

多人数講義における演習課題の TA チェック支援システムの開発と評価

古賀研究室（九州工業大学）

Support System for TA Using Mobile Devices in Large Classes

*Koga Laboratory (Kyushu Institute of Technology)

Abstract— This paper describes a web application which supports TA (Teaching Assistant) check and score management of students in large classes. TA can input data and browse them very efficiently by using mobile devices which connect to the server through wireless LAN. The result of questionnaires for evaluating the system is reported.

Key Words: Education, Teaching Assistant, Large Class, Web Application

1 はじめに

大学における講義にはクラスや学科単位で行うものがあり、学生数が 100 人近くになる場合がある。このような多人数講義では、受講生の少ない講義に比べて、学生一人一人に対して細やかな指導を行うことは難しい。特に演習を含む講義となると、教員一人では学生全員を相手にするのはさらに困難になる¹⁾。

演習を含む講義において学生一人一人に細やかに対応する方法として e-Learning システムの利用²⁾や TA (ティーチングアシスタント) の採用が挙げられる。TA とは担当教員の指導のもと、授業の補助や運用支援を行う学生のことであり、一般的は大学院生が担当することが多い。主な業務は、個別の質問に応じることや演習課題のチェックである。

e-Learning システムは、従来の通信教育で行われているような、一教室に受講生が集合せずに行われる講義を支援するためにも使用されている。e-Learning のメリットとしては、時間や場所の制約がなくなること、自分のペースで学習ができること、学習の履歴を取ることができること、資料配送の手間がかからないことなどが挙げられる。しかし一教室に集合せずに演習を行うため、学生は教員や他の学生との交流がとりにくくなり、質疑がしにくく、問題解決が難しくなる。また、演習時間が講義時間に含まれるような場合には不向きであると考えられる。

一方 TA の採用では、複人数態勢で学生の質問に個別に対応することができるため、教員一人の場合よりも学生の理解度向上の効率を上げることができる。しかし、授業内容によっては受講している学生数に対して TA の数が足りないこともあり、学生は質問対応に不満が残ってしまう³⁾。そのため TA の業務を効率的に行い、多くの学生に対応できるようにする必要がある。

演習時間における TA の主な業務には以下の 3 点がある。

- 学生の出欠や遅刻のチェック
- 学生の演習課題のチェック
- 学生の個別の質問への対応

参考文献³⁾や⁴⁾では、出席管理についてのシステムや手法の提案が行われているが、演習課題のチェックの手法については触れられていない。

本論文では、演習チェックを含めた TA チェック業務を支援するためのシステムの開発について述べ、実際にシステムを利用した TA へのアンケートをもとにその評価を行う。

2 TA 支援システム

2.1 紙による TA チェックの問題点

演習課題付きの講義の成績の評価項目には、出席状況、演習課題の提出状況、期末試験の結果などがある。このうち TA は出席状況や演習課題の提出状況のチェック業務を行うことが多い。

従来、TA がチェック作業を行うときには、紙媒体が用いられている。紙を用いる場合、以下のような問題点やデメリットが挙げられる。

- チェック結果が分散され、集計の手間がかかる
- 過去のチェック結果を探すのに手間がかかる
- チェック結果のデータ量が膨大になる可能性がある

TA 業務は複数人で行うことが多い。紙を用いて作業を行うと、各 TA がばらばらに自分のチェックした結果を持つことになるので、演習時間中に他の TA のチェック結果を把握できず、TA はどの学生がどこまで提出できているのか把握するのが難しくなる。また、各 TA がチェック結果をそれぞれ持つことで、各講義の後の集計作業が必要となる。

演習時間には、以前の講義に課された演習課題をチェックすることもあるので、TA は毎回、全講義のチェック表を持参しなければならない。以前の課題をチェックする場合は、該当するチェック表を探す手間がかかる。

また例えば、講義が全 14 回、各講義の演習が 3 題、学生数が 80 人であるとする、その講義の成績データの量は 3360 個になる。出欠データを含めたり、演習課題数や学生数が増えたりすればデータ量は更に増加する。教員が学生の最終成績の評価のために最終データを入力するときに負担が大きくなり、成績の入力ミスが起こる可能性が高くなる。



