ヒューマンインタフェース 73－2
（1997. 7. 11）

GUI 設計の一貫性を評価するツール「GUI テスタ II」の提案

岡田英彦，旭敏之

NEC 関西 C&C 研究所

ユーザビリティ（使いやすさ，操作性）要素のうち，従来の操作履歴分析ツールでは評価が困難であった一貫性を評価するツール「GUI テスタ II」を提案する．本ツールを用いれば，部品のラベル，位置，サイズ，色など画面設計の一貫性を評価できる．本稿では，これらの項目の一貫性評価方法と，必要な GUI 設計データの取得方法を示す．GUI 部品検出・データ記録・入力イベント発行のサイクルによりアプリケーションの実行プログラムを網羅的に自動操作して GUI 設計データを取得する．このため他研究のツールと比べ，ソースプログラムが不要で開発言語に依存しないという優位性がある．今後はツールの開発および有効性検証とともに，シンプルさや見やすさの評価も実現していく．

GUI-tester II: a Tool for Computer-Aided GUI Design Consistency Evaluation

Hidehiko Okada, Toshiyuki Asahi

Kansai C&C Res. Labs., NEC Corp.
h-okada@obp.cl.nec.co.jp

GUI-tester II is a tool for evaluating consistency of GUI widget properties such as label, typeface, position, size and color. We propose methods for the consistency evaluation and GUI design data collection. For example, buttons with the same size (width or height) are grouped and listed so that a evaluator can easily find buttons with inconsistent sizes (i.e., buttons in minority size groups). GUI design data are collected by the cycle of 1) detecting widgets on the screen, 2) recording their property values, and 3) sending an input event to each operable widget to have the GUI application executable pop-up/close its dialog boxes. An advantage of GUI-tester II in comparison with another tool in a related work is that the data collection is independent of the program development languages. We will develop the tool and evaluate its effectiveness. In addition, we plan to develop evaluation functions of other usability attributes such as screen simplicity and readability.

1 はじめに

情報処理システムの急速な普及とユーザ層の拡大に伴い，ユーザビリティ（使いやすさ，操作性）の重要性が高まっている．使いやすいユーザインタフェース(UI)を開発するには，ユーザビリティ評価と UI 改善からなる反復的設計のサイクルが有効である[1]．このユーザビリティ評価を支援する様々な計算機ツールが研究されている[2-13]．

本研究ではこれまでに，ユーザの操作履歴をもとに GUI のユーザビリティを評価するツール「GUI テスタ」を開発した[1][14-17]．本ツールを用いてユーザの誤操作や操作時間，操作距離などを分析することで，操作手順のわかりやすさや画面レイアウトの適切さを評価できる．しかし，GUI テスタは「一貫性」や「美しさ」などの評価をカバーしていない[17]．このような項目の評価には，ユーザ行動（操作履歴）の分析ではなく UI の設計データ自体の分析が必要である．

そこで本稿では，GUI 設計データの分析により画面の一貫性を評価するツール「GUI テスタ II」を提案する．以下では，まず GUI テスタ II で評価するユーザビリティ項目を示し，各項目の評価方法を記述する．また，一貫性評価に必要な GUI 設計データの取得方法を示す．最後に，他研究の一貫性評価ツールと比較し，GUI テスタ II の特徴を明らかにする．

2 GUI の一貫性

ユーザインタフェースの一貫性とは，「同じものを同じように表現し，同じことを同じ方法で行えること」といえる．GUI 設計ガイドラインを参考に，GUI の一貫性を画面，操作，応答の 3 つにわけて次のようにとらえた．

画面の一貫性項目

用語，図・記号（アイコンなど），文字属性（フォント，サイズなど），レイアウト，色，形状，構成要素（ボタンの組合せなど）

操作の一貫性項目

方法，順序，方向，反応領域，機能割当て（デバイスイベントや部品の使い方）

応答の一貫性項目

メッセージ，状態変化，デフォルト，音

これらのうち，本稿では画面の一貫性をとりあげ，「図・記号」と「形状」を除く項目の評価の計算機支援手法を提案する（前記 2 項目を除く理由はまとめで述べる）．

3 GUI テスタ II

2章で挙げた各項目の評価手法と，評価に必要な GUI 画面設計データの取得方法を記述する．

3.1 評価手法

用語 (1)ラベルリスト

アプリケーションで使用されている部品のラベル（文字列）の一貫性を評価する．例えばボタンでは，「中止」「閉じる」「終了」といった異なるラベルを同じ機能のボタンにつけていないかどうかを調べる．このようなラベルの一貫性を評価するために，ラベルをすべて抽出し，同じ種類の部品ごとに（または部品全体について）リストで表示する（図 1）．部品種としてはボタン，メニュー，チェックボックスなどがある．評価者はリストを見てラベルのばらつきを発見する．

