Moodle と学務情報システムのデータ連携
ー山口大学のケーススタディー
Case Study of Sharing Data Between Moodle and Educational Support System at Yamaguchi University

王躍†，小柏香穂理†，久長穣†，為末隆弘†，小河原加久治†
YUE WANG†, KAHORI OGASIWA†, YUTAKA HISANAGA†,
TAKAHIRO TAMASUE†, KAKUJI OGAWARA†

wangyue@yamaguchi-u.ac.jp, ogashiwa@yamagchi-u.ac.jp,
hisa@yamaguchi-u.ac.jp, tamesue@yamaguchi-u.ac.jp,
ogawara@yamaguchi-u.ac.jp

†山口大学・メディア基盤センター

†Media and Information Technology Center, Yamaguchi University

概要

本稿では山口大学においてメディア基盤センターが教育支援システムとして運営している Moodle サイトと，大学教育センターが運営してる学務情報システムとのデータ共有による Moodle コースの自動作成について述べる．これにより Moodle の利用手順が簡素化され，利用者の利便性が大きく向上されることになる．

キーワード

LMS，学務情報システム, Moodle，データ共有

1. はじめに

学習管理システム (Learning Management System : LMS)として Moodle は世界中の教育機関に活用されている．山口大学では，初期バージョンの Moodle の試験運用を経ってから，現在，教育支援システムとして，メディア基盤センターが Moodle サイトを運営している．Moodle はコースベースでの学習管理を行うため，講義の担当者が Moodle コース作成の申請を提出して，Moodle サイト管理者がそのコースを作成する．また，Moodle コースの参加者の登録は，基本的にコースの担当者に任せるのが一般的である．しかし，これらの作業は，サイトの管理者にとっても，コースの担当者（特に初めて Moodle を利用する者）にとっても一定の負担になる．一方，ICT とインターネットの発展に伴い，多くの大学では，履修情報・シラバス・成績管理などを行う学務情報システムは，基幹業務システムの１つとして広く利用されている．学務情報システムに登録されている講義履修情報を利用して Moodle コースの自動生成ができれば，Moodle の利用手順が簡素化さ

れ, 利用者の利便性が大きく向上されることになる[1]．本稿では，山口大学修学支援システムと呼ばれる学務情報システムと Moodle とのデータ共有による Moodle コースの自動作成について述べる．両方のシステムは，それぞれ，大学教育センターと大学情報機構メディア基盤センターが運用されているものである．また，メディア基盤センターでは ISMS 認証を取得しているため，本稿は，ISMS の観点で言えば，ISMS 適応範囲でのシステムと ISMS 適応範囲外のシステムとのデータ共有の事例になる．

2. 要件定義

本稿の目的は，学務情報システム（修学支援システム）に登録されている講義履修情報に基づいて Moodle コースの作成やコースの参加者登録を自動的に行うことである．なお，本稿での情報システムのユーザ ID はすべて統一されているとする．つまり，システム間のユーザマッピングは不要である．

2.1 コース作成

講義担当者（一人に限らない）が Moodle にログインする際，それぞれの担当講義に対して，もし対応する講義の Moodle コースが存在しない場合は．該当コースを新規作成する．ただし，自動作成されたコースの公開について担当者に任せるため，初期状態では非公開になっている．

2.2 ロール登録

講義担当者が Moodle にログインする際，それぞれの担当講義に対して，もし対応する講義の Moodle コースが存在しない場合は，該当コースを新規作成してから, そのコースの教師ロールに追加する．もし，既に該当 Moodle コースが登録されて，かつ，教師ロールとしてそのコースにまだ登録されていない場合は, そのコースの教師ロールに追加する．講義履修学生が Moodle にログインする際，それぞれの履修講義に対して，もし該当講義の Moodle コースが既に登録されて，かつ, 学生ロールとしてそのコースにまだ登録されていない場合は, そのコースの学生ロールに追加する．

