ソフトウェアエンジニアリングシンポジウム 2020
IPSJ/SIGSE Software Engineering Symposium (SES2020)

完全遠隔講義におけるソフトウェア開発演習支援ツールの開発とその評価

山田侑樹[1]　古川貴一[1]　櫨山淳雄[2]

概要：COVID-19 感染拡大により多くの高等教育機関ではオンラインでの遠隔講義を余儀なくされた．演習を実施する科目においては従来の方法が適用できず実施方法の見直しが必要になるものもある．我々はソフトウェア工学教育において，開発環境を受講生が構築した後に Web アプリケーション開発を行う演習を実施していた．オンラインで演習を行うためには開発環境構築への対応，演習においてトラブルが発生した場合の質疑応答を円滑に遂行する方策が必要になった．本報告ではオンラインでのソフトウェア工学教育実施に向けたソフトウェア開発演習支援ツールの開発とその評価について述べる．

キーワード：新型コロナウィルス感染症，完全オンラインソフトウェア工学教育，開発環境自動構築，Web アプリケーション開発

Development of a Support Tool for Full Online Implementation of a Software Engineering Course with Web Application Development and Its Evaluation

YUKI YAMADA[†1]　KIICHI FURUKAWA[†1]
ATSUO HAZEYAMA[†2]

Abstract: On account of the COVID-19 outbreak, a number of educational institutes are forced to execute online distributed education. In traditional subjects conducting not only lectures but also practices, the instructor and teaching assistants walk around the class, check how the students are doing, find students who have troubles on their practices and guide them. However, in distributed education, it is difficult to do the same things as the collocated environment, therefore some measures are required. We have been holding an introductory software engineering education that conducts practice of web application development in addition to lectures. In full online implementation of this subject, we need to consider the following two issues: (1) creation of a mechanism to mitigate troubles regarding building a software engineering environment, and (2) creation of the process of question and answer during application development practice under full distributed circumstances on a remote meeting system. We develop a tool to solve the abovementioned issues. We evaluate the tool from the viewpoint of quality based on the ISO/IEC 25000 quality model.

Keywords: COVID-19, Online software engineering education, building of a software engineering environment, web application development

1. はじめに

新型コロナウィルス感染症 (COVID-19)感染拡大により多くの高等教育機関ではオンラインでの遠隔講義を余儀なくされた．講義のみでなく演習を伴う科目では，従来，教員や Teaching Assistant (TA) が演習中に机間巡視を行い，問題を抱える受講生を見つけ，個別に対応していた．遠隔講義では，演習の場面では，対面講義と同じことを行うことは難しく，何らかの対応に迫られる．

我々はソフトウェア工学の講義を実施している[2]．この科目では教員による講義に加えて，Web アプリケーション開発の演習を行っている．演習において受講生が各自保有する PC に開発環境の構築が必要となる．受講生は開発環境を構築した上で，教員が配布したプログラムを自身の PC 上に配置して動作確認を実施する．その後，受講生は教員から与えられた課題に対して各自で Web アプリケーションの開発を行い，レポートとして提出することが求められている．受講生の開発環境は複数のオープンソースソフトウェアにより構成されており，各自が保有する PC の個体差から従来から開発環境の構築支援は問題となっていた．

この科目を完全に遠隔で実施するにあたり，(1)受講生の開発環境を整えること，(2)教員および TA が受講生の状況を把握することの 2 つが必要となる．我々はこの 2 つの問題を解決するツールを開発した．ツールは遠隔講義実施までの短期間で開発する必要があり，またツールの導入によって遠隔講義の進行に影響が生じないためにもツールの品質を高めることは重要である．

本稿では，このツールの開発と開発したツールのソフト

1 東京学芸大学大学院
Graduate School of Education, Tokyo Gakugei University
2 東京学芸大学
Tokyo Gakugei University

ウェア品質の製品品質モデルと利用者品質モデルに基づく評価について述べる．

2. 関連研究

関連研究としてオンラインでのソフトウェア工学教育に関する文献を調査した．Ellis は遠隔分散ソフトウェア開発教育の実施について述べている[1]．遠隔分散ソフトウェア開発実施前に対面での教育を実施している．完全遠隔教育となった本研究とは対象が異なる．

