Moodle と JupyterHub を用いた Web 型プログラミング教育環境の改善事例: 複数科目・複数講師による Nbgrader の共同利用

齊藤 智也 [1)], 王 躍 [1)], 西井 淳 [2)], 末長 宏康 [1)], 大平 康旦 [1)], 西村 世志人 [1)], 金山 知余 [1)], 村永 聡 [1)], 爲末 隆弘 [1)], 岡田 耕一 [1)], レール マルク [1)]

1) 山口大学 情報基盤センター

2) 山口大学 大学院創成科学研究科

info-cc@ml.cc.yamaguchi-u.ac.jp

Case Study of Improvement of Web-based Programming Educational Environment using Moodle and JupyterHub: Shared Use of Nbgrader by Multiple Subjects and Multiple Instructors

Tomoya Saito[1)], Yue Wang[1)], Jun Nishii[2)], Hiromichi Suenaga[1)], Yasuaki Ohira[1)], Yoshito Nishimura[1)], Chiyo Kaneyama[1)], Satoshi Muranaga[1)], Takahiro Tamesue[1)], Koichi Okada[1)], Marc Loehr[1)]

1) Center for Information Infrastructure, Yamaguchi Univ.

2) Graduate School of Sciences and Technology for Innovation, Yamaguchi Univ.

概要

山口大学情報基盤センターでは、Moodle といくつかの周辺システムを連携させた学習管理システムを整備・運用している。Moodle 経由で活用する JupyterHub は、全学部を対象としたデータサイエンス関連科目等において標準的なプログラミング演習環境として期待されている。特に、JupyterHub の機能拡張の一種である Nbgrader が持つ課題の配布・回収・採点の機能が注目されている。しかしながら、従来のシステムでは多数の科目が自由にかつ同時に JupyterHub 及び Nbgrader を利用するための仕組みを備えていない。本稿では、Moodle と JupyterHub を連携させたプログラミング教育環境を多数の授業科目が共同利用する場合に生じる課題、並びにそれらの改善事例について述べる。

1 はじめに

近年、大学等における情報科学の関連科目を中心として、Jupyter Notebook 及び JupyterHub [1] の活用が広まりつつある [2]。特に、Moodle 等の学習管理システムと JupyterHub を連携させたプログラミング教育環境の構築が進められている [3–5]。

山口大学情報基盤センターでは、Moodle といくつかの周辺システムを連携させた学習管理システムを整備し、全学的な講義支援サービスを運用している [6,7]。現行の Moodle 3（バージョン 3.9）システムは Kaltura、Maxima、VPL（Virtual Programming Lab）、Learning Locker に加え、JupyterHub とも連携している [8]。

山口大学では、全学部に「データサイエンス技術」等の専門教育科目を設けている。Moodle と連携した JupyterHub は、これらの授業科目における標準的なプログラミング演習環境として期待されている。医学部及び理学部では、その他の共通教育科目及び専門教育科目においても JupyterHub が活用されている。

山口大学情報・データ科学教育センターは、JupyterHub の拡張機能である Nbgrader [9,10] が持つ課題の配布・回収・自動採点の機能に着目し、前述のデータサイエンス関連科目、及び理学部等の専門教育科目における Nbgrader の活用を検討している。Nbgrader を大学の授業に導入した事例は存在するが [11]、先行事例では単独もしくは少数の教職員が各自の授業科目に利用しているのみである。複数の学部・学科にわたり任意の授業科目が自由に JupyterHub 及び Nbgrader を利用可能な教育環境の構築事例は報告されていない。