2.3 必要なデータ

上述の「コース作成」と「ロール登録」を行うには学務情報システムからの各講義に関する「ユーザ ID」，「コース ID」，「ロール」などの情報が限必要である．なお，これらの情報は参照（ReadOnly）のみに使われるので, 学務情報システムでの編集権限は一切不要である．

3. システム設計件

本稿では, セキュリティ上や運営上などの考慮で, Moodle から修学支援システムに直接にアクセスするのが望ましくないため, 修学支援システムとデータ同期を行うシステムを用いて Moodle との連携を実現する．この同期システムは，修学支援システムと Moodle との間にクッションの役割を果たすもので, 修学支援システムに１日１回アクセスし, データの同期を行う．実際に毎日の AM3:00～AM6:00 の時間帯は修学支援システムのメンテナンス時間となっているので, この時間帯を利用してデータの同期を行っている．また，Moodle から同期システムへのアクセスは，Moodle の内部プラグインを通してリアルタイムに行う（図_１）．

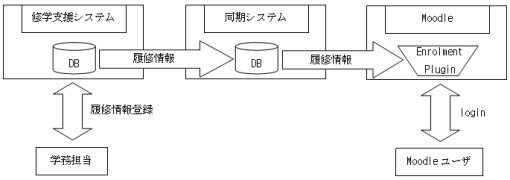

図_1．システム構成図（概略）

3.1 同期システム DB のスキーマ

修学支援システムに登録されている履修情報を Moodle 登録用の１つの DB テーブルにまとめて，テーブルの各レコードは１ユーザ・１科目の履修情報を表す．通常，１人のユーザは複数のレコードをもつことができる．具体的に，レコードに含まれるフィールドは表_1 に示す．

なお，フィールド「shortname」の値は Moodle ではユニークでなければならないため, その科目の[開講年度].[開講学期].[時間割コード]から構成されることにした．また，コース要約としてのフィールド「note」の値は，その講義を行う「建物名」「教室」「曜日」「時限」などの情報が含まれ

ている．

表_1. データ項目一覧

フィールド名	意味
username	ユーザ ID
role	ロール (student or editingteacher)
fullname	講義題目名（コース名）
shortname	コース省略名
faculty	部局名
department	学科名
year	開講年度
term	開講学期
note	コース要約

3.2 登録コースのカテゴリ

修学支援システムの履修情報に基づいて自動登録される Moodle コースは，次ようなカテゴリに従って作成される．

- 修学支援
  - [開講年度]
    - [開講学期]
      - [部局名]
        - [学科名]
          - <コース名>

ただし，これらのカテゴリ（「修学支援」を除く）の作成は，講義担当者が Moodle にログインしたとき，Moodle コースを登録する前に自動的に行う．

3.3 Moodle データの追加制約

修学支援システムの履修情報を基づいて自動作成された Moodle コースは，コース管理者（教師）に最大の自由度を与えるため，修学支援システムとのデータ連携がデータの追加のみにする．つまり，ユーザが Moodle にログインするとき，ユーザが所属しているそれぞれの Moodle コースに対して，まだ登録されていないデータがあれば，それを追加するが，Moodle コースにあって修学支援システムにないデータの削除はしない．なお，修学支援システムのデータを参照するためのキーはコース省略名の「shortname」である．従って，自動作成された Moodle コース名の変更は自由にできるが，コース省略名の変更は基本的にできない．（コース省略名は変更されると，同じコースが再登録されるだけではなく，他のコースと衝突される可能性もある．）

3.4 データのライフサイクル管理

自動登録される Moodle コースは当年度のものに限る．また，カテゴリ「修学支援」に登録されているコースは，過去４年までである．それ以外のコースは，一旦「過年度」カテゴリに移行してから必要に応じて削除を行う．