完全遠隔分散教育としては MOOCs (Massive Open Online Courses)がある．MOOCs で開発環境構築や Web アプリケーション開発を含むソフトウェア工学教育の実施に関して報告された研究は我々が調べた限りでは見つかっていない．

3. 講義概要

本稿が対象とする科目は情報教育コース 3 年次に開講している「情報システム設計」という選択科目である．この科目では，ソフトウェア工学の基礎知識を学ぶとともに，JSP/サーブレット技術を用いた Web アプリケーション開発の演習を行っている．本コースでは 2 年次までにプログラミング，データベースなどの科目を開講している．

本学は全学生にノート PC を保有することを求めており，本科目では各自の PC にソフトウェア開発環境を構築する．開発環境は，プログラミング言語として Java 言語，統合開発環境 Eclipse，関係データベース管理システム MySQL，ビルドツール Gradle から構成されている．

以下では，前年度までの対面講義における演習の実施形態の概要と今年度の遠隔講義を実施する環境とその場合に想定される問題点について述べる．

3.1 前年度までの対面講義における演習の実施形態

2019 年度までは，上記で述べたソフトウェア開発環境を受講生が各自でソフトウェアのインストールを行い構築していた．具体的にはインストール手順を配布し，講義時間中に作業を実施していた．トラブルが発生した際には，受講生同士で解決を試みることや机間巡視している TA によってサポートしていた．実際，インストールするソフトウェアの細かなバージョンの違いなどの何件かのトラブルが発生していた．全受講生のソフトウェア開発環境の構築が完了した後に，MySQL を使った RDB の作成，教員から配布された Java ソースプログラムと手順書に基づいて Eclipse 上で Gradle プロジェクトを作成するといった演習活動を行っていた．ここでもソフトウェア開発環境構築時と同様にトラブルが発生した際には机間巡視をしている TA を中心にサポートしていた．

3.2 遠隔講義実施環境

遠隔講義は大学が提供する Microsoft Teams[1]を用いて週 1 回，同期的に行うこととした．講義資料を含むプログラムの配布と手順書の配布はこれまで同様に LMS (Learning Management System)を用いて配布する．

3.3 遠隔講義で演習を実施する上での問題点

遠隔講義では，3.1 で述べたような形態で演習を行うことはできない．遠隔講義で演習を実施するにあたり，特に問題となるのは，受講生のソフトウェア開発演習のための環境を整えること，演習時のトラブル対応を行うことである．

4. 開発演習支援ツール

講義を遠隔で完全な形で実施するために 3.3 で述べた問題点を解決する．具体的には，以下の 2 つのことを実施する開発演習支援ツールを開発する．

1. すべての受講生のソフトウェア開発環境の構築支援と教員，TA による構築状況の把握
2. ソフトウェア開発演習時のトラブルシューティングの機構の構築

これらを満たす開発演習支援ツールは 2 種類のスクリプトと Function as a Service (FaaS)環境で動作する 1 つのアプリケーションからなる．2 種類のスクリプトとは受講生の環境構築を実行するスクリプトと演習時にトラブルが起きた際に実行するスクリプトである．開発演習支援ツールの全体像を図 1 に示す．受講生がスクリプトを実行すると，その結果が FaaS 環境で実行されるアプリケーションから遠隔講義環境である Teams に通知される．