従来の JupyterHub 及び Nbgrader は、多数の科目

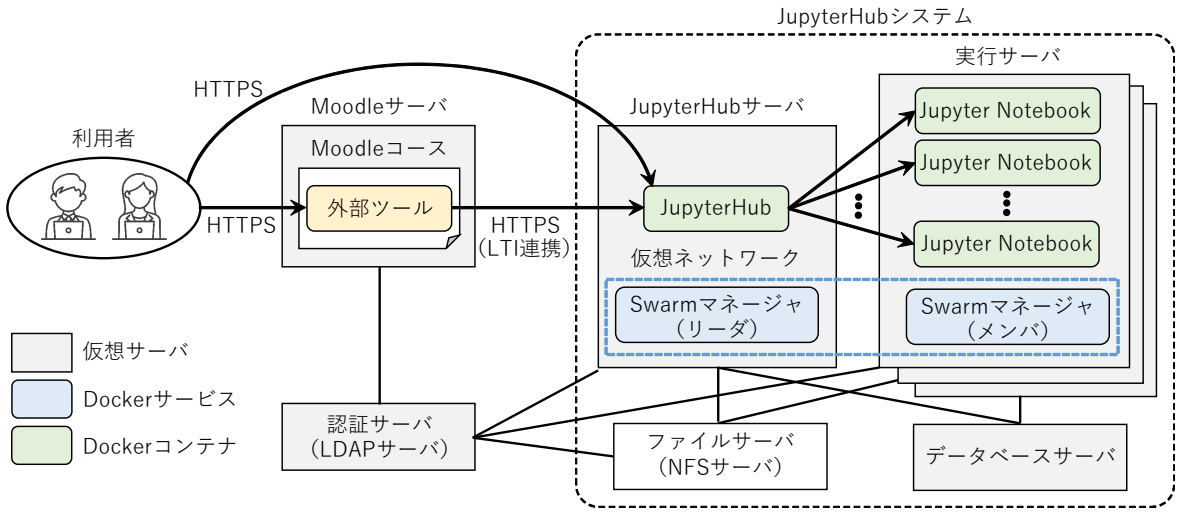


図 1: JupyterHub システム

がシステムを同時に利用しながら演習を進めるための仕組みを備えていない。加えて、Nbgrader では、各教員が担当／非担当を問わず全科目・全受講生に関連するファイル（課題内容や提出物等）の閲覧・書き込みが可能である。この仕様は、多数の学部・学科の科目で活用される環境においては、情報セキュリティ及びプライバシー保護の側面で問題となる。

そこで我々は、システム設定ファイルの修正、利用者ごとの Jupyter 環境を起動するクラス（プログラム）の修正、並びに Moodle から取得した情報の活用により、任意の授業科目・研修会等が自由に JupyterHub 及び Nbgrader を活用する仕組みを実現した。また、Nbgrader のソース・コードの修正により、Nbgrader が活用する共有ディレクトリのアクセス権限を改善した。これらの改善により、システム管理者が授業科目・授業担当者・受講生等の情報をその都度設定することなく、多数の科目が自由に JupyterHub 及び Nbgrader を活用することが可能になった。Nbgrader で取り扱われる課題や提出物等については、各教員は自身の担当科目に関するデータのみ読み書きが可能になった。各学生は、受講科目の課題内容及び自分宛てのフィードバック情報の読み込み、並びに自身が作成した提出物についての読み書きのみが可能になった。

2 JupyterHub 及び Nbgrader

2.1 JupyterHub

図 1 に、山口大学における JupyterHub システムの概要を示す。本システムは JupyterHub サーバ、実行サーバ、ファイルサーバ、及びデータベースサーバから構成される。これらのサーバは、オンプレミス環境に仮想サーバとして構築されている。仮想サーバの OS は Rocky Linux 8.6、主要なソフトウェアの各バージョンは JupyterHub 1.4.2、Nbgrader 0.6.1、及び MariaDB 10.5.15 である。JupyterHub の標準のデータベースは SQLite であるが、クライアント間の排他制御の仕組みを備えていないため、複数の実行サーバ間でデータベースを共同利用する形態には適していない。そのため、本システムでは MariaDB を採用している。

JupyterHub にはユーザ認証やコンテナ起動のための各種クラスが用意されており、各々のクラスは Python のパッケージとして提供されている。本システムでは、LTI（Learning Tools Interoperability）を用いて Moodle と JupyterHub を連携させるため、LTI 認証クラス（LTIAuthenticator クラス）を用いている。ただし、4.1 節で示すように、我々は LTIAuthenticator クラスに独自の改修を施している。

Moodle 側に特別なプラグイン等は導入せず、Moodle 3 に付属の「外部ツール」プラグインに JupyterHub のサーバに関する情報を登録している。Moodle コースの教師は、各自の Moodle コースに外部ツールのインスタンスを貼り付けておく。学生がこのインスタンスをクリックすると、JupyterHub システムの Web サイトにジャンプする。ことのき、LTI 認証の働きにより自動的に JupyterHub へのログインが完了する。Moodle との間の LTI 認証のみを有効にしているため、利用者は Moodle を経由せずに JupyterHub にログインすることはできない。

本システムでは、利用者ごとの Jupyter 環境を JupyterHub サーバとは独立した実行ノード上に

Docker コンテナとして起動する。Docker コンテナの起動・終了等を管理するためのコンテナオーケストレーションツールとして、Docekr Swarm を採用している。JupyterHub サーバ及び実行サーバでは Swarm マネージャが起動している。リーダー役の Swarm マネージャはメンバの中から 1 つを選択し、そのメンバに Docker コンテナを起動させる。

利用者の Jupyter 環境を起動する仕組みは、Spawner クラスとして提供される。Docker Swarm に対応するための SwarmSpawmer クラスでは、利用者の Docker コンテナ内におけるユーザ・アカウントは一律に「jovyan」となるため、各利用者のデータをファイルサーバに記憶する際に、ファイル等の所有者を実際のユーザ・アカウントに設定することが困難である。そこで我々は、実際のユーザ・アカウントとの対応付けが可能な SystemUserSpawner クラスと SwarmSpawner クラスのソース・コードを応用した独自の Spawner クラスを作成している。大学の認証サーバと連携することにより、各利用者が Moodle 等を利用する際のユーザ名、ユーザ ID に加え、教職員／学生の別をアクセス権限の設定に活用している。これらの改修は、システム管理者が各利用者のファイルを識別しやすくするためだけでなく、後述する Nbgrader の問題点の改善のためでもある。

