メディア教育研究　第1巻　第1号
Journal of Multimedia Aided Education Research, 2004, No. 1, 115－122

論　文

教育用計算機システムシミュレータED21の設計と評価

三浦　義之[1)]・金子　敬一[2)]・中川　正樹[2)]

本研究では、計算機ハードウェアの動作原理と同時に高水準プログラミング言語で記述されたプログラムの実行原理を学習することができる計算機システムシミュレータを設計した。このシステムは、WWW環境で利用できる機械語シミュレータ上に、新たに設計した教育用の高水準プログラミング言語のためのコンパイラ、エディタ、コンパイルブラウザを追加して構成した。コンパイルブラウザは、高水準プログラミング言語で記述されたプログラムがアセンブリコードに変換される過程を視覚化するものである。システムの有効性を評価するため、本学の計算機科学における導入教育で実験を行った。その結果、学生がプログラミング言語の実行原理を理解するのに有効であることが確かめられた。

キーワード
シミュレータ学習、コンパイラの視覚化、プログラミング教育、システム評価

１．はじめに

今日、計算機の動作の仕組みを学習するために、教育用の計算機シミュレータが広く使われている[1, 5, 8]。しかしながら、現在提案されているシステムの多くは、ハードウェアの動作原理を学習するものがほとんどである。これにより、ハードウェアの動作原理と高水準プログラミング言語を同時に学習していても、高水準プログラミング言語で記述されたプログラムがどのようにして実際の計算機で実行されているか、ということを学生が理解できないことが報告されている[3, 4]。この問題を解決するため、Deckerらはプログラムの断片がどのようにアセンブリコードに変換されるかを視覚化するシステムを開発した[3]。しかしながら、このシステムが視覚化を行うのは代入文のコンパイル過程のみであり、プログラム全体の翻訳・実行過程を理解するには、不十分である。そこで本研究では、高水準プログラミング言語で記述されたプログラムを完全に翻訳するコンパイラと、その過程を視覚化するコンパイルブラウザを備えた、教育用の計算機システムシミュレータED21を設計した。また、システムの開発には、メディア教育開発センターによるメディア教材開発支援を受けた。ED21は、大学1年生程度の計算機の初心者を対象としている。さらに、ED21の学習効果を評価するため、高水準プログラミング言語の実行過程の理解について、CSコースの大学1年生を対象に評価実験を行い、その有効性を確認した。

本論文は、以下のように構成される。まず、第2章では、ED21の設計原理について説明する。さらに、第3章では、コンパイルブラウザについて述べ、第4章で評価実験とその結果について説明する。最後に、第5章で、結論と今後の課題を示す。

２．ED21

本章では、教育用の計算機システムシミュレータED21の設計について詳細に述べる。

2.1　システム構成

ED21は、ED9900[9] をベースにし、以下の拡張を加えた：

－教育用高水準プログラミング言語EL21の設計
－エディタEE21の実装
－コンパイラEC21の実装

ED9900はJavaAppletで実装されているので、これらの拡張も同様にJavaAppletで行った。

ED21では、学習者は最初にEE21でEL21のプログラムを編集する。プログラムが完成したら、EC21を呼び出しED21のアセンブリコードに翻訳する。最後に、ED21上で翻訳されたアセンブリコードを実行し、その際学習者はプログラムの実行過程を観察することができる。図1にED21の概観を示す。

