話せば変わる GUI：ユーザの不満を反映した自然言語処理による GUI 自動修正システム

栗原佑真[†1]　　宮下芳明[†1]

ソフトウェアの使い方はユーザごとに様々であり状況によっても変化するため，それに応じて GUI も柔軟である必要があると考えられる．そこで，ソフトウェアが「気遣い」としてユーザの状況に合わせて主体的に GUI を変更する環境が必要であると考えた．本稿では，ユーザがソフトウェアを使いながらその GUI に対しての不満を声や文字で伝えることで，GUI をその場で自動修正するシステムを試作した．例えば，機能が選びづらいと伝えれば，各機能のボタンや選択リストを大きくする，機能が分かりづらいと伝えれば機能の説明を加えるなどである．システムの実装においては GPT-4 を用い，不満の理解，解決法の立案，ソフトウェアのスクリプトの修正といった一連の作業をすべて行わせている．

1. はじめに

ソフトウェアの使い方はユーザごとに様々であり，各ユーザでもタスクの種類や状況によって使い方は異なる．そのため，適した GUI デザインはユーザごとに異なり，状況によって変化すると考えられる．したがって，ソフトウェアの GUI は極めて柔軟な存在であるべきだと考えられる．ただし，ユーザが設定画面などで直接的に操作を行って，状況に合わせて GUI を変更可能にするだけでは不十分である．本来ユーザは GUI を使用して作業を行うはずであるのに，自ら GUI を変更するための作業を課してしまうからである．そこで，ユーザが GUI をカスタマイズするのではなく，ソフトウェアがユーザの状況に合わせて主体的に GUI を変更する環境が必要である．つまり，人間が状況を判断して他の人間に気を遣うように，ソフトウェアがユーザに対し「気遣い」をする環境が必要である．

しかし，現在のソフトウェアは「気遣い」をするどころかユーザ自身が GUI を容易に変更できるように設計されていることは少ない．GUI をユーザが容易に変更できるソフトウェアにおいても，ユーザは作業を中断し，理想的なデザインを自分で細かく設定する必要がある．

そこで本稿では，「気遣い」としてユーザの状況に合わせて主体的に GUI を変更するソフトウェアを作ることを目的とする．第一歩として，ユーザがソフトウェアを使いながらその GUI に対しての不満を自然言語で伝えることで，GUI をその場で自動修正するシステムを試作した．不満の例として，ボタンが選びづらい，機能の表示がわかりづらい，入力欄が小さすぎる，ウィンドウやタブ，リボンなどの場所を変えたい，などが挙げられる．自然言語を入力として GUI の修正を行うために，大規模言語モデルである GPT-4 を使用した．GPT-4 を使用することで，ユーザが自然言語で入力した不満をもとに，ソフトウェアのコードを自動で修正し，その場でソフトウェアに反映することができる．

†1 明治大学

2. 関連研究

2.1 ソフトウェアのあるべき姿に関する研究

西本[1]は創造活動に用いる道具のあるべき姿として，「創造活動のためのユニバーサルな道具」を挙げた．説明として，「可能な限り付随的要素の処理を道具が吸収し、人に余分な負荷を課さず、本質的要素の処理にのみ集中することを可能とする」ものと述べている．

2.2 GUI の変更，改善に関する研究

増田ら[2]はユーザが GUI に対して徹底的に手が加えられるような環境がなければならないと考え，マウスの選択操作によってアプリケーション内のレイアウトやビューのツリー階層を動的に変更できるシステムである DEVO を提案した．早川ら[3]は簡単な選択操作によって Web ページ上の特定のコンテンツのサイズを大きくしたり，背景色を変えたり，あるいは，非表示にするなどのカスタマイズを可能にするインタフェースである DOMinGO を提案した．山本ら[4]は情報探索の効率化を目的として，Web ページの UI について，視線停留判定を行い，ユーザの興味があるコンテンツを目立たせるようにパーソナライズするシステムを提案した．田島ら[5]は Web フォームにおける BADUI（Bad User Interfaces）の問題を解決するために，ユーザが入力したデータを分析し，入力エラーの発生頻度や入力時間などの指標を用いることで BADUI を特定し，改善するシステムである WePatch を提案した．

2.3 GPT を用いた自動化に関する研究

Hao ら[6]は GPT を用いて Android モバイルアプリケーションとのインタラクションを自動化するツールである DroidBot-GPT を提案した．Zhe ら[7]は GPT を用いてアンドロイドモバイルアプリの自動 GUI テストを行うシステムである GPTDroid を提案した．上記 2 つのシステムはどちらも，GUI ページ情報と利用可能なアクションを自然言語プロンプトに変換し，LLM（Large Language Model）にアクションの選択を求めることで自動化を行っている．

