3P-6

授業支援システムに対するプラグインによる授業単位での機能拡張の提案

熱田 智士† 松浦 佐江子‡

芝浦工業大学工学研究科電気電子情報工学専攻† 芝浦工業大学システム工学部電子情報システム学科‡

1. はじめに

e-Learning の普及に伴い近年、数多くの LMS (Learning Management System) が実用化されている。LMS は e-Learning におけるインフラ的機能を果たすソフトウェアであり、コース管理、受講者の管理、教材コンテンツの配布、連絡事項の通知、テストやアンケートの実施、コミュニティ機能などを提供することで e-Learning における教員、学生双方の利便性の向上を目的としている。また、通常 1 つの LMS で複数の授業を管理するため、LMS は特定の授業に傾倒・依存しない汎用的な機能のみで構成されるのが一般的である。しかし、これは裏を返せば LMS が支援する対象はあくまで全ての授業に共通する性質だけであり、授業固有の性質に関して LMS は基本的に何らサポートを行わないことを意味する。よって、仮に授業固有の性質がその授業を実施する上での重要な要素であったとするならば、LMS 導入よる恩恵を受けることは難しくなる。

本稿では授業固有の性質を支援するために、本学における Java によるプログラミング演習科目の支援を目的に開発したシステムを例にとり LMS に対する授業単位での機能拡張の必要性とそのモデルを提案する。

2. 授業における固有の性質

では、実際に授業固有の性質とはどのようなものが存在するのか、ここでは本学で実施されている対面型の講義など実例を交えて述べる。

まず本学で実施されている Java プログラミング演習科目を例に挙げる。この科目の特徴的な性質はレポートの提出方式及び採点方式にある。LMS によるレポートの提出は単一の文章ファイルによって行われるのが一般的である。然しながらプログラミング演習科目の場合、レポートとはプログラムのソースコードそのものであり、採点を行う際にはプログラムの正当性、妥当性を検証する必要がある。その場合ソースコードを単一の文章ファイルに纏めて提出を行うよりも、ソースコードそのものを提出形式とした方が教員側の負担は軽減される。更に採点を効率的に進めるためにはシステム上でプログラムの正当性をチェックが行えるよう、少なくともコンパイル結果の表示が行え、可能ならばテストプログラム実行でき、その結果を閲覧できることが望ましい。また、この演習科目はレポートの採点方式として、学生のプログラムに対する理解力を測るために、実際に提出したプログラムをローカルの環境でコンパイル・実行した後にアルゴリズムや処理の流れなどに関する説明を行って貰い、その内容によって採点を行う方式（以下審査方式）を設けている。これらの特徴は一般的な授業にない特別な性質である。

続いて例として挙げるのは同じく本学で実施されている「ソフトウェア設計論」という講義科目である。この科目もレポートの採点方式に特徴がある。この講義ではレポートは 2 部提出し、内 1 部は教員が採点し、もう 1 部は提出した本人以外の学生にランダムで配布し、学生自身に評価を行って貰う。そしてその評価の仕方を、評価を行った学生の成績の評価に加えるといった採点方式を採用している。この講義の採点方式も他の科目にはない固有の性質である。

Java 演習科目における授業固有の性質とは Java のソースコードを扱うための要件であり、それは他の科目に含まれない要素であったため標準的な授業に対する明確な違いとして捉えることが出来た。またソフトウェア設計論の採点方式に関してもその特殊性から他の科目との違いが明確であった。しかし授業固有の性質はこのような明確な違いとして現れる項目だけではなく、一見共通の性質として捉えられる項目であっても授業ごとに微妙な差異として含まれる場合が少なくない。例えば、LMS の機能として標準的な成績の管理機能を考えてみる。成績管理機能によって、これまでに実施されたレポートやテストの結果、出席率などを元に最終的な成績をつける際、各項目の成績における比重や、集計方法は授業ごと独自に定められており明確な基準を持たないケースが多い。このような成績管理の方式の違いなども授業固有の性質として捉える必要がある。

