Vol.2019-GN-106 No.19
Vol.2019-CDS-24 No.19
Vol.2019-DCC-21 No.19
2019/1/24

WePatch2: 無数のユーザによる半自動的な Web フォームの BADUI 改善システム

田島一樹[†1]　中村聡史[†1]

概要：Web 上の入力フォームにはユーザに全角形式やハイフンを含めた入力といった制約を求めるものが多く存在する．そういった制約がユーザに伝わりづらい場合，誤入力の原因となる使いづらいユーザインタフェース（BADUI）となってしまう．我々はこれまでユーザが Web フォームを利用する際のユーザの試行錯誤の過程から，適切な改善機能をシステムからユーザに推薦し，複数のユーザが BADUI を改善可能とする手法を提案してきた．しかし過去の提案システムは，正解の入力操作を考慮していないという点で推薦アルゴリズムが不適切であるという問題があった．そこで本稿では，提案システムの推薦アルゴリズムを改良し，その上で Web フォームの BADUI 検知精度について検証する．結果として，Web フォームを 7 人以上利用することで，システムが対象とする BADUI の 8 割以上の改善を期待できることが明らかとなった．

キーワード：ユーザビリティ，Web，BADUI，半自動改善

1. はじめに

インターネット技術の進歩により，現在の Web サービスではユーザの操作に合わせた動的な表現や幅広いデザイン等が実現可能となり，日々ユーザに恩恵をもたらしている．この Web サービスの中でも，Web フォームはユーザが最初にサービスとの接点となることが多く[1]，また，ユーザがフォームを通して送信する情報は，運営するサービスの今後の方向性の決定や，発展のうえで重要なものである．そのため，Web フォームはユーザが容易に目的を達成するために使いやすくわかりやすいことが重要である[2]．しかし，Web フォームにおいて，例えば図 1 のような，名前の入力項目の手がかりがひらがなであるにも関わらず，カタカナで入力しなければいけないためにユーザの誤入力を誘発してしまうユーザインタフェース（以下，UI と表記する）や，郵便番号の入力項目でハイフンを含めなければいけないにも関わらず，それを参考にするための手がかりが欠如している UI が存在する．このようなユーザが何かしらの目的を遂行するためにオブジェクトを操作する際，期待した動作と異なる動作をしてしまうようなユーザインタフェースは BADUI（Bad User Interface）[3]と呼ばれている．BADUI はユーザの失敗やストレスを招くだけでなく，サービス運営者の利益や信頼を損なう致命的なトラブルに繋がるため問題である．

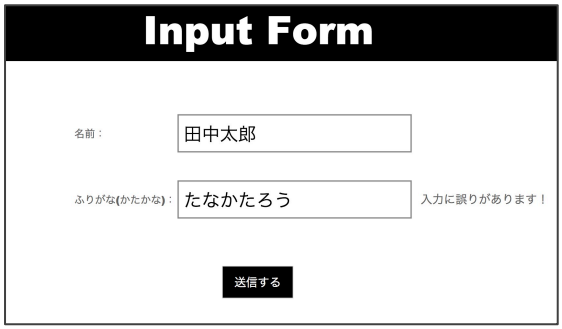

図 1　Web フォームの紛らわしい BADUI

こうした BADUI をなくすため，そもそもの開発段階で BADUI となってしまうことを防ぐ目的とした研究がこれまでに盛んに行われている．具体的には，複数の専門家がサイトのユーザビリティを評価し，UI の問題点を発見する手法や，Web フォームのユーザビリティを向上させるための汎用的なガイドラインが多く提案されている[4]．実際，Web フォームはユーザビリティの改善を行うことで，ユーザが Web フォームの一連の操作を完了する割合が 10〜40%増加する可能性があることが明らかになっている[2]．しかし，こうした研究が有用であるにも関わらず，サービス運営側の予算などのリソースの不足や，ユーザビリティを重視しない運営者の存在，経営者のメンテナンスへの意識の低さなどにより，改善されずに放置されることも珍しくない．

我々はこれまでの研究[5][6][7]において，放置された Web フォームの BADUI の問題を解決するため，無数のユーザの手によって Web フォームの BADUI を手動で改善する手法や，半自動的に改善可能とする手法を実現してきた．半自動的な手法では，入力ボックスにおけるユーザの失敗および修正行動から，内部のプログラムやデータベースが本来求めている入力形式を推定し，入力補正を行う変換フィルタの推薦を行うものであり，提案手法を実現するプロトタイプシステム WePatch 2 を開発した．しかし，これまでの研究には，変換フィルタの推薦アルゴリズムにおいて，ユーザの正解の入力行動を考慮できていなかったため，誤った変換フィルタを推薦してしまうことと，WePatch 2 の

†1 明治大学
Meiji Univ.

