研究最終報告

MVC Web アプリケーションの
機能縮退を考慮した振舞い検証方法の提案

指導教授　　国立情報学研究所 アーキテクチャ科学研究系
本位田 真一 教授　印

担当教授　　(日工専) 情報工学科
高橋 宜孝 教授　印

報 告 者　　(日工専) 情報工学科
第 53 期研究科生　澤野 宏貴　印

研究期間　　2013 年 4 月 1 日 ～ 2014 年 3 月 21 日
報告年月日　　2014 年 3 月 26 日

目次

MVC Web アプリケーションの
機能縮退を考慮した振舞い検証方法の提案

澤野宏貴[†1]　前澤悠太[†2]　高橋竜一[†3]

概要：Web アプリケーションを保守する時，保守担当者が機能を部分的に制限することがある（機能縮退）．保守担当者は，機能縮退に応じたナビゲーション要求（例えば，Web ページへの到達可能性）が満たされていることを期待するが，近年の動的な Web アプリケーションの振舞いを人手で確認することは難しい．既存手法を用いると実装コードを入力に振舞いを自動検証できる．しかし，機能縮退後の振舞いに対しては，それに応じたコード修正が必要であり，誤った縮退操作を計画すると不要なコード修正が発生するため，保守コストの増大につながる．そこで本論文では，MVC Web アプリケーションからナビゲーションモデル（NM）を抽出し，この NM 上で機能縮退を表現するツールを提案する．縮退対象の機能に関する情報は，保守担当者から与えられる．また本ツールは，検証式テンプレートを用いてナビゲーション要求を半自動で検証する．保守担当者はアプリケーション依存の値（例えば，Web ページ間の最小リンク数）を入力できる．オープンソースな実アプリケーションである CandyCane を用いてケーススタディを実施し，提案ツールが誤った縮退計画に対し，要求違反を出力できることを確認した．したがって，保守担当者が提案ツールによって縮退計画を改善し，正しく機能縮退できると考えられる．

キーワード：機能縮退，MVC Web アプリケーション，リバースエンジニアリング，静的解析，モデル検査

Verifying Failsoft-Related Invariants
on Navigation Model Extracted from MVC Web Applications

HIROKI SAWANO[†1]　YUTA MAEZAWA[†2]
RYUICHI TAKAHASHI[†3]

***Abstract*:** When maintaining Web applications, maintainers make the application functionalities partially unavailable (Failsoft). Although they expect that navigation requirements (e.g. Web page reachability) according to the failsoft are satisfied, manually verifying modern Web application behavior is difficult due to its dynamicity. Existing techniques can automatically verify the behavior corresponding to the implementation of the code. However, addressing the failsoft requires modifying the code, resulting in increasing maintenance costs because of unnecessary code modification caused by improper failsoft plans. In this paper, we propose a tool that extracts a navigation model (NM) from MVC Web applications and represents the failsoft on the extracted NM. Failsoft-related functionalities are given by maintainers. Our tool can semi-automatically verify navigation requirements by using verification templates. Maintainers can input particular parameters (e.g. the minimum number of links between Web pages) depending on their applications. We conduct a case study by using an open-source, real-world application called CandyCane and demonstrate that our tool outputs the requirement violations against improper failsoft plans. Hence, we conclude that maintainers can leverage our tool for improving their failsoft plans and effectively realizing the proper failsoft.

***Keywords*:** Failsoft, MVC Web applications, Reverse Engineering, Static Analysis, Model Checking

1. はじめに

近年，Web アプリケーションは広く普及し，その数は増加の一途を辿っている．Web アプリケーションはインターネット上で電子商取引などの様々なサービスの提供を可能にした．その結果，Web アプリケーションは我々の生活の一部として欠かすことができなくなってきている．

Web アプリケーションを保守する時，保守担当者が機能を部分的に制限することがある．例えば，Jackson County Oregon[1]，LSE[2]や三菱東京 UFJ[3]のように，システムメンテナンス時に利用できない機能へのアクセスを抑止した上で運用を続けることがある（機能縮退）．機能縮退によって Web アプリケーションの Web ページとそれらのリンク関係を表すナビゲーションが変わる．Web アプリケーション開発では，Web ページの到達可能性などのナビゲーション要求があるが[1, 2]，縮退運転時もこのような満たすべき要求があると考えられる．誤った機能縮退では，例えばユーザに停止した DB へのリクエストを許可し，アプリケーションがクラッシュするなどの致命的な問題につながる．

Web アプリケーションのナビゲーションはナビゲーショ

†1 (株)日立製作所 インフラシステム社
Hitachi, Ltd., Infrastructure Systems Company
†2 東京大学
The University of Tokyo, Bunkyo, Tokyo 113-8656, Japan
†3 早稲田大学
Waseda University, Shinjuku, Tokyo 169-8555, Japan

[1] http://www.co.jackson.or.us/News.asp?NewsID=1344
[2] http://www.lse.ac.uk/aboutThisWebsite/Home.aspx
[3] https://entry11.bk.mufg.jp/ibg/dfw/APLIN/loginib/login?_TRANID=AA000_001

ンモデル(NM)で表現することができる. NM は Web ページを状態とみなし, ユーザアクション(リンクやボタンなど)によって状態を遷移する状態遷移図である[2].