2.2 Nbgrader

Nbgrader は、JupyterHub において課題の配布・回収・採点を効率的に行うための機能拡張である [9,10]。

図 2 及び 3 は、教師側の Nbgrader の画面例である。図 2 は課題一覧の画面を示している。教師はこの画面を利用して、課題（Notebook ファイル）の作成、課題の公開、提出物（提出された Notebook ファイル）の取得、及びフィードバックの作成・公開の操作を行う。設定の難易度は高いが、自動採点の機能も有している。

図 2 において、教師が提出数の数字をクリックすると、図 3 に示す提出物一覧の画面が表示される。この画面では、個々の提出物について採点、及びフィードバックの作成・公開が可能である。

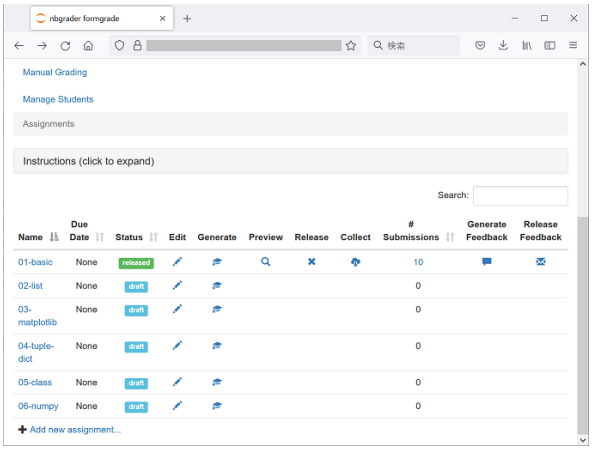


図 2: Nbgrader の課題一覧

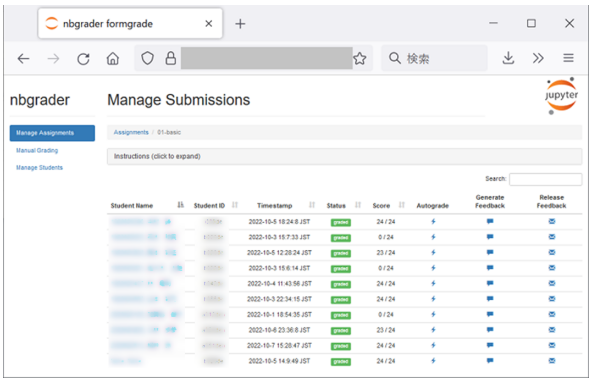


図 3: Nbgrader の提出物一覧

3 従来のシステムの課題

3.1 複数科目への対応の問題点

Nbgarder を利用する場合、システム設定ファイルのパス（例えば「/etc/jupyter」）、及び教師のホームディレクトリ以下の所定の場所に「nbgrader_config.py」という設定ファイルを作成する必要がある。図 4 は、「nbgrader_config.py」に記載すべき設定項目の中から本研究に関連する項目を抜粋したものである。共有ディレクトリのパスはシステムのすべての利用者に共通した設定内容となり、運用中に変更されることは稀である。従来のシステムでは学生向けに定義される設定項目は共有ディレクトリのパスのみで、その他の設定項目は主に教師用の設定ファイルに記載される。ただし、個人ディレクトリのルート、成績ファイルのパス、及びコースの学生一覧は授業科目ごとに異なるため、教師は授業の度に設定内容を書き換えなければならない。

多数の科目が自由に JupyterHub 及び Nbgrader を使用するためには、授業科目毎の設定ファイルを自動生成する仕組みが必要である。

Nbgrader の公式ドキュメントでは、JupyerHub の設定ファイルの記載内容によって、Nbgrader のコース毎に Formgrader サービスを起動し、複数の科目に対応する手順が紹介されている [12]。この手順では、システム管理者は JupyeteHub の設定ファイルを編集し、コース毎の Formgrader に対し、専用の通信ポート番号、及びそのポート番号にアクセス可能なユーザ（教師及び学生）の一覧を記述する。大学全体の

```
# 共有ディレクトリのルート
c.Exchange.root = '/.../exchange'
# コース名
c.CourseDirectory.course_id = 'course1'
# 個人ディレクトリのルート
c.CourseDirectory.root = '/home/instructor1/course1'
# 成績ファイルのパス
c.CourseDirectory.db_url
    = 'sqlite:////home/instrcutror1/course1/gradebook.db'
# コースの学生一覧
c.CourseDirectory.db_students = [
  dict(id="r001aa", first_name="Jiro", last_name="Yoshida", ...),
  dict(id="r002aa", first_name="Hanako", last_name="Tokiwa", ...),
  dict(id="r003aa", first_name="Saburo", last_name="Kogushi", ...),
  ...
]
```

