N-035

実習の要件の差異の吸収に着目したプログラミングの実習システムの提案と試作

A proposal for a system of programming practice focused on absorption of differences in requirements

竹山 祐太郎† 横井 翔太‡ 杉田 基樹‡ 早川 智一†
Yutaro Takeyama Shota Yokoi Motoki Sugita Tomokazu Hayakawa

1. はじめに

大学などの教育機関で行われるプログラミングの実習（以下，実習）では，学生のプログラムを自動で評価するシステム（以下，実習システム）が用いられることがある．この理由の 1 つとしては，教員の間接的な業務（演習課題の採点やフィードバックなど）の一部を実習システムに任せることで教員の工数を削減し，教員の直接的な業務（質問対応や補足説明など）に注力できる点がある．例えば，Moodle [1] や Sakai [2] などの学習管理システムでは，プログラムを自動で評価する機能が提供されており，複数の大学 [3] [4] で実習システムとして利用されている．

しかし，既存の実習システムは，実習の要件の差異を直ちに吸収することは容易ではない．これは，実習の要件（学生のプログラムに対する評価方法・取り扱うプログラミング言語（以下，言語）・履修者のプログラミングの習熟度など）が実習によって異なるためである．例えば，基礎的なプログラミングの学習を目的とした実習では，学生のプログラムの動作確認などの基本的な評価のみに注力されがちだが，堅牢なソフトウェアの開発手法の学習を目的とした実習では，学生のプログラムそのものの詳細な品質評価が追加で求められる．

実習の要件に合わせて新しいシステムを開発したり，既存のシステムを改良したりすることは，教員の工数の増大につながる [5] 場合がある．教員の工数の増大を防ぐには，実習システムが実習の要件の差異を吸収することが望ましいと我々は考えるが，これに着目した研究は我々の調査の限り存在しない．

我々の研究の目的は，1 つの実習システムで実習の要件の差異を吸収することで，実習の運営にかかる間接的な教員の工数の一部を削減することにある．本論文では，各大学の実習の要件の調査を行い，それを踏まえて評価方法を柔軟に選択・追加可能な実習システムを提案する．

2. 関連研究・関連技術

長ら [6] は，オンラインプログラミング学習環境 Bit Arrow を提案している．Bit Arrow は，複数のプログラミング言語が Web ブラウザ（以下，ブラウザ）上で動作可能となるように独自の処理系を提供し，ブラウザ上でのプログラミング学習を可能にしている．Bit Arrow はプログラミング学習に際しての環境構築の工数の削減を目的としており，教員の工数の削減という目的は本研究と共通しているが，削減する工数の内訳が異なる．

新田ら [5] は，オンラインプログラミング学習・試験配信システム track を提案している．track は，複数の言語に対応した実行環境や自動採点機能を提供しており，実習を運営する側の工数を削減することを目的としている点で本研究と類似している．しかし，プログラムの品質を詳細に評価するための一部の評価方法に対応していないため，本研究とは異なる．

Moodle や Sakai などの学習管理システムは，実習に利用できる UI（User Interface）などの機能が提供されており，複数の大学で実習システムとして運用されている．しかし，これらのシステムでも同様に，プログラムの品質を詳細に評価するための一部の評価方法を容易に追加できないため，本研究とは異なる．

3. 実習の要件の調査

3.1 日本の 86 の国立大学のシラバスの調査

我々は，実習の要件の差異を特定するために，日本の 86 の国立大学のシラバス [7] を調査した．また，調査にあたって可能な限り客観的な尺度が必要であったため，調査結果の分類は情報処理学会の提案するカリキュラム標準 J17 のソフトウェアエンジニアリング領域（以下，SE 領域）のシラバス [8] を参考にした．SE 領域を参照した理由は，（1）SE 領域のシラバスが実習科目に言及していること，（2）SE 領域が幅広い領域と密接に関わる [9] こと――から，より多くの実習の要件の差異を吸収するという本研究の目的と合致するためである．

まず，実習における言語の使用率を表 1 に示す． 使用言語の内訳としては，プログラミング初学者を対象とした実習の多くでは C・Python が，オブジェクト指向やア

表 1 国立大学のシラバス調査に基づく実習でのプログラミング言語の使用率

プログラミング言語	使用率（%）
C	58.19
Python	21.41
Java	12.60
C++	3.78
JavaScript	2.94
その他	1.08

†明治大学理工学部 School of Science and Technology, Meiji University
‡明治大学大学院理工学研究科 Graduate School of Science and Technology, Meiji University

ルゴリズムの学習を目的とした実習の多くでは C・Java・Python・C++が，堅牢なソフトウェアの開発手法の学習を目的とした実習では主に Java・JavaScript が用いられていた．しかし，開発するシステムの要件定義や技術選定を学生に委ねるためにシラバスで言語の指定が無いものも存在したため，実際の使用率は前後すると予想される．