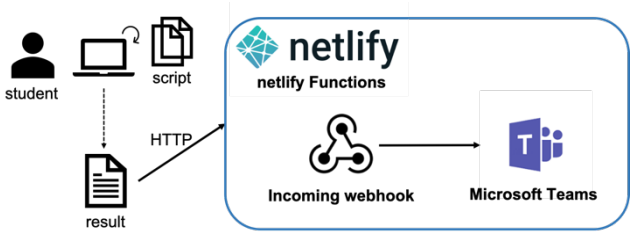

図 1 提案する開発演習支援ツールの全体像

以下では，ツールを構成するアプリケーションとスクリプトの詳細について述べる．

4.1 遠隔講義環境への通知を行うアプリケーション

我々は受講生の環境構築の結果を遠隔講義環境である Teams に通知することで教員，TA が構築状況の把握が行えるようなアプリケーションを開発した．このアプリケーションは Netlify[2]が提供する FaaS 環境である Netlify Functions を利用して実現した．FaaS はイベント駆動型のサービスであり，HTTP 通信による Web リクエストをトリガーとして様々な処理を記述できる．Netlify Functions を用いて，具体的な処理として，テキストファイルを含む Web リクエストを受け取った際に，テキストファイルの検証とテキストファイルの中身を取り出し，遠隔講義を行う環境である Teams で提供される Incoming Webhook を利用して Teams 上に受講生の環境構築状況を通知することを行っている．また Webhook のリクエストが失敗した場合でも実行結果の確認が漏れないようにデータベースにも保存している．こ

のアプリケーションによって教員, TA は講義を行う環境である Teams 上で各受講生が実行したスクリプトの結果を把握することができる.

4.2 受講生のソフトウェア開発環境構築

受講生のソフトウェア開発環境を整えることは開発演習まで見据えた上で最も重要なことである．なぜならばプログラミング言語のバージョンの違いといった開発環境のわずかな違いによってトラブルが発生するからである. 3.1 で述べたように前年度までは受講生自身にソフトウェア開発環境の構築を TA による対面式でのサポート付きで行っていた．遠隔講義でこのことを実施することは困難であり，またソフトウェア開発環境を整えることは講義の主たる目的ではない．そこで受講生の開発環境の構築の自動化を試みることとした. 自動化における要件は以下の 2 つである.

要件 1 すべての受講生のソフトウェア開発環境を構築できること

要件 2 教員，TA が受講生のソフトウェア開発環境の構築状況を把握できること

以上の要件を満たすように，受講生の環境構築を行うスクリプトの開発した.

4.2.1 環境構築スクリプト

前提として受講生の PC の OS は Windows または macOS である．この OS の違いに対応するために Windows 利用者には PowerShell スクリプトを macOS 利用者には bash スクリプトを用意した.

受講生の PC に Java，MySQL，Gradle といった本講義で使用するソフトウェア類のインストールはパッケージ管理ソフトウェア (例 : Chocolatey[3]，Scoop[4]，Homebrew[5])を用いて行うものとした．パッケージ管理ソフトウェアがインストールされていない場合にはインストールが行われるようにスクリプトに記述した.

さらに遠隔で講義を行う以上，教員，TA がスクリプトによって各ソフトウェアのインストールが成功し，各受講生の環境構築が正常に行われているかを把握できるようにする必要がある．そのためにスクリプトでは各ソフトウェアのインストール後にソフトウェアのインストール中にエラーが発生していないかとソフトウェアのバージョンを検証し，それらの結果をテキストファイルに出力する処理を記述した．スクリプトの最後の処理でこのテキストファイルを HTTP 通信により 4.1 で述べたアプリケーションに送信することで教員, TA が受講生の環境構築状況を把握できるようにした.

4.2.2 環境構築スクリプトの実行方法

スクリプトの実行方法にも工夫が必要である．他の講義資料と同様にスクリプトファイルも LMS によって配布し受講生がそれをダウンロードして実行することが考えられるが，スクリプトの実行はスクリプトが記述されるテキストファイルの文字コードに依存する. そして多くの LMS ではテキストファイルの文字コードの指定までをサポートしていない．そこでスクリプトは LMS を用いて配布するのではなく，ソースコードホスティングサービスである GitHub[6]上にスクリプトを配置し, それぞれの PC から HTTP 通信によってスクリプトのコードを取得し実行する方式とした．この方式は複雑に思えるが受講生と教員の双方に利点がある．受講生側はコマンドラインからわずか 1 行の命令を実行するだけで，自身の PC にスクリプトファイルをダウンロードすることなく実行できるということである．教員側は文字コードの問題に対処できるだけでなく，スクリプトに何らかの不具合が生じた場合にも GitHub 上のソースコードを更新するだけで対応可能であり，スクリプトファイルを再度配布するような手間を省くことができる.

