エンドユーザコンピューティングを意識した新たなプログラミング教育の一考察

――リレーショナル型データベースを利用したシステム開発を通じて――

習志野市立習志野高等学校教諭　**岡松　英雄**

1．はじめに

近年の大きな変革の中，商業教育は高度情報通信社会の発展に寄与する人材を育成してきた。産業界で必要とされる知識や技術・技能の高度化等を踏まえ技術革新，国際化，情報化等による社会の変化や産業の動向に適切に対応していくために，これからも教育現場では努力を続けていかなければならない。

こうした中，現在の商業教育では，情報化を利用者側の立場から積極的に推進するエンドユーザコンピューティング（以下，EUC）やエンドユーザデベロップメント（EUD）の重要性が日増しに高まっている。従来のプログラミング言語中心の教育に変わり，EUC の重要な担い手であるシステムアドミニストレータの育成教育に力を入れ始めた学校も多く見受けられるようになってきた。これらの EUC の積極的な推進は，同時に利用者側の立場に立ったシステム構築が，これまで以上に重要になるということを意味する。すなわち EUC の重要性が増すほど，システム開発に必要不可欠なプログラミング教育も重要になるのである。そこで本研究は EUC を意識したプログラミングの指導を主題とした。データベースソフトウェアをシステム開発のツールとして利用しながら，設計から運用に至るまでの開発を行う。EUC の存在を常に意識し，必要なプログラミングとは何かを考え，業務の効率化，合理化を積極的に推進することを目指す。その過程で，問題解決能力や自発的，創造的な学習態度を育てられるよう探求し，今後のプログラミング教育のあり方について考えていきたい。さらにこの研究が，高度な情報技術者の育成や新たな産業領域の形成に役立つような人材の育成につなげられるよう課題を持って進めていく。

2．EUC を意識したプログラミング教育

（1）　今後のプログラミング教育

本題に入る前に，今後のプログラミング教育がどのような方向に進むべきかを考えていきたい。

社会経済全体における ICT（Information and Communications Technology －情報通信技術）利用の拡大及び，ICT 産業全体に占めるソフトウェアやサービスによる付加価値の増大につれ，単なる ICT の使い手としてのみならず価値創造者としての高度 ICT 人材育成の必要性が増大している。こうした中で，特にソフトウェアに関し，システム全体の設計・構築や管理・運営を担当するなどの高度な情報技術者の育成や新たな産業領域の形成に役立つような人材の育成が重要な課題となっている。これらは，システムアドミニストレータのような利用者側の立場にいる人々の利用すべきソフトウェアやツールを提供すべき立場の人間が不足しているということを意味している。すなわち，誰もが ICT 技術を使えるようになることも重要な課題であるが，それらを誰もが使えるように提供すべき人材の育成は，さらに重要な課題である。この点からも，今後のプログラミング教育が，いかに重要であるかを垣間見ることができる。最近では，「日本の情報教育・情報処理教育に関する提言 2005」が情報処理学会情報処理教育委員会から出され話題になっている。ここでは，「国民全体の ICT 水準を底上げし，ICT 化を担う人材を継続的に輩出しなければ，わが国の将来にわたる発展は不可能である」と述べられている。

（2）　文法・機能中心のプログラミングから実践的なシステム開発へ

次にプログラミング言語を学ぶことへの高い意識に基づいて，具体的な指導方法について論じていく。従来のプログラミング教育は言語の文法や機能を習得することが中心になってきた。EUC を意識したプログラミングでは，その知識を更に発展させていく必要がある。実務的・実践的な問題を解決することを通じ，エンドユーザ（以下，EU）の立場から評価されるプログラミングとはどのようなものかを考えていきたい。開発では，初期の段階から，プロ

グラムの品質を意識させ，課題の意味，重要性を認識させることが大切である。課題の意味・重要性を認識させ，課題に集中するために最も効果的方法は，生徒自身がEUになり，評価していくことである。これらを解決するためにある程度大きいシステムを作り上げることを最終課題とし，はじめはその課題の一部である，比較的容易に作成できる課題から学習する方法をとる。これまでのものを復習しながら，新たな文法や機能を習得していく。学習が進むにつれ，徐々に課題は難しくなるが，過去のシステムを修正することで対応できるため負担は少ない。またシステムの改良，メンテナンスも同時に体験できる。最終的にシステムで開発されるプログラムのステップが大きくなれば，その開発過程で，プログラムの品質についても学習する機会ができる。