Vol.2019-GN-106 No.19
Vol.2019-CDS-24 No.19
Vol.2019-DCC-21 No.19
2019/1/24

改善精度やどの程度の人数を用いた際に有用なものであるかといった，システムの有用性が明らかになっていないという問題があった．そこで，本稿では WePatch 2 のフィルタ推薦アルゴリズムにおいて，ユーザの正解の修正行動を考慮可能とする改良を行なった上で，入力タスクを通した BADUI 検知精度に関する評価実験を行う．

2. Web フォームの BADUI

ここでは，Web フォームにおいてユーザが誤った文字形式で入力してしまう BADUI について記述する．

まず，Web フォームはユーザがデータを送信するための UI，データベースに値を受け渡すためのプログラム，入力されたデータを格納するためのデータベースの 3 種類のレイヤーに分けられる．この中でも内部のプログラムとデータベースが求める文字形式に関する情報は UI では見えない部分であるため，内部プログラムとデータベースが求める文字形式と異なった文字を入力してしまうような UI が BADUI となることが多い．具体的には，図 2 の UI 側の住所の入力項目では，入力例の文字列から半角入力も許容すると予測できる．しかし，その通りに入力すると，内部では全角入力のみ通すようなプログラムとなっているため，送信エラーが出力されてしまう例が挙げられる．これは，ユーザから半角入力が可能に見える UI と，ユーザからは見えない内部の全角入力を要求するプログラムやデータベースとの間にギャップが生じているためと言える．

そもそも Web フォームが BADUI となる原因の例としては，開発者が過去に自社で利用していたデータベースや，海外の会社で利用されていたデータベースを再利用して新たな Web フォームを作ろうとしたことで，その再利用によって開発者が UI やプログラム側でデータベースに合うようなシステム構築を行った結果，利用対象のユーザを混乱させてしまうものとなってしまったことも考えられる．また，単に開発者が UI 側の入力例においてカタカナで表記するところを誤ってひらがなで表記するといった誤字が原因で UI と内部プログラムの間にギャップが生じてしまう場合も考えられる．

ここで，これまでの膨大なユーザが内部のプログラムが求めている文字形式と，異なる文字形式を入力してしまいやすい BADUI は表 1 のように分類できる．なお，表 1 の先頭行の BADUI 分類はユーザが誤った文字形式を入力してしまう原因の種類を表しており，単一は特定の一つの入力ボックスが原因で誤った文字形式を入力してしまうもの，複数間は複数の入力ボックスがあるときにその関係性により誤った文字形式を入力してしまうものを表している．

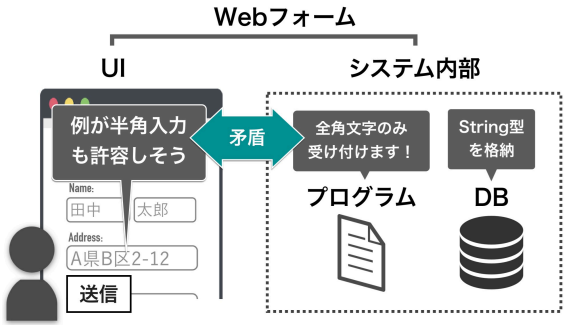

図 2　Web フォームの BADUI の原因

表 1　入力形式に関する BADUI 分類

BADUI 分類	単一の入力ボックス	複数の入力ボックス間
ミスリード	内部が求める文字形式と異なる表記	前後の入力形式に一貫性がない
手がかりの欠如	内部が求める文字形式を表記していない	入力の手がかりが不十分
慣習との乖離	大多数のユーザにとって慣れない入力形式を要求するもの	入力順番が慣習と異なるために紛らわしいもの
一苦労	特殊な入力形式を要求するもの	余計な入力項目を設けているもの

また，これまで収集された膨大な BADUI から，テキスト入力ボックスの制約については，下記の 5 つに分類できる．

- **文字種の制約**：ひらがな，カタカナ，数字，漢数字，アルファベットのいずれか，またはその組み合わせ
- **記号に関する制約**：ハイフン必須／不要，スラッシュ必須／不要，使用可能な記号の制約（例：アンダーバーのみ，ハイフンのみ，$#!?のみ）など
- **半角・全角**：半角文字のみ，全角文字のみ，両方を許容するなど
- **数値データに関する制約**：整数，実数，桁数に応じて先頭に 0 を埋める（例：8 月を 08 月と入力）など
- **文字数制限**：8 文字ちょうど，10 文字以内，140 字以内，1000 文字以内など

なお，過去に実装した WePatch 2 の改善可能な入力形式に関する BADUI 分類は「ミスリード（単一）」「ミスリード（複数）」「手がかりの欠如（単一）」「慣習との乖離（単一）」「一苦労（単一）」であり，改善可能な入力ボックスの制約は「文字種の制約」，「記号に関する制約」，「半角・全角」である．

Vol.2019-GN-106 No.19
Vol.2019-CDS-24 No.19
Vol.2019-DCC-21 No.19
2019/1/24

3. 関連研究

3.1 ユーザの操作支援

Web 上において，ユーザ同士で誤操作を支援する研究は数多く行われている[8]．代表的なものとしては，ユーザが Web サイトに操作時の注意事項が記載された付箋型のアノテーションを付与し，他ユーザとの共有を可能とするシステムが実現されている[9]．しかし，我々の過去の研究[6]において，付箋型アノテーションは Web フォームの BADUI において有効でないこと，また，自動で入力形式を修正する機能（以下，変換フィルタと記述する）は有効であることが明らかになっている．そのため，我々はこうした付箋型アノテーションでなく変換フィルタに着目し，無数のユーザによる BADUI 改善を目的とする．