Fig. 1: 紙によるチェックの問題点

2.2 システムに必要な要素

2.1 節で挙げた問題点を解決するための TA チェック支援システムに必要な要素はモバイル端末、データベース、ウェブアプリケーションであると考ええる。

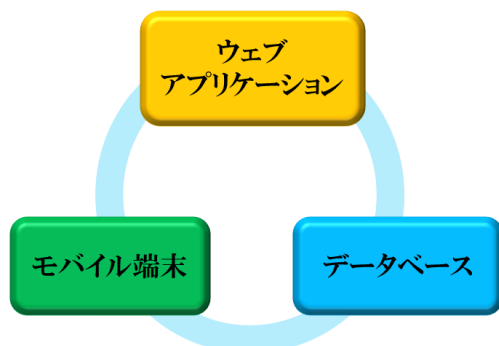


Fig. 2: TA チェック支援に必要な要素

2.2.1 モバイル端末

TA は多くの学生の質問に応じたり、チェックを行ったりするため講義室内を歩き回る。システムを利用するためにノート PC のような大きな端末を用いると、通路が狭い場合に不便に感じる。そのため、TA が利用する機器を無線 LAN を搭載したモバイル端末とすることで移動が容易になる。利用するモバイル端末は無線 LAN を搭載していること、モバイル性に優れていることが重要であり、タッチパネルによる操作のものとすることで直感的に操作できる。

2.2.2 データベース

データベースを利用すれば、TA のチェック結果を一元管理することが可能となる。複数の TA でチェック作業を行う場合、データベースを利用することで、他の TA のチェック結果を共有でき、学生の課題提出状況の把握が容易になり、TA の業務の効率性を向上できる。また、チェックする度にデータベースに保存されるので、講義後の集計作業を行う必要もなくなる。

2.2.3 ウェブアプリケーション

システムがウェブアプリケーションであることも重要な要素であると考ええる。iPhone アプリやアンドロイドアプリのような専用アプリケーションやデスクトップアプリケーションではなく、ウェブアプリケーションとすることで、ブラウザがある端末ならば、どの端

末でも同じ環境を再現することができる。TA が個人的にスマートフォンや Apple 社の iPod touch⁵⁾、タブレット PC などを持っていれば、システムの URL にアクセスするだけで利用することができる。また、端末にインストールする必要がないので、ユーザーはシステムのバージョンの違いによる不具合について心配する必要がない。2.2.3 節で説明する Ajax 技術を利用することでより直感的なアプリケーションにすることも可能である。

システムをウェブアプリケーションにすることでセキュリティの面も重要となる。システムを利用するためにログイン機能を持たせることや、ログイン後のアカウントによる権限を管理すること、ログインパスワードの暗号化を行うことなどが必要である。

Ajax

Ajax(Asynchronous JavaScript + XML) は、ウェブブラウザ内で非同期通信とインターフェースの構築などを行う技術の総称のことである。従来のウェブアプリケーションでは、サーバーへ送信したリクエストのレスポンスの結果として画面遷移が発生していた。しかし Ajax を利用することで、ウェブページの更新を伴わずにサーバーと XML 形式のデータのやり取りを行い、処理を進めていく対話型のウェブアプリケーションにすることが可能となる。

Fig. 3 に Ajax によるクライアントとサーバーのやり取りを示す。

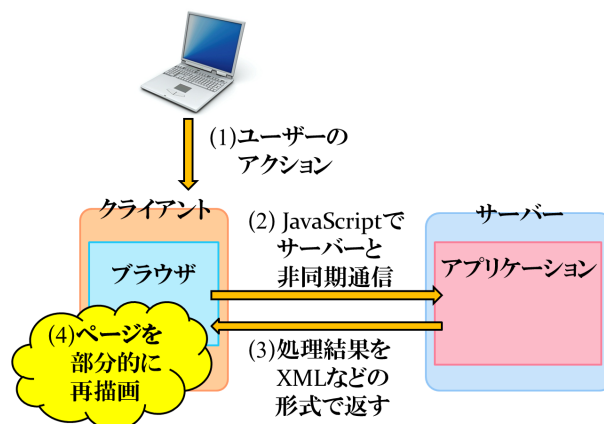


Fig. 3: Ajax によるクライアントとサーバーのやり取り

有名な利用例としては Google マップがある。従来であれば、ウェブ上の地図は表示を変更するためにページを切り替えなければならなかった。しかし Ajax を利用すれば、ページの切り替えをすることなく、マウス操作によって表示範囲の移動や拡大/縮小などの直感的な操作が Flash のような技術を利用せずとも標準的なブラウザ機能だけで実現できる。