これらの固有の性質を持つ対面型授業の一部、もしくは全てを e-Learning 化する際にはこれらの性質を補助しなければならない。

3. LMS による授業固有の性質の支援

当時、我々は対面形式の授業である Java 演習科目の補助を目的とした独自の LMS を開発した。ここではその開発したシステムの概要について述べる。システムはクライアントのインターフェースに Web ブラウザを用いる Web アプリケーションであり、一般的な LMS の基本的な機能に加え、Java 演習科目を支援するために以下の機能を実装している。まず Java のソースコードはディレクトリの階層構造に依存していることから、ソースコードによるレポート提出を可能にするために提出時にディレクトリ構造を構築できるレポート提出機能を実装した。次に教員側でソースコードのコンパイル結果を閲覧できるように、教員側にはユーザーインターフェース（以下 UI）に専用の表示領域とソースコードのコンパイル機能を、学生側の UI にはソースコードをコンパイルする際のコンパイルオプションを指定するための機能を付加した。そして審査方式による採点は、提出したレポートのダウンロード機能を用いることでこれをサポートした。テストプログラムの実行に関しては、現行の Web アプリケーションの形態でこの機能を実装しようとすれば、サーバー上でシステムの利用者が記述したプログラムを実行しなければならず、悪意のあるコードがプログラムに含まれていた場合その実行を阻止するための何らかのセキュリティ上の対策が必要になるため今回は実装を見合わせた。なお、このシステムの詳細については[1]を参照して頂きたい。

4. 授業固有の性質の問題点と授業単位での機能拡張の必要性

システムの実装に際し、可能な限り LMS の持つ汎用的な機能によって授業固有の性質を支援するように心掛けた。結果、演習科目を支援するために実装した機能の内、複数のファイルによるレポート提出機能と提出したレポートのダウンロード機能は比較的汎用性のある機能であるため、そのまま他の科目にも流用可能であった。しかし、Java のソースコードを扱うための機能に関しては、この科目以外に活用することは恐らく不可能である。LMS に LMS の汎用的な機能のみでの支援が困難な授業ごとの性質が含まれていた場合、このような一部の利用者以外には意味を成さない機能をシステムのインターフェースに実装しなければならなくなる。このような機能はインターフェースの実装を**汚す**ため、操作性を損なわせる要因となる。

また、先ほど述べた成績管理など授業ごとの微妙な差異を LMS で吸収しようとうするならば、LMS で授業の多様性に適応するように考え得る様々な選択肢（成績管理ならば集計方法のバリエーションなど）を提供しなければならず実装上、現実的でなくなる。また過度の機能追加はインターフェースを複雑にするという点でも好ましくない。

授業形態の多様性を網羅するために、システム側での機能提供に限界があるとすると、利用者は授業のシステムに上手く適応しない領域でのシステムの利用を諦めるか、もしくは授業の形態をシステムにあわせるかの何れかを選択しなければならないが、授業形態をシステムにあわせることは、授業の個性や授業を実施する上で蓄積されたノウハウが生かされなく可能性を孕んでいるため、安易な摺り寄せは行うべきではないと考える。

LMS が提供する機能の限界と授業形態をシステム側に摺り寄せることで生じる欠点、これらの問題を解決するには機能拡張を利用者側に委ねることが必要であり、LMS にはシステム側でシステムに利用者側が後付けて機能を拡張できるようなプラグインのような枠組みを提供することが求められる。

5. 機能拡張のモデル

現在我々が開発中の LMS に対する授業固有の性質を支援するための 2 つの機能拡張モデルについて述べる。1 つ目のモデルは Web ブラウザベースのシステムに対する機能拡張である（図 1）。システム全体は 3 層アーキテクチャモデルによって構成され、機能拡張の対象となるのは 3 層の内上位 2 層、つまり大きく分けてプレゼンテーション層への拡張と、ビジネス層への拡張に分類される。そして各層への機能拡張は各層のコンポーネントへアクセスするための専用のフレームワークを通じてのみ行われる。特にデータモデルフレークワークはビジネス層のコンポーネントを包み込み、データモデルの整合性を保つという重要な役割を担う。

Proposal of enhancing function for unit of class by plug-in to class support system

† Satoshi Atsuta ‡ Saeko Matsuura

† Department of Electrical Engineering and Computer Science, Graduate School of Engineering, Shibaura Institute of Technology