（3） システム開発の手法

上記（2）で論じたとおり，本稿では課題を漸進的に展開し，徐々にシステムを進化させながら，学習を進めていく展開を提案する。現在のシステム開発の手法では，プロトタイピングを用いたスパイラルアプローチによる開発ということができる。これらの手法を用いる理由は，EUCを意識したプログラミングを行うことを研究主題としているのはもちろんのこと，現在のシステム開発の主流が，これらの方式であることも意識している。プロトタイピングでは，システム開発の早い段階で試作品を作成するので，利用部門と開発部門との認識のずれやあいまいさを取り除くことができる。EUCの意向に合わないものは開発できないため，EUCを意識した開発が進むこととなる。スパイラルアプローチでは，システムの一部の開発プロセスを繰り返していくので工程の後戻りが起こりにくい。授業という限られた時間の中での開発は，後期の段階に入ってからの後戻りは困難であるので，この方式が望ましい。また演習課題の形で漸進的に展開し，徐々にプログラムを進化させることが可能である。システム開発だけでなく，授業そのものがスパイラル方式の学習として成立する点も意義深い。EUCの視点からは「使いやすさ」や「高い保守性」や「効率の良さ」などが期待され，現在のプログラミング教育では，初期の段階からプログラムの品質を意識させることが望ましい。この学習アプローチでは開発するプログラムは，規模が徐々に大きくなるため，その過程でプログラムの品質を意識させることも可能になろう。基本的なアルゴリズムや文法を理解したうえで，利用者の要求を真に満足し，仕様拡張への対応を意識したプログラミングを教えていく必要もあるだろう。これまでプログラミングや表計算ソフトウェアを学んできた生徒は，言語の文法や機能を習得することを中心に行ってきた。個々のテーマごとに例題や演習課題を行う形式は，市販されている教材でも多く見られる。このような細切れの課題対応型だとコードの書き方は習得できても，どのようなプログラムが良いプログラムであるかの判断や課題の意味・重要性の認識を持たせることは困難である。今回はこれらの問題の解決を図り，実習・体験を通じ利用者の立場からの評価を常に心がけるよう開発する。

（4） 指導内容の概要

今回の開発では，学校における講座選択についての作業を省力化することを題材に，システム開発を行うものと設定した。生徒が選択する各講座が，適切に選択されているかをチェックし，人数の集計や諸表の印刷・講座の管理までを行えるものを目指す。使用するツールとして，リレーショナル型データベースソフトウェアである「Microsoft Access 2003」（以下，Access）を使用する。開発言語はAccessに付属の「Microsoft Access Visual Basic」を使用する。この言語は，市販の「Visual Basic 6.0」相当の機能を持つデータベースプログラミング用にカスタマイズされた言語である。システム開発の実際の現場に立つと，言語の選択権は開発側にはないのが実情である。その中でAccessを選択した理由は，システム開発ツールとして基本的な内容から，高度な開発まで柔軟に対応することができる優秀なソフトウェアとしての実績が認められるためである。またEUCのツールとして実務でも使用されることが多く，ワープロ・表計算とともに各学校にも導入されている実績が多いのもその選択理由である。

最終的な設計イメージとしては次の（図４－１）のような形を想定している。

クラス・番号・氏名を入力後，選択講座を入力し，「登録ボタン」を押すことにより，各人が入力した講座をデータベース化する。その際，「同じ講座は取れない」や「英語Ⅱは英語Ⅰを履修した後でないと選択できない」などの条件を満たしているかを確認する。諸表としては，「クラス別選択講座一覧」「選択講座名簿」「選択人数集計表」などを出力する。