1) 東京農工大学大学院工学府
2) 東京農工大学大学院共生科学技術研究院

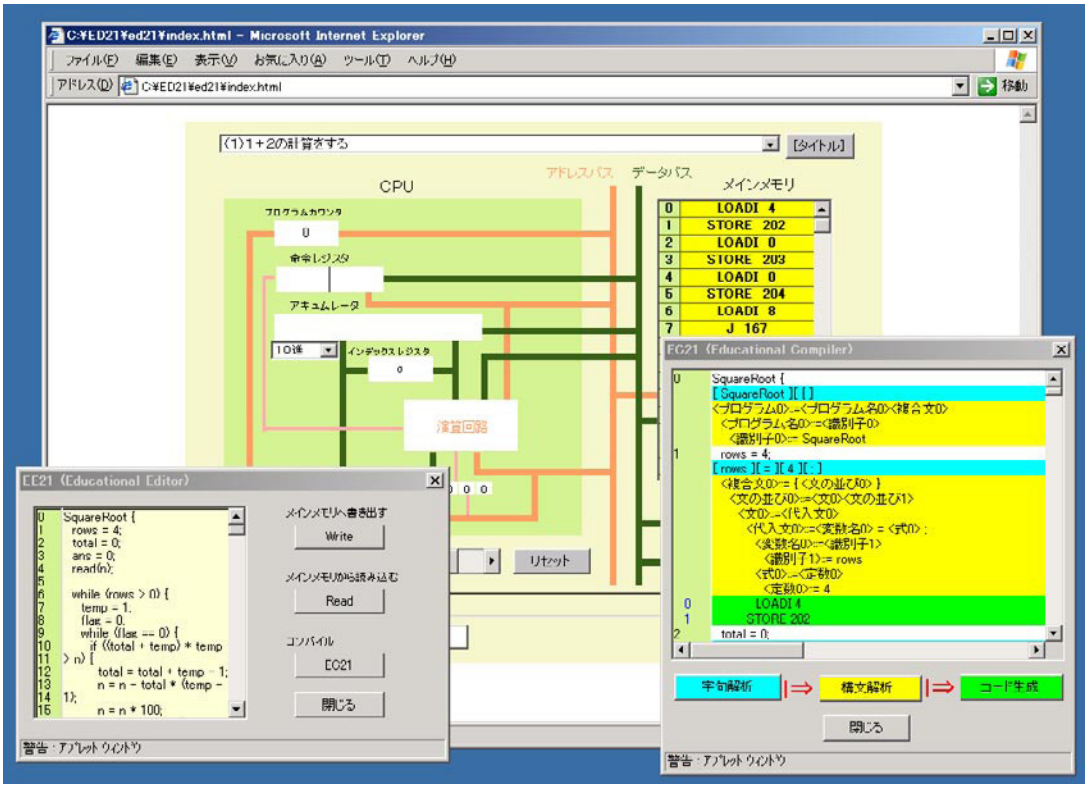

図1　ED21の概観

各拡張項目については、次節以降で詳しく述べる。

2.2　EL21

EL21は、計算機の初心者がプログラムの実行過程を学習する目的で設計された、単純なプログラミング言語である。教育用のプログラミング言語として、他にLogoやBasicが有名である[2, 6, 7]。しかしながら、Logoは子供向けに開発されたもので、論理的な思考が発達した大学生のプログラミング教育には不向きである。またBasicについても、実行時のライブラリが必要であり、それによりプログラムの実行が不明瞭となってしまう。そのため、新しい言語を設計することにした。EL21の言語仕様は、手続き型プログラミング言語を参考にして設計した。これは、学習者がいずれCなどの手続き型プログラミング言語を学習する段階へ移行することを考慮してのことである。

ED21は初心者を対象にしているので、EL21の言語仕様も学習者が一目見て理解できる程度に簡単なものにした。一方で、学習者がプログラミングを楽しめる程度の機能を有することも目指した。このため、EL21は5つの文（代入文、if文、while文、read文、print文）を持つ設計にした。一般に他のプログラミング言語にはfor文が含まれている。しかしながら、for文はwhile文に比べて構造が複雑であり、複数の意味論が関連付けられる可能性があることから学習が困難であることと、繰り返し処理はwhile文があれば実現できることから、EL21の言語仕様からfor文を除外した。関数呼び出しについても、引数渡しに関する複雑な意味論があるため、言語仕様から除外した。図2に、EL21の言語仕様を示す。

```
<プログラム> ::= <プログラム名> <複合文>
<プログラム名> ::= <識別子>
<文> ::= <複合文> | <if文> | <ifelse文> | <while文> |
                <read文> | <print文> | <代入文>
<代入文> ::= <変数名> = <式>;
<if文> ::= if ( <式> ) <複合文>
<ifelse文> ::= if ( <式> ) <複合文> else <複合文>
<while文> ::= while ( <式> ) <複合文>
<read文> ::= read(<変数名>);
<print文> ::= print(<式>);
<複合文> ::= { <文の並び> }
<文の並び> ::= <文> <文の並び> | <文>
<変数名> ::= <識別子>
```

