Web行動リプレイシステムに基づくWebアプリケーション動作検証システムとその応用

Performance Testing of Web Applications based on Web Usage Mining and its Applications

柿元宏晃*1 Hiroaki KAKIMOTO　佐野博之*1 Hiroyuki SANO　大囿忠親*1 Tadachika OZONO　新谷虎松*1 Toramatsu SHINTANI

*1名古屋工業大学大学院工学研究科情報工学専攻
Dept. of Computer Science and Engineering, Graduate School of Engineering, Nagoya Institute of Technology

Conventional studies about inspection of the Web applications have focused on a penetration test and optimization of the page structure. This paper focuses on regression test of Web applications towards to a promotion of efficiency of movement inspection and testing. We make use of an automatic approach to generate an instruction set by recording interface usage in a Web application. Our automatic approach reduces time and effort for manual creation of instruction sets. This paper sheds light on user browsing behaviors that can be automatically generated and introduces a method to generate them.

1. はじめに

本稿では，Web アプリケーションのテストおよび動作検証の支援を目的とした，Web アプリケーションの動作検証システムの提案とその実現方法について述べる．

Web アプリケーションの検証に関わる研究はこれまで，ペネトレーションテスト [Jonathan 07] やページ構造の最適化 [Frank 02] を目的としたものが行われてきた．本研究では，Web アプリケーションのリグレッションテストに焦点を当て，動作検証およびテストの効率化を目的としている．

現在，Web アプリケーションの動作検証は，検証に必要な Web アプリケーション上での行動 (以降，テストシナリオと呼ぶ) を人手で試みる手動テスト，または動作検証を行うテストツールによってテストシナリオを自動的に行う自動テストによって行われている．自動テストでは，繰り返し行うテストシナリオを自動的に行うプログラム (以降，命令セットと呼ぶ) としてテストツールに与える．また，自動テストのためには，テストシナリオに沿った命令セットを適切に記述する必要がある．複雑な Web アプリケーションでは，これらの命令セットを記述することに手間がかかる点が課題である．

本研究では，システムがユーザの行動を記録して自動テストに用いる命令セットを生成することで， ユーザが記述する手間が省き，効率的なテストおよび動作検証が可能にする．命令セットは，ブラウザ上で発生するイベントを強制的に発生させることで，ユーザの行動を擬似的に再現する．実現のためには，ユーザの Web ブラウジング時の行動を記録し，得られたイベント列から命令セットを生成，そしてユーザが行った行動をブラウザ上で再現することが必要である．また，動作検証を様々な環境で行うことを考慮し，Web ページのレンダリング結果による表示の違いやクロスブラウザに対応した行動の再現を行う必要がある．本稿では，命令セットの自動生成が可能なブラウジング時の行動を明らかにし，動作環境に影響を受けにくい行動の取得，再現方法を示す．また，生成した命令セットの応用について述べる．

連絡先: 柿元宏晃，名古屋工業大学大学院 工学研究科 情報工学専攻，〒 466-8555 愛知県名古屋市昭和区御器所町，Tel:052-733-6550，Fax:052-735-5584，E-Mail:kak@toralab.ics.nitech.ac.jp

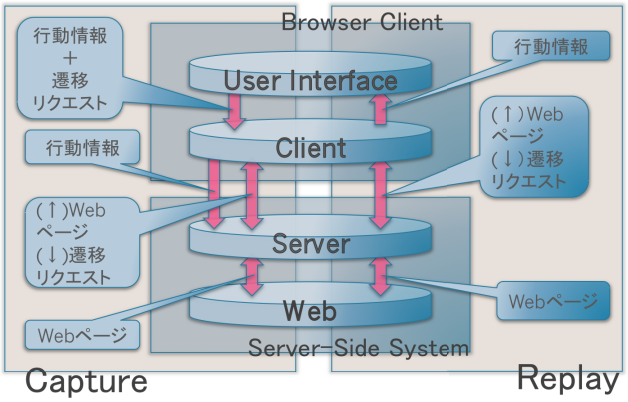

図 1: Web 行動リプレイシステム

2. Web 行動リプレイシステム

Web ブラウジング時のユーザの行動を記録し，ユーザの行動を再現するシステムを Web 行動リプレイシステムと呼ぶ．本システムは，ユーザのマウスの動きも再生可能である．