図 4: Nbgrader の設定ファイルの一部

多数の授業で Nbgrader を用いる場合、システム管理者は JupyerHub を使用し得るすべての授業に対して Formgrader のポート番号を設定し、それらの科目の受講生を記述しなければならない。学期途中での受講生の増減、及び Nbgrader の利用科目の増加に応じて設定ファイルを編集する必要があるほか、設定の変更をシステムに反映させるためには、JupyterHub サービスの再起動が必要になる。また、この手順では、コースの全参加者の Jupyter 環境から該当の Formgrader の通信ポートにアクセス可能にする必要がある。設定ファイルに登録する科目数に応じて多数の通信ポートを開き、そのポートへの接続要求を受け入れなければならない。このような仕組みは、重大なセキュリティ・ホールになる恐れがある。

以上の理由により、本研究では公式ドキュメントにおいて紹介されている手順は採用していない。

3.2 共有ディレクトリの権限設定

Nbgrader では、教師・学生のホームディレクトリに中に配置される個人ディレクトリと、共有ディレクトリの間のファイルのやり取りによって課題の配布・回収等の機能が実現されている。図 5 は、共有ディレクトリの概要を示している。なお、共有ディレクトリの利用形態及びアクセス権限に焦点を当てるため、教師の個人ディレクトリ及び学生の課題ディレクトリの詳細は省略している。

Nbgrader では、課題に関連するファイルは共有ディレクトリ（図 5 中の exchange）の配下に作成されるコース・ディレクトリ（図 5 中の course1 及び course2）の中にまとめて記憶される。標準ではコース・ディレクトリは自動的に作成されないため、教師は JupyterHub 上で端末（ターミナル）を起動した後、コマンド操作により任意の名称のコース・ディレクトリを作成する。

JupyterHub 上で教師が課題の公開の操作を行うと、課題の Notebook ファイルは教師のホームディレクトリの配下から、コース・ディレクトリの配下にある outbound ディレクトリの中にコピーされる。教師の個人ディレクトリには一般にコース名が使用され、その中に課題ごとのファイルが保存される。標準では、学生の個人ディレクトリにはコースの区別が無く、また、ホームディレクトリそのものが個人ディレクトリとなる。学生が課題の取得の操作を行うと、ホームディレクトリの直下に課題名のディレクトリが作成され、取得したファイルはその中に保存される。

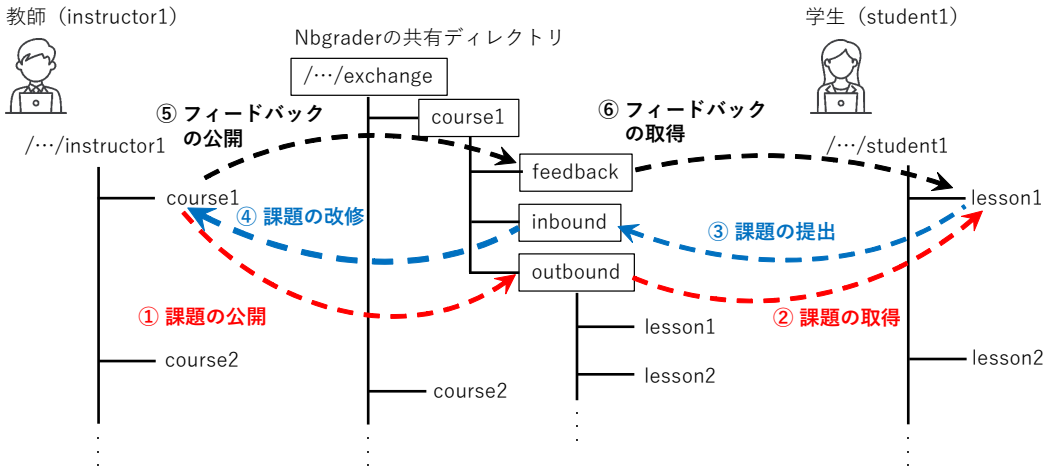
学生は取得した Notebook ファイルを編集して解答などを記入した後、課題提出の操作を行う。このとき、Notebook ファイルは提出物として、inbound ディレクトリの中にコピーされる。教師が提出物の回収の操作を行うと、提出物の Notebook ファイルが inbound ディレクトリから教師の個人ディレクトリにコピーされる。これにより提出物の手動／自動採点が可能になる。

教師が作成・公開した各学生へのフィードバック・ファイルは、feedback ディレクトリに保存される。学生は、自分宛てのフィードバックが公開されている場合には、JupyterHub の画面においてそれらの情報を閲覧することが出来る。