‡ Department of Electronic Information Systems Shibaura Institute of Technology

システムへのプラグインとして機能するモジュールは、ベースとなるシステムが ASP.NET により実装されていることから.NET Framework のモジュールとして実装される。プラグインは機能的に大きく分けて 2 つに分類される。1 つは UI を拡張するためのプラグインで ASP.NET カスタムコントロールとして実装される。もう 1 つはインターフェースを持たないプラグインである。それぞれのプラグインの適応箇所は、例えば先ほどの述べた Java のコンパイルオプションの指定、コンパイル結果の表示機能のように UI を必要とする性質を支援する機能拡張には前者の利用し、成績の集計ルールや自己学習教材における問題遷移ロジックなど、UI を必要としない性質への支援には後者を用いる。

プラグインの登録は、各科目の教員用の UI を通じて行われ、クライアントからのリクエストがあった際に.NET Framework のリフレクション機能によって動的にシステムに読み込まれる。また、システムに読み込まれるモジュールや型の名前の衝突をさけるためにシステム側でモジュールの名称や名前空間を指定し、それに準拠するもののみをシステムに登録する。

このモデルの欠点は ASP.NET によるアーキテクチャ上の制約が厳しいことである。UI は本来プログラム中でインターフェースを構築するカスタムコントロールより、タグによってインターフェースを記述できるユーザーコントロールを用いた方が容易に実装することができるが、確認した限りでは ASP.NET のアセンブリ配置に関する制約からユーザーコントロールを任意の場所から動的に読み込むことができないため、UI の拡張にユーザーコントロールを用いることができない。

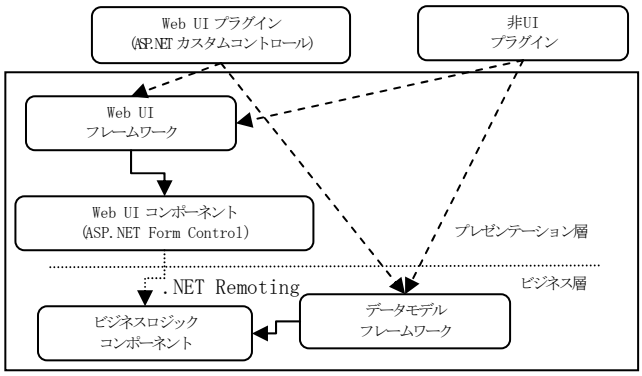

図 1.Web ブラウザベースのシステムへの機能拡張モデル

2 つ目のモデルではクライアントのインターフェースをブラウザベースのものからスマートクライアントベースのものへ移行し、プラグインによる機能拡張はこのスマートクライアントに対して行う（図 2）。スマートクライアントとは Microsoft が提唱する管理、運用の問題を解決する Web ベースのシンクライアントと、プラットフォーム性能を活かしユーザーの生産性を最大化するファットクライアントの利点を両立させるクライアントソリューションである[2]。スマートクライアントは Web サービスのクライアントアプリケーションとして動作し、特定の Web サーバーから自動的に展開・更新される。またクライアントのローカル環境で動作し、オフライン環境でも動作もサポートされる。

スマートクライアントを用いたモデルは先に述べたソフトウェア設計論の学生によるレポートの評価を支援する場合などに効果的である。学生によるレポート採点を支援するには様々な方法が考えられるが何れにしても教員、学生双方の UI に対する大幅な変更が必要となることが予想される。UI 対する大幅な変更をプラグインによって実現するのは難しいと考えられ、そのようなケースにおいてスマートクライアントを用いたモデルは有効である。スマートクライアントはビジネス層以下と Web サービスを通じて非常に疎に結合されているため、UI そのものの換装を容易に実現できる。これによって授業に最適な UI を提供することができる。

また、UI の換装を必要としない程度の小規模の UI に対する機能拡張や、UI を必要としない機能拡張であれば、Web ブラウザベースのシステムに対する機能拡張のときと同様に、システムが提供する Web ブラウザベースのシステムが提供する機能と同等の機能を有するスマートクライアントに対してプラグインを導入する。

更に、用途に合わせてスマートクライアントを使い分けることで利便性が向上する。アンケートや成績の管理など集計作業が必要な操作を行いたい場合は MS Excel をクライアントに用いれば Excel の持つ豊富な表計算機能をそのまま利用できる。テストやアンケートを実施したい場合は、クライアントに MS Word を用いることで数式やオートシェイプを利用した設問の記述が可能になる。

