つまり，依存関係を考慮するのは依存先のテストケースのみであり，依存元のテストケースは考慮しない．

4.3.3 設計文書とテストケースの関連性の定義

関連性は，設計文書とテストケースのタグ文字列一致個数で計算する．関連度は，アサインしているかどうかで以下のように計算方法を変える．また，依存関係にあるテストケースについても，関連度を算出する．

・アサインしている場合

関連度の初期値は0とする．設計文書とテストケースのタグを比較し，アサインしている場合は，タグ一致個数をそのまま関連度とする．例えば，アサ

インしていてタグが２個一致した場合は，関連度を２とする．アサインしていない場合は，関連度を０とする．

・アサインしていないが，設計文書と依存関係にあるテストケースが存在する場合

間接的な関連性があるとみる．タグ一致個数/2 を関連度として計算する．設計文書にアサインしているテストケースの依存先がアサインしていない場合が，この例に当たる．

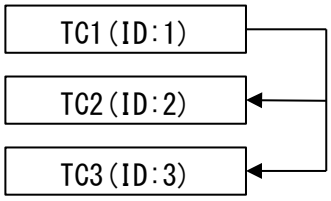

図３：TC 依存関係の例

4.3.4 再利用可能なテストケースの抽出

設計文書にアサインされているテストケースが，最利用可能なテストケースの候補として確認できる．また，アサインされていないが依存関係にあるテストケースも，候補として提示する．提示するテストケースは，関連度によって再利用性を確認できる．

仕様変更が起きた場合，設計文書の内容及び付与されているタグ文字列を，必要があれば再調整する．これにより，テストケースの関連度が再計算され，関連性の高いものを選択可能となる．

5. 提案手法の実装

5.1 実装内容

提案手法の確認として，PHP と MySQL を用いた Web ベースのテストケース管理システムを試作した．このシステムは，設計文書が作成されていることを前提に，テストケース作成者が使用することを想定している．本論文では，設計文書とテストケースをテキストデータとして入力する機能，依存関係を解析する機能，関連性があるテストケースを表示する機能を実装した．

システム構成図を図４に示す．本システムは，設計文書表示部，テストケース表示部，データベース接続部，テストケース取得部，依存関係抽出部で構成する．データベースは，設計文書を格納するテーブルと，テストケースを格納するテーブルがある．タグ情報は，各テーブルへ文字列として格納する．タグは，単語と単語の間に区切り文字’,’を挿入し，名前空間を’::’で分割した一つの文字列として扱う．

画面左に設計文書，画面右にテストケースが表示される．設計文書の項目をクリックすると，関連性のあるテストケースを抽出し，その文書番号を画面に表示する(図 5)．テストケースの文書番号をクリックすることで，文書を開くことができる．画面には，関連度や依存関係にあるテストケースを列挙する．また，タグ情報なども表示する．