Dong[10]らは，Web サイト運営者のユーザビリティ評価に対する専門知識とリソースの不足を解消するため，Web 拡張を用いて分かりづらいテキストの修正や，ツールチップの付与をユーザのコミュニティが行うことによって問題を改善可能とするシステムを実現している．これらの研究では主に Web 上のテキストを読むことを中心に行うページにおいて誤った操作をしやすい要素を改善対象としているが，本稿で実現するシステムは入力を間違えやすい Web フォームを改善対象としている点で異なる. Sergio [11]らは Web 上でユーザ同士のリアルタイムなコミュニケーションを実現しており，操作方法を質問及び回答可能であるが，こうしたサービスではその場，その時間に対話できるユーザが必要である．一方，我々の手法では Web サイト上に機能を残しておくことができるためリアルタイムにユーザを支援する必要がないという利点がある．

Web フォームはユーザにとって入力量が負担になるものが多いという問題点から，ユーザの入力ボックスに全ての単語を打ち込む前に，打ち込む単語を予測し，入力を支援する研究が行われている[12]．このような Web フォームにおけるユーザの入力を支援するというアプローチは，我々の研究と類似するものである．しかし，我々の研究はユーザの試行錯誤から，ユーザが誤って入力してしまいやすい BADUI を修正することを目的としている点がこれらの研究と大きく異なる．

3.2 自動 UI 分析

人手によるユーザビリティ評価にかかるリソースや，人手では発見することが難しいユーザビリティ上の問題点を発見するため，自動的な UI 分析や BADUI 検知を目的とした研究はこれまで盛んに行われている[13]. また，人手による評価での発見が難しい潜在的な問題を，ユーザのページ遷移やページ内移動，インタラクションを可視化することによって発見を促す研究が行われている[14][15]．さらに，画像認識技術や検出モデルの制作によって BADUI 検知を試みる研究も行われている[16]. 一方，クリックやスクロール等の操作ログを取得し，ユーザビリティの問題点を自動的に検出する研究もいくつか提案されており，ユーザのイベントデータをクライアント側で取得, データ処理を行い，自動的にユーザビリティの問題点の検知を可能とするシステムが開発されている[17][18]．このようなシステムでは，開発者はシステムから送られてきたログデータを参考に BADUI を改善可能である．

これらの研究において，ユーザのページ上の操作を分析することで BADUI を検知している点については我々の手法の目指すところと同じであるが，我々は入力操作から BADUI を検知することに焦点を当てている点において異なる．また，Web フォームの検知から改善まで行えるという点が異なる．

3.3 人手のユーザビリティ評価

これまでに Web サイトのユーザビリティの計測や問題点の特定を行うことを目的とした研究は多く行われている．代表的なユーザビリティ評価法の 1 つとしてヒューリスティック評価法[19]が挙げられる．これは 3〜5 人の専門家によってユーザビリティの問題点の評価と改善を繰り返し行うものである．また，より幅広い意味でのユーザビリティの評価尺度を用いた SUS（System Usability Scale）[20]では，低コストで信頼性の高いユーザビリティ評価を可能としている．さらに，本稿の改善対象である Web フォームのユーザビリティ評価を行うための FUS（Form Usability Scale）[21]というアンケートが提案されている．このアンケートによって Web フォームのユーザビリティを手軽に測定可能である．これらの Web ユーザビリティ評価を効率的に行うためのツール[22]が公開されており，開発者がユーザビリティ評価を行うための環境は整っていると言える．しかし，これらのユーザビリティの問題点を早期発見可能な手法が存在するにも関わらず，ユーザビリティを重視しない運営者の存在により, BADUI は放置されてしまうことが多い．また，これらは手軽に行える手法であるが，多くの人手を要することから同様の理由で評価が実施されるないことは少なくない．一方，我々の手法では自動で BADUI の検出から改善まで行うため，放置されてしまっている BADUI であっても改善可能という利点がある．

4. 評価実験

ここでは，WePatch 2 を改良した上で，入力タスクを通して実験協力者が適応した変換フィルタや入力データから，どの程度の精度で WePatch 2 が BADUI を改善可能であるかを明らかにする．

4.1 WePatch 2 の改良

過去の研究[7]で実装した WePatch 2 では BADUI の検知と変換フィルタの推薦をユーザの入力中に行なっていた．そのため，ユーザの失敗の入力と正解の入力の区別がつか

Vol.2019-GN-106 No.19
Vol.2019-CDS-24 No.19
Vol.2019-DCC-21 No.19
2019/1/24

ないことが原因で誤った変換フィルタを推薦してしまう問題があった．具体的には，全角入力が必須の電話番号の入力項目において，ユーザが「09012341234」,「090-1234-1234」,「０９０１２３４１２３４」の順番で試行錯誤した場合，本来「半角から全角に変換するフィルタ」が正解のところを，過去の WePatch 2 ではハイフンを追加した時点で「電話番号にハイフンを補完するフィルタを」推薦してしまっていた．そこで，ユーザが入力を完了し，次のページへの遷移が成功するタイミングで BADUI の検知とフィルタ推薦を行う．その改良によって，先述した試行錯誤例においても，試行錯誤の最後の入力形式を正解の入力形式，それ以外を失敗の入力形式と区別することが可能である．この場合，半角とハイフンを含む形式を失敗の入力，全角を正解の入力とし，「半角から全角へ変換するフィルタ」の条件に一致するため，正解の変換フィルタを推薦可能である．