図 6 は、共有ディレクトリの所有者、グループ、及びアクセス権限を示している。なお、Nbgrader では教師と学生を含む全ユーザが同一のユーザ・グループに所属する必要があり、図 6 ではそのグループ名を users としている。

従来の Nbgrader では、共有ディレクトリの権限設定に大きな問題点がある。教師が自由に Nbgrader のコースを作成するため、exchange ディレクトリはすべてのユーザからの書き込みが許可されている。Linux 系 OS では、いわゆる「777」のディレクトリについては、全ユーザがその中にあるサブディレクトリを削除することが可能である。教師はコマンド操作によりコース・ディレクトリを作成するため、JupyterHub の端末（ターミナル）等を用いてコマンドライン上から exchange ディレクトリを操作する。コース設定や課題の設定の不備により、教師が既存のコースのディレクトリを削除するケースもあるが、このとき、操作を間違えると他の授業科目のコースを削除してしまう。学生にも JupyterHub の端末（ターミナル）の利用を許可している環境においては、学生がコース・ディレクトリを削除することも可能である。

inbound ディレクトリは全ユーザから書き込み可能



※「course1」及び「course2」はNbgraderにおけるコース名。
「lesson1」及び「lesson2」はcourse1の中の課題名。

図 5: Nbgrader の共有ディレクトリ

であるが、ユーザはこのディレクトリ内にあるファイルの名称等を取得できないため、すべての提出物を一括削除することはできない。また、inbound に保存される提出物のファイル名は課題名、ユーザ名、提出日時に加えてランダムな文字列が追加されているため、ユーザは特定のファイルの上書きや削除を行うことは出来ない。

feedback ディレクトリについてはグループ・ユーザもその他のユーザも、ディレクトリ内にあるファイルの名称等を取得することは出来ない。提出物と同様に、フィードバック・ファイルのファイル名にもランダムな文字列が使用されている。学生側のユーザ・インターフェースは、データベースを介してその学生宛てのフィードバック・ファイルの名称を取得する。

Nbgrader を多数の科目で利用するためには、共有ディレクトリの権限設定を改善しなければならない。また、学生の利用環境ではホームディレクトリの直下に課題名のディレクトリが作成されるため、複数の科目で課題名が重複した場合には課題提出の操作に不具合が生じる。そのため、個人ディレクトリの作成規則に関する改善も必要である。

4　改善事例

4.1　複数科目に対応するための改善

複数の授業科目が自由に JupyterHub 及び Nbgrader を利用するため、我々は LTI 認証クラス（LTIAuthenticator クラス）を独自に改修した。

図 7 は、作成した認証クラスの機能を示している。利用者が Moodle コースに配置された外部ツールを経

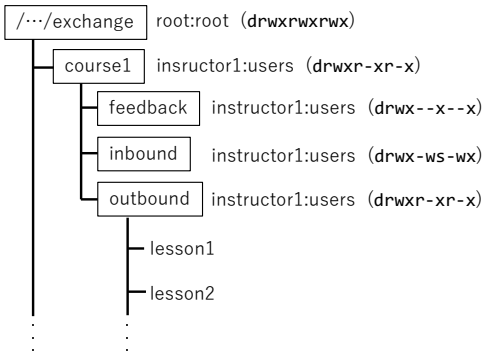


図 6: 共有ディレクトリの権限設定

由して JupyterHub にログインすると、本クラスは、Moodle から引き渡されたパラメータの値を確認する。Moodle コースにおいて「教師」のロールを持つユーザの場合、roles パラメータの値が「Instructor」となっている。その場合、Moodle サーバからコースの学生一覧を取得する（図 7①）。

次に、本クラスは大学の認証サーバ（LDAP サーバ）から、利用者の認証サーバにおけるユーザ ID、及び教職員・学生の別を取得する（図 7②）。大学の認証サーバに登録されてないアカウント（Moodle の手動アカウント）が JupyterHub へのログインを試みた場合には、ここでエラーが発生し、ログインに失敗する。

続いて、本クラスは、利用者のホームディレクトリが存在しなければ新規に作成する（図 7③）。ホームディレクトリを新規作成する場合には、認証サーバ取得したユーザ ID を使用する。また、ユーザの教職員／学生の別に応じて、ディレクトリの所属グループを「teachers」もしくは「students」に設定する。