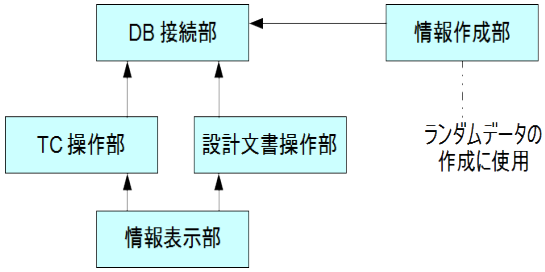

図４：システム構成図

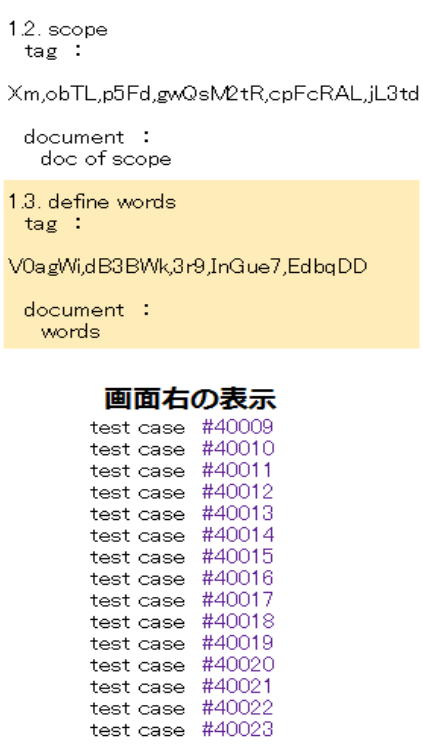

図５：画面の表示設計

5.2 利用方法

事前にテキストデータの設計文書とテストケースを用意する．このテキストデータは，付録の DB スキーマの並び順の通りに記述されているものとする．項目ごとに，区切り文字を挿入する．これは，テキストデータを解析する時に利用する．テキストデータをアップロードし，データが正しく格納されると，成功と表示される．これを確認した後，データ表示ページを開き，テストケースの依存関係を確認する．設計文書をクリックすると，アサインされているテストケースを表示し，更に依存関係にあるテストケ

ースも表示する．また，テストケースや設計文書をクリックすることで，データの編集を行える．

6. 実験と検証

6.1 依存関係抽出実験

テストケースの依存関係を正しく解決できているかを確認するため，依存関係抽出実験を実施した．実験用に設計文書を 20，テストケースを 100 用意した．文書内容は，PHP によるランダム文字列の自動生成である．他に，タグ文字列や依存関係の記述，アサイン先も，ランダム英数字を利用している．実験内容は，次の操作を行った．

まず，設計文書とテストケースを DB へ登録する．次に，登録した設計文書を選択し，設計文書にテストケースがアサインされていることを確認する．最後に，依存関係にあるテストケースが列挙されることを確認する．

6.2 依存関係抽出実験結果

実際に挙動を確認したのが図 6 である．依存関係が正しいかどうか，DB 内容と照らし合わせて視認した．その結果，同じテストケースを重複表示せずに，依存関係を正しく抽出できることを確認した．

この実験で，テストケースが循環参照に陥るケースを確認した．循環参照とは，依存先テストケースを辿っていくと，解析を始めたテストケースへたどり着く無限ループ状態である．これを回避するため，循環参照が発生した場合は解析を打ち切るようにした．今回の実験は件数が少なかったので視認による確認ができたが，実際の開発では確認できない可能性があるので，より確実な確認方法を検討する．

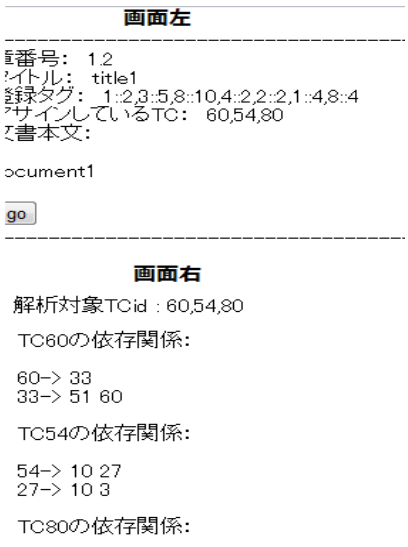

図 6：実際の表示画面

6.3 関連度算出と TC 抽出実験

5.1 節とは別の設計文書とテストケースを用意して，関連度算出実験と TC 抽出実験を行った．文書内容は実際の開発例を想定し，本研究で使用するテストケース管理ツールの，テキスト文書を入力し解析を行う機能をテストするものを用いた．実験内容は次の操作を行なった．

まず，5.1 節と同様の操作を行う．次に，アサインしている TC に関連度が表示されることを確認する．依存先 TC の関連度に補正がけされていることを確認し，最後に，設計文書の内容を変更して TC の変化を観察する．