なお，WePatch 2 の BADUI 検知と推薦プロセスと，ユーザに提示する推薦用の UI を図 3, 4 に示す. 図 4 の WePatch とタイトルが表記されている区画が推薦されたフィルタを適用するための UI である．その中でも真ん中より左のウィンドウから前ページの入力フォームを確認可能であり，右側で適用するフィルタの選択と送信を行うことが可能である．なお，推薦されたフィルタが表記されたボタンの上をユーザがマウスホバーすると，適応対象の入力ボックスがハイライトするため，ユーザはどのフィルタがどの入力ボックスに適応されているかを把握可能である．また，フィルタ適応後の入力ボックスは緑色に変色するため，適応されたことが容易に把握可能である．

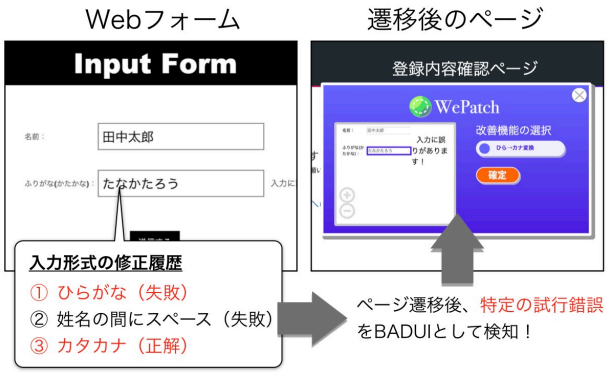

図 3 変換フィルタ推薦の様子

図 4 変換フィルタ推薦の様子

4.2 実験準備

実験の準備として，まず実験協力者 17 名を募集した．実験協力者は 10〜20 代の男性 12 名，女性 5 名であり，PC の使用頻度はほぼ毎日使用する実験協力者が 15 名，3 日に 1 回程度が 2 名であった．タッチタイプを習得している実験協力者は 1 名，半分程度タッチタイプ出来るのは 14 名，キーボードを見ながら入力するのは 2 名であった．

次に，入力タスクを行うための実験用フォームを用意した. この実験用 Web フォームは過去の実験[7]で用いた Web フォームの入力項目を 2〜5 項目に増やした点以外は同様のものである．入力項目を増やした理由としては，過去の実験で用いた Web フォームは入力項目が 2〜3 個という UI が単純すぎるものであり，協力者の BADUI における誤入力を誘発しづらい状況となってしまったためである．本実験では，この実験用 Web フォームのページを 70 ページ用意し, その内の 20 ページに BADUI が含まれるようにした．表 2 に用意した BADUI 一覧，図 5 に BADUI の一例を示す．

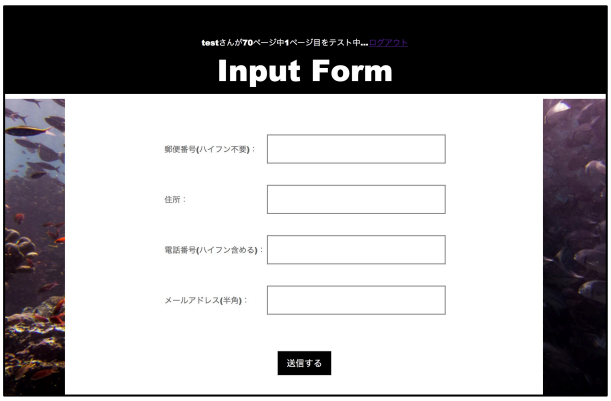

図 5 実験用 BADUI 例（表 2 の H）

4.3 実験手順

実験を行った手順について記述する．まず，入力準備として普段から利用している PC に WePatch2 をインストールしてもらった上で，実験中に WePatch2 以外の拡張機能はオフの状態にしてもらった．また，WePatch 2 の使い方を説明することで BADUI 改善方法について把握してもらった．さらに，ダミープロフィールの入力に慣れてもらうため，練習用の Web フォームでそれを 3 回入力してもらった．

入力タスクでは実験協力者にダミープロフィールを参考にしながら，提示される複数の Web フォームで入力するよう依頼した．その際，もし WePatch2 から変換フィルタを推薦された場合，適切だと思われるフィルタを付与してもらった．最後に，入力タスク終了後は実験協力者にシステムのフィードバックを得るためのアンケートに回答してもら

Vol.2019-GN-106 No.19
Vol.2019-CDS-24 No.19
Vol.2019-DCC-21 No.19
2019/1/24

った.

ここで, 実験を通して得られるデータを以下に記述する.