4. システム実装

図_1 に示したデータの流れに従ってシステムの実装を述べる．

4.1 修学支援システムから同期システムへ

同期システムは，修学システムから特定のデータのみを抽出する役割を果たすと同時に，公開できるデータの制御やアクセス制限なども行う．

なお，修学支援システムとのデータの同期は Cron スクリプトで一日一回自動的に取るようにしている．

4.2 同期システムから Moodle へ

本稿で実装に利用されている Moodle のバージョンは 1.9 となっているが，Moodle2 においても同様に実現できることが確認済みである．Moodle のコースユーザ登録は，外部データベースを使用して行うことができる．本稿は Moodle1.9 の外部データベース登録プラグインをカスタマイズしてコースの自動登録を実現する．オリジナルの外部データベース登録は，ユーザが Moodle にログインした時点で外部データベースのデータに基づき，すべての該当するコースに対してユーザの自動登録の試みを行う．コースが存在しない場合，（設定で指定されているカテゴリに）空のコースを作成してから登録を行う．また，ユーザがデータベースに存在しない場合，この処理ではコースからユーザを登録解除する．なお，このプラグインは Moodle サイトに登録されているユーザのみに対して動作する．

しかし，標準の Moodle 外部データベース登録プラグインは次のような２つの問題点がある [2]．(1) コースが本来所属しているカテゴリの作成機能はないので，全てのコースは同じカテゴリ

に登録されていること．(2) 外部データベースのデータと完全同期を取るので，外部データベースのデータに強く依存していること．そのため，データベースからデータが喪失した場合，ユーザが一部またはすべてのコースから登録解除されてしまう可能性がある．

前者の問題はプラグインにカテゴリの自動作成機能を追加して解決される．後者の問題はユーザ登録解除を行わないようにプラグインの仕様を変更して解決される．これによって例え外部データベースのデータを参照できなくなってもMoodle が通常に動作することができる．その代わりに，コースのユーザ登録解除はコースの担当者に任せることになるが，それほど問題はないと思われる．

4.3 ソースのカスタマイズ

Moodle1.9 の外部データベース登録プラグインのソースプログラムは，Moodle のディレクトリ「moodle/enrol/database」にある３つのファイル

- config.html
- enrol.php
- enrol_database_sync.php

から構成されている．ここで，config.html はサイト管理者用の外部データベース登録プラグインの設定ファイルである．enrol.php はこのプラグインの本体で，その中に定義されている関数 setup_enrolments はユーザが Moodle サイトにログインしたとき，Moodle コアに（一度だけ）自動的に呼び出されて実行される．なお，enrol_database_sync.php は本稿では利用しない．

4.3.1 config.html のカスタマイズ

同期システム DB のスキーマに含まれるフィールドマッピングのため，config.html（150 行あたり）に必要な項目を追加する．例えば，「開講年度」（year）に対応する項目は次のようになる．

```
<tr>
  <td align="right">
    enrol_db_remoteNENDOfield:
  </td>
  <td>
    <input
      size="15"
      type="text"
      name="enrol_db_remoteNENDOfield"
      value="<?php
      echo $frm->enrol_db_remoteNENDOfield
           ?>"
    />
  </td><td></td>
</tr>
```

変更後のプラグイン設定画面（一部）は図_2 に示す．

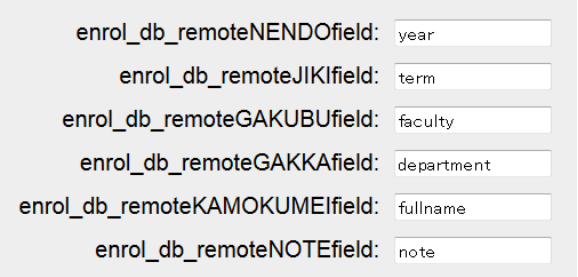

図_2．外部データベース登録プラグインの設定画面（一部）

4.3.2 enrol.php のカスタマイズ

本稿での enrol.php ファイルのバージョンは，$Id: enrol.php,v 1.42.2.8 2010/12/22 07:49:11 moodlerobot Exp $ となっている．enrol.php に定義されている関数 setup_enrolments（54 行あたり）において，同期システム DB テーブルからコース登録データを抽出するために，次の SELECT ステートメント（64 行あたり）を実行する．

```
if ( $rs = $enroldb->Execute("
    SELECT * FROM {$CFG->enrol_dbtable}
      WHERE {$CFG->enrol_remoteuserfield}
       = " . $useridfield .
        ( isset($remote_role_name,
          $remote_role_value) ? ' AND ' .
          $remote_role_name . ' = ' .
          $remote_role_value : ' ' ) ) )
{
 // We'll use this to see what to add and remove
      .....
}
```