6.4 関連度算出と TC 抽出の実験結果

実際に挙動を確認したのが図 7 である．関連度が正しいかは視認によって確認した．実験の結果，次の３つことがわかった．

１つ目はタグ一致個数をそのまま関連度とすると情報の粒度が低い．現在はタグを最低個数を５個以上つけると定義しているが，タグの付与は思った以上に時間がかかるため，５個以上つけることを想定しづらい．しかし５個のタグで一致個数を比較すると，関連度は平均２～３程度になると予想される．また，依存関係にあるテストケースは，アサインしていなければ関連度の上限値は 2.5 である．そのため，関連度３以上のテストケースは稀な存在になり，重要なテストケースが他の物の影響で埋もれやすい事がわかった．

２つ目は，微細な変更だと関連度に影響を与えづらい．設計文書の内容を変更する時，機能の追加や削除を行わない場合は，タグを変更しなくて良い事がわかった．そのため，小さな変更は関連度に影響を与えられず，また文書の編集とタグの編集はほぼ同時に行わなければならないことがわかった．

３つ目は，そもそもタグ文字列が一致しづらい．タグ文字列は事前に合議している前提だが，それでも選択肢が増えると関連度は低くなりがちだった．実際の開発では，さらにタグの増加が予想でき，関連度が０のままのテストケースが増えることが懸念される．これを避けるため，文中の単語をタグとして利用するように強制するか，重要語の自動抽出を検討する等の措置が必要があることがわかった．

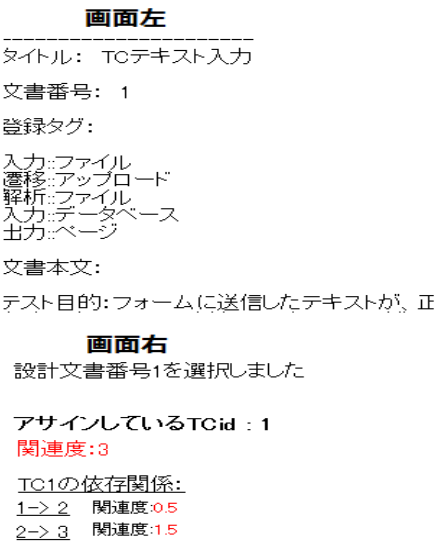

図 7：実際の表示画面

7. 考察

7.1 依存関係抽出実験の考察

今回の実験では，循環参照の回避策として，テストケースを辿る時に個々のテストケースにフラグを立てておき，フラグのたったテストケースへ辿り着いた時にループと判定している．そのため，循環参照の先に依存関係のあるテストケースが続いてても，検知せず打ち切ってしまい，正確な依存関係を解析できない． そこで，循環参照が発生した場合，その部分を一つのテストケースに置き換えて解析を続行する手法を提案する．循環参照が発生する場合は，ループへの入り口・出口となるテストケースが存在する(図 8)．このループを構成するテストケース群を扱うテストケースを新たに作成し，置換することでループを取り除く．これによって，同じ遷移を繰り返すストケースであっても，通常と同じような扱いができるようになる(図 9)．

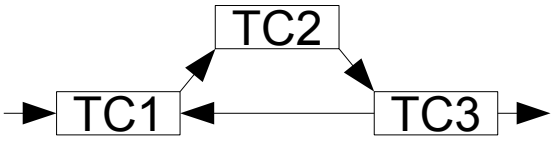

図 8：循環参照の発生例

7.2 関連度算出と TC 抽出の考察

実験結果から，タグの個数を最低 5 個とすることは，人手での付与数としては多すぎるものの，機械処理にかけるには少なすぎることがわかる．また，関連度の粒度を上げるためには，タグの一致個数を関連度とせずに，統計処理による重みがけが必要である．タグ個数の少なさが関連度の精度を下げているため，重要な用語をタグとして付与するのではなく，なんらかの自動判別処理が必要である．対応策として次の 3 つを検討している．