図2　EL21の言語仕様

2.3　EC21

EC21は、EL21で書かれたプログラムをED21のアセンブリコードに翻訳するコンパイラである。EC21は4つのフェーズ（字句解析、構文解析、中間コード生成、目的コード生成）から構成される。字句解析は、バックトラックに基づく単純なアプローチをとっている。構文解析系は、予測再帰下降型である。また、中間コード生成には、解析木を使用している。目的コード生成には、単純な再帰アルゴリズムを用いている。

通常、ほとんどの実用的なコンパイラには、コードを

```
Factorial {
  f = 1;
  read(n);
  while (n <> 0) {
    f = f * n;
    n = n - 1;
  }
  print(f);
}
```

図3　EL21ソースプログラム

```
//Factorial
  0:LOADI 1
  1:STORE 109
  2:LOADI 4
  3:J 74
  4:STORE 110
  5:LOAD 110
  6:STORE 111
  7:LOADI 0
  8:SUB 111
  9:JZ 23
 10:LOAD 109
 11:STORE 111
 12:LOAD 110
 13:MUL 111
 14:STORE 109
 15:LOAD 110
 16:STORE 111
 17:LOADI 1
 18:STORE 112
 19:LOAD 111
 20:SUB 112
 21:STORE 110
 22:J 5
 23:LOADI 27
 24:STORE 28
 25:LOAD 110
 26:J 29
 27:STOP 0
 28:0
//Print
 29:STORE 70
 30:LOAD 70
 31:JZ 58
 32:JM 35
 33:LOAD 68
 34:J 38
 35:LOADI 45
 36:OUT 9
 37:LOAD 69
 38:STORE 67
 39:LOAD 70
 40:DIV 67
 41:STORE 71
 42:SHIFTR 16
 43:STORE 70
 44:LOAD 67
 45:DIV 65
 46:STORE 67
 47:JZ 56
 48:LOAD 71
 49:ADD 73
 50:JZ 55
 51:SUB 73
 52:ADD 66
 53:OUT 9
 54:STORE 73
 55:J 39
 56:STORE 73
 57:LOAD 71
 58:ADD 66
 59:OUT 9
 60:LOAD 72
 61:ADD 28
 62:STORE 63
 63:0
 64:-1
 65:10
 66:48
 67:0
 68:10000
 69:-10000
 70:0
 71:0
 72:55296
 73:0
//Read
 74:ADD 107
 75:STORE 100
 76:LOADI 0
 77:STORE 101
 78:IN 0
 79:SUB 106
 80:STORE 108
 81:JZ 84
 82:ADD 106
 83:J 85
 84:IN 0
 85:SUB 105
 86:STORE 102
 87:JM 95
 88:ADD 104
 89:JM 95
 90:LOAD 101
 91:MUL 103
 92:ADD 102
 93:STORE 101
 94:J 84
 95:LOAD 108
 96:JZ 99
 97:LOAD 101
 98:J 100
 99:SUB 101
100:0
101:0
102:0
103:10
104:32758
105:48
106:45
107:55296
108:0
//Variable f
109:0
//Variable n
110:0
```

図4　EC21により生成されたコード

最適化するフェーズがある。しかし、最適化しないコードの方が、初心者には分かりやすい、余力がある学習者には、自分でコードを最適化させる課題に取り組ませたい、という理由からEC21にはコード最適化フェーズを実装しないことにした。