実際のスクリプトの実行画面を図 2 に示す．実行時には誰がスクリプトを実行したかを把握するために学籍番号の入力を促すプロンプトを表示する．学籍番号入力後は 4.2.1 で述べた内容が実行される．実行後は図 3 に示す情報が Teams に通知され教員, TA が受講生の開発環境の構築状況を把握することができる.

図 2　環境構築スクリプトの実行画面

図 3　環境構築スクリプトから Teams への通知画面

4.3 ソフトウェア開発演習時のトラブルシューティング

受講生が Web アプリケーション開発演習時に直面するエラーに対し，その原因を突き止め解決することは重要なことである．しかしながら，受講生は初学者であることから自らエラーへの対処方法がわからないことも多く，教員や経験豊富な TA によってトラブルシューティングをすることも重要である．このトラブルシューティングを適切にまた迅速に行うことで受講生の負担が軽減される．今回のような完全遠隔環境下では 3.1 で述べたような方法では行えない．従って，教員や TA が効率的にトラブルシューテ

ィングを支援するための機構を用意する必要がある．教員や TA は受講生のトラブルシューティングを行う際にエラーログを確認することで実行状況を確認する．従って，トラブルシューティングの機構を実現するための要件として以下があげられる．

要件 受講生のソフトウェア開発演習時のエラーログを教員，TA が容易に把握できるようにすること

これを 4.2 と同様にスクリプトによって，開発しているアプリケーション実行時のログを収集する．

4.3.1 アプリケーション実行時のログ収集スクリプト

受講生は, Gradle の Tomcat プラグインを導入したプロジェクトで Web アプリケーション開発を行う．このアプリケーションの起動と Gradle のロガーによるログを抽出し保存するスクリプトを開発し，4.2.1 と同様の方法でリモートに送信することで受講生が直面しているエラーログの情報を Teams に送信する．

4.3.2 アプリケーション実行時のログ収集スクリプトの実行方法

4.2.2 と同様にアプリケーションログ収集スクリプトも GitHub 上に配置する．受講生がエラーに直面した際に受講生の PC から HTTP 通信によってスクリプトのコードを取得し実行することで図 4 に示す情報が Teams に通知される．これにより教員，TA は受講生が直面しているエラーを容易に把握することができ，迅速にトラブルシューティングを行うことができる．

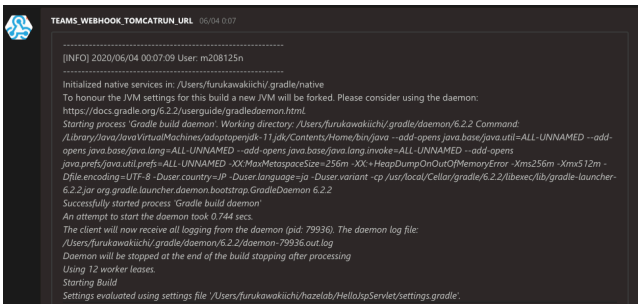

図 4　ログ収集スクリプトから Teams への通知

5. 評価

ソフトウェアの品質の観点からツールの評価を行う．4.2 と 4.3 で述べたスクリプトの構造は似ているため，スクリプトについては 4.2 の環境構築スクリプトに焦点を当てて評価を行う．我々はツールの品質を高めるためにツールの開発時に Continuous Integration (CI)を導入した．CI は変更が加わるたびにコードを検証し，不具合の発見を容易にするという考え方が根底にある．CI を実施するには CI をサポートする環境を用意する必要がある．我々はこの CI 環境には GitHub が提供する GitHub Actions を採用した．GitHub Actions は GitHub 上でホスティングされるプロジェクトに設定ファイルを記述することで CI 環境を用意することができる．これにより各スクリプトに変更が加わるたびにスクリプトを検証する．GitHub Actions を用いて環境構築スクリプトに実施した検証プロセスを図 5 に示す．スクリプトに変更が加わるたびに CI 環境で Windows OS と macOS の仮想マシンを立ち上げそれぞれのスクリプトの検証を行う．スクリプトの検証は静的解析ツールによる構文エラーなどの検証と仮想マシン上でスクリプトを実行することで期待する動作をしているかのテストを行った．4 節で述べたように各スクリプトは結果をリモートに送信するためにテキストファイルが生成される．生成されたテキストファイルを GitHub Actions の artifact 機能を利用して GitHub 上にアップロードすることにより CI 環境で生成されたテキストファイルの内容も確認できるようにしている．