MVP パターン

MVP パターンはウェブアプリケーションを構築する上で注目を集めているデザインパターンである。Fig. 4 に示すように Model, View, Presenter で構成される。

- Model
ビジネスオブジェクトを司り、View や Presenter に依存しない。

- View
画面表示やユーザーの入力の受付を担当する。
- Presenter
View を通じてユーザーから入力イベントを受け取り、その後 Model や View の操作を行う。

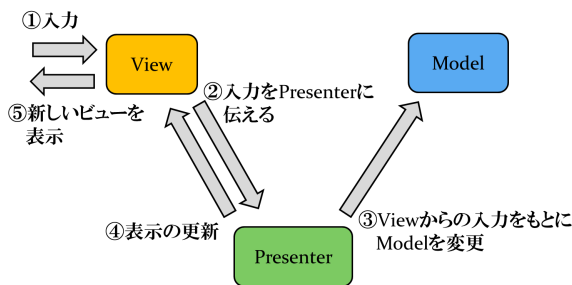


Fig. 4: MVP パターン

MVP パターンを適用することで端末ごとにビューの切り替えが容易になる。また、プログラムの再利用性が高まり、講義ごとの仕様変更にも柔軟な対応が可能になる。

3 TASKit

開発した TA チェック支援システムは TASKit という名称である。タッチパネルにより直感的に操作できる点、無線 LAN を搭載している点、モバイル性に優れている点、廉価な点から、TA が利用する端末は iPod touch やサムスン電子株式会社の Galaxy S⁶⁾ のような大きさであることを前提した。2.2.3 節で述べたように、非専用アプリケーションにすることで、スマートフォンのような他のモバイル端末や PC 上から操作できるようにした。この章では TA チェック支援のために開発した TASKit の詳細について述べる。

3.1 TASKit のシステム構成

Fig. 5 に TASKit のシステム構成を示す。TA は主に iPod touch のようなモバイル端末のウェブブラウザを通して、出欠チェックや演習チェックの操作を行う。教員は主に PC のウェブブラウザから出欠や演習チェックに加えて、TA や講義、演習課題、学生の登録などを行える。チェックしたデータや登録した講義などの情報は、HTTP に基づいてウェブサーバーへと通信される。送られた情報はサーバー側で SQL を実行することによりデータベースに格納される。また Ajax によって非同期通信を行うことで、他の端末上で変更された出欠や演習のチェックデータを一定時間毎に同期するようにしている。