コースカテゴリを作成するには次のようなコードを追加する．例えば，[開講年度]カテゴリは以下のコードで登録される．

```
// let's create the parent_category of the course
// if necessary
$template = array(
            'sortorder'          => 999,
            'coursecount'        => 0,
            'visible'            => 1,
            'timemodified'       => time(),
);
$NENDO_category =
get_record('course_categories',
   'parent', $CFG->enrol_db_category, 'name',
   addslashes($fields_obj->{
      $CFG->enrol_db_remoteNENDOfield}));
if (!is_object($NENDO_category))
{ // let's create it
   $NENDO_category = new StdClass;
   $NENDO_category->name =
         $fields_obj->{
            CFG->enrol_db_remoteNENDOfield};
            $NENDO_category->description =
         $fields_obj->{
            CFG->enrol_db_remoteNENDOfield};
            $NENDO_category->parent =
                        $CFG->enrol_db_category;
   // overlay template
   foreach (array_keys($template) AS $key) {
      if (empty($NENDO_category->$key)) {
                  $NENDO_category->$key =
                                       emplate[$key];
      }
   }
   if ( !($NENDO_category->id =
        insert_record('course_categories',
        $NENDO_category)) )
   {
        error_log("Could not insert the
                    new category
                    $NENDO_category->name");
        continue; // next foreach course
   }
   $NENDO_category->context =
                    get_context_instance(
                        ONTEXT_COURSECAT,
                           $NENDO_category->id);
   mark_context_dirty(
            NENDO_category->context->path);
   // Required to build course_categories
   //.depth and .path.
   fix_course_sortorder();
   add_to_log($NENDO_category->id,
        "NENDOcategory", "new",
        "view.php?id=1",
        "enrol/database:
                    $NENDO_category->name");
}
```

コースの登録は以下のコードで行う．

```
// ok, now then let's create it!
// prepare any course properties we actually have
$course = new StdClass;
$course->{$CFG->enrol_localcoursefield} =
                                        coursefield;
$course->password = random_string(8);
$course->fullname =
                $fields_obj->{$CFG->
            nrol_db_remoteKAMOKUMEIfield};
$course->summary = ':' . $fields_obj->{$CFG->
                    enrol_db_remoteNOTEfield};

if ( !empty($fields_obj->{
        $CFG->enrol_db_remoteGAKKAfield}) )
{
    $course->category =$GAKKA_category->id;
} else {
    $course->category=
                        $GAKUBU_category->id;
}
$course->shortname = $coursefield;

if (!($newcourseid =
                $this->create_course($course, true)
   and $course =
                get_record( 'course', 'id',
                                $newcourseid)))
{
   error_log(
            "Creating course $coursefield failed");
   continue; // nothing left to do...
}
```

なお，Moodle のソースプログラムについては文献[3]に参照されたい．

5. 今後の課題

本稿では，Moodle の利便性を向上させるために山口大学の学務情報システム（修学支援システム）とのデータ共有による Moodle コースの自動作成について述べた．

今後の課題としては，学習データ（成績）の共有やシステム間の SSO を実現しさらなるユーザエクスペリエンスの向上などがあげられる．

参 考 文 献

[1] 戸田英貴，江木啓訓，須田良幸，品川徳秀：Moodle と学務システムのデータ連携の設計と課題，情報処理学会研究報告，コンピュータと教育研究会報告 2008(64)，49-54，2008-07
[2] http://docs.moodle.org/19/en/External_database (accessed by 2013/7/24)
[3] Jonathan Moore, Michel Churchward: Moodle 1.9 Extension Development, Packt Publishing, Chapter 9 (2010-04)