図3と図4に、入力した数の階乗を計算するEL21のプログラムと、そのプログラムをコンパイルした結果得られるED21のアセンブリコードを示す。なお、ED21のアセンブリコードにある行番号とコメントは、著者らが書き加えたものである。readとprintルーチンは、比較的長いコードになっているため、その分メモリを消費してしまう。そのため、コンパイルされたコードでは、readとprintの少なくともどちらかが複数ある場合には、サブルーチン化することが望ましい。しかしながら、ED21ではスタック機能は、初心者には難しいという理由からサポートされていない。そこで、readとprintルーチンが呼ばれるときには、戻り番地を自己書き換えする形式のプログラムを採用した。なお講義では、自己書き換えプログラムの細かい説明は省いている。

2.4　EE21

ED21は、計算機の初心者を対象としており、そのような学習者はファイル操作にも慣れていない可能性がある。そこで、我々はED21上で動作するエディタEE21を用意した。すなわち、学習者がファイルの概念を学んでいなくても、このEE21を使うことにより、プログラムの編集、コンパイル及び実行をすることができるようになっている。図5にEE21の概観を示す。

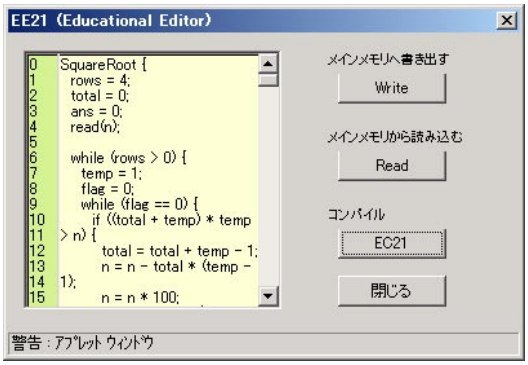

図5　EE21の概観

学習者はEE21を使ってEL21のプログラムを編集し、プログラミングが終わったら、右側のEC21ボタンを押すことで、プログラムがEC21へ転送される。EC21がプログラムをコンパイルしてED21のメモリに生成されたコードを書き出すとともに、コンパイルブラウザを起動する。コンパイルブラウザは、EC21が行ったコンパイル過程を表示する。

またEE21は、ED21のアセンブリコードも編集することができる。そのために、EE21にはED21のメモリエリアとEE21のテキストエリアの間で、アセンブリコードとデータを読み書きする機能が備わっている。この機能を使うことで、学習者はアセンブリコードをプログラミングする他、手動で最適化を行うために利用することもできる。

3．コンパイルブラウザ

本章では、コンパイルブラウザの詳細について述べる。コンパイルブラウザは、学習者がコンパイルの役割を認識するために導入されたものである。

3.1　目　的

前節で述べたように、EL21で書かれたプログラムは、EC21によってED21のアセンブリコードに翻訳される。コンパイルブラウザは、このEC21による翻訳過程を視覚化する。その目的は、学習者に以下の点を理解させることである。

- ソースプログラムをコンパイラがアセンブリコードに翻訳して、その形式になってから計算機が実行すること
- 翻訳過程は、字句解析、構文解析、コード生成から構成されること
- 翻訳が機械的に実行可能であること

コンパイルブラウザは、学習者が厳密に翻訳過程を理解することを意図するものではない。学習者がコンパイラの存在とその役割を大まかに認識することで、高水準プログラミング言語の実行方式について理解を深めることが狙いである。

3.2　コンパイル過程の視覚化

本節では、コンパイルブラウザにより視覚化される内容について述べる。コンパイルブラウザは、EC21のプロセスを視覚化する。このEC21は、4つのフェーズ（字句解析、構文解析、中間コード生成、目的コード生成）から構成される。これらのフェーズのうち、字句解析、構文解析、目的コード生成の視覚化を行う。単純化のため、中間コード生成の視覚化は省いた。

