様　式　Ｃ－１９、Ｆ－１９、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

平成　２８　年　　６　月　１４　日現在

機関番号：　３２６０１
研究種目：　若手研究(B)
研究期間：　2013～2015
課題番号：　２５７３０２０３

研究課題名（和文）情報システム学に基づくプログラミング教育システムの再構築

研究課題名（英文）Restructuring the Programming Education from the Viewpoint of Information Systems

研究代表者
松澤　芳昭（Matsuzawa, Yoshiaki）

青山学院大学・社会情報学部・助教

研究者番号：４０５１７０１７

交付決定額（研究期間全体）：（直接経費）　　3,100,000円

研究成果の概要（和文）：本研究ではプログラミング教育を「情報システム学」の観点から再構成し，一般国民を対象とする次世代プログラミング教育基盤の構築を行った．主要な成果は，１）ブロックエディタからテキストプログラミングへのシームレスな移行についての学習時間計測による実証，２）オブジェクト指向プログラミングの理解支援ツールの開発，および　３）コンパイルエラーの修正時間減少指標とその視覚化ツールの開発，である．それぞれの項目について実践を行うカリキュラムも開発された．

研究成果の概要（英文）：In this research, we tried to develop the next generation of programming education platform designed for all citizens, by restructuring the current education from the viewpoint of Information Systems. The results are summarized by the following three achievements: 1) the demonstration of a seamless migration from block-based to text-based programing language by measuring time spent by learners in both languages, 2) the development of a visualization tool to support debugging for object-oriented systems 3) the development of a visualization tool to promote learners to reflect their compile error correction experiences. We also developed the new curriculum to demonstrate these results.

研究分野：　情報教育，情報システム

キーワード：　プログラミング教育　Computational Thinking　ビジュアルプログラミング　抽象化技法　メトリクス

１．研究開始当初の背景

IT 技術の発展によって，一般国民がコンピュータを使いこなすためのプログラミング教育は影を潜めていたが，近年英米ではプログラミング教育が再評価されている．英国では操作偏重教育への反省から 80 年代のプログラミング教育が再評価され，プログラミング教育の復活が議論されている[1]．米国では，Computational Thinking ([2], 以下 CT)と呼ばれる「世の中の様々な問題について, 抽象化, 自動化の観点からモデルを組み立て, 解決するスキル」の概念が提唱され，NSF(National Science Foundation)はこの CT に関する教育研究助成を行っている[3].

Wing 氏が提唱する CT とは，21 世紀の知識社会に生きるすべての人に必要な基本的スキルとされており,「コンピュータサイエンスの基礎概念に基づいた問題解決，システムデザイン, 人間活動の理解」を含むものであり，複雑な問題を解決するための抽象化技法の利用が強調されている[2]．これらの特徴は，筆者がこれまでの研究の論拠としてきた「情報システム学(Information Systems : IS)」の概念にほかならない．情報システム学では，人間活動システムとの調和を目指した情報システムのデザインが主目標として据えられてきた[4]. IS における中心テーマは元よりアルゴリズム開発ではなく, アルゴリズムを隠蔽し, 抽象の壁を構築してシステムの複雑性を制御する抽象化技法である．筆者はこのコンセプトに基づく教育の要素技術を提案してきた．

これまでに IS（または CT）を指向するプログラミング教育システムの開発は試みられてきたが，明確な成功事例は存在しておらず，教育効果の測定法も確立していない．抽象化技法育成を指向した著名なプログラミング教育の教科書として,「計算機プログラムの構造と解釈（SICP）」[5]がある．この教科書は長年 MIT（米国マサチューセッツ工科大学）の入門プログラミングコースで使用されてきた．しかしながら，SICP には(1)関数型言語のスタイルが実用言語(C/Java 等)とかけ離れている，(2)MIT 以外の大学では難しくて使えない，という欠点が指摘されており, CT が対象とする万人向け教育への適用は不可能である．この欠点を改善する「How to Design Programs: HtDP」という教科書の提案や，HtDP から Java に移行する「Teach Scheme / Reach Java」プロジェクト,「Program by Design」プロジェクトの提案も行われている．これらのプロジェクトのプログラミング教育目的は汎用的な問題解決や抽象化技法の獲得であり，本研究の目的に最も適合する研究と考えられるが，カリキュラムは構築中であり，効果測定も未発達である．

プログラミング教育の万人への普及を目指した，ビジュアルプログラミング言語の開発によるプログラミング教育改善の試みも進められている. しかしながら, 抽象化技法の獲得のための機能を有したものは存在しない. 唯一, 世界に普及しているビジュアルプログラミング環境 Scratch に対して SICP のコンセプトを組み込んだ BYOB[6] や snap!というシステムは提案されているが，これは単に scheme 言語の要素をブロックに置き換えただけであり，データ抽象（オブジェクト指向）に対応しておらず, 教育効果検証の試みも行われていない.