- **ユーザの入力データ** : ユーザが入力した文字列, また, 名前や郵便番号といった入力された項目名
- **推薦フィルタデータ** : WePatch2 が BADUI を検知し, その際に推薦された変換フィルタ名, また, それが推薦された入力項目名
- **適応済みフィルタデータ** : WePatch2 から推薦されたフィルタの中でもユーザによって適応が行われた変換フィルタ名, また, それが推薦された入力項目名
- **フィードバック** : **使いやすかった点 / 使いづらかった点についての自由記述)**

表 2　実験用 Web フォームに含めた BADUI

分類		説明
ミスリード 単一	A	ふりがなの手がかりはひらがな表記だが, カタカナ入力が必要
	B	ふりがなの手がかりはカタカナ表記だが, ひらがな入力が必要
	C	電話番号の入力例では半角スペースが必要に見えるが, 必要ない
	D	郵便番号の入力例はハイフンが必要な表記だが, ラベルの隣にはハイフン不要と書いてあり, 実際はハイフン不要
ミスリード 複数間	E	住所のみ全角入力だが他は全て半角入力
	F	年齢のみ全角入力だが他は全て半角入力
	G	誕生日のみ全角入力だが他は全て半角入力
	H	郵便番号はハイフン必要ではないが, 電話番号は必要
手がかりの欠如 単一	I	フリガナをカタカナで入力する必要があるが, 入力例やラベルといった手がかりが欠如している
	J	郵便番号でハイフン必要だが表記がない
	K	生年月日でスラッシュ必要だが表記がない
	L	電話番号でハイフン必要だが表記がない
慣習との乖離 単一	M	姓名の間にスペースが必要
	N	誕生日がスラッシュでなくハイフンで区切らなければいけない
	O	電話番号を国番号から記述する必要がある
	P	郵便番号をスラッシュ区切りで記述しなければいけない
一苦労 単一	Q	名前で半角カタカナ入力が必要
	R	名前英字を全角英字入力が必要
	S	メールアドレスで全角英字入力が必要
	T	住所フリガナで半角カタカナ入力が必要

4.4 評価指標

実験ではクラスは2つあり, BADUIであるクラス（正例）と非BADUI（負例）である. この2クラスにおける評価尺度として適合率（Precsion）と再現率（Recall）を用いる. ここで, クラスCiについての適合率と再現率は以下のように算出される. 以下のTBは入力(テキスト)ボックスを表す.

$$\mathrm{Precision}(C_i) = \frac{\text{正しく}C_i\text{が改善された}TB\text{数}}{C_i\text{として改善された}TB\text{数}}$$

$$\mathrm{Recall}(C_i) = \frac{\text{正しく}C_i\text{が改善された}TB\text{数}}{C_i\text{に属する}TB\text{数}}$$

正しく改善された基準は, こちらがあらかじめ用意した表 2 の BADUI 入力ボックスに対し, あらかじめ定義した正解の変換フィルタが付与されている場合とした. 例えば, 「ふりがなの手がかりはひらがな表記だがカタカナ入力が必要」の BADUI であれば, ユーザはカタカナ形式で入力しなければいけないところを, 誤ってひらがなで入力してしまうと考えられる. この場合の正解フィルタは「ひらがなからカタカナへ変換」のフィルタを正解とする. また, BADUI の入力ボックスに正解フィルタ以外のフィルタが付与されている場合も Web フォームを改悪してしまうフィルタでない限り正解とした. ここで, 入力タスクにおいて, システムが BADUI を正しく検知したにも関わらず, ユーザが推薦されたフィルタは BADUI を改善できないと判断した場合や, フィルタの適応忘れによって正解のフィルタの適応漏れが起きてしまう可能性が考えられる. そのため, 上記の改善された BADUI についての適合率と再現率とは別に, ユーザが関与しない部分である, システムが検知した BADUI についての適合率と再現率も算出する. システムが検知した BADUI についての適合率と再現率は以下のように算出する.

$$\mathrm{Precision}(C_i) = \frac{\text{正しく}C_i\text{と検知された}TB\text{数}}{C_i\text{として検知された}TB\text{数}}$$

$$\mathrm{Recall}(C_i) = \frac{\text{正しく}C_i\text{と検知された}TB\text{数}}{C_i\text{に属する}TB\text{数}}$$

5 章以降では, システムから推薦された変換フィルタをユーザが改善を行うことを半自動改善, また, システムがユーザの試行錯誤情報から特定の BADUI であると検知することを自動検知と呼ぶことにする. なお, ユーザの試行錯誤情報をサーバに送信すると抵抗を感じるユーザが多いと考えられるため, 自動検知はあくまで理想的な仕組みとしてのものを意味している.