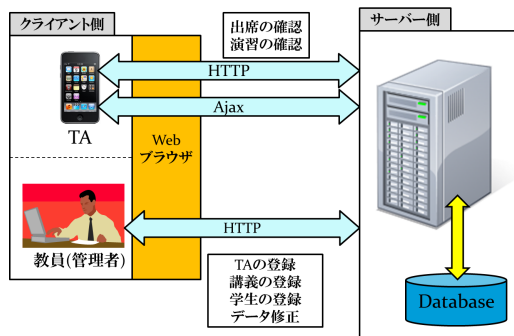


Fig. 5: TASKit のシステム構成

3.2 TASKit の処理フロー

Fig. 6 で本システムにおけるクライアントとサーバーのやり取りについて説明する。Fig. 6 では学生の成績のチェックを例にしている。

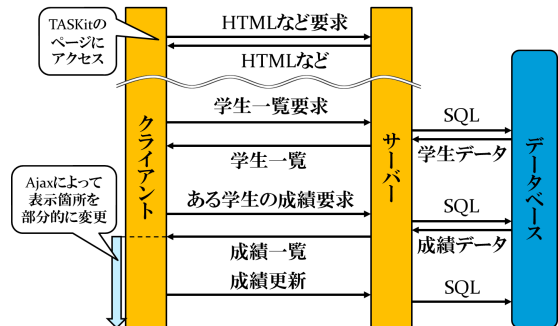


Fig. 6: 演習チェックの処理フロー

1. TASKit のページにアクセスする
TASKit は一つの HTML ファイルで構成されており、JavaScript によって表示画面を変更させている。
2. ユーザーがログインを行う
クライアントサイドでは HTML ファイルを取得するとログイン画面が表示されるので、ログインを行う。
3. クライアントサイドが学生一覧を要求する
ログイン後は学生一覧を表示するステップに移るため、クライアント側がサーバーに対して学生一覧を要求することになる。画面の表示用プログラムはクライアント側が保持しているので、サーバーはデータベースから学生一覧のデータのみを送信する。
4. ユーザーは表示された学生一覧からチェックしたい学生を選択する
先程と同様に、クライアント側はサーバーに対してチェックしたい学生の成績データを要求する。サーバーはデータベースからその学生の成績データのみを送信し、ブラウザ上にはその学生の全成績が表示される。

チェック画面では他のユーザーによって変更された箇所がある場合、Ajax によって一定時間毎に該当する箇所のラジオボタンやドロップダウンリストなどの値が部分的に変更される。

3.3 TASKit の開発

3.3.1 開発ツール

本システムは GWT(Google Web Toolkit)⁷⁾ を利用して開発した。GWT は Google 社が提供しているウェブアプリケーション開発用ツールキットである。ウェブアプリケーションの開発は、面倒でエラーの発生しやすいプロセスであり、開発者はブラウザ固有の動作への対処に多く時間を費やすことがある。また、大規模な JavaScript のコードや Ajax を利用したコンポーネントの作成は困難である。

GWT では Java 言語で開発を行うので、統合開発環境 Eclipse や単体テスト用フレームワーク JUnit といった使い慣れた Java 開発ツールを利用でき、生産性を向上できる。

GWT を用いれば Ajax の実現や MVP パターンの適用を容易に行うことができ、本システムでも Ajax の利用、MVP パターンの適用を行っている。

Fig. 7 に Google Web Toolkit プロジェクトについての図を示す。

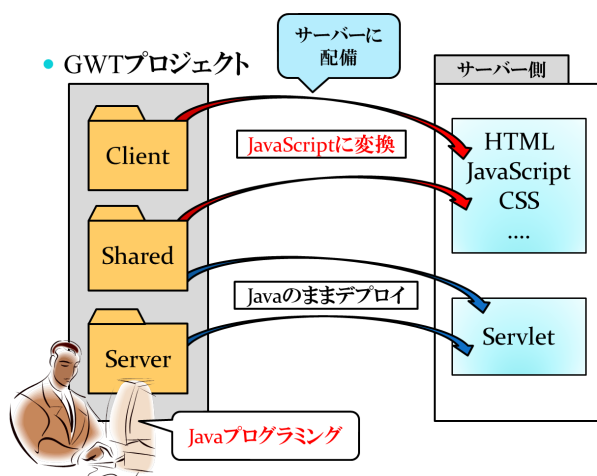


Fig. 7: Google Web Toolkit

GWT プロジェクトではクライアント側のコード、サーバー側のコード、またクライアントとサーバーが共通に利用するインターフェースなどのコードはすべて Java 言語で記述する。この GWT プロジェクトを実際にサーバーにデプロイすると、サーバー側のコードは Java でそのまま機能するが、クライアント側のコードは JavaScript に変換される。

3.3.2 利用するデータベース

本システムは Java とデータベースを結びつけるための API として JDBC を利用し、リレーショナルデータベースを利用している。有名なリレーショナルデータベース管理システムには MySQL⁸⁾ や PostgreSQL⁹⁾, SQL Server¹⁰⁾, Oracle¹¹⁾ などがある。本システムではオープンソースであり、多くのオープンソースウェブアプリケーションで使われている実績がある MySQL を選択した。