参考文献）

[1]BBC.: Back to the future of computer science,http://news.bbc.co.uk/2/hi/programmes/click_online/9707886.stm
[2]J. Wing.: Computational thinking. Communications of the ACM, 49(3):33–35, 2006.
[3]NSF.: Computing Education for the 21st Century, http://www.nsf.gov/pubs/2012/nsf12527/nsf12527.htm
[4]浦昭二, 細野公男, 神沼靖子, 宮川裕之, 情報システム学へのいざない:人間活動と情報技術の調和を求めて，培風館, 1998.
[5] Abelson, H. , Sussman, G.J. , JulieSussman : Structure and Interpretation of Computer Programs (邦題 : 計算機プログラムの構造と解釈), MIT Press (1996).
[6]Harvey B., Monig J.:Bringing“No Ceiling” to Scratch: Can One Language Serve Kids and Computer Scientists? Constructionism, 2010, 2010

２．研究の目的

このような背景を踏まえて，本研究ではプログラミング教育を「情報システム学」の観点から再構成し，一般国民を対象とする次世代プログラミング教育基盤を構築することを提案する．上記で問題として取り上げた，(1)カリキュラム，(2)ビジュアルプログラミングツール，(3)教育効果測定法の 3 つの観点から教育システム全体の再構成を行い，理論とツールの確立を目指す．

本研究では以下 3 点を目標とする．

(1)．情報システム学に基づく要求駆動型カリキュラムの開発

SICP のように抽象化技法の獲得を目的としつつ，例題が（数学ではなく）情報システムの一般例で構成されたテキストを開発する．目標はプロセスの抽象化技法を扱う初級編，データの抽象化技法（オブジェクト指向）を扱う中級編の開発である．

(2)．抽象化を促進するビジュアルプログラミングツールの開発

抽象化技法を主軸とするカリキュラムを初学者に展開するために，学習者が足場かけとして利用できるビジュアルプログラミング言語を開発する．目標はプロセス抽象化，データ抽象化（オブジェクト指向）の双方について，学習者が（指導者の強制によるものではなく）自らの意志で自然に抽象化が行えるユーザビリティを持つツールの開発である．

(3)．実証データに基づく教育効果検証理論の開発

本研究の評価実験においては，当研究者らが開発したプログラミング過程記録装置によって測定された実証データに基づく教育効果検証理論を構築する．目標は，(1)コンパイルエラーの修正時間減少指標とアルゴリズム開発への集中度指標の開発と評価，(2)ビジュアルプログラミングから，テキスト型プログラミング言語へのシームレスな移行について，学習者の言語の選択率や利用時間の指標の開発と評価，(3) 抽象化技法の効果的な利用を測定するためのプログラム評価メトリクスの開発の 3 点である．

３．研究の方法

(1)．情報システム学に基づく要求駆動型カリキュラムの開発

著者が開発している初等中等向けプログラミング教育カリキュラムがある．このカリキュラムでは，コメント・関数化からオブジェクト指向まで，初期の段階から抽象化技法を中心に議論するものとなっている．些末な言語仕様やアルゴリズムを隠蔽し，ライブラリは極力扱わず，小さな要素から大きな構造を構築する技法を扱う．本研究では，このカリキュラムをベースとして，プロセスの抽象化技法を扱う初級編，データの抽象化技法（オブジェクト指向）を扱う中級編の 2 編のカリキュラムを開発する．

(2)．抽象化を促進するビジュアルプログラミングツールの開発

現在，本教育カリキュラムを支える開発環境を開発中である．現在のプロトタイプでは，米国 MIT で開発された OpenBlocks を利用して開発しているブロック方式のビジュアルプログラミング言語がある．本研究開始後の主たる拡張開発は，プロセス抽象化機能の拡張（引数への対応），およびデータ抽象化（オブジェクト指向）の記述機能である．動的なオブジェクトの可視化についても検討している．