Vol.2019-GN-106 No.19
Vol.2019-CDS-24 No.19
Vol.2019-DCC-21 No.19
2019/1/24

5. 結果

用意した BADUI の妥当性を検証するため，実験協力者の誤った操作に着目したものを表 3 に示す．表 3 の 1 行目の BADUI 分類とアルファベットは表 2 に示した BADUI と対応しており，エラー回数はユーザが誤った入力を行った平均回数，誤答率はユーザが誤った入力を 1 回以上した平均回数を表している．まず，表 3 の誤答率を全て平均すると 49.11%となり，用意した BADUI のページにおいて実験協力者のほぼ半分が間違えた入力をしたことがわかる．そのため，本実験では半分程度のユーザが誤った操作をしてしまうような適切な BADUI を設定できていたと言える．次に，自動検知，または半自動改善が成功していたかどうかについて，各 BADUI 分類についてまとめたグラフを図 6 に示す．図 6 の横軸は各 BADUI 分類，縦軸は BADUI の再現率を表している．また，青色のグラフは半自動改善，オレンジ色のグラフは自動検知が適切に行われた BADUI の再現率の平均を表している．表 3 と図 6 より，BADUI 分類の中でも誤答率が高かった「ミスリード（単一）」と「慣習との乖離(単一)」の BADUI 改善または検知はしやすく，一方で，誤答率の低かった「ミスリード（複数）」の BADUI 改善または検知は難しいことがわかる．

表 3　各 BADUI におけるユーザの入力試行錯誤データ

分類		エラー回数	平均	誤答率(%)	平均(%)
ミスリード(単一)	A	1.00	**1.43**	41.2	**58.8**
	B	1.00		58.8	
	C	2.71		100.0	
	D	1.00		35.3	
手がかりの欠如(単一)	E	0.00	**1.19**	0.0	**35.3**
	F	1.00		29.4	
	G	1.57		41.2	
	H	1.00		35.3	
慣習との乖離(単一)	I	1.00	**7.91**	82.4	**66.2**
	J	3.25		47.1	
	K	26.38		94.1	
	L	1.00		41.2	
一苦労(単一)	M	1.50	**2.32**	11.8	**51.5**
	N	3.81		94.1	
	O	2.64		64.7	
	P	1.33		35.3	
ミスリード(複数間)	Q	1.00	**1.07**	64.7	**33.8**
	R	1.29		41.2	
	S	1.00		5.9	
	T	1.00		23.5	

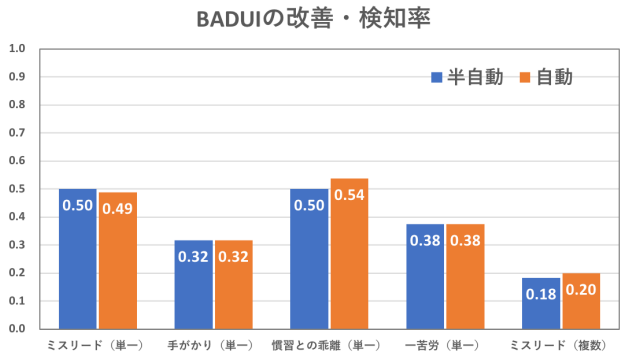

図 6　BADUI 分類ごとの改善・検知率

次に，半自動改善を行った BADUI が正解かどうかを，4.4 節で記述した基準に則り人手で分類した．その結果として得られた人数ごとの再現率の遷移を表すグラフを図 7，8 に示す．横軸は実験協力者の人数であり，縦軸は実験協力者数ごとにその人数の全ての組み合わせから得られる再現率を平均化したものを表している．

図 7，8 より，1 人が利用するときの再現率は 4 割程度であるが，7 人が利用した時点で 8 割程度，16 人時点では 9

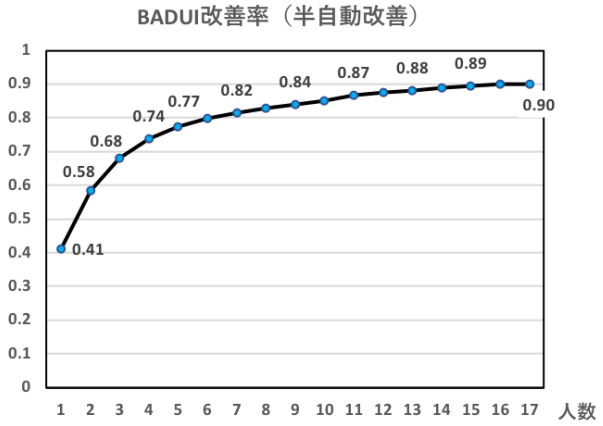

図 7　半自動改善による BADUI の再現率

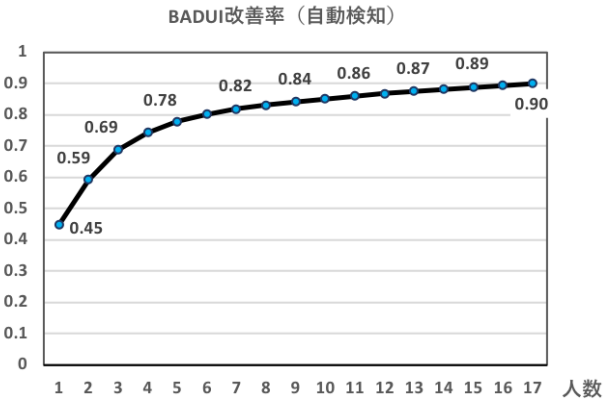

図 8　自動検知による BADUI の再現率

割に達していた．また，実験協力者全員による BADUI の適合率は半自動改善が 0.75，自動検知では 0.64 であり，半自動改善の方が，誤った改善の可能性は低いことが明らかになった．さらに，実験協力者全員により半自動改善または自動検知された BADUI はどちらも 20 件中 2 件であったが，1 件はそもそも 1 人も間違った形式で入力しない BADUI であったため，BADUI の選定が不適切であったことが原因であった．残り 1 件は誤入力が起きていたが，異なる形式が混在する文字列から 1 つの形式の文字列へのユーザの修正行動（中野区中野 4-15（半角と全角形式の混在）から中野区中野４−１５（全角形式のみ）へ修正等）をシステムが BADUI として検知出来ていないことが原因であった．