1 つ目は，タグを付与するのではなく，文章にマークしておく方法である．文中にタグとしてしようする単語が登場した場合，その単語を[]などでマークしておく．例えば，[タグ]として使いたい[単語]とマークしたら，"タグ,単語"をタグ文字列として扱う．

2 つ目は，文書内容を構文解析して単語をサジェストする方法である．文書内の単語を利用し，かつ単語内容を変更可能にすることで，タグ文字列の選択にかかる時間を軽減させる．

3 つ目は，文章同士の類似度を比較する方法である．レーベンシュタイン距離等で文字列を比較し，類似度が高いものを関連度の高さとして採用する．ただし，文章が似ていることが意味のある関連性として評価できるかは，検証が必要だと考えられる．

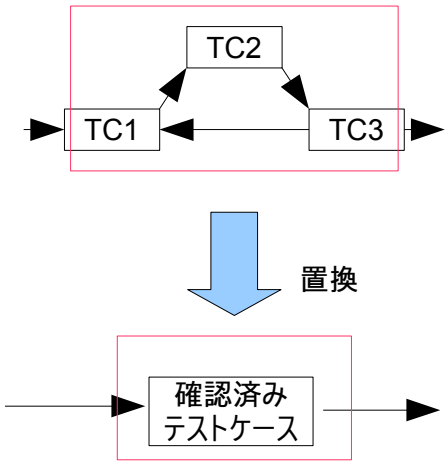

図 9：循環参照の置換

8. まとめ

本研究は，仕様変更に伴う回帰テストをサポートした Web テスト管理手法を提案した．開発で生じる設計文書とテストケースを，文字列として解析処理し，情報を付与することにより，設計変更情報を定義する．設計文書に変更が生じたら，修正が必要と考えられるテストケースを検出する．設計文書とテストケースそれぞれに，検索タグを文字列として付与する．関連があるテストケース全てに対応するため，テストケースの実行順序を解析して抽出する．関連度を設定し，それぞれの設計文書とテストケースごとに計算する．提案手法の確認として，PHP と MySQL を用いた Web ベースのテストケース管理システムを試作した．設計文書が作成されていることを前提に，テストケース作成者が使用することを想定している．テストケースの依存関係を正しく解決で

きているかを確認するため，依存関係抽出実験を実施した．実験用にPHPを用いて仮文章を作成した．また，関連度算出実験とTC抽出実験も行った．文書内容は実際の開発例を想定し，本研究で使用するテストケース管理ツールの，テキスト文書を入力し解析を行う機能をテストするものを用いた．実験結果から，タグの個数を最低５個とすることは，人手での付与数としては多すぎるものの，機械処理にかけるには少なすぎることがわかった．また，関連度の粒度を上げるためには，タグの一致個数を関連度とせずに，統計処理による重みがけが必要である．タグ個数の少なさが関連度の精度を下げているため，重要な用語をタグとして付与するのではなく，なんらかの自動判別処理が必要である．

今後の方針として，タグを付与するのではなく，文章にマークしておく方法，文書内容を構文解析して単語をサジェストする方法，文章同士の類似度を比較する方法をテストケース抽出方法の代案として検討する．

参考文献

[1]史寧, 八重樫理人 , 高木智彦 , 古川善吾, “**機能木を用いたテストケース管理方法の提案**,” 2011

[2] **梶野 由貴子**, “TestLink **はじめの一歩**,” JaSST **北海道** , 2008

[3]TestLink Community
http://www.teamst.org

[4]TestLinkJP
http://testlinkjp.org

付録

表１：テストケースのスキーマ

テストケース番号
用語
テスト構成
テスト環境
テスト実施手順
テスト合格条件
入力
期待出力
テスト結果
TCの依存関係
タグ文字列
アサインする設計文書の番号
参考資料
備考

表２：設計文書のスキーマ

章番号
タイトル
本文
文書番号
タグ