(3)．プロセス測定とデータ検証による教育効果検証

情報システム学（あるいは CT）に基づくプログラミング教育について，学習者の学習効果をテストで測定する方法は提案されていない．したがって，成果物と過程の分析による学習効果測定方法論を整備する必要がある．本研究における教育効果検証には，筆者ら開発しているプログラミング過程のデータ記録・分析ツールを利用する．アルゴリズム開発への集中度指標への展開が期待される．本研究ではこのようなデータ分析指標の研究を積み重ねることで抽象化技法の効果的な利用を測定するためのメトリクスの開発を行う．

４．研究成果

本章では，３つの主要な研究成果を抜粋し，それらの成果の概要を報告する．

（１）ビジュアル-Java 相互変換によるシームレスな言語移行を指向したプログラミング学習環境の提案と評価

本研究ではプログラミング初学者を対象としたプログラミング教育支援システム「BlockEditor」（図１）を提案する．BlockEditor はブロック型言語からテキスト記述型言語へのシームレスな移行を支援する．BlockEditor の支援対象として想定している学習者は，初めてプログラミングを学習する者（初学者）である．

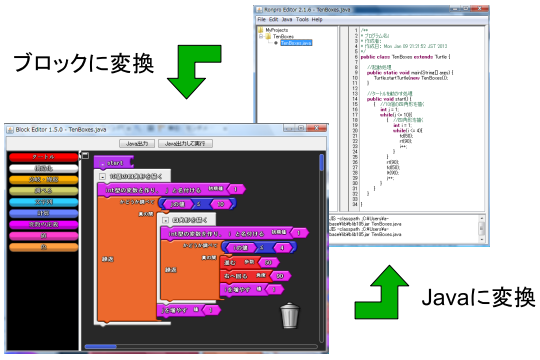

図１．ブロックエディタ

本システムは大まかに以下の 3 つの機能を持つ．

［機能 1］ ブロック型言語によるプログラム記述機能：ユーザがブロック型言語を用いてプログラムを作れる．

［機能 2］ブロック型言語と Java の相互変換機能：ユーザが任意のタイミングで，Java プログラムを用いてプログラムを作ることができる（図１）．

［機能 3］ 抽象化ブロック機能：ユーザがブロック型言語を利用して部分プログラムをまとめることができる．

実験目的は，提案システムの利用によってブロック型言語から Java へシームレスな移行が可能になるかを評価することである.「実際の教育現場で，ブロック言語，Java を学生が任意に選択できるようにしたら，学生はどのように言語を選択するのか」というのが本研究の Research Question で，その仮説は以下の 2 つである.

[仮説 1]最初はブロック言語を選択し，理解進行に伴って，交ぜ書き状態を経て，最終的に Java を選択するようになる.

[仮説 2]移行のタイミングには個人差がある.苦手意識を持つ学生ほど，ブロック言語を選択する.

エディタに付属の操作記録保存機能を用いてプログラミングの過程の記録を行い，2 つの言語の使用時間を計算した.

各学生の移行の状況，および個人差の分析を目的として，学生毎の BlockEditor 利用状況について図示を試みた．その結果を図 2 に示す.図 2 においては，x 軸の要素が各課題を表現し，y 軸は学生を表現している.各格子要素の濃淡はその区画（課題×学生）の BlockEditor 利用率を表現する.濃色が利用率が高く，淡色は利用率が低いことを表現する.つまり，横一列の帯を左から右へ順に追っていくと学生一人の利用率の推移が分かるようになっている．y 軸はカリキュラムを通して利用率が高い学生順に整列してある.

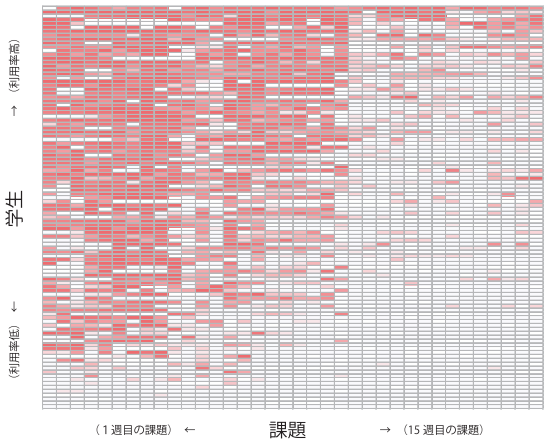

図 2．ブロックエディタ利用率推移

利用率が低い 10%程度の学生は，最初から BlockEditor を全く利用していない.逆に最後まで BlockEditor を主として利用している学生が 10%程度いる．残りの 80%は，全体的に緩やかに濃から淡へ模様が遷移している．平均の推移グラフと同様に中間課題からメソッドの学習についての急激な利用率の低下が観察されるが，その後も全く利用されていないわけではなく，部分的に利用され，その数が徐々に低下している．個人単位で観察しても利用率は徐々に低下し，クラス単位での BlockEditor 利用の人数比も徐々に低下し，その時期の個人差が大きいことも観察できる.