用語 (2)ウィンドウタイトル

ボタンやメニューの選択により新しいウィンドウが開く場合，選択した部品のラベルとウィンドウタイトルとの間の一貫性を保つべきである．例えば，「設定」－「パスワード」メニューを選択した場合，ポップアップするウィンドウのタイトルが「パスワードの登録」では「設定」と「登録」が一致せず，「パスワードの設定」とするのが適当である．このような部品ラベルとウィンド

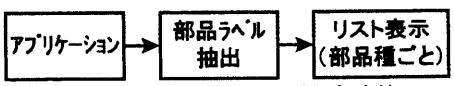

図1：部品ラベルの一貫性評価方法

[1] フリーソフトとして公開中．http://www.labs.nec.co.jp/freesoft/GUIT/guit.html.

ウタイトルの一貫性を評価するために，両者をアプリケーションから抽出して比較し，一貫性のない組合せを表示する（図 2）．ウィンドウタイトル中の単語が部品ラベルに含まれていれば○，含まれなければ×と判定する．

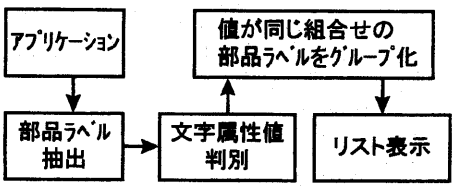

図3：文字属性の一貫性評価方法

文字属性

部品ラベルの文字属性の一貫性を評価する．例えば，ボタン全体を通してラベルのフォントなどが統一されているかどうかを調べる．文字属性の種類はフォント，サイズ，色，装飾（強調，斜体，下線）からなる．

文字属性の一貫性を評価するために，部品ラベルの文字属性の値を判別してその組合せを同じ種類の部品ごとに比較し，同じ組合せをもつ部品をグループ化してリスト表示する（図 3）．このリストから，属性値を変更すべき部品を発見する．

レイアウト (1)部品の位置

同じラベルをもつ部品は複数のウィンドウ間で統一して同じ相対位置に配置すべきである．このような位置の一貫性を評価するために，ウィンドウ上での部品の相対位置を算出しグラフ表示する（図 4）．

部品のウィンドウ上での相対位置は次のように求める．ウィンドウの縦／横サイズを x_win／y_win とし，部品の中心点のウィンドウ上での相対座標を(x_obj, y_obj)とする．このとき，部品の相対位置を次の Pos(x)と Pos(y)の値で表す．

$$Pos(x) = x\_obj / x\_win$$

$$Pos(y) = y\_obj / y\_win$$

Pos(x)と Pos(y)はどちらも 0～1 の実数値をとる関数で，Pos(x)／Pos(y)の値が大きいほど，その部品がウィンドウの右端／下端に近い位置にある．

部品の各ウィンドウ上での相対位置を正方形上の点で示すことで，その部品の位置のウィンドウ間でのばらつきがわかる．またグラフ上の各点がどのウィンドウ上での位置であるかを調べ，位置を変更すべき部品を発見する．

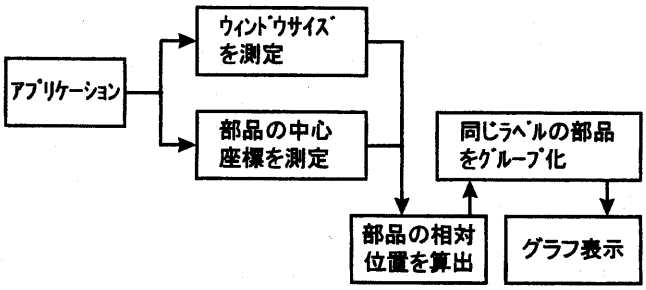

図4：部品の位置の一貫性評価方法

レイアウト (2)部品のサイズ

同じ種類の複数の部品について，サイズが統一されているかどうかを評価する．例えばボタンでは，複数の部品全体を通して縦のサイズを統一すべきである（横方向は，ラベルの長さに依存するので必ずしも一致しなくてもよいが，可能なものは一致させるべき）．一行エディタ（テキストフィールド）も同様である．