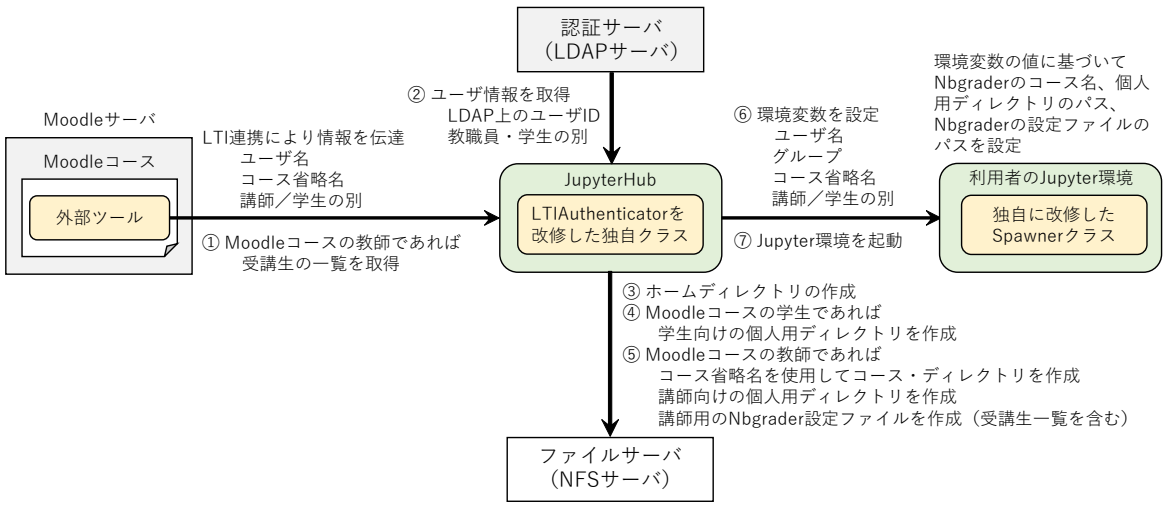


図 7: 独自に改修した認証クラスの機能

続いて、利用者が Moodle コースの学生の場合、そのコース用の個人ディレクトリが存在しなければ作成する（図 7④）。

利用者が Moodle コースの教師で、かつ、共有ディレクトリの配下に Nbgrader 用のコース・ディレクトリが存在しない場合、コース・ディレクトリが新規に作成される（図 7⑤）。コース・ディレクトリの名称には Moodle コースにおけるコース省略名を用いている。山口大学では教務システムとの連携により、通常の授業科目に対応する Moodle コースのコース省略名には年度及び時間割番号が含まれるため、授業科目の間で Nbgrader のコース名が重複することは無い。また、ホームディレクトリの配下にそのコースの個人ディレクトリが存在しない場合は新規に作成される。加えて、学生一覧の情報、及び成績ファイルのパスに関する情報等を含む教師用の Nbgrader 設定ファイルを、そのコースの個人ディレクトリの中に作成する。新たな履修登録に伴う学生の新規登録や、履修登録の取り下げに伴う学生の登録解除に対応するため、この設定ファイルは教師が JupyterHub にログインする度に新規に作成される。

本クラスは、利用者の Jupyter 環境で用いられる環境変数を設定した後、利用者の Jupyter 環境を起動する（図 7⑥及び⑦）。

Docker Swarm 環境において利用者毎の仮想的な Jupyter 環境を起動することと、Jupyter 環境内のユーザ・アカウントとシステム外（ファイルサーバ等）におけるユーザ・アカウントを対応させることを両立するため、我々は従来の SystemUserSpawner クラスと SwarmSpawner クラスのソース・コードを応用した独自の Spawner クラスを作成している。

図 8 に、Moodle コースにおける教師・学生が共通で読み込む設定ファイルの一部を示す。この設定ファイルは「/etc/jupyter/nbgrader_config.py」として配置されている。MOOCLECOURSE（Moodle コースのコース省略名）、COURSEROLE（利用者の Moodle コース内におけるロール）、及び HOME（利用者のホームディレクトリのパス）といった環境変数を用いることにより、ユーザ毎の個人ディレクトリのパスが設定される。

従来のシステムでは教師・学生共に共有ディレクトリのパスは設定されるが、その他の設定項目については各教師が自身のホームディレクトリの配下に置いた設定ファイルに記載するため、利用者が学生の場合には Nbgrader のコース名は設定されない。学生用の Nbgrader の画面では共有フォルダの配下にあるすべてのコース・ディレクトリを閲覧可能である。学生はリストボックスに並ぶコース名の一覧の中から現在の授業に応じたコース名を選択し、その後に表示される課題について各種の操作を行う。Nbgrader の利用科目が増加した場合、学生は大量に表示されるコース名の中から正しいコース名を選択しなければならない。そこで本研究では教師・学生に共通の設定ファイルを作成し、学生に対しても Nbgrader のコース名を設定している。これにより、Jupyter 環境の起動時に Moodle コースに対応した Nbgrader のコースが選択済みとなり、他のコース名は表示されなくなるため、誤ったコースが選択されることが無くなる。