次に，実習システムが持つ主な機能を表 2 に示す． 提出されるファイル形式には，テキスト形式や画像形式など，フィードバックの表示形式は，プレーンテキストによるものや HTML によるものなどの種類がある．また，学生のプログラムの評価方法は様々であったため，SE 領域のシラバスも合わせて調査した．その結果，実習システムに必要と判断できる評価方法は次の 6 種類である：（1）動作確認，（2）静的解析，（3）単体テスト，（4）結合テスト，（5）メトリクス解析，（6）システムテスト．

3.2 既存の実習システムの課題点

既存の実習システムの工数削減の余地のある利用例を図 1 に示す． 調査の結果，多くの実習システムは，学生が提出したファイルに対して任意の評価を行い，それをフィードバックするといった構成であった．しかし，3.1 節で述べたように，学生のプログラムの評価方法は様々である．そのため，実習 A と実習 B とでそれぞれ実習システムを用意し，評価方法を設定する必要がある．この結果，両システムで共通している機能（ファイルの提出やフィードバック）がそれぞれで実装されている．そこで，学生のプログラムの評価方法の差異を吸収することで，実習の要件の差異の吸収につながり，教員の工数の削減が期待できる．

表 2 国立大学のシラバス調査に基づく実習システムが持つ主な機能

主な機能	実習ごとで差異があった点
ファイルの提出	ファイル形式
フィードバック	表示形式
学生のプログラムの評価	評価方法

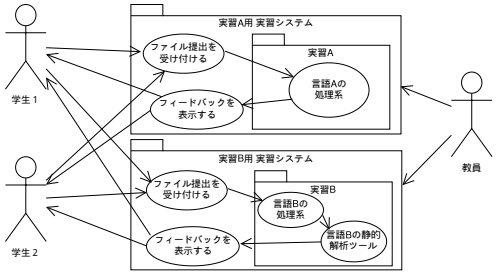

図 1 既存の実習システムの利用例

4. 提案手法

提案システムは，プログラムの評価方法の差異を吸収するために，各評価方法をモジュールとして扱う．教員は，3.1 節で挙げた評価方法に該当するソフトウェアや，教員が独自に用意したツールなどをモジュールとして追加できる．

提案システムにおけるモジュールの使用例を図 2 に示す． 各モジュールは，共通の入出力インタフェースを持っているため，学生のプログラムに対して用いる評価方法を任意の組み合わせで教員が問題ごとに設定できる．

提案システムを用いた場合の想定利用例を図 3 に示す．問題ごとに評価方法を設定可能にしたことで，評価方法の差異を吸収でき，提案システムは単独で複数の実習に対応可能となる．また，ファイルの提出機能やフィードバック機能を実習ごとに作り直す必要がなくなる．

5. 設計

5.1 モジュールがもつインタフェース

我々は，3 章の調査をもとに，モジュールの入出力インタフェースとして必要と考えられるものを抽出し，表 3 のように設計した． 学生のプログラムのソースコードは全ての評価方法，テストケースは単体テスト・結合テスト・システムテストで必要となる．終了ステータスとエ

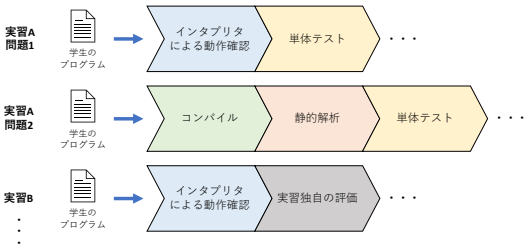

図 2 モジュール化した評価方法の使用例

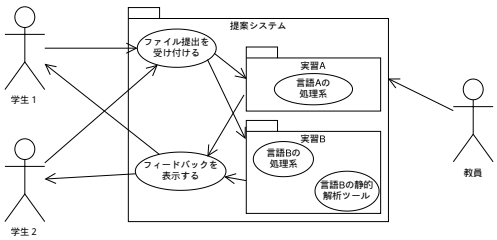

図 3 提案システムの想定利用例

表 3 評価モジュールが持つインタフェース

入出力	項目
入力	学生のプログラム
	テストケース
出力	終了ステータス
	エラーメッセージ

ラーメッセージは，必ずしも両方が必要ではないが，学生へのフィードバックコメントの提示や成績評価のためにはいずれかが必要となると考えられる．

5.2 コンテナ仮想化ツールの使用

我々は，IaC（Infrastructure as Code）が可能なコンテナ仮想化ツールを用いてモジュールを作成できるように設計した．これは，プログラムの実行環境を少量のコードで構築することで，モジュールの作成にかかる教員の工数を削減するためである．

提案手法では，コンテナ仮想化ツールにおけるイメージ（仮想コンテナの生成に必要な命令などを記述したもの）を用いて仮想コンテナを生成する．そして，構築したコンテナの中で任意のコマンドを実行することで，プログラムの評価を行う．