最後に，実験終了後に収集したシステムについてのフィードバックについて記述する．まず，システムの使いやすかった点，または分かりやすかった点の質問項目では，「自分が何の入力を間違えたのかを提示してくれるので，「ここを直してほしい」とすぐに選べる点は使いやすかったです」「ポインターを乗っけているときにどこのフォームについてか左の画面で示されるのが良かった」といったような，どの入力ボックスが BADUI として検知され，修正すると良いかを把握しやすかった点や，「ボタンを押すだけでいいので便利」「ワンクリックで選択できる点」といったような改善の手間が少ないことを示唆するような意見が得られた．システムの使いにくかった点，または分かりづらかった点の質問項目では，「左画面内の文字が小さく少し見えづらかった」「改善すべきページが縮小されていて文字が小さく，改行も多かったので見づらい」といったように前ページの Web フォームを再現した箇所が見えづらいといった意見が得られた．また，「どのように修正されるかを左でデモして欲しかった」「補完の前後での変化が確認できたらいいと思った」といったような，適応するフィルタによって，どのような文字修正が行われるかが把握しにくいといった意見が得られた．

6. 考察

実験結果より，システムは少数のユーザの手で多くの BADUI が改善可能であることが明らかになった．特に Web フォームの BADUI の中でも「ミスリード（単一）」と「慣習との乖離（単一）」の BADUI に対して有用であることが明らかになった．これらの BADUI では，ユーザが特に誤った入力をしてしまいやすいものであるため，そういった致命的な UI における誤操作を防ぎやすい点は有用であると考えられる．一方で，「ミスリード（複数）」の BADUI に対しては有用でないことが結果から明らかになった．これは，こういった一貫性のない BADUI は，ユーザは形式を使い分ける必要がある点で使いづらいが，誤って入力してしまうほどの要因になりづらく，BADUI の検知が起きづらい原因になったと考えられる．また，適合率の結果から，BADUI でない入力ボックスを誤って BADUI と検知され，変換フィルタが推薦されてしまったものが数件あったが，いずれの推薦された変換フィルタはユーザの誤入力を増やしてしまうものではなかったため，WePatch 2 がユーザビリティに悪影響を与える可能性は少ないと考えられる．

ここで，ユーザの手を介さずとも，自動的にシステムが変換フィルタを適用する手法も考えられるが，システムが自動的にデータを送信することはユーザにとって不安であると考えられる．また，図 6，7，8 から半自動改善と自動検知の結果に差があまり見られなかったため，半自動な改善であっても問題なく BADUI を改善できるものであり，これは自動検知手法におけるどんな情報が送られているのか心配であるというユーザの不安を減らすことが出来るものであるため，半自動 BADUI 改善手法が有用であると考えられる．

本章では表 1 の Web フォームの BADUI の 8 分類のうち，5 種類のみ扱って評価実験を行ったが，今後は複数の入力ボックス間の「手がかりの欠如」「慣習との乖離」「一苦労」の BADUI についても，改善機能を実現する必要があると考えられる．具体的には，複数の入力ボックスに対して住所 1，住所 2 しか手がかりがないために，どこで区切ればいいか分かりづらい BADUI では，正例のユーザ群の入力した文字数や入力形式以外の文字列に対してアラートを表示するといった支援方法が考えられる．また，今回用意した手法は，2 章で触れた文字種の制約や記号に関する制約，半角・全角に関する制約や，数値データに関する制約，文字数制限の全部をカバーできているわけではなかった．そこで今後は，こうした様々な入力について考慮しつつ，BADUI を改善していく予定である．

ここで，今回提案する手法は，多くのユーザに対して入力を監視されているのではという不安を抱かせるものとなっている．実際にはユーザが変換フィルタ付与ボタンを押してはじめて，そのフィルタに関する情報のみが送られるようになっている．しかし，なんらかの情報を送っているというのは，ユーザに対して積極的に使おうと思わせるには弱いものであるといえるため，ユーザに対して不安感を抱かせないものにするようシステムを改良する予定である．

最後に，主な WePatch 2 のメリットを以下に記述する．

- BADUI 改善のためのユーザにかかる負担がほとんどないこと
- ユーザビリティ専門家の知識が必要ないこと
- 放置された BADUI が改善可能になること
- 開発者が自動変換フィルタの適用された箇所を BADUI 改善の参考にすることが可能になること

