B-005

動的テスト自動化による GUI プログラミング課題採点手法の検討
GUI Dynamic Testing Method for Programming Exercises

立石 良生† 井上 潮†
Yoshiki Tateishi Ushio Inoue

1. はじめに

近年，人工知能（AI）や IoT などのテクノロジーが飛躍的に進化していることからプログラミング教育が重要視されている．プログラミング言語の学習には C 言語や Java を用いてコマンドラインインタフェース（CLI）のプログラムを作成する他に，オブジェクト指向プログラミングを習得するためにグラフィカルユーザインタフェース（GUI）のプログラムを作成する．プログラミング言語の習得は実際にソースコードを記述することが非常に重要なため，プログラミングの授業において毎回数個の簡単なプログラムを作成する課題が課されることが多い．しかし，数百人分のプログラミング課題を手作業で採点するのは非常に手間がかかる．特に，GUI プログラミング課題を採点する場合はマウスの操作や文字の入力などが必要なため，より多くの時間を必要とする上に採点ミスも発生しやすくなる．そこで本研究では学生が作成した GUI プログラミング課題を自動で評価，採点するシステムを開発する．これにより，教員の負担を減らすだけでなく，学生も自ら作成した GUI をテストしながら誤りの修正を行うことで，学生のプログラミング能力を向上させることを目的とする．

2. 関連研究

Nguyen ら[1]は，「GUITAR」と呼ばれる GUI アプリケーション自動テスト用ツールを開発した．これはモデルベース方式のフレームワークとなっており，GUI テスト対象のコンポーネントやイベントを容易に拡張することができる．これにより，従来の手法と比較して様々な GUI アプリケーションの自動テストに柔軟に対応できるようになった．しかし，手動での作業が多いことや誤検出，予期せぬエラーが発生するなどの問題が挙げられていた．また，使用するには特別な環境を必要とするため学生が使用するには適していない．本研究では Java の知識のみで機能を拡張できるようにするためにスクリプトベースで自動テストを行う．

3. 前提条件

3.1 対象のプログラミング課題

本研究の対象となるプログラミング課題は Java の GUI ライブラリである「JavaFX」で開発されたアプリケーションとする[2].

3.2 テスト方式

GUI のテストは大きく分けて「静的テスト」と「動的テスト」に分けられる．静的テストは GUI のボタンやラベルなどの部品が正しく配置されているかを検証し，動的テストではボタン押下などのイベント処理が正しく行われているか検証する．本研究では静的テストに筆者が開発した静的テスト自動システムを用いる[3]．ここでは動的テストの手法に焦点を当てて説明していく．

†東京電機大学 工学研究科 情報通信工学専攻
Information and Communication Engineering, Graduate School of Engineering, Tokyo Denki University

4. 提案手法

4.1 システム構成

プログラミングの授業で演習課題を課す場合，教員は事前に模範解答のプログラムを作成し，動作確認を行う．この模範解答を用いて自動採点するシステムについて説明する．学生が作成した GUI の採点は以下の手順で行う．

① 教員は模範解答と学生の解答のプログラムを用意する
② これらの GUI からレイアウト情報をそれぞれ取得し，XML 形式でファイルを自動生成する
③ 2 つの XML ファイルを用いて静的，動的テストを行う
④ 採点結果を出力する

この手順で採点を行うことで学生の課題を自動で採点することができ，他の GUI を採点する際も新たにテストスクリプトを作成する必要がなくなるので，採点者の負担を大幅に減らすことができる．このシステムの構成図を図 1 に示す．

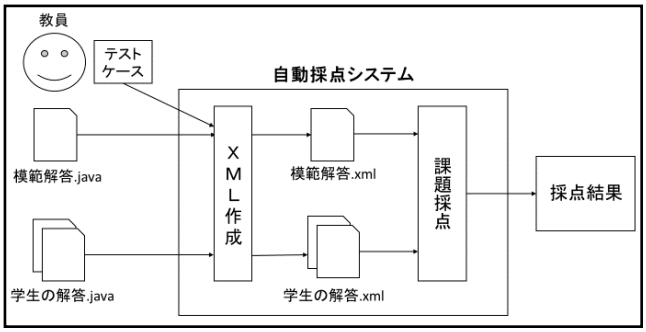

図 1　自動採点システムの構成図

4.2 動的テスト手法

レイアウト情報が保存された XML ファイルを用いて動的テストを行う．教員は入力と出力のコンポーネントを選択し，XML に属性として書き込む．また，テストケースとして教員が用意した入力値と出力値も XML に記述する．動的テストを行う際に入力となるコンポーネントを XML から抽出し、入力値を学生の GUI に入力させる．出力された値を XML に保存してある出力値と比較することで動的テストを行うことができる．XML ファイルを複数用意すれば様々なテストケースで動的テストを行うことができる．