図 9 に、教師用の Nbgrader の設定ファイルを示す。このファイルにはそのコースに応じた学生一覧、及び成績ファイルのパスが含まれている。このファイルの内容は、図 7①に示した動作により、教師が

```
# 共有ディレクトリのルート
c.Exchange.root = '/.../exchange'
# 環境変数を使用してコース名を設定
c.CourseDirectory.course_id = os.environ.get('MOODLECOURSE')
# 学生の個人ディレクトリのルートを設定
c.CourseDirectory.root = os.environ.get('HOME')
   + os.environ.get('MOODLECOURSE')
# Moodleコースにおけるロールが教師の場合
if os.environ.get('COURSEROLE') == 'Instructor':
   # 教師用の個人ディレクトリのルートを設定
   c.CourseDirectory.root = os.environ.get('HOME')
      + '/nbgrader/' + os.environ.get('MOODLECOURSE')
```

図 8: 改善後の共通の設定ファイルの一部

```
# 成績ファイルのパス
c.CourseDirectory.db_url
   = 'sqlite:////home/instrcutror1/course1/gradebook.db'
# コースの学生一覧
c.CourseDirectory.db_students = [
  dict(id="r001aa", first_name="Jiro", last_name="Yoshida", ...),
  dict(id="r002aa", first_name="Hanako", last_name="Tokiwa", ...),
  dict(id="r003aa", first_name="Saburo", last_name="Kogushi", ...),
  ...
]
```

図 9: 改善後の教師用設定ファイルの一部

JupyterHub にログインする度に自動的に作成され、その時点での Moodle コースの学生一覧が適用される。

以上に示した仕組みにより、本システムでは多数の科目が自由に JupyterHub 及び Nbgrader を利用することが可能になる。

4.2 アクセス権限の改善

図 10 は、改善後の共有ディレクトリの概要を示している。「course1」というコースにおいては、教師の個人ディレクトリの例は「/.../instructor1/nbgraer/course1」となり、学生の個人ディレクトリの例は「/.../student1/course1」となる。学生用の課題ディレクトリがホームディレクトリの直下に作成されることを防止し、複数の科目で Nbgrader が利用される場合の利便性の向上を図っている。山口大学では教職員を対象とした JupyterHub に関する研修会も開催されるため、教職員が Moodle コースでは学生として JupyterHub 及び Nbgrader を利用する機会もある。そのため、Moodle コースにおけるロールが教師か学生かにより、個人ディレクトリのパスが別々となるように設定されている。

図 11 は、改善後の共有ディレクトリの権限設定を示している。ルートとなる exchange ディレクトリの権限を「755」に変更し、教師や学生によるコース・ディレクトリの削除を回避している。この権限設定では、教師は手動でコースファイルを作成することが出来なくなるため、前節で述べたように、Moodle コースから取得した情報に基づいて自動的にコース・ディレクトリが作成される。

コース・ディレクトリは授業担当教員のみが書き込貴可能となることが望ましい。教師のユーザ・アカウントが teacher1 のとき、コース・ディレクトリの所有者及び所属グループは「instructor1:teachers」となる。feedback、inbound、及び outbound の各ディレクトリについては、所属グループを students としている。これは、Nbgrader では inboundd ディレクトリに対して、学生が授業担当教員の権限を用いて提出物を書き込む必要があるためである。

以上の対策により、本システムでは多数の科目が Ngrader を共同利用する場合における共有フォルダのアクセス権限の問題を改善している。

5 まとめ

本稿では、Moodle と JupyterHub を用いた Web 型プログラミング教育環境の改善事例について述べた。JupyterHub の機能拡張である Nbgrader を用いることにより、教師は JupyterHub の GUI を経由して課題の配布・回収・採点が可能になる。従来のシステムでは、Nbgrader を利用するクラスのコース名及び受講者一覧を JupyterHub のシステム設定ファイルに記述する必要がある。また、複数の科目が同時に Nbgrader を利用する場合、システム設定ファイルにおいて科目ごとに固有の通信ポートを割り当てる必要がある。そのため、任意の科目が自由に Nbgrader を利用することが出来なかった。

そこで本研究では、システム設定ファイルを大学の教職員やシステム管理者が編集することなく、多数の科目が自由に Nbgrader を利用するための改善を行った。改善したシステムでは、利用者が経由した Moodle コースに応じて、利用者ごとの Jupyter 環境に自動的にその授業に適した設定が施される。また、Moodle から取得した授業情報及び及び認証サーバから取得したユーザ情報に基づき、利用者の所属グループ及び共有ディレクトリのアクセス権限が自動的に設定される。これにより、従来のシステムが抱えていたセキュリティ面の課題が改善されている。

山口大学では 2022 年度より、全学部において「データサイエンス技術」等の専門教育科目のデータサイエンス関連科目が開講されている。本研究により開発されたシステムは、これらの授業科目における標準的なプログラミング演習環境として活用されている。