Web 行動リプレイシステムは Capture 部と Replay 部から成る（図 1）．Capture 部，Replay 部ともに，Web プロキシが Web エージェントを付加した Web ページを返す．この Web エージェントは JavaScript で記述されている．Capture 時の Web エージェントは主に，付加された Web ページ上で行われた行動情報の取得を行う．また，取得した行動情報や遷移リクエストを一定時間毎，ページの unload イベント発生時にサーバへ送信する．サーバ側ではそれらを記録し，遷移リクエストに対しては新たに Web エージェントを付加した Web ページを返す．Replay 時の Web エージェントは，実行スクリプトと Capture 時に取得した行動情報から成る．実行スクリプトとは，行動情報を時系列に再現するための関数群である．遷移リクエストに対しては Capture 時と同様に，Web プロキシが Web ページを取得し，Web エージェントを付加して返す．

2.1 Web ブラウザ上での行動記録と再現

Web ブラウザの実装や規格上の都合により，表 1 に示される行動は記録や再現に制限がある．

ファイルのアップロードは，type 属性が file である input タグの value 属性を監視することで観測できる．さらに，Web プロキシとして実装している Web 行動リプレイシステムでは，アップロードされた時間とそのファイルまで取得することが可能である．しかし，ファイルをアップロードする行動を再現することはセキュリティ面での制限により不可能である．

また，Web ブラウザの HTML 描画領域外で行われる操作は，プラグインや，OS レベルでの行動追跡によってしか取得・再現を行うことはできない．Web ブラウザのプラグインを利用しようとすると，クロスブラウザでの Web リプレイシステムは実装できず，また，OS レベルでの行動追跡では HTML の構文解析を行うことができないという制約をそれぞれ持つ．そのため，汎用性の高い Web 行動リプレイシステムを実装しようとする場合，描画領域内に限定して操作を取得するのが妥当であると考えられる．

ウィンドウサイズの変更も，HTML 描画領域外で行われる操作ではあるが，JavaScript によってウィンドウサイズを取得することができるため，行動の取得は可能である．しかし，取得できるウィンドウサイズは Web ブラウザごとに仕様が異なり，ずれが生じる可能性がある．ここでいうずれとは，ブラウザのツールバーやステータスバーなどのサイズの影響であったり，ウィンドウの内側のサイズ，外側のサイズの差によって発生するサイズの違いである．

本稿で述べる手法では命令セットを生成するために，Web アプリケーションのテストシナリオを 1 度人手で試み，発生したイベントを Web 行動リプレイシステムによって記録する．ファイルのアップロードや描画領域外での操作など，再現することが不可能な行動の取得は行わない．描画領域外で行われる操作は，ページの保存・印刷，ブックマーク，ページを進む・戻る，リロードが主なものであり，Web アプリケーションの操作に直接関わる機能でないため，行動の取得，再現を行う必要性は低いと考えられる．ウィンドウサイズはオブジェクトの位置に大きく影響を与える要素であるが，本手法では後に述べるように，オブジェクトの相対位置を基準にした行動の記録を行い，ウィンドウサイズのずれの影響を軽減しているため，高い精度でサイズを取得する必要はあまりない．その他，表 2 に示すイベントを取得し行動の再現を行う．

3. 行動記録からのテストケース生成

3.1 関連システム

本手法のように，Web 行動リプレイシステムに基づいて，行動の記録・再現を行うシステムの一つに，デスクトップアプリケーションのロギングツールがある．イベントの発生した位置を OS から取得し，記録・再現を行うというアプローチである．Web 上ではレイアウト上部オブジェクトがレイアウト下部のオブジェクトを押し出すという表示規則があるため，サイズが表示ごとに不定のオブジェクト（広告など）や，インターフェースの部分的な変更によって下位のオブジェクトの位置も不定になることがある．取得したイベント発生位置を絶対位置として取得している場合では，イベントの再現時にオブジェクトの位置とイベント発生位置のずれが生じ，正確に再現することができない点が課題である．

表 1: 再現が困難な行動

再現が困難（または不可能）な行動	行動の取得
ファイルのアップロード	可能
HTML 描画領域外で行われる操作	困難
ウィンドウサイズの変更	ブラウザ依存

表 2: 取得・再現するイベント

イベントハンドラ	トリガー
click	クリック
dblclick	ダブルクリック
mousedown	マウス押下
mousemove	マウス移動
mouseout	マウスが外れる
mouseover	マウスが重なる
mouseup	マウスが離される
scroll	スクロール
keydown	キー押下
keyup	キーが離される
resize	ウィンドウのリサイズ