最初の段階では，このように完成され，すでに稼

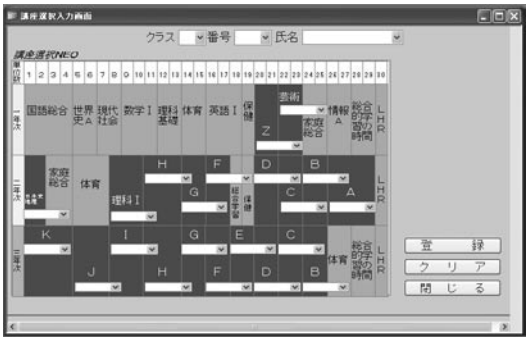

図４－１

動しているシステムを提示することが重要である。この作業こそまさにプロトタイピングであり，今後の開発を進めていく上で，各自がイメージを持って作業を進めることができるのである。しかし実際の指導で，いきなりここまでの形を，生徒に求めることは困難であるため，実際の開発は，この例示を簡略化したものから進めていく。

紙面の都合で，具体的な指導内容の詳細については割愛するが，実際には Access のモジュール機能を活用しながら，目標とするシステム開発を漸進的に進めていく。途中，プロトタイプの相互評価を生徒間で行い，使用感を確認させながら，開発を進めていくことが有効である。

３．新たなプログラミング教育

（１） ブラックボックスである自動化ツール

今日では ICT 化に伴い，既存の手作業の多くはコンピュータを用いた自動化ツールにより置き換え可能となり，作業効率は画期的に向上した。また以前であれば，方法は分かっていたがあまりに時間がかかり不可能だったような解決方法も，実現可能となった。また一方で，従来であれば適用方法について十分理解した人だけが利用できた解法を，その詳細は理解せず，ツールの操作ができるだけの人でも利用可能になった結果，解法をその適用範囲外にまで不適切に使用して無意味な結果を得たり，そのような結果を「コンピュータが計算したものだから正しい」という盲目的な信頼にもとづいて実地適用したりするなどの問題も生じている。

「日本の情報教育・情報処理教育に関する提言 2005」の中で，上記の問題を克服していくために，「実践力を伴う情報処理の理解が必要だ」と提言している。そこでは，コンピュータの本質について，「手順的な自動処理」であることを，体感的かつ具体的に理解することが重要であると述べられている。すなわち，Office ソフトの使用法のような情報リテラシー教育だけでは，コンピュータの本質を理解できたとは言えず，コンピュータに何ができて何ができないかといったことがらを判断できるようにはならないというのである。コンピュータをブラックボックスとして扱い，中に一定のメカニズムが内蔵されているという意識を持つことなく，使用法だけを習得することでは，論理的な力が身につくはずもない。高等学校では論理的な力が発達する時期であるので，抽象化の考え方に触れさせながら，そのメカニズムを体験的に学ぶ必要がある。知識だけの理解にとどまるのでなく，実際にプログラミングに代表される「手順的な自動処理」に接して体験し，その特性を身体的に納得することが重要だということである。このような体験的理解をもった人であれば，学んだことがなく全く新たに遭遇するような場面においても，体験的理解を土台として「コンピュータはこのように動作するだろう」「したがってこういうことはできるが，こういうことはできないだろう」という判断をある程度的確に行えるはずである。これが「体感的かつ具体的」の意味するところである。

自動化ツールを使う人は，上記の意味での体感的・具体的であり実践力を伴う「情報処理の理解」があってはじめて，その中で「何が起きているか」を理解できるはずである。解法の詳細までは熟知しないとしても，感覚的に納得し，そこに一定の制約があり，それを外れたことをすれば出て来る結果が無意味であることも理解できるはずである。

（２） アルゴリズミックとヒューリスティック

本研究では，今後のプログラミング教育のあり方についてテーマを設定し，探求を続けてきた。プログラミングはシステム開発の一領域であるため，システム開発の手順を踏みながらあるべき姿を模索した。私はこの研究を始める前に，納得できないひとつの課題があった。それは“プログラミング言語離れ”とでもいうべき問題である。教育現場では，従来のプログラミング言語教育からシステムアドミニストレータ教育へとシフトしている学校が増えていると聞く。プログラミングの技術が，社会で全く必要とされていないのなら理解できるが，社会ではこれらの技術者が圧倒的に不足し，深刻な状況を招いている。平成 17 年に（社）日本経済団体連合会か