3.3.3 O/R マッピング

Java 言語のようなオブジェクト指向とリレーショナルデータベースはデータモデルの設計思想が違う。オブジェクト指向設計では、現実世界のモデルに即したデータモデルを定義する。一方、リレーショナルデータベース設計では、リレーショナルデータベースの検索や登録などを最適に行うことを目的とするため、主に正規化や非正規化といった方法でデータモデルを定義する。この設計思想の違いからオブジェクト指向とリレーショナルデータベースのマッピング作業は煩わしい問題とされている。この問題を解決する手段に O/R マッピングがあり、本システムでは Java 言語で O/R マッピングを行うためのツールである Hibernate¹²⁾ を利用して、Java と MySQL の溝を埋める。

3.4 主な機能

3.4.1 ユーザー管理

特定のユーザーが利用できるように、TASKit はログイン機能を持つ。Fig. 8 にログイン画面を示す。教員と TA のユーザー ID とパスワードは事前に登録し

ておくことで、TASKit 利用前にアカウントのチェックを行う。登録されたパスワードは平文ではなく暗号化されており、セキュリティを高めている。ログインに成功すれば学生一覧の画面が表示される。アカウントタイプ (TA か教員) によって利用できる機能が変わり、TA アカウントでは出欠のチェック、演習のチェック機能が利用できる。教員アカウントでは TA アカウントの権限に加えて、全学生の演習結果一覧表示や、講義、演習課題、TA の登録機能が利用可能となる。



Fig. 8: ログイン画面

3.4.2 TA による出欠チェック機能

Fig. 9 に TA による出欠チェック用の画面を示す。上部のドロップダウンリストからチェックする講義回を選択すると、該当の講義の学生の出席状況が表示される。出欠のタイプは、「出席」、「欠席」、インフルエンザのような必ず休まなければならない病気の時に利用する「病欠」、「公欠」に対応しており、ラジオボタンでチェックを行う。



Fig. 9: 出欠画面

3.4.3 TA による演習チェック機能

Fig. 10 と Fig. 11 にそれぞれ学生リストの表示画面と TA による演習チェック用の画面を示す。TA が演習のチェックを行うには、まず Fig. 10 の学生リストから

チェックする学生を選択する。選択すると、Fig. 11 の演習チェック画面に変わる。演習チェック画面には選択した学生の全成績が表示され、該当する演習課題をチェックすることになる。現在は、「○」、「△」、「×」によって成績を評価できるようになっている。



Fig. 10: 学生リスト表示画面



Fig. 11: 演習チェック画面

3.4.4 教員による講義の管理機能

教員としてログインすることで全学生の演習結果一覧表示や、講義、演習課題、TA の登録を行うことができる。教員用の機能の例として Fig. 12 に演習結果一覧表示画面を示す。

Fig. 12: 演習結果一覧表示画面

4 評価

九州工業大学情報工学部システム創成情報工学科のプログラミング講義で本システムを実際に TA に利用してもらった。受講生は 85 名、TA 数は 5 名、演習時間は 90 分である。システムのネットワーク図は Fig. 13 のようになっている。

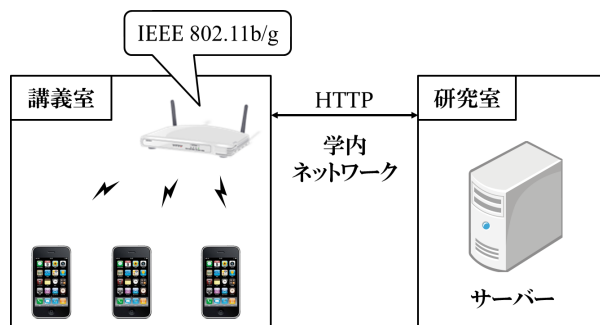


Fig. 13: ネットワーク環境

Fig. 14 と Fig. 15 に実際に iPod touch を用いて、TA が演習のチェックを行っている様子を示す。

TASKit を利用した TA にアンケートにより、本システムの評価を行う。本論文執筆時のアンケート対象者は 5 名である。



Fig. 14: TASKit の利用風景 1



Fig. 15: TASKit の利用風景 2

まず、Table.1 に 5 段階評価のアンケートの結果を示す。点数は TA 5 名の平均であり、数字が大きいほど評

価が高いことを表す。

Table 1: アンケート結果

アンケート項目	評価
表示画面がわかりやすい	4.4
集計作業がないので楽である	4.0
操作しやすい	3.4
紙よりも TASKit の方がよい	2.8

表示画面のわかりやすさや集計作業の必要がないことに対しては高評価だが、操作性に関しては評価は低く、紙ベースのチェックに必要性を感じる結果となった。これは現在の TASKit ではボタンが小さいことや、画面の解像度に合わせて表示できていないことなど、ユーザーインターフェースの設計が十分でないからであると考えられる。