近年の Web アプリケーションは, ユーザリクエストに応じて実行時に Web ページを生成するといった動的性を持つため, この振舞いを理解し NM を作成することは難しい. そこで，既存手法では実装コードから NM を抽出する[3, 12]. 取得した NM を形式的に記述することで，ナビゲーション要求を検証することができる．しかしながら，既存手法は実装コードを入力に NM の抽出をするため，縮退運転時の性質検証には，機能縮退のための実装（縮退実装）が必要である．この場合，縮退実装と既存手法の実行は，機能縮退後に満たすべき性質をすべて満たすまで繰り返さなければならず，非常にコストが高い.

そこで我々は，Web アプリケーションから抽出した NM を操作することで，機能制限された NM（RNM）を自動的に抽出・検証するためのツールを提案する．提案ツールの概要を図 1 に示す．本研究では，MVC デザインパターンで設計された Web アプリケーションを対象とし，制限される機能の単位を MVC の構造から，コントローラのアクションとする．提案ツールのワークフローは主に次の 3 つのステップに分かれる．各ステップの詳細は 3 章で述べる．

ステップ 1（抽出）：MVC Web アプリケーションを静的に解析し，NM を抽出する．この NM ではビューを状態とみなし，状態はアクション呼び出しによって遷移する．各遷移のラベルは，その遷移中に実行されたコントローラとアクションの組および参照されたモデルの集合である．また，NM から不変条件を抽出する．**ステップ 2（制限）**：保守担当者によって与えられた制限情報（制限するビューなど）をもとに，抽出した NM に制限を加え，RNM を得る．**ステップ 3（検証）**：RNM をモデル検査器 NuSMV[4]の記述に変換し，検証する．検証性質は，NM から抽出した不変条件と与えられた要求から定義する.

保守担当者は検証結果より，機能縮退がナビゲーションに与えるインパクトを分析し，縮退計画を改善できる．最終的に，要求を満たす縮退計画で機能縮退を完遂できる.

本研究では，提案ツールが扱う研究課題を次のように設定する.

RQ1　縮退運転時の振舞い検証によって縮退計画の成否を判断できるか？

RQ2　実用可能な時間で検証できるか？

我々の貢献は次の通りである.

・機能縮退のための MVC Web アプリケーションの静的な RNM 抽出・検証手法の開発

・提案手法を実装したツールの提供

・実アプリケーションの誤った縮退計画に対して，提案ツールが実用的な時間で要求違反を検出できたこと

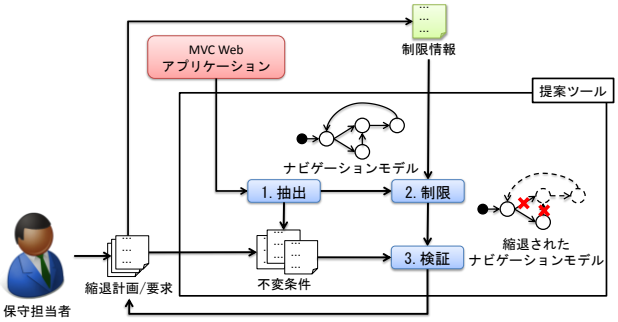

図 1　提案ツールの概要

Figure 1　Overview of our tool

本論文の構成は次の通りである．まず 2 章で本手法の対象となる MVC Web アプリケーションの特徴および機能縮退のナビゲーション要求への影響を述べ，動機付けの例を示す．続いて 3 章で提案する手法のアプローチを述べる．4 章では CandyCane[5]を用いたケーススタディとその評価結果を示す．5 章では関連研究を紹介する．最後に 6 章で結論および今後の課題を示す.

2. 背景

本章では，本手法の対象とする MVC Web アプリケーションの特徴および機能縮退のナビゲーション要求への影響を述べ，動機付けの例を示す.

2.1 MVC デザインパターン

MVC デザインパターンは Web アプリケーションの設計に広く利用されている[4, 12]. MVC Web アプリケーションでは，そのデータと手続き（モデル）をユーザが直接参照する情報から分離する（ビュー）. コントローラはリクエストを処理し，モデルおよびビューを制御することで，それぞれ主にデータの取得・保存，描画するビューの決定を果たす．コントローラは複数のアクションで構成される.

MVC Web アプリケーションではユーザリクエストをアクションが処理する．この仕組みから機能縮退はアクションを制限すると考える．提案ツールでは，MVC デザインパターンを採用したフレームワーク CakePHP[6]で実装した PHP Web アプリケーションを対象とする.

2.2 機能縮退とナビゲーション要求

Web アプリケーションを保守する時，保守担当者が機能を部分的に制限することがある．例えば，Jackson County Oregon，LSE や三菱東京 UFJ のように，システムメンテナンス時に利用できない機能へのアクセスを抑止した上で運用を続けることがある．機能縮退のために機能，すなわち MVC Web アプリケーションにおけるアクションを制限す

