2K-01

Moodle と JupyterHub を用いたプログラミング環境の構築

池田裕希† 長尾和彦†
弓削商船高等専門学校†

1. はじめに

本校情報工学科では，問題解決手法やアルゴリズムなどのプログラミング技術の習得を目標として，複数の授業でプログラミング教育を実施している．図１に授業の一連のプロセスを示す．まず，教師は紙や Moodle などを用いて課題を作成・配布する．次に，生徒は学内 PC 内の Eclipse などの開発環境を用いて課題へ取り組み，Moodle などの専用サーバにプログラムを提出する．次に，教師は提出されたプログラムをローカル環境にダウンロード・実行し，手動で採点する．最後に，教師は採点結果を手動で Moodle へ反映させる．

これらのプロセスにおける問題点として，授業における一連のプロセスで使用する環境が一つに統合されていないことによる教材管理の不便さや，学外での自学自習のための環境構築の手間が想定される．これは，生徒の負担の増加や授業進行の非効率の要因となる．

本発表では，問題点の解決を目標に JupyterHub を用いた環境構築の負担軽減，nbgrader を用いたプログラミング教育プロセスの統合・効率化，LTI_Authenticator を用いたユーザ管理の負担軽減について述べる．

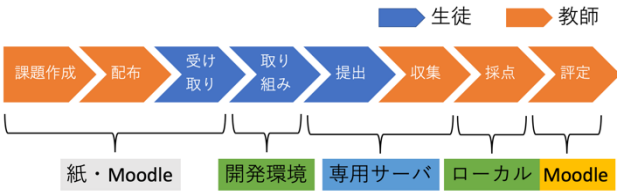


図１　プログラミング授業における一連のプロセス

2.　プログラミング環境の概要

2.1 JupyterHub によるプログラミング環境の構築

教材管理や環境構築などの負担を軽減するため，
① データのサーバ保存可能
② オンライン上で各ユーザ専用の環境が提供可能
③ プログラムの部分実行・段階的な学習ができる
という条件の元，ブラウザ上で動作する開発環境を検討した．
プログラムは条件分岐や繰り返し文など細かな要素の組み合わせで構成されている．プログラム内の要素ごとの開発を繰り返すことで，プログラミングの理解度や意欲を向上できる積み上げ式のプログラミング学習が有効であるとされる。課題の提示順序を工夫したり、インタプリタ言語を採用するなどの工夫が必要となる。

JupyterNotebook[1]などの開発環境を提供するユーザ認証付きサーバである JupyterHub[2]，Moodle 上において Quiz 形式で動作する CodeRunner[3]と VPL[4]，paiza 株式会社が提供する paiza.IO[5]を候補とした．検証環境には，リソースの消費が少なく開発の容易性から，コンテナ型仮想化プラットフォームである Docker を用いた．

全ての候補において Web ブラウザ上で対話的にプログラミングが可能である．CodeRunner と VPL は Moodle との親和性が高く手軽に導入できるが，プログラムの部分実行ができない．paiza.IO は積み上げ式での学習が困難なことに加え，テキストを交えた教材を作成できない．JupyterNotebook はセルコーディング方式によるトライ&エラーの容易性，テキストとプログラムを混在させた教材を作成可能，オープンソースや豊富なライブラリによる拡張性の高さが挙げられる．これらの理由から，JupyterHub が最適であると考えた．

2.2　LTI_Authenticator を用いた Moodle と JupyterHub 間のシングルサインオンの実装

本校学生は，学習管理用 LMS サーバである Moodle を日常的に使用している．JupyterHub は Moodle とユーザ管理が別になるため新たな負担の増加が想定される．LTI_Authenticator は各システム間の連携機能を実現するプラグインである．図２に Moodle と JupyterHub での LTI_Authenticator の実行画面を示す．Moodle 上の外部サイトリンクを押すことにより JupyterHub に自動ログインできる．ただし，LTI_Authenticator は Moodle に対応しておらず，ソースコードを変更し Moodle のユーザ情報に対応させる処理を行った[6]．

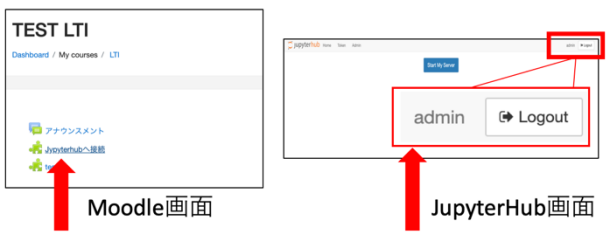


図２　LTI_Authenticator の実行画面

Building a programming environment using Moodle and JupyterHub
†Y.Ikeda,K.Nagao, National Institute of Technology , Yuge College

Copyright ©2022 Information Processing Society of Japan.
All Rights Reserved.