3. 提案システム

本稿の提案システムは，ソフトウェア使用中のユーザがGUIに対して感じた不満に対し，その場で自動修正を行うシステムである．GUIを修正する対象のソフトウェアは文書作成ソフトとし，Webアプリとしてブラウザ上で動作する．システムはJavaScriptとOpenAIのAPIを使用して実装し，GPT-4のモデルはGPT-4-0613を使用した．図1にシステムイメージを示す．

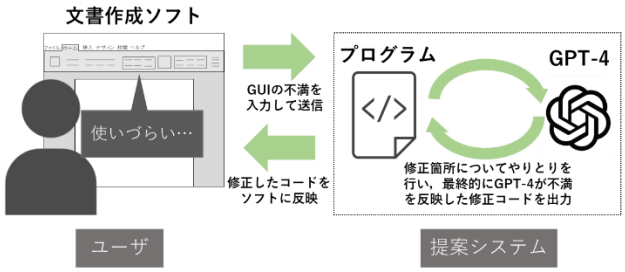

図 1　システムイメージ．ユーザはソフトを使いづらいと感じたら，提案システムに対して不満を送信する．提案システムはGPT-4を用いて修正コードを出力する．その後，修正コードをソフトに反映し，GUIの修正を行う．

3.1 不満の入力

入力手法はテキスト入力と音声入力である．入力フォームに音声もしくはキーボード操作で入力を行い，エンターキーを押してシステムに送信する．音声入力にはWeb Speech APIを使用している．GUIへの不満の入力例としては，「機能が選びづらい」，「複雑すぎる」，「機能が分かりづらい」，「テキストボックスが大きすぎる」，「ボタンを中央に並べたい」，などが挙げられ，指示内容が多少抽象的であっても問題ない．

3.2 GPT-4との通信とGUI変更

ユーザから不満が送信されると，プログラムはまずその不満を読み取り，コードを修正する適切なプロンプトを作成するように要求するプロンプトをGPT-4に送信する．具体的なプロンプトは「(ユーザの入力)。という不満が先ほど送ったコードのテキスト編集アプリに対してあります。この不満を解消するように修正したコードの出力をChatGPTにお願いしたいです。修正箇所を具体的にした適切なプロンプトを1つだけ出力してください。」とした．これによって，ユーザの不満をコードの修正を依頼する明確なプロンプトに変換することができる．なお，プログラムの修正を適切に行うために，ページ更新時に文書作成ソフトのHTML, CSSコードを前もってGPT-4に送信している．提案システムでは文書作成ソフトのGUIを修正することを目的としており，JavaScriptコードは別ファイルにし，GPT-4への送信は行っていない．

次に，GPT-4からプロンプトが出力され次第，プログラムは先ほどGPT-4で出力したプロンプトに基づきコードを修正するように依頼するプロンプトをGPT-4に送信する．具体的なプロンプトは,「ではそのプロンプトに基づいて私が送ったコードもしくは前回出力してもらったコードを修正して出力してください。条件が2つあります．①出力コードはhtmlエンティティを使わないで使えるコードにしてください。②改行もしないでください。」とした．プログラムの都合上，コードの記述に関する条件をプロンプトの後半部分で指定した．

プログラムからプロンプトが送信されると，GPT-4はユーザの不満を変換した適切なプロンプトをもとに，修正したHTMLコードを出力する．GPT-4からの出力が終わると，修正ボタンが表示され，ユーザが修正ボタンを押すことで修正したコードを文書作成ソフトに反映し，GUIの修正を行うことができる．

4. 対象ソフトウェアの概要とシステム使用例

4.1 対象ソフトウェアの概要

本稿ではGUIを修正する対象のソフトウェアを文書作成ソフトとした．GUI修正前の文書作成ソフトの全体図と細部図をそれぞれ図2, 3に示す．文書作成ソフト自体はリボンとテキストボックスで構成されていて，提案システムのGUIとして不満入力フォームが備わっている．リボンには保存，文字色変更，文字サイズ変更，下線挿入，太字変更，フォント変更，画像挿入，図形挿入の計8種類の機能が設けられていて，図3上で示すように，それぞれボタンやオプションが表示されている．不満入力フォームは，図2の右上のはてなマークを押すと，図3下に示すように表示され，入力フォームの右のマイクのボタンを押すと音声入力が可能になる．