教員は，以下の 2 通りの方法で，モジュールを提案システムに追加することが可能である：(1) 教員が新規イメージを構築する方法，(2) ベースイメージをそのまま利用する方法．これにより，モジュールの汎用性を確保しつつ，教員が記述するコードを最小限に抑える．それぞれの方法について，以下で詳細を述べる．

5.3 モジュールの追加

5.3.1 教員が新規イメージを構築する方法

教員が新規イメージを作成することで，任意のソフトウェアの実行環境を作成できる．具体的には，教員は設定ファイル・イメージ・スクリプトファイルを記述する必要がある．例えば，C 言語のプログラムに対し，Splint を用いて静的解析を行う場合に教員が記述する設定ファイルを**コード 1** に，イメージを**コード 2** に，スクリプトファイルを**コード 3** に示す．コード 2 では，Ubuntu22.04 の環境をベースイメージとし，Splint をインストールするまでのイメージ構築の流れを記述する．コード 3 では，Splint を用いて実際にプログラムの静的解析を行うコマンドを記述している．

5.3.2 ベースイメージをそのまま利用する方法

主要な環境を定義したイメージは，レジストリにて公開されている場合が多い．提案システムでは，公開されているイメージをそのまま使用可能である場合に，教員によるイメージの構築に関する記述を省略することができる．例えば，Python で記述されたプログラムの動作確認を行う場合に教員が記述する設定ファイルを，**コード 4** に，スクリプトを**コード 5** に示す．ここでは，コード 4 において，ベースイメージ `python:latest` の使用を指定している．これにより，モジュールの作成にかかる教員の工数の削減に貢献している．

コード 1 使用するイメージ構築ファイル・スクリプトファイルの設定

```
{
  "useBaseImage": false, // ベースイメージを不使用
  "customImageFile": "Dockerfile", // Dockerfile を使用
  "script": "run.sh" // run.sh を実行
}
```

コード 2 イメージの構築（Dockerfile）

```
FROM ubuntu:22.04
RUN apt-get update -y
RUN apt-get install splint -y
```

コード 3 Splint による静的解析のためのスクリプト（run.sh）

```
filename=$1 # 実行するファイル名
errorlog="error" # エラー出力先

splint "$filename" 2> "$errorlog"
```

6. 実装

我々は提案システムを**表 4** のソフトウェアを用いて実装した．プログラミング言語には，JavaScript を採用した．これは，提案システムを Web アプリケーションの形態としたため，ブラウザ上で動作可能な事実上唯一の言語をクライアントとサーバとで用いることで，開発・保守の工数を低減できるためである．

7. 提案システムの評価

我々は，提案システムが実習の要件の差異を吸収し，教員の工数の削減に貢献できるかを評価するために，3 章で調査した日本の 86 の国立大学のシラバスより読み取れる実習や，本学に設けられている実習に対して提案システムの適用が可能かを検証した．

7.1 国立大学の実習への適用

まず我々は，日本の 86 の国立大学のシラバスを参照し，提案システムが全実習 463 件に対して，(1) 要件の全てに

コード 4 使用するイメージ・スクリプトファイルの設定

```
{
  "useBaseImage": true, // ベースイメージを使用
  "baseImage": "python:latest", // python の最新版を使用
  "scriptFile": "run.sh" // run.sh を実行
}
```

コード 5 Python による動作確認のためのスクリプト（run.sh）

```
filename="$1" # 実行するファイル名
errorlog="error" # エラー出力先
stdin="stdin" # 標準入力先
stdout="stdout" # 標準出力先

python3 "$filename" < "$stdin" > "$stdout"  2> "$errorlog"
```

表 4 実装に用いたソフトウェア

ソフトウェア	バージョン	備考
Node.js	18.16.0	提案システムの実行環境
Nest.js	9.0.0	ルーティング
Bull	4.10.4	ジョブキュー
Docker	20.10.21	コンテナ仮想化ツール

適用可能，（2）要件の一部に対して適用不能，（3）要件の全てに適用不能――のどれに該当するかの評価を行った．

評価結果を表 5 に示す．我々は，全体の 96.96%の実習では適用可能であると判断した．また，全体の 3.02%の実習では，一部または全てに適用不可であると判断した．この理由としては，実習の一部または全体で扱われる題材が UNIX コマンドや VBA（Visual Basic for Applications）についてであったことが挙げられる．我々は提案システムが要件の差異を吸収する実習を，学生が提出したファイルに対して任意の評価を行い，それをフィードバックするといった構成のものに限定した．しかし，UNIX コマンドについて取り扱う実習の内容はファイルの提出を伴わない形態であった．また，VBA は独自の処理系に依存しているため，提案システムの評価方法としてモジュール化して組み込むことができない．よって，これらの実習の提案システムは適用不可であると判断した．