2.3 nbgrader を用いた課題支援環境の構築

現状の構成では，JupyterHub で作成した課題を Moodle 経由で管理する必要がある．図３に現状の課題使用までの流れを示す．まず，教師は JupyterHub で課題を作成し Moodle にアップロードする．次に，生徒は Moodle から課題をダウンロードし JupyterHub で課題を使用する．また，JupyterHub では課題を作成できるが，採点などの成績評価は行えない．

nbgrader は JupyterHub を用いた課題の管理，採点を容易にするプラグインである．JupyterHub のみで教育プロセスを実施できる課題支援環境を構築した．図４に nbgrader の課題管理画面，図５に課題採点画面を示す．課題の作成や配布，採点などを GUI 操作で容易に行える．また，nbgrader は学習記録の CSV ファイル作成機能が実装されている．Moodle の評定フォーマットに対応していないため，出力する CSV ファイルの構成を Moodle の評定用に変更する実装を行った．また，パワーポイントを用いて使用方法のチュートリアルを作成した．

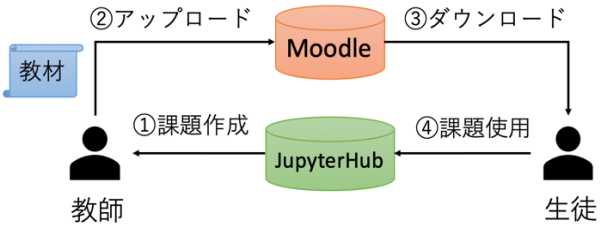


図３　現状の課題使用までの流れ



図４　nbgrader の課題管理画面

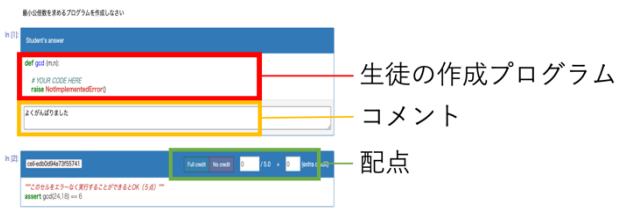


図５　nbgrader の課題採点画面

2.4 プログラミング教育担当教員へのヒアリング

作成したプログラミング環境をプログラミング教育の担当教員に説明し，

① 現状の評価
② 改善策

についてヒアリングを行った．その結果，学習支援プラグインである nbgrader による課題作成は，操作が複雑であると指摘された．また，課題作成の効率化の目的からテキスト形式などの別フォーマットファイルの課題変換機能が要望された．

3. まとめ

本研究では，プログラミング教育におけるプロセスが統合されていないことや，プログラミング開発環境の汎用性の低さから，Moodle と JupyterHub を用いたプログラミング環境を構築した．専用端末に限定されない汎用性や，各ユーザ専用の開発環境の提供，データのサーバ内保存，Moodle と JupyterHub 間のシングルサインオン，JupyterHub 上で教育プロセスが実施可能などの特徴を持つ．また，教師へのヒアリングを通じて，現システムの評価と改善策について検討した．

今後は，別フォーマットファイルの課題変換機能や学習記録の Moodle への自動反映機能の開発，授業を想定しての運用テストを行う．

参考文献

[1] Project Jupyter, “Project Jupyter|Home”, 2021, https://jupyter.org/ (11/25 閲覧)
[2] Project Jupyter “Project Jupyter|JupyterHub”, 2021, https://jupyter.org/hub (11/25 閲覧)
[3] Richard Lobb, Tim Hunt, “CodeRunner”, 2021, https://github.com/trampgeek/moodle-qtype_coderunner-code-runner (11/23 閲覧)
[4] Juan Carlos Rodríguez-del-Pino , “Virtual Programming Lab”, https://moodle.org/plugins/mod_vpl, (11/23 閲覧)
[5] paiza 株式会社 , “paiza.IO” , 2014 , https://paiza.io/ja, (11/23 閲覧)
[6] 石坂徹，桑田喜隆，合田憲人，政谷好伸，横山重俊，浜元信洲，“Moodle と JupyterNotebook の連携によるプログラミング教育環境の構築”，日本ムードル協会全国大会発表論文集，Vol. 7, No. 316 (2019), pp. 32-37.
[7] 桑田喜隆，石坂徹，政谷好伸，長久勝，横山重俊，浜元信州，“クラウドを利用した対話的なプログラミング教育環境とその評価手法の提案”，情報処理学会研究報告 2008-CE-93 (5)

Copyright ©2022 Information Processing Society of Japan.
All Rights Reserved.