また，Web 上のシステムによってイベントの記録を行っている既存のシステムも存在する．これらのシステムではリンクやボタンなど，粒度の粗い行動のみに絞ってイベントの監視を行い，それらに発生したイベント列を命令セットとして記録している．Web 技術が発展した現在の Web アプリケーションでは，マウスカーソルの軌跡，mousedown，mouseup に連動するアクション（オブジェクトのドラッグ&ドロップや軌跡による描画など）や mouseover に連動するアクション（プルダウンメニューなど）が使われているものがある．既存のシステムのように，リンクやボタンによって発生するイベントを監視するだけでは，これら複雑なアクションを記録することはできないという課題がある．

3.2 ユーザの環境に影響を受けにくいイベント発生座標の取得

本手法では，Web 上で動作するシステムによってイベントの記録し，DOM 構造に基づく行動情報の取得を行うため，オブジェクト位置の変動や，ユーザの環境による影響に柔軟に対応することが可能である．

Web ページ上でマウスカーソルによるイベントが発生した場合，Web ページの左上を基準とした座標としてイベントの発生位置を取得する．本研究ではこの座標をイベント絶対座標と呼ぶ．得られた座標情報をそのまま再現時に利用した場合，前述のようにサイズの可変なオブジェクトに対応することができなくなる．

そこで，イベント絶対座標を，イベントが発生した座標直下のオブジェクトを基準として表現する (図 2)．イベントの絶対座標直下のオブジェクトをイベント直下オブジェクト，イベント直下オブジェクトの左上座標とイベントの絶対座標との差をイベント相対座標と呼ぶ．イベント絶対座標は，イベント直下オブジェクトの左上座標とイベント相対座標の和で表すことができる．本手法ではイベント直下オブジェクトの位置を座標の代わりに XPath で表現する．この表現により，再現時に可変長のオブジェクトによってイベント直下オブジェクトの位置が

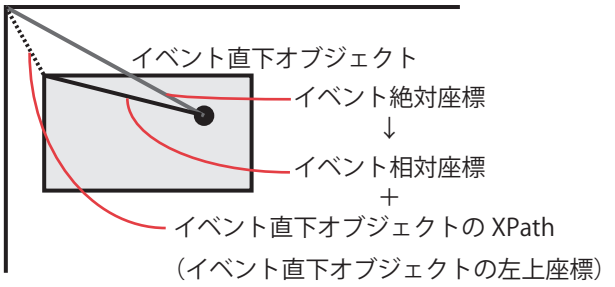

図 2: DOM 構造に基づく行動情報取得

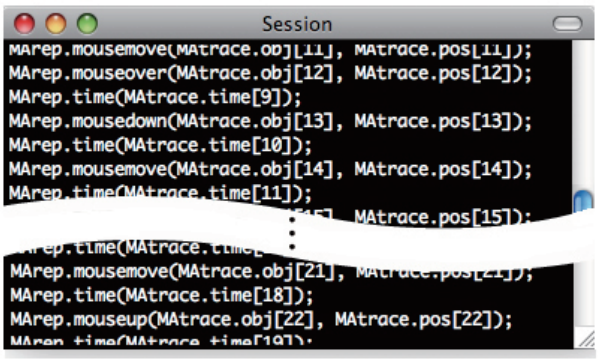

図 3: 命令セットの一部

変動した場合でも，XPath に変更がなければ，取得時と同様に再現することが可能になる．

3.3 細粒度の行動記録を用いた行動分析

本研究では，行動情報の抽象化レベルをを粒度と呼ぶ．既存のシステムで生成される命令セットは粒度が粗く，「ページ X を読み込む」や「ボタン Y を押す」など，Web アプリケーションに対してアクションを起こす可能性の高いイベント列から構成される．マウスの移動に関わるイベント (mousemove,mouseover など) は他のイベントに比べ，Web アプリケーションに対してアクションを起こす可能性が低いため，これらの命令セットに含まれない．本システムでは，細粒度の行動情報として，テストシナリオの実施によって発生したマウスイベントをタグによらず取得するため，複雑な Web アプリケーションの操作にも対応可能である．

取得したイベントは一定時間ごとにまとめてサーバへ送信することで，通信回数を減らし，負荷を軽減している．また，取得するイベントの種類を再現に影響が出ない範囲で極力減らし，取得する情報の削減を行っている（表 3）．change，keypress などのイベントは keyup と keydown イベントから発生の有無を判断できる．そのため，サーバ上で，keyup や keydown の発生時に合わせて change や keypress が発生したかどうか判定し，行動の再現項目に追加する．