[4]http://nusmv.fbk.eu/

[5]http://yandod.github.io/candycane/

[6]http://cakephp.org/

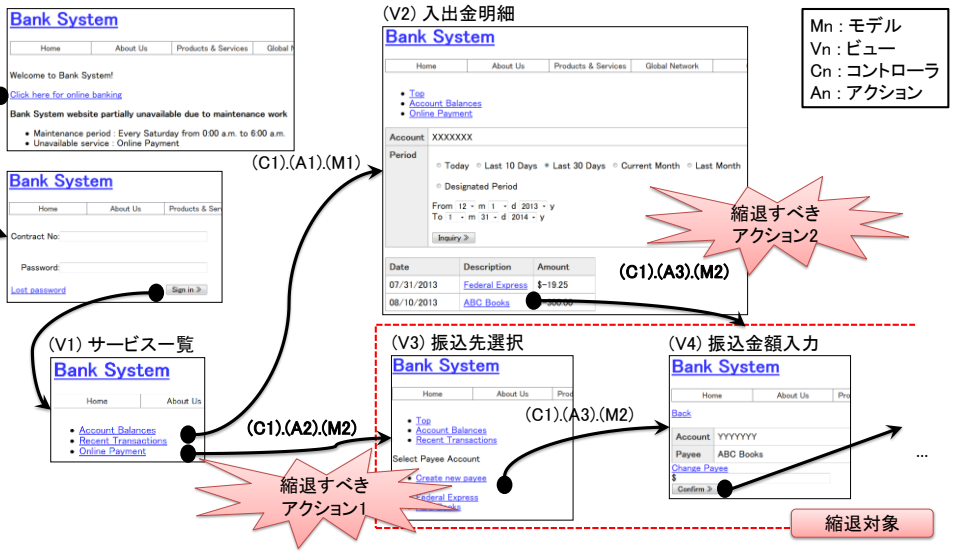

図 2　動機付けの例のスクリーンショット

Figure 2　Screenshots of our motivating example

る時, Web アプリケーションのナビゲーションが変化する．ナビゲーションとはユーザが訪れることのできる Web ページの列であり，通常，次のページは現在のページとユーザアクションによって決定される[2].

Web アプリケーション開発では，ナビゲーションは豊かなユーザエクスペリエンスを与えるために重要視される[5]．ゆえに，設計者はナビゲーション構造の構築に多大な労力を費やす[6]．この時，Web ページの到達可能性やナビゲーションのパスの長さなどの要求が関心事となる[1, 2]が，縮退運転時もこのような満たすべき要求があると考えられる．もしナビゲーション要求を無視した縮退運転を行えば，例えばユーザが停止した DB にリクエストを送信する Web ページに到達できてしまい，アプリケーションがクラッシュするといった致命的な問題が発生する．したがって，保守担当者は機能縮退を行う場合，このようなナビゲーション要求を満たしていることを保証しなければならない．

ナビゲーションをモデル化した NM は要求を明確にし，実装の振舞いを特定するために役立つ．また，NM を形式的に記述することで，ナビゲーションに関わる性質検証に使うことができる．しかし近年の Web アプリケーションは，ユーザリクエストに応じて実行時に Web ページを生成するといった動的性を持つため，この振舞いの理解を助ける NM を作成することは難しい．NM がなければ，多くのナビゲーション要求の実現はできない．そこで，実装から設計を回復するリバースエンジニアリング技術が役立つ．

Web アプリケーションから特定の側面をモデル化する手法の多くは動的解析に基づく[8, 9]．しかし，動的解析では実行環境に依存した結果しか得られず，ナビゲーション要求の違反を引き起こす遷移や状態を見落としかねない．そこで，静的解析による NM の抽出・検証手法が提案されている[3]．しかし既存手法[3]は，我々が取り上げる機能縮退の問題には取り組んでおらず，実装コード上に存在する振舞いのみを検証する．したがって，縮退実装なしには縮退運転時の性質検証ができない．保守のために縮退実装と既存手法の実行を繰り返すことは現実的ではない．そこで我々は，縮退実装なく RNM を獲得し，縮退運転時の性質検証が可能なツールを提案する．

2.3　動機付けの例

本研究の動機付けの例として，インターネットバンキング Web アプリケーションを示す．図 2 にスクリーンショットを示す．この例では，ユーザは主に次の 3 つのサービスを利用することができる．1）残高照会サービス：口座の残高を照会する．2）入出金明細照会サービス：指定した期間内の入出金明細を表示する．明細から振込先を指定できる．3）振込みサービス：指定した振込先に振込みを行う．過去の振込先を指定できる．

この例に対して，保守のために振込みサービスが一時的に利用できないシナリオを与える．この縮退運転中，次の要求を満たさなければならない．**要求 1：**振込みサービスに関わる Web ページ(図 2 の赤枠内)に到達できないこと．**要求 2：**残高照会サービスと入出金明細照会サービスに関わる Web ページに到達できること．

図 2 より，振込みサービスへの経路は次の 2 通りある．一方の経路では，(V2) から過去に利用した振込先を指定することで (V3) を迂回し，(V4) に至る．他方では，(V3) から振込先を選択し，(V4) に遷移する．

保守担当者は，該当する遷移を特定し制限するアクションを決定するために, すべてのリンク構造の把握に努める．その際，着目の対象はプログラムフローやブラウザ上で視覚化された情報である．しかし通常，そのプログラムの複雑さのために，ブラウザ上で実際に見える情報に強く依存すると考えられる．その結果，見落としが生まれやすい．

図 3 にコード片を示す．ここでは，3 つのファイルを取り上げる．それぞれ，次の役割を持つ．なお，ここで登場するアクションはすべて（C1）のアクションである．**（V1）サービス一覧ビュー**：ユーザが利用可能なサービスの一覧を提供する．3 つのリンクを持ち，それぞれ balance アクション（1.2 行目），transaction アクション（1.4 行目），selectPayee アクション（1.6 行目）を呼び出す．**（V2）入出金明細ビュー**：入出金履歴を提供する．過去に取引があれば（2.1 行目），その明細を全件表示する（2.2 行目～2.7 行目）．明細は payment アクションへのリンクを持つ（2.4 行目～2.5 行目）．**（C1）バンクコントローラー**：transaction アクション（3.1 行目）は，Transaction モデルを参照してユーザが指定した期間内の入出金明細の配列を取得する（3.2 行目）．取得したデータをビューに受け渡す準備をし（3.3 行目），（V2）を描画する（3.4 行目）．selectPayee アクションは，（V3）を描画する（3.5 行目）．payment アクション（3.6 行目）は，リクエストされた振込先の妥当性を判定し（3.7 行目），不正である場合は，selectPayee アクションにリダイレクトする（3.8 行目）．さもなければ（V4）を描画する（3.9 行目）．