このようなサイズの一貫性を評価するために，部品の 2 対角点の座標からその部品の縦／横サイズを算出し，同じサイズの部品をグループ化してリスト表示する（図 5）．

このリストを見てサイズ間のばらつきを分析し，サイズを統一するために変更すべき部品を発見する．

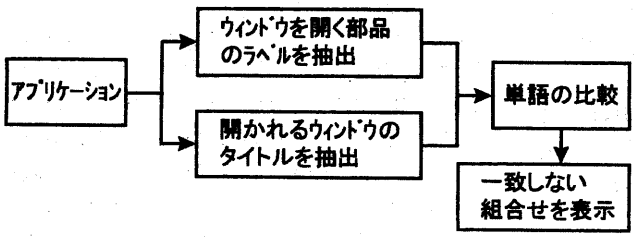

図2：ウィンドウのタイトルの一貫性評価方法

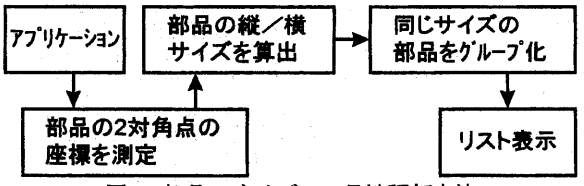

図5：部品のサイズの一貫性評価方法

レイアウト (3)部品間のスペース

隣り合う部品間のスペースの大きさも一貫性の評価要素の1つである．例えば，複数のウィンドウ上でボタンが横一列に並べられている場合，これらのボタン間のスペースを統一させる．

スペースの一貫性を評価するために，ウィンドウ上で縦または横に整列された複数の部品を抽出し，これらの部品の座標からスペースを算出する．このスペースが同じ値である部品対をグループ化しリスト表示する（サイズの場合と同様）．このリストを見てスペースのばらつきを調べ，位置を変更すべき部品を発見する．

色

同じ種類の複数の部品の色が統一されているかどうか評価するために，部品の色を判別し，同じ色の部品をグループ化してリスト表示する（サイズ，スペースの場合と同様）．このリストを見て色のばらつきを調べ，色を統一するために変更すべき部品を発見する．

構成要素 (1)ウィンドウ整理部品の組合せ

ウィンドウは最大／最小化やサイズ変更，移動，クローズなどの整理操作が可能であり，これらの操作の部品（ボタン，メニュー，枠）がつけられている．このような整理部品の付け方がウィンドウ間で統一されているかどうかを評価するために，それぞれのウィンドウの整理部品を抽出し，その組合せが等しいウィンドウをグループ化してリスト表示する（文字属性の場合と同様）．このリストを見て整理部品の付け方のばらつきを調べ，組合せを統一するために変更すべきウィンドウを発見する．

構成要素 (2)ウィンドウ上の部品の組合せ

複数のウィンドウ間で共通して使用されている部品の集合について，その組合せが統一されているかどうか評価する．例えば，「OK」「キャンセル」「ヘルプ」の3つのボタンを複数のウィンドウで共通して使用している場合に，この組合せと統一されていないウィンドウ（例えば「ヘルプ」ボタンを付けていないウィンドウ）の有無を調べる．このような部品の組合せの一貫性を評価するために，それぞれのウィンドウ上での部品の組合せを抽出し，組合せが等しいウィンドウをグループ化してリスト表示する（文字属性やウィンドウ整理部品の場合と同様）．このリストを見て部品の組合せのばらつきを調べ，追加／削除すべき部品を発見する．

3.2 必要なGUI設計データの取得方法

以上のGUIの一貫性評価のためには，部品の種類，ラベル，位置，サイズ，色やラベル文字の属性などのデータが必要である．部品の種類は操作可能な部品（操作部品）と表示だけの部品（表示部品）に大別でき，操作可能な部品にはメニューやボタンなど，表示だけの部品にはスタティックテキストなどがある．

本稿では，これらのGUI画面設計データをアプリケーションの実行プログラムから取得する方法を提案する．具体的には，評価対象アプリケーションの画面上の操作部品を自動検出する「オフスクリーンモデル生成ツール」[18]を利用し，部品操作を網羅的に自動実行して（操作イベントを部品に対して自動発行して）アプリケーションの画面を遷移させ，各ウィンドウ上の部品データを記録する．言い換えると，画面／操作をそれぞれノード／リンクとする（操作により画面が遷移する）ツリー構造でGUIをモデル化し，部品検出・データ記録・イベント発行のサイクルを繰り返すことでGUI構造モデルをアプリケーションの実行プログラムから自動的に生成する．評価対象アプリケーションの実行プログラムからGUI設計データを取得するため，ソースプログラムは不要であり，開発言語に依存せず一貫性を評価できる