フォームの送信とリセットは，送信やリセットを引き起こすその他の要因（ボタンのクリックなど）とページ遷移リクエスト，innerHTML のクリアの情報から発生したと判断することができる．

3.4 テストケースの生成

本手法では，相対的な位置関係でイベントを発生させるための命令セットを，「イベント発生時間」「イベントタイプ」「イベント直下オブジェクトのパス」「イベント相対座標」から生成する．イベントを発生させる位置は命令セットの実行時にリアルタイムに決定する．前述のように，命令セットを実行する環境ごとに

表 3: 他のイベントで代替可能なイベント

イベントハンドラ	トリガー	代用
change	内容の変更	keyup，keydown
keypress	キーが押下中	keyup，keydown
reset	フォームリセット	リセットボタン
submit	フォーム送信	送信ボタン
unload	ページ遷移	リンクなど

ウィンドウサイズ，オブジェクトの表示位置に誤差が生じるため，実行時にリアルタイムに計算を行うことで，正確なイベント発生位置を決定することができる．そのため，生成する命令セットは，時系列に並んだ命令群だけでなく，命令セットの実行時にイベント直下オブジェクトのパスから座標を計算する関数を含んだものとなる．図 3 は，命令群を時系列に並べた命令セットの一部であり，オブジェクトのドラッグ&ドロップを行った場合のものである．MArep.mousemove，MArep.mouseover，MArep.mousedown，MArep.mouseup はそれぞれ，mousemove，mouseover，mousedown，mouseup イベントを発生させる関数である．MAtrace.obj はイベントの発生した位置直下のオブジェクトのパスを，イベント発生順に格納している．MAtrace.pos はイベントの発生位置を，その直下のオブジェクトからの相対位置として格納している．MArep.time はイベント間の遅延を発生させる関数である．MAtrace.time はイベント間の時間を格納した変数であり，MArep.time への入力として与えられる．

再現時のイベント絶対位置は，イベント直下オブジェクトのパス（MAtrace.obj）から計算した座標とイベント相対座標（MAtrace.pos）の和によって計算できる．この計算を時系列に沿ってイベントごとに行うことで，オブジェクトサイズの変更にも柔軟に対応し，Caputure 時と同様の行動を再現できる．

3.5 動作検証

Web アプリケーションに対してテストケースを実行する際，本システムは class='MAtrace.watch' というクラス属性を持つタグを監視する．このクラス属性をもつオブジェクトを検証オブジェクトと呼ぶ．Capture 時には，イベントの発生時に検証オブジェクトの innerHTML をイベント発生前と比較する．差分がある検証オブジェクトについては，イベントと同様に時刻と innerHTML を記録する．Replay 時にも同様に検証オブジェクトの innerHTML を観測し時刻とともに記録する．Replay が終了した時点で，Capture 時，Replay 時の検証オブジェクトの innerHTML の変化をそれぞれ時系列に表示する．

ユーザは予め，Web アプリケーション中の監視を行うタグに対し，class='MAtrace.watch' とクラス属性を付加する必要がある．

3.6 応用

本手法によって生成された命令セットは，Web アプリケーションのリグレッションテストのテストケースとしてだけでなく，負荷テストにも応用ができる．Web アプリケーションにおける負荷テストで最も重要ものの 1 つとして，同時アクセ

表 4: Gmail での操作の比較

操作	本システム	Selenium IDE
受信メールの選択	可能	不可能
メールの新規作成	可能	可能
メール本文の入力	可能	不可能
ファイルの添付	不可能	不可能
メールの送信	可能	可能

ス，同時作業に対する耐久性評価が挙げられる．特に不特定多数が利用するような Web アプリケーションであれば，同時アクセスは必至であり，必要不可欠なテストと言える．本手法によって生成した命令セットは，単体で動作可能なスクリプトであり，Web ページ中に付加するだけで実行可能である．この命令セットを仮想的に複数用意したブラウザ上で実行することで，同時アクセス，同時作業を行う命令セットとして利用可能である．我々は以前に，Web ページ上へリアルタイムに一斉配信を行うシステムを実装した [Mukai 06][Nishi 05]．これは，番組表と呼ばれるコンテンツ配信時間管理システムにより，Web ページ上のコンテンツやスクリプトを任意の時間帯に，指定した Web ページ上へ配信を行うことが可能になる技術である．この配信技術によって，本稿で述べた手法による命令セットを同時刻に一斉配信することで，複数セッションによる同時作業のテストを行うことが可能である．