スマートクライアントの採用によってプレゼンテーション層の実装が全てクライアント側に移行されることで以下のような利点が生まれる。

- ネットワークトラフィックが大幅に軽減される。
- 従来、サーバーサイドで行っていた Java ソースコードのコンパイルなどサーバーマシンへの負荷が大きい操作もリッチクライアントの導入によってローカルの環境に移行できる。
- 従来 Web ブラウザベースの UI ではセキュリティ上の理由から、Java のテストプログラムを実行するなどの機能を実装することができなかったが、スマートクライアントの場合テストプログラムの実行をクライアントサイドに移行できるため、このような機能も実装することが可能になる。

このようにスマートクライアントを用いたモデルは、授業形態の多様性を実現するため、授業ごとにクライアントに個別の振る舞いを持たせなければならない場合に非常に効果的である。

このモデルの欠点はクライアントサイドに求められるシステム要件が厳しくなる点である。サーバーとの通信は Web サービスによって行われることから、サーバーに求められるシステム要件は Web ブラウザベースのシステムと何ら変わりは無い。しかし、クライアントに求められる要件は Web ブラウザベースのシステムでは Web ブラウザだけであったが、スマートクライアントの場合はスマートクライアントの実装に合わせた実行環境が求められる。

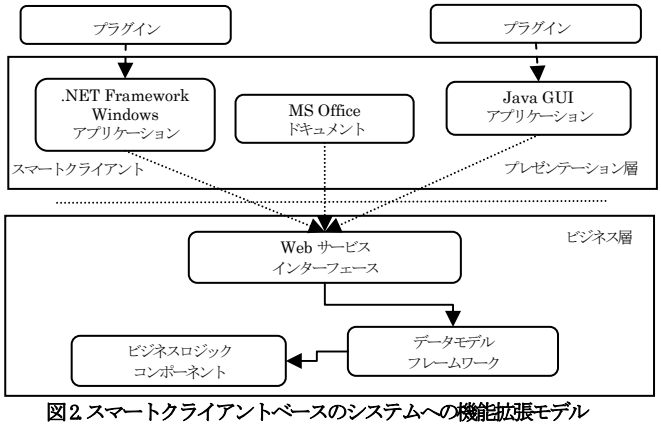

図 2. スマートクライアントベースのシステムへの機能拡張モデル

以上を纏めると、UI を必要としない機能拡張であれば、2 つのモデルの何れかのプラグインによる機能拡張を行う。成績の管理方法など授業ごとの僅かな差異をシステムに盛り込むにはこの方法が適している。小規模の UI に対する拡張が必要な場合も両モデルのプラグインによる機能拡張に実現可能であるが、クライアントのシステム要件の点では Web ベースシステムが有利であり、実装の容易性の面ではスマートクライアントが有利である。プラグインによる機能拡張でカバーできないような大幅な UI に対する拡張が求められる場合は、スマートクライアントの換装を行う。他の授業には存在しない授業固有の性質として明らかな項目を支援する場合にこれらの UI に対する拡張を用いる。その際、UI に対する拡張の規模に合わせて、プラグインか、スマートクライアントの換装かの何れかを選択する。

6. むすび

LMS に対する授業単位での機能拡張の必要性とそれを実現するためのモデルについて述べた。機能拡張が可能な LMS は WebCT[3]など極一部しか存在しないが、LMS にとって重要なのは汎用的な機能を詰め込むことではなく、後付けでの機能拡張の容易性の確保こそが重要であると考える。また、LMS の実装にスマートクライアントと Web サービスを用いることでプラグインによる機能拡張に比べ、より柔軟に授業への対応が可能なる。今後、このモデルにより構築したシステムを今回取り上げた演習科目以外の特徴的な性質を持つ授業に適用し、その効果を検証する予定である。

参考文献：

[1] 熱田 智士, 松浦佐江子, “Java プログラミング演習向け課題レポート提出・管理機能を付加した授業支援システム”, FIT2004 情報科学技術レターズ, 2004

[2] “スマートクライアント情報センター”, http://www.microsoft.com/japan/partner/isv/community/smartclient/

[3] “WebCT”, http://www.webct.com/