今後の課題として、複数の教師が 1 つの科目を担当する場合における、Nbgrader の共有ディレクトリのアクセス権限に関する検討が挙げられる。

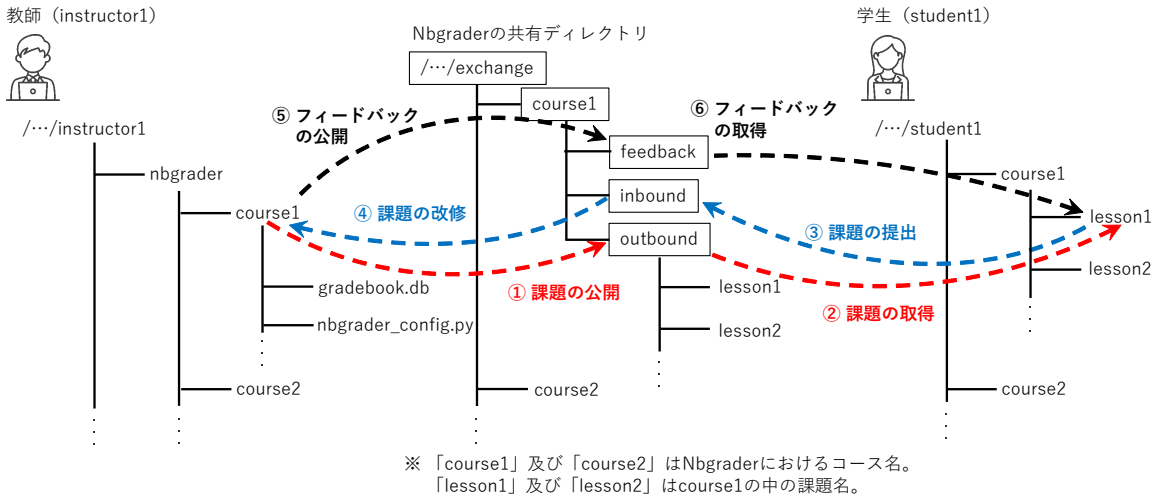


図 10: 改善後の共有ディレクトリ

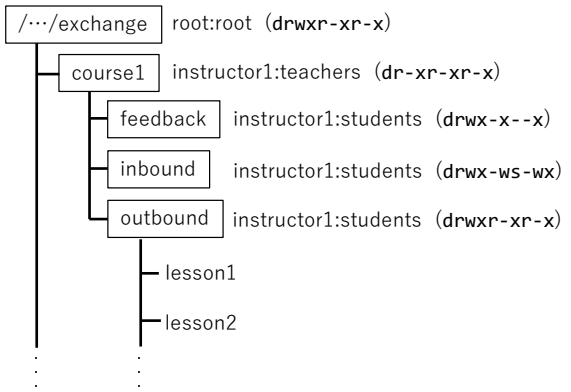


図 11: 改善後の権限設定

参考文献

[1] Project Jupyter: JupyerHub, https://jupyter.org/hub （2022 年 10 月 5 日確認）.

[2] 山口健二,「Google Colaboratory による自宅学習教材の作成と遠隔授業方法の検討」, 高等教育と学生支援 : お茶の水女子大学紀要, vol.10, pp.59-63 （2019）.

[3] 井関文一 他,「LTI カスタムパラメータによる Moodle - JupyterHub 連携に関する研究」, 日本ムードル協会全国大会（2022）発表論文集, pp.20–25 （2022）.

[4] 池田裕希 他,「Moodle と JupyterHub を用いたプログラミング環境の構築」, 情報処理学会第 84 回全国大会 （2022）.

[5] 石坂徹 他,「Moodle と Jupyter Notebook 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（2019）発表論文集, pp.32–37 （2019）.

[6] Moodle: Open-source learning platform, https://moodle.org/ （2022 年 10 月 5 日確認）.

[7] 齊藤智也 他,「山口大学における遠隔授業の増加に伴う授業支援システムの性能改善」, 第 24 回学術情報処理研究集会（2020）.

[8] 齊藤智也 他,「山口大学における Moodle システムの性能改善及び機能拡張」, 日本ムードル協会全国大会 （2022）.

[9] nbgrader, https://nbgrader.readthedocs.io/en/stable/ （2022 年 10 月 5 日確認）.

[10] Nik Klever,「Jupyter Notebook, Jupyter-HJub and Nbgrader」, https://klever.hs-augsburg.de/nb/OWL/LAB_2020_02_Klever_Jupyter_Notebook_JupyterHub_Nbgrader.pdf （2022 年 10 月 5 日確認）.

[11] 田浦健次朗,「Jupyterhub と nbgrader で授業・試験・レポートを DX してみた件」, 第 39 回大学等におけるオンライン教育とデジタル変革に関するサイバーシンポジウム （2021）.

[12] Using nbgrader with JupyterHub, https://nbgrader.readthedocs.io/en/stable/configuration/jupyterhub_config.html (2022 年 10 月 5 日確認).