また，Web アプリケーションの操作ガイドとしても応用可能であると考えられる．本システムではマウスカーソルの軌跡を記録することもできるため，行動の再現時に仮想的なマウスカーソルを表示することで，Web アプリケーション上での操作を誘導，または代わりに行うといったサービスも考えられる．

4. 評価・考察

4.1 既存システムとの比較

本システムと既存の動作検証システム Selenium IDE[*1]において，Gmail[*2]での操作を行った場合の比較を行った．その結果を表に示す．ファイルの添付は 2.1 節で述べたように両システムとも再現は不可能であった．本システムと Selenium IDE で大きく違う点は，用いている行動記録の粒度の差である．本システムでは，ボタンだけでなく，対象に関わらずマウスクリックイベントも取得しているため，ボタン以外で，マウスのクリックイベントによって発生するような機能（受信メールの選択など）でも再現が可能であった．Gmail の本文入力欄はテキストエリアではなく，div と iframe の複合によって形成されている．このように複雑な構造をしているアプリケーションでも，細粒度の行動記録を用いることで，行動の追跡が可能になっていることが確認できた．

4.2 Web アプリケーションの仕様変更に対する柔軟性

命令セットを用いて Web アプリケーションのテストを行う場合，Web アプリケーションの仕様変更に柔軟に対応し，テストを実行することが重要である．本システムによって生成した命令セットは，第 3.2 節で述べたように，イベント絶対座標をイベント直下オブジェクトの XPath とイベント相対座標で表現する．この表現により，各オブジェクトのサイズの変更がある場合（例えば，ボタンのイメージやサイズが変更された場合）でも，パスを辿ることで正確な座標でイベントの再現を行うことが出来る．また，第 2 章で述べたように，イベント直下のオブジェクトパスとイベント相対座標はイベント発生時に計算する．既存のシステムでは，Web ページの読み込み時にタグを走査し，パスをそれぞれの属性に付加しておくといったアプローチが取られている．しかし，appendChild や insertBefore などの DOM 構造に影響するような，Web アプリケーションの動作が発生すると，DOM 構造が変化し，予め付加したパスがとの不一致が発生する．本手法では，リアルタイムにパスを計算することで，DOM 構造の変化後のパスとして取得し，正確に行動を再現できる．以上から，本システムは既存のシステムと比較し，オブジェクトサイズの変更や動的な DOM 構造の変化に対しての柔軟性が向上しているといえる．

5. おわりに

本稿では，Web 行動リプレイシステムに基づき，Web アプリケーション上で人手によって実施されたテストシナリオの行動記録から，Web アプリケーションのテストおよび動作検証に用いる命令セットを生成する手法を提案した．本手法では，発生したイベントの座標を，イベント直下オブジェクト（イベント発生座標直下に位置するオブジェクト）とその相対座標に分解し相対的に扱うことで，ウィンドウサイズが正確に取得できない場合や，広告などによって表示位置が変動してしまうような場合でもテストケースの再現を正確に行うことを可能にした．また，細粒度の行動情報を取得を行うことで複雑な構造のインターフェースを持つ Web アプリケーションの操作にも対応可能である．細粒度の行動情報の取得では最低限の情報のみを取得し，サーバ側でそれらの行動情報を補完することで情報量の削減を行い，通信への負担を軽減している．

本システムの課題として，Web アプリケーションのインターフェースなどの変更が静的に DOM 構造レベルで行われた場合，同一のオブジェクトを参照できなくなる点がある．これは，イベント直下オブジェクトの表現に XPath を用いていることに起因する．オブジェクトの属性値や innerHTML，ノード情報などを統合的に扱い，各ノードを比較する手法を今後の研究課題とする．

参考文献

[Jonathan 07] Jonathan Wilkins, “ProxMon Automating Web Application Penetration Testing”, International Superconductive Electronics Conferenc, Mar, 2007.

[Frank 02] Frank Heidmann, Jurgen Ziegler, “Web-SCORE - A Structured Method for Evaluating Web Applications”, Work With Display Units, May, 2002

[Mukai 06] 向井康人，大囿忠親，伊藤孝行，新谷虎松，“Web における情報配信の最適化のためのユーザ行動の分析手法の提案”，第 68 回情報処理学会全国大会論文集，Mar, 2006.

[Nishi 05] 西健太郎，大囿忠親，伊藤孝行，新谷虎松，“Web エージェント MiSpider に基づく広告制御機構の実装”，第 19 回人工知能学会全国大会論文集，June, 2005.

*1 https://addons.mozilla.org/ja/firefox/addon/2079
*2 https://mail.google.com/