南山大学 数理情報学部 情報通信学科 2007 年度卒業論文要旨集

プログラミング教育における教育支援システム

～機能の追加・拡張の観点からの提案と実験～

2003MT033 片田賢世

指導教員 真野芳久

1. はじめに

情報化社会が高度になるにつれて、作業にパーソナルコンピュータ(以下、PC)を用いることはもはや必須と言える時代となった[1]昨今では、一般にシステムエンジニアやソフトウェア開発者と呼ばれる職種に就く人が増え、またそれに付随して、教育機関の中にもプログラミングを教える機会、現場が増えてきている[2]。

本論では、このようなプログラミングを教える教育現場において有用である教育支援システムの提案と開発を拡張性という点に的を絞って述べる。

2. 拡張性を持つということの教育現場での有用性

前提として、教育現場をここでは大学とし、基礎的なプログラミングを教える教育の場とする。そのため、教育者は大学教員とし、受講者は PC を所有した学生を指すものとする。現在のプログラミング教育では Linux か Windows 上で学習することが大半であり、ここでは Windows 上で教育を行っているものとする。加えて以後、ユーザという用語は教育者を指すものとする。

2.1 eラーニングとは？

PC 上での学習というと、現在 eラーニングを用いることが多くなっている。そこではサーバにある教育プログラムを受講者が各々の PC で走らせることで、時間的、空間的な制限なく学習ができるため、ブロードバンドの発展と共に急速に利用が高まった[3]。メリットは、前述の時間的・空間的制約の緩和に加え、従来の学習に比べて、音や動画、イラスト等、視覚や聴覚を伴った教材作りができる。加えて、各々の理解の進捗度に合わせて学習が進められるため、ストレスを感じにくいこと等が挙げられる。逆にデメリットは、医学等の実物に触れて学習しなければならないような分野では、動画や 3DCG などでは教材として限界があることや、強制力が弱いため、学習意欲の維持が困難であること等が挙げられる[4]。そのため、実際には従来どおりの教育と併せて eラーニングが用いられる(ブレンディング教育)ことが多い[5]。

2.2 プログラミング教育現場での eラーニング

現在プログラミング教育の場で主流の eラーニングのシステム形態は、章立てされた教科書のようなページがあり、最後にラジオボタンによる選択問題が設置されているタイプがほとんどである。

2.3 eラーニングとの差別化

PC を用いて学習する以上、広義では本論も eラーニングの一部になるが、本論ではもう少し改善の余地がないか探っていく。従来の eラーニングは、最初にプログラムが作成されると、後は問題を追加する程度の拡張性しか持たない。それは、多くの eラーニング教材が、その講義の完成形を eラーニング教材とするため、それ以上の拡張、変更にはあまり配慮しないからである。ここに改善の余地があるのではないかと考え、本論で提案するシステムでは、教育者が自分の教育スタイルに沿う形でシステムを改善できるようにする。

3. システムの概要

3.1 拡張性のある教材とは？

拡張性の実現のためには、既にあるシステムに対して、後からユーザがシステムの製作者の手を借りずに機能を追加、実装できなければならない。そのため、システム自身にあらかじめ外部からの要素を受け入れる機構がなければならない。その機構を持たず、製作された時点で完成としてしまっているのが、現在の eラーニング教材である。労力の軽減、追加の効率化として、思い浮かぶのがプログラミングにおけるモジュラープログラミングである。そこで、本論ではモジュラープログラミングのように、教材の各機能を部品としてまとめてしまうという方法を考えた。

3.2 教材の部品化

システムを構成する機能群を全てまとめてしまうのではなく、機能はスクリプト言語で個別にまとめ、部品とする。さらに、部品化した個々の機能を統括する機構が必要であるため、その役割を果たすコントローラをシステムに設ける。部品化した機能はシステムからコントローラを介して直接呼び出すことができる。

また、外部ツールを用いる場合、必要な情報をパイプを用いてスクリプトが得る。

3.3 メタスクリプト

しかし、上記の方法だと、逐一機能を呼び出さなくて

はならないため、面倒である。そこで、一連の処理の流れをあらかじめ記述しておき、その流れの記述どおりにシステムが自動で処理を行う機構を設ける。こうすれば、一度処理の流れを記述してしまえば、あとは必要な時にそれを呼び出すだけで済み、ユーザの労力が軽減される。そこで、スクリプト言語により個別に部品化した教材を、【部品の呼び出し】、【条件分岐】、【繰り返し】を記述できる独自のメタスクリプト（表 1）で組み合わせる。つまり、本論におけるメタスクリプトとは、個々の機能であるスクリプトを用いた作業の流れを記述しておくスクリプトのことである。