5. 実験

5.1 実験方法

実際に授業で扱った演習課題を用いて実験を行う．実験で用いる演習課題は図 2 で示すチケット価格計算アプリケーションで行う．料金と枚数を選択すると，合計金額が表示され，リセットボタンを押すと初期画面になるアプリケーションである．

図２　チケット価格計算アプリケーション

5.2 模範解答から XML を生成

教員が事前に作成した図 2 の GUI から XML を自動生成すると図 3 のようになる．ライブラリの仕様上，BorderPane の場合真ん中，上，下の行の順で部品の情報が格納される．図 3 を見ると，図 2 の GUI のレイアウトが正しい順番で配置されていることが確認できる．ここでは料金と枚数を選択する ComboBox が「入力」，金額を表示する TextField が「出力」となる．

```
<Scene ID="0">
<BorderPane ID="1">
<HBox ID="2" Pos="CENTER">
<Label ID="3">チケット計算機</Label>
</HBox>
<HBox ID="4" Pos="CENTER">
<Label ID="5">料金</Label>
<ComboBox ID="6" type="Input">200</ComboBox>
<Label ID="7">枚数</Label>
<ComboBox ID="8" type="Input">2</ComboBox>
</HBox>
<HBox ID="9" Pos="CENTER">
<Label ID="10">金額</Label>
<TextField ID="11" type="Output">400</TextField>
<Button ID="12">リセット</Button>
</HBox>
</BorderPane>
</Scene>
```

図３　自動生成された XML

5.3 自動テスト結果

図 2 で示したアプリケーションの自動テスト結果を図 4 に示す．図 4 を見ると，誤った出力に対して「NG」と出力されているので，学生が作成した GUI アプリケーションに対して正しく自動テストが行えていることが確認できる．

```
HBox：OK
Label：OK
TextField：OK
Button：OK
静的テスト：成功
動的テスト開始
入力：200
入力：2
出力：200
NG　出力は200ではありません。
動的テスト：失敗
```

図４　自動テストの結果画面

6. 評価

学生が作成した演習課題を用いて採点精度と採点時間の比較を行う．対象となる GUI アプリケーションは 5 章で紹介したチケット価格計算アプリケーションとする．採点対象は提出した 95 名の課題のうち，静的テストに成功した 91 個である．採点精度は手動採点の結果を 100%とした時の提案手法の結果の一致率を表し，採点時間は 91 個のプログラムに対してプログラムを実行してからテスト終了までの時間を合計した時間を表す．7 種類のテストケースでテストしたときの自動採点の精度，時間を比較した結果を表 1 に示す．採点の結果が一致しなかったプログラムは，GUI 上では正しく動作していたが，コンソール上でエラーが出ていたのが原因であった．表 1 を見ると，手作業は人によって採点時間に差があるが，提案手法での採点時間は手作業での採点時間と比較して大幅に短縮でき，高い精度で自動採点できていることがわかる．

表 1　採点時間の比較

採点手法	採点時間[s]	採点精度[%]
手動採点	2321.3	
提案手法	426.2	92.3

7. まとめ

GUI プログラミング課題の動的テストを自動化することで，大幅に採点時間を短縮することができた．また，XML ファイルを用いることで自動採点の事前準備の手間を減らすことができた．しかし，現状ではすべての演習課題を自動テストできるわけではない．また，入力と出力を XML に書き込む作業やテストケースを考える作業を手動で行わなければならない．特に，テストケースは教員が考えて用意したものでは全てのパターンをテストするのが難しく，テスト漏れが発生することがある．今後は対応するアプリケーションを増やしながら、手動で行っている作業を自動化する手法について検討する必要がある．

参考文献

[1] Nguyen, B.N.,Robbins,B., Banerjee, I.,Memon, A., “GUITAR: an innovative tool for automated testing of GUI-driven software”, Autom. Softw. Eng. (Springer) 21(1), p65–105 (2014)
[2] JavaFX の概要 - Oracle Docs, https://docs.oracle.com/javase/jp/8/javafx/get-started-tutorial/jfx-overview.htm
[3] 立石良生, 井上潮, “GUI プログラミング課題自動採点方式の検討”, 第 11 回データ工学と情報マネジメントに関するフォーラム (2019)