図 2　文書作成ソフトの全体図．

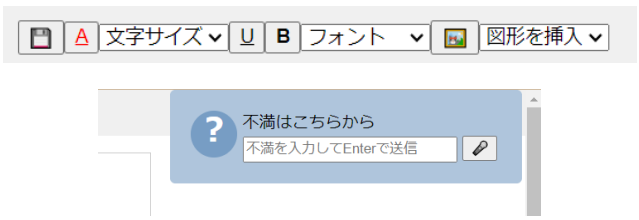

図 3　文書作成ソフトの細部図．上がリボン，下が不満入力フォームを示している．

4.2 システム使用例

4.2.1 機能が選びづらいと感じたとき

使用例の 1 つ目として，現状の文書作成ソフトは機能のボタンが小さすぎて，選択しづらいという不満を想定した．フォームに「機能が小さすぎて選びづらい」と入力し送信したところ，約 1 分後にコードの出力が終わり，修正ボタンが表示された．修正ボタンを押すと，図 4 に示すように機能のボタンが大きくなり，それに伴いリボンも太くなった．実際のコードでも，図 5 に示すようにボタンを示すタグ button と選択リストを示すタグ select に対し，フォントサイズを既定の 1.5 倍にするようにコードが追加されている．また，図では示していないが，リボンの余白を定める padding やテキストボックスの上の余白を定める margin-top が大きい値に変更されていた．この修正によって，ユーザは機能が選択しやすくなったと考えられる．

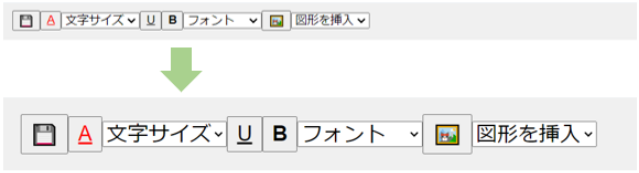

図 4　修正前と修正後のリボンの比較．機能のボタンが拡大され，選択がしやすくなった．

```
button {font-size: 1.5em; padding: 5px 10px;}
select {font-size: 1.5em;}
```

図 5　修正の際に追加されたコード．button と select タグに対し，フォントサイズが既定の 1.5 倍になるように定められた．

4.2.2 機能が多すぎると感じたとき

使用例の 2 つ目として，現状の文書作成ソフトは機能が多すぎるという不満を想定した．フォームに「機能が多すぎる」と入力し送信したところ，約 40 秒後にコードの出力が終わり，修正ボタンが表示された．修正ボタンを押すと，図 6 に示すように機能のボタンが少なくなり，シンプルになった．実際のコードでも，保存ボタンと太字ボタン，下線ボタン以外の機能の HTML 要素が削除されていた．この修正によって，余計だと思われる機能を減らし，ユーザは作業がしやすくなったと考えられる．

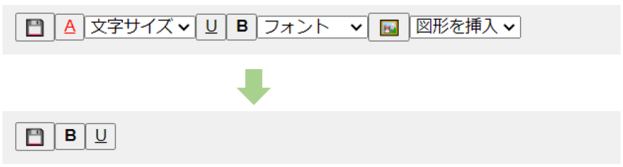

図 6　修正前と修正後のリボンの比較．機能が少なくなり，シンプルになった．

4.2.3 機能の表示が分かりづらいと感じたとき

使用例の 3 つ目として，現状の文書作成ソフトは機能がデフォルメされていて何の機能か分かりづらいという不満を想定した．フォームに「機能が分かりづらい」と入力し送信したところ，約 1 分 20 秒後にコードの出力が終わり，修正ボタンが表示された．修正ボタンを押すと，図 8 に示すように機能の説明がボタンに追加され，ボタンが横に広がった．実際のコードでも，各ボタンに対し図 9 に示すように span タグで機能の説明がテキストとして加えられていた．この修正によって，ユーザが機能を理解しやすくなったと考えられる．

図 7　修正前と修正後のリボンの比較．機能の説明がボタンに追加された．

```
<button onclick="downloadText()" title="保存">💾</button>
<button onclick="downloadText()" title="保存">💾<span>保存</span></button>
```

図 8　修正前と修正後の一部コードの比較．修正後は span タグで機能の説明がテキストとして加えられている．

5.　議論

提案システムでは，ユーザがテキストあるいは音声で文書作成ソフトの GUI への不満を入力することで，その不満を反映して GUI を自動で修正することができた．しかし以下のような課題も見つかった．

5.1　入力から反映までの遅延

現状，ユーザが不満を送信してからソフトに修正可能になるまで 1 分ほどの遅延が生じてしまう．これは GPT-4 に修正箇所に関係なく最初から最後まですべての修正コードを出力するように要求しているためである．この問題の解決策としては，GPT-4 には修正した箇所のみを出力してもらい，修正前のコードと比較したうえで修正箇所を元のコードに適切に挿入することが考えられる．ただ，修正箇所は毎回異なり，GPT-4 が修正箇所と指定する範囲も毎回異なるので，すべての修正において元のコードに適切にコードを挿入することは難しい．