表 1: メタスクリプトの持つ機能

・スクリプトファイル、メタスクリプトファイルの呼び出し
・メッセージボックスの表示とその選択による分岐
・メタスクリプト独自の変数による分岐や繰り返し
・テキストファイルの表示

これにより、欲しい機能があった場合、まさにその部分のみを新たに記述すれば、それ以外の部分は既存の部品を再利用することができるため、拡張にかかる労力を大幅に削減することができる（図 1）。

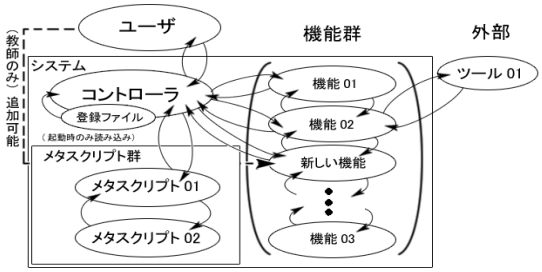

図 1:システム構成図

3.4 実現方法

必要最低限のファイル構成は以下の通りである。システム本体であるコントローラ、機能群を登録しておくファイル、各機能を担当する VBS ファイルとメタスクリプトである VSC ファイル、またそれらを入れるディレクトリである。機能は個々に VBS ファイルで小分けして、コントローラは各 VBS ファイルを統括する。このようにすることで自作の機能がプラグインという形で新規に追加できるため、使用者が機能拡張が行える。また、一個の機能である VBS ソース内で別の VBS を呼び出すことができるため機能の連携を容易に行うことができる。教育者がプラグインとして機能を追加する場合、VBS ディレクトリにソースファイルを入れ、登録ファイルを変更し、システムを再起動すれば追加は完了する。

4. 評価

実験として、新しく機能を追加するという場面を設定して、実際に追加作業を行い、その労力を検証する。

場面設定は現在のシステムに新しい機能を追加するものとし、

A：本システム上で既存の部品を利用しながら作業を行った場合

B：本システムを使わず、一つのソースファイル内に全て記述した場合

C：C/C++言語で記述してバイナリファイルを作成した場合

の 3 パターンを行い、比較検討する。労力の目安として、ソースのテキスト量を用いる。これに加えて、実行時間を考慮して評価する。

結果を表 2 に示すが、労力（ソース量）は 50%～60%ほどに収めることが可能になった。しかし、やはり複数のスクリプトに渡りスクリプトを実行するアプリケーションを起動・終了しているため、実行時間が B、C に比べて長くなってしまっている。これからの課題として、実行時間の短縮が挙げられる。

表 2: 実験結果

ソース量（文字数）	4,659	7,508	9,293
軽減率	-------	62.0%	50.1%
実行時間（対象ファイル 12 個）	20 秒	6 秒	2 秒

5. おわりに

本研究で、スクリプト言語とメタスクリプトを用いることで拡張可能な支援システムをより手軽に構築できることがわかった。しかし、いまだ問題も多い。

まず、スクリプトによってはサーバ上で走らせることができるものもあるが、その場合の機能実現が未検証である。また、機能を担当するコードがスクリプトファイルとして外に出ているため、学生用のシステムの方では、差し替えや改ざんを防止するような工夫をしなければならない。最後に、繰り返して同じ機能を使用する場合、何度もスクリプトを走らせるアプリケーションを起動・終了させるため、実行時間が長くなることである。実用に耐えうる支援システムとして確立させるには、少なくとも上記の問題点は解決しなければならないと考える。

参考文献

[1]経済産業省ホームページ（http://www.meti.go.jp/）

[2]情報サービス産業売上高と就業者数の推移（http://www.jisa.or.jp/statistics/download/jittai_chart2004.pdf）

[3]小塚沙織「e ラーニングによる生涯学習」, 富山大学, 2006（http://www3.u-toyama.ac.jp/frukawa/Seminar/Y2006/Koduka.pdf）

[4]㈱ネットラーニング（http://www.netlearning.co.jp/index.html）

[5]岡本敏雄, 小松秀圀, 香山瑞恵:e ラーニングの理論と実際, 丸善(2004.11)