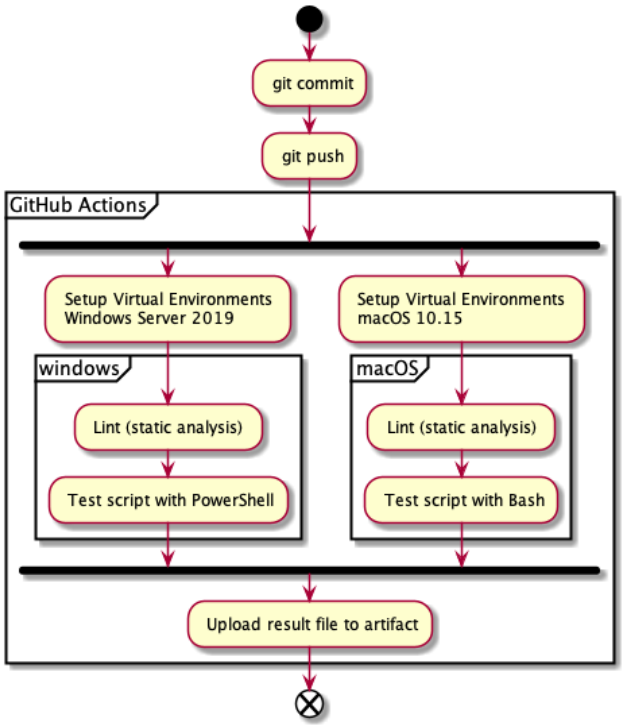

図 5　環境構築スクリプトに対する検証フロー

これらの取り組みに対して品質モデルによる評価を行う．評価に用いるソフトウェアの品質の観点は ISO/IEC 25000 シリーズ[3]の製品品質モデルと利用者時の品質モデルに基づく．

5.1 製品品質モデルに基づく評価

製品品質モデルは機能適合性，機能効率性，互換性，使用性，信頼性，セキュリティ，保守性，移植性の 8 つの品質特性からなる．使用性に関しては利用者品質と関わるため使用性を除く 7 つの品質特性に対して評価を行う．

機能適合性

機能適合性は，機能を十分に提供する度合いである．こちらは CI によって各スクリプトの検証結果で確認できる．CI による検証結果は図 6 に示すように Status Badge で確認することができる．これらが “passing”，“Success” であることよって機能適合性を満たしていると言える．

図 6　CI の検証結果を示す Status Badge

性能効率性

性能効率性は，使用する資源の量に関係する性能の度合いである．こちらは CI でのスクリプトの実行時間をもとに評価を行う．実行に最も時間がかかる環境構築スクリプトの直近5回でのスクリプトの平均実行時間を表1に示す．Windows と macOS で比較すると macOS の方が時間を要することが確認できたが実行時間は 1 分程度であり環境構築に要する時間としては十分であると考えられる．

表 1 OS 別の環境構築スクリプトの平均実行時間

OS	実行時間(s)
Windows	40.2
macOS	66.6

互換性

互換性とは，同じハードウェアやソフトウェア環境を共有しながら，他の製品やシステム，コンポーネントと情報を交換したり，必要な機能を実行したりすることができる度合いのことである[3]. スクリプトは OS ごとに作成しているため OS 間の互換性を保証できない．アプリケーションは AWS lambda のラッパー上で動作する Netlify Functions を使用しているため互換性はあると言える．

信頼性

信頼性とは，システムや製品，部品が指定された機能を指定された条件で指定された期間実行する度合いのことである．4.2.2 で述べたように，スクリプトは GitHub から取得し実行する形式であるため受講生は任意のタイミングでスクリプトを実行することができる．またスクリプトの実行結果やアプリケーションの実行ログの収集は FaaS 環境で実現しているため信頼性は高いと言える．