（V2）では入出金明細が空でない場合に限り，その情報を表示する．つまり，取引の有無や期間の指定によっては payment アクションへのリンクが現れない．このような遷移の発見には，プログラムを正確に把握する必要がある．しかし，これは多大な労力を要し，とりわけ実装に精通していない保守担当者にとっては困難を極める．

2.4 問題提起

Web アプリケーションの保守時，縮退運転が必要とされる．その際，ナビゲーションへのインパクトを明確にし，要求を満たす縮退を履行しなければならない．既存手法は実装から NM を抽出・検証することができる．しかし，適用のために縮退実装を繰り返すことは非常にコストが高い．そこで，我々は MVC Web アプリケーションから抽出した NM 上で縮退操作をすることで，縮退実装なく縮退運転時の性質検証をすることを目指す．提案ツールの導入によって，保守担当者は機能縮退がナビゲーションに与えるインパクトを分析することが可能となる．そして，ナビゲーション要求を満たす縮退計画を実現できる．

3. 提案ツール

本論文で我々は，MVC Web アプリケーションから RNM を獲得・検証するツールを提案する．本章では，まず，NM の静的な抽出手法について述べる．続いて，RNM を取得するための NM の制限方法について述べる．さらに，ナビゲーション要求の検証方法について触れる．最後に，動機付けの例を用いた提案ツールの実行結果を示す．

```
1.1 <div id="bankMenu"><ul><li><?php
1.2 echo $this->Html->link('Account Balances','/Banks/balance');
1.3 ?></li><li><?php
1.4 echo $this->Html->link('Recent Transactions','/Banks/transaction');
1.5 ?></li><li><?php
1.6 echo $this->Html->link('Online Payment','/Banks/selectPayee');
1.7 ?></li></ul></div>
```

(V1) Banks/top.ctp

```
2.1 <?php if (!empty($transactions)): ?>
2.2 <?php foreach ($transactions as $transaction): ?>
2.3 <tr><td><?php echo $transaction['date']; ?></td>
2.4 <td><?php echo $this->Html->link($transaction['name'], array('controller'
2.5 => 'banks', 'action' => 'payment', 'id' => $transaction['id'])); ?></td>
2.6 <td>$<?php echo $transaction['amount']; ?></td></tr>
2.7 <?php endforeach; ?>
2.8 <?php endif; ?>
```

(V2) Banks/transaction.ctp

```
3.1 public function transaction() { /*(A1)…*/
3.2   $transactions = $this->Transaction->getInfo($request);
3.3   $this->set('transactions', $transactions);
3.4   $this->render('transaction'); }
3.5 public function selectPayee() { /*(A2)…*/$this->render('select_payee'); }
3.6 public function payment() {/*(A3)…*/
3.7   if (!$this->Payee->isValidPayee($request)) {
3.8     $this->redirect('selectPayee'); return;
3.9   } else { $this->render('payment'); } }
```

(C1) BanksController.php

図 3　動機付けの例のコード片

Figure 3　Code snippets of our motivating example

3.1 MVC 構造に着目したナビゲーションモデルの抽出

提案ツールではまず，NM を静的解析によって抽出する．NM の抽出に先だって，解析する情報を決定するために制限プロセスの入力となる対象を特定する．そこで，MVC の構造から，モデル，ビュー，そしてコントローラがそれぞれ制限の対象であると仮定を置いた．NM にこれらオブジェクト間の関係を表現することで，縮退計画に応じた制約を課すことができる．詳細は 3.2 節で述べる．

上記の理由から，抽出する NM では，ビューを状態とみなし，状態はユーザアクションが呼び出すコントローラのアクションによって遷移する．各遷移のラベルは，その遷移中に実行されたコントローラとアクションの組および参照されたモデルの集合を保持する．初期状態はひとつだけ与えられ，外部サイトへの遷移は扱わない．

CakePHP の仕様[7]に基づき，モデル，ビュー，そしてコントローラをそれぞれ次のように解析する．

モデル：モデル同士は，アソシエーション機能を通じて，$hasOne や$hasMany などで関連を持ち，他のモデルにアクセスできる．そのため，関連モデルを参照するメンバを解析する．さらに，関連モデルのエイリアスを解決する．

ビュー：開発者はリンクやフォームを生成するために，普通 HtmlHelper::link()や FormHelper::create()などのコアヘルパーのメソッドを利用する．これらのメソッド呼び出しを解析することで，リンク先のアクションを決定する．指定に省略がある場合，規約から推定する．これに加え，ユーザ実装のヘルパーメソッドがリンクを生成することがある．しかし，ヘルパーの静的解析では，その挙動の判断が難しい．そこでユーザメソッドについては，生成するリンク毎にその方法に応じて，次のいずれかの情報を与える．1）戻り値となるコントローラとアクション．2）内部で使用されるリンクを生成するヘルパーメソッド名．3）リンク情報を渡す引数の位置．このようなメソッドは通常，その内部