という利点がある．また，アプリケーション操作の大部分を自動化できる（操作スクリプトを記録・編集する必要がほとんどない）ため，動作自動検査や操作マクロ自動生成といった別分野にも応用可能性がある．提案するデータ取得手法の流れ図を図6に示す．

アプリケーション起動：評価者により登録された評価対象アプリケーションの実行ファイル名（絶対パス）をもとに，そのアプリケーションを(再)起動する．

ウィンドウ／部品検出：画面上に表示されているウィンドウとそのウィンドウ上の部品（操作部品と表示部品の両方）を検出し，ウィンドウ／部品データを抽出する．この処理はオフスクリーンモデル生成ツールによって実行される．

ウィンドウ／部品データ記録：現在の画面について検出されたウィンドウと部品のなかで新規なもの（まだデータを記録していないもの）を判別し，データを記録する．また，直前に実行された操作によりウィンドウの開閉があった場合，開かれた／閉じられたウィンドウのタイトルを，実行された操作部品のデータとして記録する．

初期化判定：1 つ前のデータ記録処理で新規な操作部品が 1 つも記録されなかった（現在の画面には新規な操作部品がなかった）場合，アプリケーションを初期化させるために終了処理に進む．

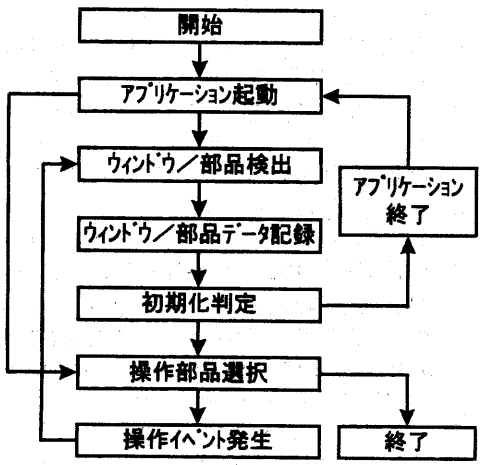

図6：GUI 設計データ（GUI 構造モデル）取得手法の流れ図

記録されていた場合は操作部品選択に進む．

アプリケーション終了：実行中のアプリケーションを強制終了させる．

操作部品選択：記録された操作部品のなかからまだ操作を実行していないものを 1 つ選択する．

操作イベント発生：選択された操作部品を操作するための入力イベントを発生させる．

4 他ツールとの比較

GUI の一貫性を評価するためのツールとしてこれまでに「SHERLOCK」と呼ばれるツールが提案されている[10]．このツールは，あらかじめ変換プログラムで GUI のソースファイルから部品ラベルなどのデータを抽出して定型ファイルに出力しておき，このファイルを読み込んで GUI テスタと同様にリストを生成する．

SHERLOCK と GUI テスタ II を比較すると，評価用データの取得方法と一貫性の評価方法についてそれぞれ次のことが言える．

評価用データ(GUI 設計データ)の取得方法

- SHERLOCK では GUI のソースプログラムが必要であり，かつ開発言語に応じたソース解析プログラムを要する．一方，GUI テスタ II では実行プログラムを用いて設計データを取得するため，ソースプログラムは不要であり，開発言語に依存しない．

一貫性の評価方法

- SHERLOCK と GUI テスタ II の両方がカバーしている評価要素としては，ラベルリスト，文字属性，位置，サイズ，スペース，色，部品の組合せの 7 つがある．
- しかし，文字属性，サイズ，スペース，色，部品の組合せの 5 要素については，SHERLOCK はデータをそのままリスト（表）に出力しているだけで，データ間の比較（同じデータのグループ化）は行っていない．一方，GUI テスタ II では部品間でデータを比較して同じものをグループ化しており，部品数のばらつきが容易にわかる．
- 部品の位置については，SHRELOCK では座標値を出力するだけのため，ウィンドウの「右

上」や「左下」といった相対的な位置が直感的にはわからない．一方 GUI テスタ II では，部品の位置をウィンドウサイズで正規化しており，かつその位置をグラフで表示するため，同じラベルをもつ部品の相対的な位置のばらつきが直感的にわかる．