7.2 本学の実習への適用

次に我々は，複数の実習に対して適用可能であることをより詳細に評価するため，本学に設けられている実習にて出題されている問題が再現可能であるかの評価を行った．また，評価にあたって客観性を保つために，SE 領域のシラバスで提言されている実習科目のどれに再現する実習が該当するかも併せて確認を行った．

7.2.1 プログラム実習

プログラム実習は，C を用いてプログラミングの基礎を学ぶ本学の 1 年次の実習であり，SE 領域のシラバスで提言されている実習では「プログラミング基礎実習」に該当する．この実習では，学習管理システム Sakai をもとに構築された実習システムが用いられており，学生のプログラムに対して以下の評価が行われている：C のコンパイラ GCC によるコンパイルエラーの有無のチェック，実行時の入出力のテスト．

我々は，提案システムがプログラム実習に適用可能かを検証するため，既存のシステムにて出題されている問題 477 件を提案システムで再現した．その結果，471 件の問題を再現できることを確認した．なお，再現できなかった 6 件は，UNIX コマンドについて取り扱っていた．

7.2.2 ソフトウェア工学演習

ソフトウェア工学演習は，学生同士の集団開発を交えながら堅牢なソフトウェアの開発手法を学ぶ本学の 3 年次の実習であり，SE 領域のシラバスで提言されている実習では「エンタープライズ開発実習」に該当する．この実習では，主に HTML・CSS・JavaScript などの言語を用いた Web アプリケーションの開発を行う．また，堅牢なソフトウェアを開発するための手法として，ESLint を用いた静的解析や Jest を用いた単体テストを取り扱っている．これまで，実習課題として学生にソースコードの提出を求めることがあったが，実習システムは使用せず，教員が手動でプログラムの評価を行っていた．

我々は，提案システムがソフトウェア工学演習に適用可能かを検証するため，実習課題として学生にプログラムの提出を求める問題 6 件を提案システムで再現した．その結果，手動での評価と同等の評価を提案システムで再現できることを確認した．

以上の評価結果から，提案システムは実習システムとして実習の要件に適用でき，教員の工数の削減の有用な手段たりえるという結論を得た．

表 5 提案システムを国立大学の実習に適用できる割合

適用度	件数（件）	割合（%）
適用可能	448	96.96
一部適用不能	10	2.16
適用不能	4	0.86

8. おわりに

本論文では，教員の工数を削減するため，学生のプログラムに対する評価方法をモジュール化することで，実習の要件の差異を吸収する実習システムを提案した．評価の結果から，提案システムが実習を運営する教員の工数の削減に貢献可能であるという結論を得た．今後の展望としては，7 章で適用不可とした実習も提案システムに適用可能にすることによって，実習を運営する教員の工数をさらに削減することが挙げられる．

参考文献

[1] Moodle HQ: Moodle - Open-source learning platform | Moodle.org, Moodle , https://moodle.org/.
[2] Sakai: Sakai Learning Management System | Sakai LMS, https://www.sakailms.org/.
[3] 大西淑雅，山口真之介，近藤秀樹，西野和典：ネットワークログを用いた学習活動の把握の提案，技術報告 1，九州工業大学学習教育センター, 九州工業大学学習教育センター, 九州工業大学学習教育センター, 九州工業大学教養教育院 (2019).
[4] 生田寛，中野裕司，杉谷賢一，久保田真一郎：オンライン短答式記述問題の解答に対する潜在的意味解析を用いた自動フィードバック手法の検討，技術報告 21，熊本大学大学院自然科学教育部, 熊本大学総合情報統括センター, 熊本大学総合情報統括センター, 熊本大学総合情報統括センター (2019).
[5] 新田章太，小西俊司，竹内郁雄：複数言語に対応しやすいオンラインプログラミング学習・試験システム track，情報教育シンポジウム論文集，Vol. 2019, pp. 114–121 (2019).
[6] 長慎也，長島和平，堀越将之，兼宗進，並木美太郎：オンラインプログラミング環境 Bit Arrow を用いた C 言語プログラミングの授業実践，情報教育シンポジウム論文集，Vol. 2017, No. 17, pp. 121–128 (2017).
[7] 国立大学協会：国立大学のシラバス，国立大学協会（オンライン），https://www.janu.jp/univ/syllabus/.
[8] 情報処理学会：カリキュラム標準ソフトウェアエンジニアリング領域 (SE)，情報処理学会（オンライン），https://www.ipsj.or.jp/annai/committee/education/j07/9faeag000000v1e7-att/a1526533136298.xlsx.
[9] 情報処理学会：カリキュラム標準 J07，情報処理学会（オンライン），https://www.ipsj.or.jp/annai/committee/education/j07/curriculum_j07.html.