[7] http://book.cakephp.org/2.0/_downloads/en/CakePHPCookbook.pdf

で前述のコアヘルパーを使用している．したがって，これらコアヘルパーのメソッド名をキーワードに検索することで，設定が必要なメソッドを特定することができる．部品化されたビューであるエレメントは element()によってビューに読込まれる．エレメントが持つリンクはそのエレメントを読込むビューが持つ．多くのインタフェースをラップするレイアウト（$layout）は，レイアウトが適用されるビューからアクセスできる．アクション名などが変数を評価する場合，ビューでの変数定義に加え，Controller::set()でコントローラから渡されたデータを利用する．

以上により，アクションへの遷移元状態を確立する．

コントローラ：ビューからのリクエストはコントローラ内部で単一のアクションを呼び出す．各アクションは複数のビューを描画する可能性を持ち，リクエストを解釈することで動的に遷移先を選択する．指定したビューを描画するためには render()を使う．省略がある場合，ビューは規約によって決定される．フローのコントロールには redirect()が最もよく使われる．メッセージ表示後にリダイレクトするためには flash()が使われる．上記のメソッドを解析し，アクションの遷移先状態を確定する．リクエストが DOM[8] 操作を目的とした Ajax の非同期通信であってもアクション呼び出しによって状態は遷移する．その時にクライアントに出力するビューやエレメントはひとつの状態とし，同期的に遷移する状態と区別しない．

また，コントローラはモデルを参照する．利用可能なモデルは規約, $uses, そしてモデルの解析結果から得られる．モデルの参照情報をアクションごとに記憶する．

ビューとコントローラの解析結果から，状態間を結ぶ．状態間の遷移上にはラベルとして，リダイレクト先を含む実行されたコントローラとアクションの組および参照したモデルの集合を保持する．

3.2 ナビゲーションモデルの制限

我々は，MVC のそれぞれの要素が制限の対象であると仮定する．したがって，各要素の利用箇所に基づき，抽出した NM に制限を加える．

このプロセスでは，制限をアクション単位に行う．すなわち，制限されるアクションをラベルに持つ遷移を NM から除く．与えられた制限対象に応じて，次のように制限すべきアクション集合を計算する．**ABR :** 特定のアクション．**CBR :** 特定のコントローラのすべてのアクション．**VBR :** 特定のビューを呼び出すすべてのアクション．**MBR :** 特定のモデルを参照するすべてのアクション．制限の例を図 4 に示す．破線は利用できないことを表す．

この制限のために，機能要求から実装箇所が追跡可能でなければならない．そこで本手法の適用においては，要求

[8]http://www.w3.org/DOM/

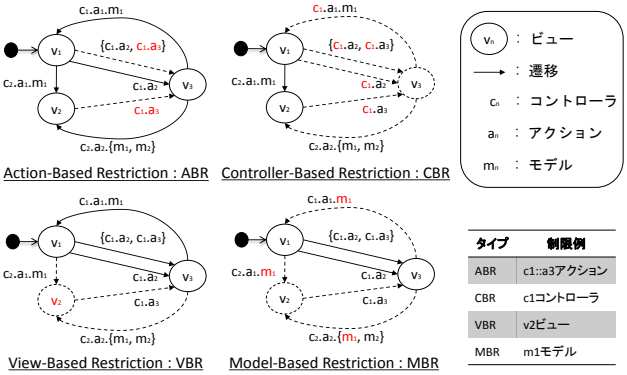

図 4　抽出したナビゲーションモデルの制限種別

Figure 4　Restriction types for extracted navigation model

表 1　検証カテゴリと CTL 式[2]

Table 1　Categories of verification and CTL formula based on [2]

ID	カテゴリ	CTL 式
P1	到達可能性	EF (*ViewVar* = *ViewName*)
P2	デッドロック	AG EF (*ViewVar* = *ViewName*)
P3		AG ((*ViewVar* = *View1Name*)->EF (*ViewVar* = *View2Name*))
P4	到達ステップ	AG ((*ViewVar* = *View1Name*)->AX (*ViewVar* = *View2Name*))
P5		AG ((*ViewVar* = *View1Name*)->EX (*ViewVar* = *View2Name*))
P6		AG ((*ViewVar* = *View1Name*)->EX EX EX (*ViewVar* =*View2Name*))

追跡マトリクス（Requirements Traceability Matrix : RTM）を入力として定義する．RTM は要求をソフトウェア機能とマッピングするために利用される．本手法の RTM は機能要求をキーにモデル，ビュー，コントローラ，そしてアクションを持つと定義する．RTM を利用し，提案ツールは要求の実装箇所を特定できる．RTM が持つ情報はアプリケーションの静的な構造であるため，開発者がこれを作成することは難しくない．さらには，本手法の適用にあたっては，完全な RTM を作成する必要はなく，制限対象の追跡のみできれば良い．加えて，RTM の妥当性検証には既存手法が利用できる[7].