セキュリティ

セキュリティとは，製品やシステムが情報やデータを保護し，人や他の製品やシステムがそれぞれの種類や権限レベルに応じた適切なデータアクセスの度合いである．遠隔講義という形式もあり IP アドレスによる実行制限などは設けていない．

保守性

ツールはモジュール性を意識した設計になっている．またすべてのコードは GitHub 上で管理されており，CI で検証されている．これにより不具合の発見から修正作業までが容易であり，保守性は高いと言える．

移植性

移植性とは，システムや製品，コンポーネントをあるハードウェアやソフトウェアなどの運用環境や利用環境から別の環境に移植することができる有効性や効率性の度合いのことである[3]. 本研究では OS ごとにスクリプトを実装しているため，この特性は考慮していない．

5.2 利用者品質モデルに基づく評価

利用者品質モデルは有効性，効率性，満足性，リスク回避性，利用状況網羅性の 5 つの品質特性からなる． なお現時点(2020 年 7 月 30 日) で講義がすべて終了していないため，環境構築スクリプトによる演習環境構築に絞って評価を行った．利用者品質モデルの効率性，満足性，リスク回避性の品質特性から評価を行う上で，環境構築スクリプトによる演習環境実施について受講生にアンケート調査を行った．アンケート調査の質問項目と質問項目に対応する利用者品質モデルを表 2 に示す. すべての質問項目に対し「非常に満足」，「満足」，「不満」，「非常に不満」の 4 段階での回答を求めた．以下では各品質特性の評価を述べる．

表 2　質問項目と対応する利用者品質モデル

No	質問項目	利用者モデル
1	環境構築スクリプトによる演習環境構築について	満足性
2	環境構築スクリプトによる演習環境構築に要した時間について	効率性
3	環境構築スクリプトの実施手順書について	リスク回避性

有効性

有効性は明示された目標を利用者が達成する上での正確さおよび完全さの度合いである．これについては収集した受講生の環境構築の実施状況から評価する．環境構築スクリプトを実施した受講生は 26 名であり，実施したすべての受講生で必要なソフトウェアの導入が成功していることを確認している．このことより有効性は十分に満たせていると言える．

効率性

効率性は利用者が特定の目標を達成するために，使用する資源の度合いである．こちらはアンケート調査の質問項目 2 の結果に基づいて述べる．なおアンケート調査は受講生から 19 件の回答が得られた. 受講生からの質問項目 2 の回答結果を図 7 に示す．19 件すべての回答で「非常に満足」，「満足」の回答が得られた．このことより効率性も満たせていると言える．

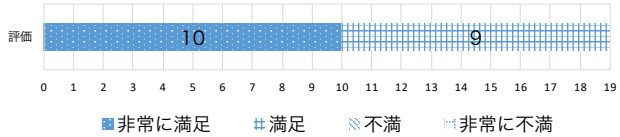

図 7　質問項目 2 の回答結果

満足性

満足性は明示された利用状況において利用者ニーズが満足される度合いである．これについてはアンケート調査の質問項目 1 の結果に基づいて述べる．受講生からの質問項目 1 の回答結果を図 8 に示す．こちらも 19 件すべての回

答で「非常に満足」,「満足」の回答が得られた．このことより満足性も満たせたと言える．

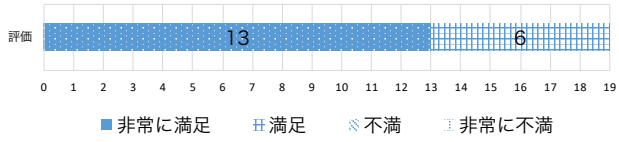

図 8　質問項目 1 の回答結果

リスク回避性

リスク回避性は潜在的なリスクを緩和する度合いである．これはアンケート調査の質問項目 3 の結果に基づいて述べる．受講生からの質問項目 3 の回答結果を図 9 に示す．19 件すべての回答で「非常に満足」,「満足」の回答が得られた．このことよりリスク回避性についても十分に満たすことができたと言える．