Vol.2019-GN-106 No.19
Vol.2019-CDS-24 No.19
Vol.2019-DCC-21 No.19
2019/1/24

7. まとめと今後の展望

本稿では，WePatch 2 の BADUI 検知と変換フィルタ推薦アルゴリズムを改良することで, BADUI 改善精度の向上を行なった．また，システムの検知精度に関する評価実験を行なった．その結果，少数のユーザによる改善であっても 8 割以上の BADUI 改善が可能であることが明らかになった．今後の展開としては，本システムを通してユーザの手で変換フィルタが付与されることによって，膨大な数の BADUI ページを収集し，BADUI データセットを構築することで, 将来的には機械学習などの技術を用いた BADUI の自動推定を可能にすることを検討している．そういった自動推定により，ユーザの手を介さなくても BADUI が検知され，開発者に BADUI の情報が送信されることや，自動的に変換フィルタ等の改善機能が付与されることで，ユーザの操作を支援することが可能になると期待される．

謝辞 本稿の一部は JST ACCEL（グラント番号 JPMJACI1602）の助成を受けたものです.

参考文献

[1] Downes, P. K.. Creating a practice website. British Dental Journal. 2007, vol. 202, p. 597-604.
[2] Wroblewski, L.. Web Form Design: filling in the blanks. Rosenfeld Media, 2008, 226p.
[3] 中村聡史: 失敗から学ぶユーザインタフェース, 技術評論社 (2015).
[4] Bargas-Avila, J, A., Brenzikofer, O., Roth, S., Tuch, A, N., Orsini, S. and Opwis, K.. Simple but Crucial User Interfaces in the World Wide Web: Introducing 20 Guidelines for Usable Web Form Design. 2010.
[5] Tajima,K. and Nakamura, S.. WePatch: A System Enabling Users to Improve Bad User Interfaces on the Web. the 29th Australian Conference. 2017, p. 448-451.
[6] 田島一樹, 中村聡史. WePatch：ユーザの手による Web 上の BADUI 改善システムの評価. 情報処理学会研究報告ヒューマンコンピュータインタラクション（HCI）.vol. 174, no. 16, p. 2188-8760.
[7] 田島一樹, 中村聡史. WePatch 2：Web フォームにおける入力ミスを利用した半自動 BADUI 改善システム. 情報処理学会研究報告ヒューマンコンピュータインタラクション（HCI）. vol. 179, no. 3, p. 2188-8760.
[8] Hartson, H.R. and Castillo, J.C.. Remote evaluation for post-deployment usability improvement. In Proc. AVI ’98. 1998, p. 22-29.
[9] Reed, D. and John,S.. Web annotator. In Proc. SIGCSE ‘03. 2003, vol. 35, p. 386-390.
[10] Dong, T., Ackerman, M. S., Newman, M.W. and Paruthi, G.. So cialoverlays:Collectively making websites more usable. In Proc. INTERACT ‘13. 2013, vol. 8120, p. 280-297.
[11] Sergio,T,S. and Ackerman,I, I.. Enabling communication between users surfing the same web page. 2003.
[12] Trinh, T.D., Wetz, P., Do, B.L., Aryan, P.R., Kiesling, E. and Tjoa, A.M.. An Autocomplete Input Box for Semantic Annotation on the Web. In Proc. ISWC ’15. 2015, vol. 1456, p. 97-103.
[13] Bakaev, M., Mamysheva, T. and Gaedke, M.. Current trends in automating usability evaluation of websites: can you manage what you can’t measure?. In Proc. of IEEE 11t International Forum on Strategic Technology (IFOST). 2016, p. 510-514.
[14] Chi., E.H.. Improving Web Usability Through Visualization. IEEE Internet Computing. 2002, vol. 6, p. 64-71.
[15] Herder, E. and Weinreich, H.. Interactive Web Usage Mining with the Navigation Visualizer. In Proc. CHI ’05. 2005, p. 1451-1454.
[16] Vigo, M. and Harper, S.. Real-time detection of navigation problems on the World ‘Wild’ Web. International Journal of Human-Computer Studies. 2017, vol. 101, p. 1-9.
[17] Bakaev, M., Heil, S., Khvorostov, V. and Gaedke. M.. HCI Vision for Automated Analysis and Mining of Web User Interfaces. In Proc. ICWE ’18. 2018, vol. 10845, p. 136-144.
[18] Grigera, J., Garrido, A., Rivero, J. M. and Rossi, G.. Automatic detection of usability smells in web applications. International Journal of Human-Computer Studies. 2017, vol. 97, p. 129-148.
[19] Nielsen, J.. Heuristic evaluation of user interfaces. In Proc. of CHI '90. 1990, p. 249-256.
[20] Brooke, J.. SUS: a “quick and dirty” usability scale. Usability Evaluation in Industry. 1996, p. 189-194.
[21] Sebastien, O., Sc, M., Prof Dr., Opwis, K. and Aeberhard, Andreas.. FUS - Form Usability Scale Development of a Usability Measuring Tool for Online Forms. Unpublished master’s thesis. 2011.
[22] Chiew, T.K., and Salim, S. S.. Webuse: Website usability evaluation tool, Malaysian Journal of Computer Science. 2003, vol. 16, no. 1, p. 47-57.