3.3 縮退運転時のナビゲーション要求検証

モデル検査を検証プロセスに導入する．本論文では，ビューの到達性およびその CTL 式[2]を扱うため，モデル検査器には CTL 式を検証可能な NuSMV を選択する．RNM を NuSMV の記述に自動変換し，検証する．本研究では特に表 1 に示す性質を扱う．*ViewVar* は滞在中のビューを示す状態変数であり，*ViewName* は検証するビューである．

表 1 の性質のうち P1（¬P1）と P2 を検証することで，それぞれ期待する状態のみ到達可能（不可能）か，他の状態に遷移できない状態がないか，を確認できる．これらの性質は，抽出した状態集合を用いて要求に依存することなく定義できる．一方，P3～P6 を定義するためには要求に依存する固有のパラメータが必要となるため，保守担当者に与えられた特定のビューの組み合わせや到達にかかる最小

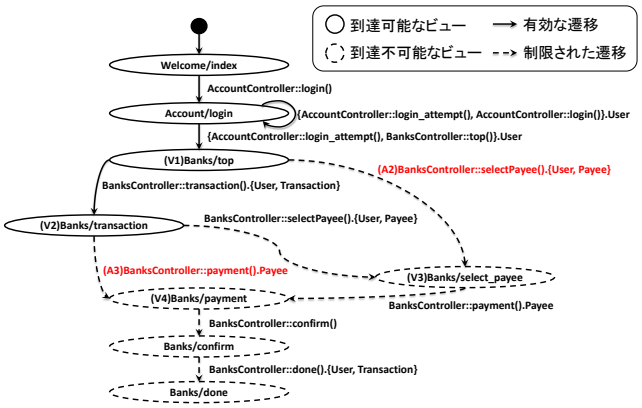

図 5　動機付けの例への適用

Figure 5　Application to our motivating example

リンク数を使用する. 表 1 以外の性質の検証には CTL パターン[9]を参照することができる.

3.4 動機付けの例への適用例

2.3 節で提示した振込サービス縮退時の要求のひとつはそのサービスに関わる Web ページに到達できないことである．そこで，制限情報として次の 4 つのビューを指定する. 1) Banks/select_payee. 2) Banks/payment. 3) Banks/confirm. 4) Banks/done．これは，図 4 の VBR に該当する．提案ツールが抽出した RNM の一部を図 5 に示す．破線は制限された遷移または到達不可能なビューを表す．自動抽出された到達可能性の検証式（表 1 の P1 および¬P1）の検証結果から，要求を満たす機能縮退であることがわかる．制限の過程から，縮退すべきアクション（図 5 の破線の遷移ラベル）を判断できる．（A3）は見逃しやすいアクションであると言及したが，正しく制限するビューから突き止められている．

4. 評価実験

次の研究課題に解答するために，我々はケーススタディを行い，提案ツールを評価した．本実験では評価対象として，実アプリケーションであるオープンソースのバグ管理システム CandyCane を選択した．

RQ1　縮退運転時の振舞い検証によって縮退計画の成否を判断できるか？

RQ2　実用可能な時間で検証できるか？

4.1 実験手順

我々は最初に, ユーザ実装のヘルパーを処理するために, 22 個のメソッド情報を定義した．そのうえで，解析できたファイル数と全体数を表 2 に示す．

続いて，図 4 の制限種別を網羅する縮退要求と RTM を表 3 に用意し，これを提案ツールに入力した．その結果から RQ2 に解答し，かつ提案ツールの有効性を計るために，表 4 が示す通り，実行時間を計測し（7 列目），本来縮退の影響調査に確認が必要な箇所をコードから確認した．2 列目は関連するモデルの数である．3 列目は制限されるアクションのリンクを持つファイルの数である．4 列目は制限されるアクションの数と制限対象外かつリダイレクト先のアクションの数である．5 列目は削除された遷移の遷移元状態で，6 列目が削除された遷移の数である．

表 2　CandyCane の構成ファイル数と解析結果

Table 2　# of files of CandyCane and analysis results

対象	全体数	解析できた数
モデル	51	51
ビュー	97	95
エレメント	64	61
レイアウト	1	1
コントローラ	29	29

最後に，RQ1 に解答するために，我々が定義したナビゲーション要求を検証し，提案ツールが誤った縮退計画の性質違反を検出できることを確認した．

4.2 結果と考察

4.2.1 縮退運転時のナビゲーション要求違反の検出（RQ1）

A)　到達可能性

到達可能性の要求定義と検証結果を表 5 に示す. ここで, 3 列目の状態以外のすべての状態は到達できなければならない．ある状態が到達可能であることは表 1 の P1，到達不可能であることは¬P1 を定義することで検証する．¬P1 を検証することで，これが満たされない場合に到達に至る経路を反例から得ることができる．反例はアクションとビューの連続であり容易に理解できる. 4 列目の検証結果は, すべての性質を満たす場合に True であり，さもなければ False である．

初期の縮退計画として，保守担当者は次のように考えると仮定する．ID1）Users/add を描画するアクションは UsersController::add()のみである．ID2）News 関連のビューへの遷移はすべて NewsController が負う責務である．ID3）メニューから直接遷移可能なすべての Issues 関連のビュー（Issues/index および Issues/new）が到達不可能であれば，Issues 関連の全てのビューには到達不可能である．ID4）コントローラは一般的に，一つのモデルのロジックを管理するために使われる．そのため，Wiki のデータを扱うのは WikiController のみであり，到達できなくなるのは Wiki 関連のビューのみである．

上記の縮退計画に則り，検証をした結果 ID3 と ID4 が性質違反を引き起こした．ここで得られた反例はそれぞれ，到達不可能であるべき複数の状態が到達可能，到達可能であるべき状態の一つが到達不可能であることを指摘していた．反例を分析した結果，例えば ID3 ではマイページやメ

[9]http://patterns.projects.cis.ksu.edu/documentation/patterns/ctl.shtml

表 3　縮退要求と要求追跡マトリクス(RTM)

Table 3　Requirements for failsoft and Requirements Traceability Matrix(RTM)