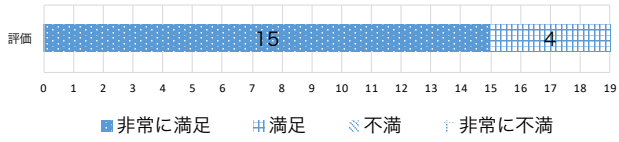

図 9　質問項目 3 の回答結果

利用状況網羅性

利用状況網羅性は，当初想定されていた状況を超えて有効性，効率性，リスク回避性および満足性を伴ってシステムが使用できる度合いであるが，想定されていた状況外での利用は確認できなかった．

6. 議論

4 節で述べた，すべての受講生のソフトウェア開発環境の構築支援と教員, TA による構築状況の把握とソフトウェア開発演習時のトラブルシューティングの機構の構築が本ツールによって達成できているかを議論する．

6.1 すべての受講生のソフトウェア開発環境の構築支援について

すべての受講生の開発環境の構築支援については, 5.2 の有効性の項目でも述べたように環境構築スクリプトを実施したすべての受講生で必要なソフトウェアの導入に成功していることが確認できている．また受講生からのツールの満足性も高く，受講生へのスクリプトの実行方法を指示する実施手順書の評価も高く開発環境の構築支援は十分に達成できたと言える．

6.2 教員，TA による構築状況の把握とソフトウェア開発演習時のトラブルシューティングの機構の構築について

ソフトウェア開発演習時のトラブルシューティングにおけるログ収集スクリプトの実行回数は現時点で少ない状況である．実行回数が少ない理由としては，スクリプト開発時の想定と実際の講義での演習方法に差異があったことが考えられる．このスクリプトは遠隔での同期的な講義時間において，複数の学生から質問があった際に対応できるように開発をしていた．しかし実際には，本講義のソフトウェア開発演習は授業時間外に行う受講生がほとんどであり Teams の個別チャットを用いた質問が多くそちらで対応をしていた．しかしながら想定していたような遠隔講義時間中に複数の受講生の質問対応を実施する際には，Teams 上に受講生の実行ログを溜めることができ，スムーズなトラブルシューティングを行う助けになると考えられる．

7. おわりに

本研究では完全遠隔によるソフトウェア開発演習を含む講義の実施について述べた．受講生の開発環境の自動構築やエラーログの収集機構は決して先進的な方法ではない．先進的な方法として，クラウドを活用し各受講生に開発環境が構築済みの仮想マシンを提供するなどがある．しかし高等教育機関において受講生の演習向けのクラウド環境の導入はコスト面の問題だけでなく，受講生にクラウドに関するドメイン知識を学ばせる必要もある．さらに COVID-19 の流行はまさに晴天の霹靂であり，我々は限られたわずかな時間で講義を完全な形で行えるように準備する必要があった．そのためシェルスクリプトや FaaS，Webhook といった方法を組み合わせることで，受講生の環境構築と演習のトラブルシューティングを円滑に行う基盤を急遽，構築した．我々のツールはあらゆる面で低コストであり，同様のソフトウェア開発演習講義においても応用可能であることを示している．

謝辞

本研究は科研費 20K12089 の助成のもとに実施している．記して謝意を示す．

参考文献

[1] Heidi J. C. Ellis, An evaluation of learning in an online project-based web application design and development course, Journal of Computing Sciences in Colleges, 21(6), pp. 217-227, 2006.
[2] Yutsuki Miyashita, Yuki Yamada, Hiroaki Hashiura, and Atsuo Hazeyama, Design of the Inspection Process Using the GitHub Flow in Project Based Learning for Software Engineering and Its Practice, arXiv:2002.02056, 2020.
[3] ISO/IEC 25000, https://www.iso.org/obp/ui/#iso:std:iso-iec:25010:en.

1 Microsoft Teams, https://microsoft.com/teams
2 Netlify, https://www.netlify.com/
3 Chocolatey, https://chocolatey.org/
4 Scoop, https://scoop.sh/
5 Homebrew, https://brew.sh/
6 GitHub, https://github.com/