5.2　修正の精度と一貫性

修正されるコードはすべて GPT-4 次第であるので，毎回ユーザの不満が完全に解消される適切な修正が施されわけではない．例えば，「ボタンが選びづらい」という不満を送信したところ，図 9 に示すようにボタンのサイズは変わらず左右に等間隔で配置された修正となった．この修正によってボタンが選びやすくなったとはあまり考えられない．

このような場合には，修正された GUI の不満をもう一度送信する必要がある．不満を解消するために送信しているのにも関わらず，修正された結果に対し再度不満を送信しなければならないのは，ユーザにとって負担である．

また，全く同じ不満の文章を送信しても，異なる結果が返ってくることがある．抽象的な不満ほど回答にランダム性が付与されてしまうが，それがどれくらい許容できるものなのかどうかは議論の余地がある．

図 9　修正が上手くいかなかった例．ボタンがリボンに均等に配置されただけで，ボタンを選びやすくはなっていない．

5.3 API の制限

GPT-4 の API はテキストをトークン（単語，句読点，特殊な記号など）に分割した個数であるトークン数を制限している[8]．提案システムで用いている GPT-4-0613 のモデルでは 2023 年 7 月現在で 8192 トークンが最大で，提案システムではコードの出力が行われるため制限に達しやすい．そのため，会話履歴をすべて保存することは出来ず，直近の出力以外は削除している．したがって，連続で不満を送る場合には，前の修正が適切に反映されないことがある．これも遅延の課題と同様に修正箇所のみを出力することができれば解決すると考えられる．

5.4 対象範囲

本稿では，試作した単純な文書作成ソフトにおける GUI の修正しか行っていないため，一般的に使用されている複雑なシステム環境において提案システムの手法が適用可能であるかは明らかではない．また，本稿では文書作成ソフトのみを対象にしていて，他のソフトウェアでの検証は行っていない．そのため文書作成ソフトよりも提案システムと相性がよいソフトウェアが存在する可能性が十分にある．

6. おわりに

本稿では，大規模言語モデルである GPT-4 を用いて，ユーザがソフトウェアを使いながらその GUI に対しての不満を自然言語で伝えることで，GUI をその場で自動修正するシステムを試作した．ソフトウェア使用中に感じた GUI への不満をユーザが送信することで，その場でシステムが GUI の修正を行い，不満を解消する新しい GUI を適用することができる．ただ，修正した GUI が反映されるまでの遅延や修正の精度と一貫性の問題，API の制限，対象とした範囲が狭いなどの課題も発見された．GPT-4 の発展や提案システムの改良によって，ユーザの GUI への不満をさらに適切に修正することができるようになるのではないかと考えられる．

しかし，提案システムではユーザが GUI の修正に際して不満を入力するという操作を行わなければならない．第 1 章で述べたように，ソフトウェアは「気遣い」としてユーザの状況に合わせて主体的に GUI を変更するようにならなければならない．このようなソフトウェアの実現を目指して今後の研究を行っていく．

参考文献

1) 西本一志: 創造活動のためのユニバーサルな道具とは，エンターテインメントコンピューティング 2006 予稿集, 7-8(2006).
2) 増田英孝, 笠原宏: アプリケーション実行時 GUI レイアウト変更機能, 情報処理学会論文誌, 35(9), 1794-1806 (1994).
3) 早川匠, 増井俊之: DOMinGO:対話的な Web ページ表示制御システム, 研究報告ヒューマンコンピュータインタラクション（HCI）, 2019-HCI-182(25), 1-5. (2019).
4) 山本ひかる, 満田成紀, 松延拓生, 福安直樹, 鯵坂恒夫: 情報探索の効率化を目的としたウェブ UI のパーソナライゼーション, 2019 年度 情報処理学会関西支部 支部大会 講演論文集(2019).
5) 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．研究報告ヒューマンコンピュータインタラクション（HCI）, 2017-HCI-172(23), 1-8 (2017).
6) Wen, H., Wang, H., Liu, J., Li, Y: DroidBot-GPT: GPT-powered UI Automation for Android, arXiv preprint arXiv:2304.07061 (2023).
7) Liu, Z., Chen, C., Wang, J., Chen, M., Wu, B., Che, X., Wang, D., Wang, Q: Chatting with GPT-3 for Zero-Shot Human-Like Mobile Automated GUI Testing, arXiv preprint arXiv:2305.09434. (2023).
8) OpenAI, (2023, 2 月), OpenAI API models, https://platform.openai.com/docs/models/overview, (参照日: 2023 年 7 月 2 日).