ID	タイプ	縮退要求	要求追跡マトリクス(RTM)				
			機能要求	モデル	ビュー	コントローラ	アクション
1	ABR	ユーザ作成機能が利用不可	新規ユーザを作成する機能	-	-	UsersController.php	add()
2	CBR	ニュース機能全体が利用不可	ニュースの追加，編集，削除などの機能を制御	-	-	NewsController.php	-
3	VBR	バグ一覧画面が到達不可	プロジェクトに登録されたバグを管理する画面	-	Issues/index.ctp	-	-
		バグ登録画面が到達不可	プロジェクトにバグを新規登録する画面	-	Issues/new.ctp	-	-
4	MBR	Wiki データにアクセス不可	プロジェクトの Wiki に関連するデータを扱う	Wiki.php	-	-	-

表 4　表 3 のシナリオを用いた実行結果

Table 4　Execution results with scenarios shown in Table 3

ID	関連モデル数	ビュー/エレメント/レイアウトファイル数	アクション数/関連アクション数	遷移元状態数	遷移数	実行時間(分)
1	-	2/1/0	1/2	27	56	6.55m
2	-	5/1/1	7/0	98	119	5.15m
3	-	6/8/1	2/2	98	1126	4.94m
4	2	8/4/1	11/3	98	257	5.51m

表 5　到達可能性の要求と検証

Table 5　Requirements and verifications for reachability

ID	到達不可能状態数	期待する到達不可能状態(¬P1)	結果
1	1	Users/add	True
2	4	News/*	True
3	2	Issues/*	False
4	9	Wiki/*	False

ニューの検索機能から Issues のビューにアクセスできることが明らかになった．ID4 ではプロジェクトのバージョン情報を表示する画面で Wiki データを参照するためその画面が到達不可能になることが判明した．

B)　デッドロック・到達ステップ

本実験の対象アプリケーションはメニューを持つ．これは各状態から初期状態への遷移を可能とするため，デッドロックの発生を防いだ．通常の Web アプリケーションはトップページへのリンクをメニューに持つため，機能縮退が原因でデッドロックが起こるケースは極めて稀であると考えられる．さらに，メニューの存在がリンク構造を浅くした結果，到達不可能になる場合を除き，到達ステップ数が変動する状況は観測できなかった．言い換えれば，この実験で扱ったアプリケーションのメニューがうまく実装されていたということである．つまり，メニューが本来持つべきリンクを持たない不適切な場合では，機能縮退によって到達性が低下することは十分に考えられる．

この結果から，提案ツールが機能縮退によるナビゲーション要求違反を検出できることを確認した．この検証によるインパクト分析を繰り返すことで，保守担当者は，より洗練された縮退計画を練ることができる．

4.2.2　実用的な実行時間（RQ2）と有効性

CandyCane は 98 の状態と 5340 の遷移を持つ中規模なアプリケーションであった．なお，この状態数とビューの解析数の差分は DOM 操作に使われたエレメントの数である．この規模に対し，表 4 の結果から提案ツールは約 6 分間で実行を終えていることがわかる．しかし，実行時間の大半は NM の抽出に費やされており，制限と検証の繰り返しは，ごく短時間で行うことができる．したがって，保守工程において十分に実用可能な実行時間であるといえる．

表 4 が示すように，従来の縮退プロセスではリンク構造や制限される遷移を把握するために，エレメントの利用箇所やレイアウトの存在，リダイレクトにも注意しなければならない．さらに，モデルの制限では，モデルの関連にも気を配らなければならない．このことから，人手でナビゲーション要求を満たす縮退計画を立てることは困難であるといえる．提案する検証手法では，上記の作業にかかる時間や労力を大幅に削減することができる．

4.2.3 ヘルパー情報設定のコスト

この実験で，我々はユーザ実装のヘルパーメソッド情報を定義した．しかし本来であれば，この作業は開発者がヘルパー作成時に行えばよく，保守担当者が苦労を被ることはない．さらには，ヘルパーはプレゼンテーション層のための頻繁な処理をまとめたモジュールであるため，修正の頻度は少ないと予想される．

4.2.4 状態・遷移の見逃しおよび不十分な制限

表 2 の結果から，本手法ではコードからファイルの特定に成功したことがわかる．しかし，コントローラ解析中に 2 つのビューを見落とした．また，ビュー解析中に 3 つのエレメント読込みを見逃した．前者の問題は，変数を用いた文字列操作によってビュー名を決定していたことが原因であった．後者は，分散したコード片を跨る複雑なデータフローが原因であった．また，モデルの利用箇所の特定では，動的なモデル読込みを解析できていない．

そこで，我々はビューやエレメントの指定に変数を使わないこと，そして動的なモデル読込みを可能な限り行わないことを規約として定めることでこの制限を緩和する．CakePHP は規約を尊重するフレームワークであるため，このような対策は妥当であると考えられる．規約に従えない場合は, 問題となる箇所に絞り込んだ動的解析による RNM の修正が有効である．

4.3 妥当性への脅威

A)　内的妥当性

ケーススタディでは，CandyCane を使用し提案ツールの有用性を示した．これはオープンソースな実アプリケーションであり，内的妥当性への脅威にはならない．しかし，利用したシナリオは我々が用意したものである．ゆえに，今後実アプリケーションの保守現場に提案ツールを導入し，実際の縮退計画に沿った実験を行うことが必要である．

さらに検証では，表 1 で示した到達可能性を検証することの有用性のみを示した．今後の課題として，例えば潜在的に機能縮退に起因するデッドロックや到達容易性低下の可能性を持つ Web アプリケーションでそれらを正しく解析できるかを評価することが求められる．