EE21でプログラムの編集が終わったら、EE21にある「EC21」ボタンを押すことで、EE21に書かれたプログラムはEC21に転送される。この際、コンパイルブラウザが起動し、新しいウィンドウが開く。プログラムの翻訳過程は、そのウィンドウに表示される。図6に、コンパイルブラウザの実行過程を示す。

図6に示したとおり、対応するボタンを順にクリックすることで3つのフェーズの実行結果を表示することができる。コンパイルブラウザのコード生成フェーズが、実際にEC21が行っている目的コード生成にあたる。

コンパイルブラウザでは、それぞれのボタンはトグル式になっており、ボタンを押すことで対応するフェーズの結果の表示／非表示が切り替わる。このボタンにより、学習者は全ての結果を非表示にしてソースコードだけを見たり、1つのフェーズの結果だけを見たり、全てのフェーズの結果を同時に見たりすることもできる。一方、EL21のプログラムにエラーが存在するときには、コンパイルブラウザがエラーメッセージを表示する。

4．評　価

本章では、教育用計算機システムシミュレータED21の有効性を検証するために実施した評価実験について述べる。

この評価実験の目的は、ED21の使用がハードウェアとソフトウェアとの関係について学習するのに有効であることを検証することである。

4.1　実験内容

本実験は、本学CSコースの1年生75名を対象にして、第1セメスター後半の4回の講義を利用して行われた。今回の実験では、コンパイル処理の存在とその役割、高水準プログラミング言語の実行過程など、ハードウェアとソフトウェアとの関係についての理解度を、テストに

(a) 起動直後

(b) 字句解析後

(c) 構文解析後

(d) コード生成後

図6 コンパイルブラウザの実行過程

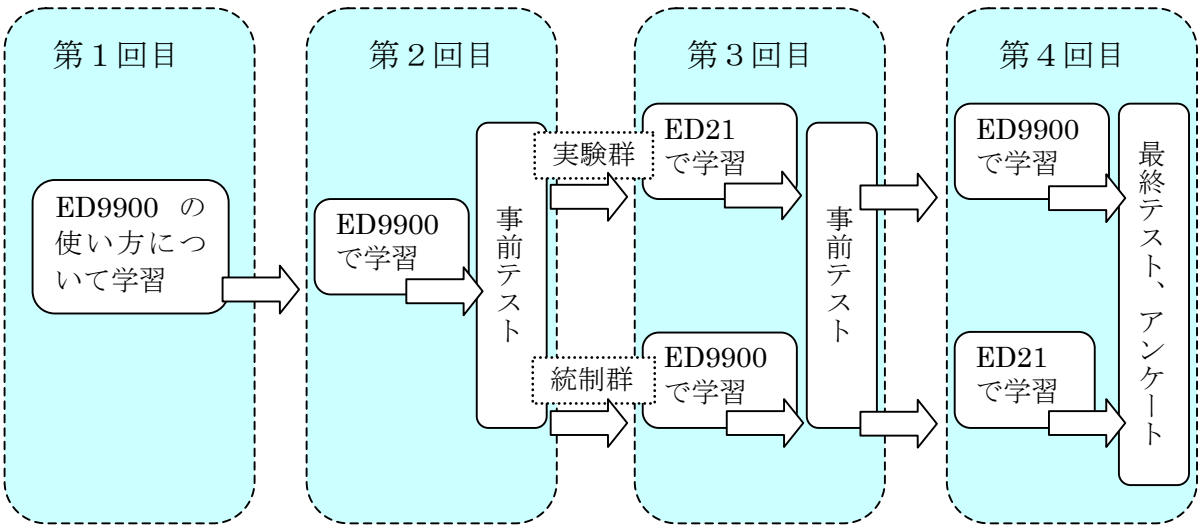

図7 実験の流れ

より評価した。

実験に際して学習者を、EL21でプログラミングをして高水準言語の動作原理を学ぶグループ（実験群）と、EL21は使わずアセンブラプログラミングについて深く学習するグループ（統制群）に分けた。この2つのグループについて、実験前後のテストの点の伸びを比較することにより、ED21の有効性を検証する。大まかな実験の流れを図7に示す。