・ ウィンドウタイトルとウィンドウ機能部品の組合せの 2 要素は，GUI テスタ II だけがカバーしている．しかし逆に，SHERLOCK では類義語の登録により，起こりやすい単語の不一致を自動的に検出できる．

5 まとめ

GUI 画面の一貫性を評価するツール「GUI テスタ II」を提案し，一貫性の評価方法と評価に必要な GUI 設計データの取得方法を示した．現在，Windows®アプリケーションを評価対象として同 OS 上で本ツールを開発している．

課題の 1 つとして「図・記号」と「形状」の評価が挙げられる．これらの評価には，本稿で提案した技術に加え画像認識・比較技術が必要となる．

今後は本ツールを用いて一貫性評価を試み，その有効性や課題を検証する．さらには画面の「シンプルさ」や「見やすさ」の評価手法を開発していく．

参考文献

1. Nielsen, J. Usability engineering. Academic Press, 1993.
2. Hoiem, D.E., Sullivan, K.D. Designing and using integrated data collection and analysis tools: challenges and considerations. Behavior & Information Technology, Vol. 13, Nos. 1&2, 160-170, 1994.
3. Siochi, A.C., Enrich, R.W. Computer analysis of user interface based on repetition in transcripts of user sessions. ACM Trans. on Information Systems, Vol. 9, No. 4, 309-335, 1991.
4. 池本. GUI の操作性評価ツール. 第 11 回ヒューマンインタフェースシンポジウム, 335-340, 1995.
5. 来住. X Window におけるユーザ操作記録および再生ツールの再設計. WISS'94, 95-104, 1994.
6. Guzdial, M., Santos, P., Badre, A.N., Hudson, S.E., Gray, M. Analyzing and visualizing log files: a computational science of usability. TR94-08, GVU Center, Georgia Institute of Technology, http://www.cc.gatech.edu/gvu/research/techreports94, 1994.
7. Cuomo, D.L. Understanding the applicability of sequential data analysis techniques for analyzing usability data. Behavior & Information Technology, Vol. 13, Nos. 1&2, 171-182, 1994.
8. Hicinbothom, J., Watanabe, M., Weiland, W., Boardway, J., Zachary, W. A toolset for systematic observation and evaluation of computer-human interaction. CHI '94 Companion, 5-6.
9. Kieras, D.E., Wood, S.D., Abotel, K., Hornof, A. GLEAN: a computer-based tool for rapid GOMS model usability evaluation of user interface designs. Proc. UIST '95, 91-100.
10. Mahajan, R., Shneiderman, B. Visual & textual consistency checking tools for graphical user interfaces. TR96-08 (Human-Computer Interaction Lab., Univ. of Maryland), May 1996. http://www.cs.umd.edu:80/projects/hcil/Research/tech-report-list.html
11. Sears, A. Layout appropriateness: a metric for evaluating user interface widget layout. IEEE Trans. On Software Engineering, Vol. 19, No. 7, 707-719, 1993.
12. 浜田, 新倉, 前川. 音声インデックス機能を持つユーザビリティテストツール. 第 8 回ヒューマンインタフェースシンポジウム, 343-346, 1993.
13. 鈴木, 金子, 大村. ユーザビリティ評価システム QUIS. 第 11 回ヒューマンインタフェースシンポジウム, 747-752, 1995.
14. 岡田, 旭, 井関. 使いやすさ評価ツール「GUI テスタ」の提案. 情処 HI 研, 59-13, 87-94, 1995.
15. 岡田, 旭, 井関. 使いやすさ評価ツール「GUI テスタ」における共通操作パタン抽出方式の提案と評価. 情処 HI 研, 63-7, 37-42, 1995.
16. 岡田, 旭, 足尾. 使いやすさ評価ツール「GUI テスタ」～省略操作検出手法の提案と評価～. 53 回(H8 後)情処全大, 分冊 6, 399-400, 1996.
17. 岡田, 旭. 操作性評価ツール「GUI テスタ」の UI 開発サイクルにおける有効性と課題. 計測自動制御学会 HI 研究会(Human Interface N&R), Vol. 12, No. 1, 77-84, 1997.
18. 山中, 岡田, 上窪, 兼吉, 井関. GUI 対応スクリーンリーダのためのオフスクリーンモデル. 53 回(H8 前)情処全大, 分冊 5, 189-190, 1996.