この5段階評価のアンケート以外にもいくつかアンケートを行った。

「TASKit を利用することでチェックミスがなくなると思うか」という問いの回答は Fig. 16 のようになった。「変わらないと思う」、「減らないと思う」と答えた理由はボタンが小さいことであった。講義後の集計プロセスがないことにより集計中のチェックミスがなくなるので、一長一短であるという回答もあった。

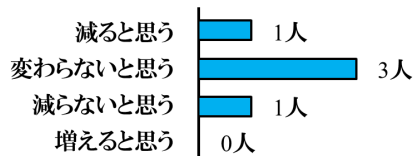


Fig. 16: チェックミスに関する質問の回答

また今回は iPod touch を利用してもらったが、5人全員が利用端末の画面サイズに満足しないという結果になった。ボタンが小さくて押しにくいことや表示される情報量が少ないことが理由である。このことから現在の表示画面を iPod touch のサイズでも満足できるようなユーザーインターフェースに変更するか、利用端末を大きなサイズに変更して試してみる必要があると考えられる。

これらの質問のほか、利用した感想についても伺い、以下のような結果が得られた。

- 過去のチェックデータがすぐに確認できる
- TASKit さえあれば全部の週のチェック表を準備しなくて済む
- チェックデータの紛失の恐れがない
- TA として講義に持ち込むものが増えてめんどくさく感じた

以上より、本システムは現状では十分に満足できるものではないが、機能拡張やユーザーインターフェースの変更によって TA チェック支援システムとして十分機能できると期待されることがわかる。

5 まとめ

本論文では、TA チェックを支援するシステムをウェブアプリケーションとして開発した。複数の TA がチェック結果を共有できるようにデータベースを利用した。TA がチェック業務を効率的に行えるように、モバイル

端末を利用することを前提とした。Ajax を活用することで、端末の画面に表示されているチェックデータが最新データでない際に、一定時間ごとに自動的に表示データの同期をとるようにした。

また、開発だけでなく、大学の講義で実際に TA に利用してもらい、アンケートによって本システムの評価を行った。表示画面のわかりやすさや、集計作業には関して高い評価を得た。

今後は他の講義でも活用してもらい、アンケートで得られた要望に応えられるように本システムの改良を行い、さらに利用しやすいシステムにしていく必要があると考える。

参考文献

- 1) 西澤泰彦, 「多人数講義における問題点と教育方法」, 名古屋高等教育研究第 6 号, 45/57 (2006)
- 2) 川崎治夫, 「e-Learning システムを利用した多人数プログラミング教育」, 国士舘大学情報科学センター紀要 (28), 57/62 (2007-03)
- 3) 井上真由美, 大即洋子, 中川正樹, 「PDA を利用したティーチングアシスタントによる演習授業支援システム」, 情報処理学会全国大会講演論文集, 第 67 号, 4 頁 427/428 (2005-03-02)
- 4) 呉靱, 葛崎偉, 「多人数情報処理教育講義における出席について」, 教育実践総合センター研究紀要 17, 1/9 (2004-03-25)
- 5) <http://www.apple.com/jp/ipodtouch/>
- 6) <http://jp.samsungmobile.com/pc/galaxys/index.html>
- 7) <http://code.google.com/intl/ja/webtoolkit/>
- 8) <http://www-jp.mysql.com/>
- 9) <http://www.postgresql.org/>
- 10) <http://www.microsoft.com/japan/sqlserver/2008/r2/default.msp>
- 11) <http://www.oracle.com/jp/products/database/index.html>
- 12) <http://www.hibernate.org/>