学習者は、4回の講義のうち前半2回でED9900を利用して計算機の動作原理やアセンブラプログラミングの基礎を学ぶ。また、彼らはC言語を使ったプログラミングの学習も、別の講義・演習で並行して行っている。したがって、事前テストを行う段階では、ある程度ハードウェアとソフトウェアについて学習を行っている、と見なすことができる。

ここで、学習者を2つのグループに分けた。グループ分けには事前テストのうちの、コンパイラ・高水準言語の動作原理に関する設問の成績を使用した。

さらに、学習内容を公平にするため、4回目の講義は、実験群はアセンブラプログラミングについてのさらなる学習を、統制群は高水準言語の動作原理についての学習を行った。さらに、第3回目と第4回目の講義の間に、レポート作成を通じてより学習が進んでいることも考えられるため、第4回の講義終了後にも最終テストを行った。

4.2　実験結果

実験群と統制群の2グループについて、事前テスト、事後テスト及び最終テストの平均点を、表1に示す。

表1　2グループの平均点（平均正答率）

	事前テスト	事後テスト	最終テスト
実験群	14.18(59.08%)	15.82(65.92%)	16.18(67.42%)
統制群	14.32(59.67%)	14.11(58.79%)	15.64(65.17%)

表1に示したとおり、事前テストと事後テストとを比較すると、統制群はわずかに平均点が下がったのに対し、実験群は上昇している。実験群と統制群の事後テストの平均点について、t検定を行ったところ、統計的に有意差があることが確認された。また、事前テストの平均点は2グループで有意差がなく、第4回目の講義の後に行った最終テストの平均点も、同様に有意差がなかった。このことから、グループ間で学力に差がなかったことが言える。

以上の実験結果から、コンパイルブラウザを実装したED21が、高水準プログラミング言語の実行原理の理解に有効であることが示された。

4.3　その他実験について

何人かの学生は熱心に取り組み、非常に面白いプログラムを作った。図8に一例を示す。このプログラムは、入力として整数を読み込み、その数の平方根を計算して表示するプログラムである。EL21は整数型しかサポートしていないので、このプログラムでは計算に固定小数点数を用いている。

```
SquareRoot {
 rows = 4;
 total = 0;
 ans = 0;
 read(n);

 while (rows > 0) {
  temp = 1;
  flag = 0;
  while (flag == 0) {
   if ((total + temp) * temp > n) {
    total = total + temp - 1;
    n = n - total * (temp - 1);
    n = n * 100;
    total = total + (temp - 1);
    total = total * 10;
    ans = ans * 10 + temp - 1;
    flag = 1;
   }
   temp = temp + 1;
  }
  rows = rows - 1;
 }
 print(ans);
}
```

図8　平方根を計算するEL21プログラム

ほとんどの学生は好印象だったが、何人かの学生からは難しすぎる、という意見が出た。しかし、この点については、我々の意図に反してコンパイルブラウザの出力を完全に理解しようとしたため、のようであった。この問題を解決するため、現在よりグラフィカルなコンパイルブラウザの実現を検討している。

5．結論及び今後の課題

従来のシミュレータに、エディタ、コンパイラ、コンパイルブラウザを拡張することで、計算機のハードウェアだけではなく、プログラミング言語の実行原理についても学習可能な計算機システムシミュレータED21を設計した。評価実験の結果、ED21が、プログラミング言語の実行原理を理解するのに役に立つことを示した。

今回実現したコンパイルブラウザでは、テキストによる視覚化を行っている。初学者にも分かりやすいように、グラフィカルに視覚化をすることが、今後の課題である。また、高水準プログラミング言語の実行原理に加えて、

操作系の実行過程についても学習可能な拡張も今後の課題である。

参考文献

［1］ Barua, S.: “An Interactive Multimedia System on －Computer Architecture, Organization, and Design－.” IEEE Transactions on Education, Vol. 44, No. 1, pp.41-46, 2001.