B)　外的妥当性

提案ツールは MVC デザインパターンの構造に着目し，NM の抽出と制限を実現した．したがって，本手法は本質的には MVC Web アプリケーション全般に適用できる．しかしながら, 提案ツールでは CakePHP フレームワークに特化した実装をしたため，今後 Struts[10]といった他のフレームワークで設計された Web アプリケーションへの適用が課題となる．

また，評価実験では中規模程度の実アプリケーションである CandyCane を対象としたが, 大規模な Web アプリケーションでも実用的な時間で解析できることを確認することが今後の課題である．

5. 関連研究

Ricca らは Web アプリケーションの状態は Web ページであると主張し，ナビゲーションの静的解析によるテストに成功した[1]．これに対し我々は，Web ページのテンプレートであるビューを状態とみなしている．

Kubo らは Struts Web アプリケーションの設定ファイルと JSP テンプレートファイルから作成した NM を SPIN[11]で検証した[3]．これは提案ツールと同様に，MVC デザインをベースに設計された Web アプリケーションの実装から NM を抽出している．しかし，縮退運転時の振舞い検証は考慮していない．

Ajax 技術の登場は Web ページ内での状態変化を可能にした．そこで，Ajax Web アプリケーションのための状態ベーステスティング・検証手法が提案されている[8, 9, 10, 11]. これらは，クライアントサイドの振舞いを解析している点で，我々の手法とは異なる．

6. おわりに

保守担当者は機能縮退を行う際に，ナビゲーション要求が満たされていることを検証しなければならない. しかし，近年の Web アプリケーションの動的性のために，この要求を人手で検証することは難しい．既存手法では自動検証に実装コードを入力とするため，縮退実装が必要であり，保守コストの増大につながる．

そこで，本論文で我々は MVC Web アプリケーションから縮退計画に応じて縮退された NM を抽出・検証するツールを提案した．本研究の目的は，縮退実装なく縮退運転時のナビゲーション構造の検証を可能にすることであった．評価実験では，本ツールが実用的な実行時間で，不適切な縮退運転における要求違反を検出できることを確認した．したがって，提案ツールはナビゲーション要求を考慮した縮退計画の策定を支援することができると考えられる．

今後の課題として，反例として出力された実行列が実在することを確認するための動的解析手法との併用が挙げられる．さらには，クライアントサイドの振舞い検証手法との連携が考えられる．また，手法の一般性を評価するために，さらなるケーススタディを実施する予定である．

謝辞

本研究にあたり指導教授を務めてくださった国立情報学研究所 本位田 真一 教授, そして高橋 竜一 特任助教に感謝申し上げます．研究期間中には，研究のみならず，トップエスイープロジェクトにおいて多大なるご指導ご鞭撻を賜りました．頂きました数々のご支援に対して，ここに厚くお礼申し上げます.

本位田研究室の皆様には，本研究に際し，多くの貴重な意見を頂きました．特に東京大学 前澤 悠太さんはアドバイザーとして私の研究を強く支えてくださいました．心より感謝申し上げます.

（日工専）で担当教授を務めてくださった高橋 宜孝 教授には，研究内容や日々の生活で丁寧かつ熱心なご指導を賜りました．ここに感謝の意を表します.

そして，（イ交通）（開発セ）氏家隆二氏をはじめ，（イ交通）の皆様にお礼申し上げます．皆様のご協力により，非常に価値のある経験ができました．心より感謝致します.

参考文献

1) F. Ricca, et al., “Analysis and Testing of Web Applications,” in *Proc.*

[10] http://struts.apache.org/

[11] http://spinroot.com/

ICSE'01, pp.25-34, 2001
2) M. Han, et al., “Modeling and Verification of Adaptive Navigation in Web Applications,” in *Proc. ICWE'06*, pp.329-336, 2006
3) A. Kubo, et al., “Automatic Extraction and Verification of Page Transitions in a Web Application,” in *Proc. APSEC'07*, pp.350-357, 2007
4) T. A. Bennett, et al., “Bridging the data integration gap: from theory to implementation,” in *ACM SIGSOFT SEN36(4)*, pp.1-8, 2011
5) J. Nielsen, “User Interface directions for the web,” in *CACM42(1)*, pp.65-72, 1999
6) V. Hollink, et al., “Navigation behavior models for link structure optimization,” in *Proc. UMUAI'07*, pp.339-377, 2007
7) A. Ghabi, et al., “Code Patterns for Automatically Validating Requirements-to-Code Traces,” in *Proc. ASE'12*, pp.200-209, 2012
8) A. Marchetto, et al., “State-Based Testing of AjaxWeb Applications,” in *Proc. ICST'08*, pp.121-130, 2008
9) A. Mesbah, et al., “Invariant-Based Automatic Testing of AJAX User Interfaces,” in *Proc. ICSE'09*, pp.210-220, 2009
10) Y. Maezawa, et al., “Extracting interaction-based stateful behavior in rich internet applications,” in *Proc. CSMR'12*, pp.423-428, 2012
11) Y. Maezawa, et al., “Automated Verification of Pattern-Based Interaction Invariants in Ajax Applications,” in *Proc. ASE'13*, pp.158-168, 2013
12) 今関雄人，高田眞吾，土居範久，“ソフトウェアの動的モデルに着目したラウンドトリップエンジニアリングの支援”，情報処理学会論文誌 Vol.49 No.7，2008