ら発表された「産学官連携による高度な情報通信人材の育成強化に向けて」という提言には，「わが国の高度情報通信人材の育成は危機に瀕し，世界的に見ても大きく後れを取っている」と述べられている。しかしそれに対応する環境は，まだ充分とはいえないのが現状である。この矛盾を考えながら，本研究を通して，私は２つの結論にたどり着いた。その１つ目には，情報活用能力の応用としてのプログラミング能力の必要性があげられる。現在，システムアドミニストレータ教育の中心は，情報活用能力を養うことに重きが置かれている。ここで教えられる比較的平易な文書作成や表計算は，各ソフトウェアの標準的機能を使用すれば実現可能な内容である。これらの活用能力の育成にはまだまだ浸透する時間が必要であろう。しかしシステムアドミニストレータにとって，システム開発への参画も任務の１つであることを忘れてはならない。ソフトウェアに内蔵されている各種ツールや簡易言語を使用しなければならない場面で，そのスキルを遺憾なく発揮できる能力の育成が必要とされてくるはずである。すなわち，情報活用能力の応用として，プログラミングの技術を学ぶ必要性が出てくると考えられる。今回の研究で，Access のモジュール機能を使ってシステム開発を行ったように，Excel ではマクロ機能を使いながら，プログラミングを行っていくという考えである。そこでは従来の情報活用能力はもちろんのこと，今回研究したシステム開発の基本的な知識も理解していく必要がある。そして結論としての２つ目にはアルゴリズミックとヒューリスティックの理解があげられる。今後も使用されるプログラミング言語は変化していくことが考えられる。新たに優秀な言語やマクロツールが開発され，淘汰されていくだろう。その流れの中で，最も大切な能力が，アルゴリズミックな論理的思考が出来る能力だと考える。人間が経験則から感覚や思い付きで行動することや，考えたりすることをヒューリスティックな思考という。しかし，たとえそれが自分自身で論拠を明確に出来ない，ヒューリスティックな思いつきの行動であったとしても，裏には必ず論理的な根拠がある。この根拠を認識し，筋道を立てて考えることが出来る能力（アルゴリズミック）を身につけることが，これからの情報教育で最も重要な要素の１つである。プログラミング教育はこのアルゴリズミックな思考能力を身につけることの出来る教育であると信じてやまない。ニーズにあったシステムを実現できるアルゴリズムが出来ていれば，プログラミング言語は何を使っても同じである。この考え方がこれからの情報処理教育に重要ではないかと考えるのである。

４．おわりに

わが国は高度 ICT 社会に向けて基盤の整備を進めているが，設備機器や行政制度などの側面と比較し，人材面の育成面では大きく遅れを取っているのが現状である。世界的に猛烈な勢いで ICT 化が進展するなか，資源に乏しいわが国にとって，今後，ICT を通じて高付加価値の製品・サービスを提供し，国際競争力を高めていくために，その基盤を支える高度 ICT 人材の育成を進めていくことがより重要になってくる。これらの問題に対処し，世界にまたがる高度 ICT 社会化の波に立ち後れないためには，過去の情報処理教育を精査し，新たな情報教育を模索していく必要がある。今日の ICT 社会が以前の社会と決定的に異なるのは，コンピュータによる自動化された情報の蓄積 / 加工 / 伝達が，あらゆる情報の短時間での高度な取り扱いを可能にし，その量的変化，および自動化による人間の介在不要という質的変化が社会に大きな変化をもたらしているという点である。そして今日のわが国の情報システム開発において，個別専門分野と情報技術の両方に精通した人材の不足が，適切なシステム開発の妨げとなり，多大な損失を招いていることに留意する必要がある。

商業教育における情報教育は新たな段階へ進むべき時期にきているが，この個別専門分野と情報技術の両方に精通した人材が育成できる教育こそが，商業教育であるといえる。これまで社会のニーズに的確に対応し，優秀な人材を送り出してきた商業教育に寄せられる期待は大きい。私自身も商業情報処理教育を担う教員として，ICT を活用し価値創造を可能とする人材の育成と社会への貢献をテーマに，更なる研究と修養に励み，諸問題の探求を続けていきたい。

参考文献

・「高等学校学習指導要領解説　商業編」　（文部省）
・「日本の情報教育・情報処理教育に関する提言 2005」
（情報処理学会情報処理教育委員会）
・「産学官連携による高度な情報通信人材の育成強化に向けて」　（(社）日本経済団体連合会）
・「スパイラルアプローチによるＣ言語プログラミング教育の実践と評価」