［2］ Dann, W., Cooper, S., and Pausch, R.: “Making the Connection: Programming with Animated Small World.” Proceedings of the 5th Annual SIGCSE/SIGCUE ITiCSE Conference on Innovation and Technology in Computer Science Education, pp.41-44, 2000.

［3］ Decker, R., and Hirshfield, S.: “The PIPIN Machine: Simulations of Language Processing.” Journal of Educational Resources in Computing, Vol. 1, No. 4, pp.4-17, 2001.

［4］ Evangelidis, G., Dagdilelis, V., Satratzemi, M., and Efopoulos, V.: “X-Compiler: Yet Another Integrated Novice Programming Environment.” Proceedings of the IEEE International Conference on Advanced Learning Techniques, pp.166-169, 2001.

［5］ Johnson, M., and Craig, B.: “Computer Systems Pedagogy using Digital Logic Simulation.” Proceedings of the International Conference on Computers in Education, pp.703-704, 2002.

［6］ Jones, A., J.: “Can ICT in Teacher Education Induce Reform in School Learning?” Proceedings of the International Conference on Computers in Education, pp.104-108, 2003.

［7］ Krumholtz, N.: “The Spiral Evolution of Technology Learning Environments: 3 Cycles.” Proceedings of the International Conference on Computers in Education, pp.608-612, 2003.

［8］ Nishida, T., Yahara, J., Masuzawa, T., and Matsuura, T.: “Development and Evaluation of ECAS: A Computer Simulator with Adjustable Degree of Abstraction Based on Educational Objectives.” Proceedings of the International Conference on Computers in Education, pp.954-958, 2003.

［9］ 山口菊子、佐藤泰助、辻　政昭、小谷善行、中川正樹：“高等学校「情報」教科向けWeb/Java教材「ED9900の開発」。”情報処理学会第60回全国大会論文集、pp.375-376、2000。

三浦（みうら）　義之（よしゆき）

平14東京農工大・工・情報コミュニケーション卒。現在、同大大学院博士前期課程在学中。工学士。シミュレーションや視覚化によるマルチメディア教材、教育システムの研究・開発に従事。

金子（かねこ）　敬一（けいいち）

昭60東大・工・計数卒。昭62同大大学院・修士課程修了。同年同大・工・計数助手、平8千葉大・工・情報講師を経て、現在、東京農工大・工・情報コミュニケーション助教授。工博。並列分散計算、部分計算、耐故障計算、関数プログラミング、マルチメディア教育等の研究・教育に従事。

中川（なかがわ）　正樹（まさき）

昭52東大・理・情報卒。昭54同大大学院・修士課程修了。同在学中、英国Essex大留学（M. Sc. with distinction in Computer Studies）。昭54東京農工大・工・数理情報助手。現在、同大・工・情報コミュニケーション教授。理博。手書きパターン認識、手書きユーザインタフェース、教育の情報化などの研究・教育に従事。

Design and Evaluation of an Educational Computer System Simulator ED21

Yoshiyuki Miura[1)] ・ Keiichi Kaneko[2)] ・ Masaki Nakagawa[2)]

In this study, we designed an educational computer system simulator by which learners can understand the execution principle of programs written in a high-level programming language as well as the execution principle of computer hardware. This system is obtained by extending a conventional computer hardware simulator on WWW by adding a compiler, an editor, and a compilation browser for a high-level programming language that we designed for educational use. The compilation browser visualizes the process how programs written in the high-level programming language are translated into the assembly codes. We conducted an experiment to evaluate the effectiveness of our system in an introductive course of computer science in our university and verified its effectiveness for students to understand the execution principle of programming languages.

Keywords
Simulation-Based Instruction, Compiler Visualization, Programming Education, System Evaluation

[1)] Tokyo University of Agriculture and Technology, Graduate School of Engineering
[2)] Tokyo University of Agriculture and Technology, Institute of Symbiotic Science and Technology