（2）オブジェクト指向言語におけるポリモーフィズムの概念を理解するためのワークベンチ

本研究では，ポリモーフィズムの概念を学習することを目的とした学習支援システム「enPoly」を開発した．enPoly とは「en-」という「~にする」という意味を持った接頭語と，Polymorphism の「Poly」を繋げることで，ポリモーフィズムがわかる人にする，という意味がある.

enPoly は，Anchor Garden のソースコードの拡張開発を行うことで実現した．Anchor Garden が提供するワークベンチ方式のインターフェイスに，ポリモーフィズム概念の学習に必要な機能を追加した．主な追加機能は，

（1）動的振る舞いを表現するアニメーション機能

（2）インターフェイスの概念理解支援機能

である． 追加機能を図示し，図 3 に示す.

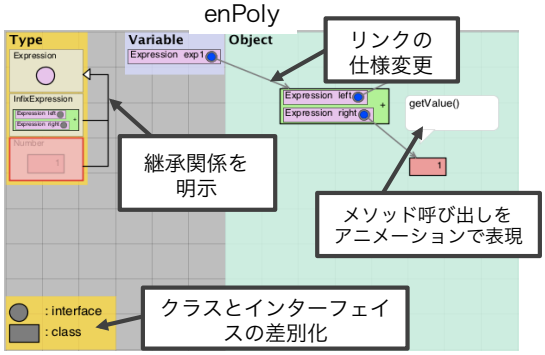

図 3．enPoly の追加機能

enPoly の評価のために実施する課題は，ポリモーフィズムを用いないと解くことができない Java のプログラム作成課題であった.

実証実験の結果，被験者のレベル上昇度（事前レベルと事後レベルの差分）では，実験群の全ての被験者が統制群を上回っていた．解答プロセスを質的に分析した結果，筆者らが想定したプロセスにより，enPoly が利用された上で解答が行われていることがわかった.

特に，解答プロセス分析において，筆者らが最も注目した点は，学習者が指示されていないのにもかかわらず，自主的にアニメーション機能を持ちいて，何度も同じメソッド呼び出しを行い，動的振る舞いの観察を行っていたことであった．筆者らは，この行動がポリモーフィズムの概念理解につながり，正答率の上昇という結果を導いたと考えている.この結論は，「アニメーションを観察することで，振る舞いの違いを認識することができた」という被験者のコメントでも補強できる.

解答プロセス分析では，enPoly のもう一つの特徴であるインターフェイス理解支援の有用性も示された．インターフェイスの概念が理解できていない被験者は，インターフェイス型の変数に，そのサブクラスのインスタンスを代入できることがわからない．しかしながら，そのような被験者が enPoly を用いると，

アンカータブの色で代入できるかできないかを判断することができ，吟味の過程が支援されて，その結果としてインターフェイスの概念理解につながり，これも正答率の上昇に寄与したと考えられる．その理解が，その問題領域にとどまるか，インターフェイスという概念に概念化されたかは測定できていないが，概念化は具体的な問題領域の理解の経験の積み重ねによって行われることが期待できるため，本研究の結果で目標を達成したと評価している．

（3）Compile Error Collection Viewer：修正履歴分析によるコンパイルエラー学習支援システム

本研究では，学習者が個人のコンパイルエラー修正履歴を観察，分析できるシステム CocoViewer(Compile error Collection Viewer)を開発した．本システムは，プログラミングの授業でコンピュータに蓄積された開発環境の操作履歴を利用してコンパイルエラーの修正時間を計算し，その推移グラフを自動生成し提示する．学習者はエラーの種類毎に集計された推移グラフの一覧や，各個のエラーが生じた状況の詳細な分析が可能である．これまで学習者各々が漠然と抱いていたコンパイルエラー修正の学習状況が定量的に把握されることで，学習の動機付け，プロセスの改善促進，およびエラーに対する恐怖感の軽減をすることが本研究の目的である．

本システムを起動すると，CocoViewer メインウィンドウが表示される．（図４）．CocoViewer メインウィンドウでは修正時間推移グラフの一覧表である修正時間推移グラフ一覧表と，全体のコンパイルエラー修正に関する情報が得られる．この画面に表示されているグラフをクリックすると，そのコンパイルエラーの種類について詳細な修正情報が表示される（図５）．

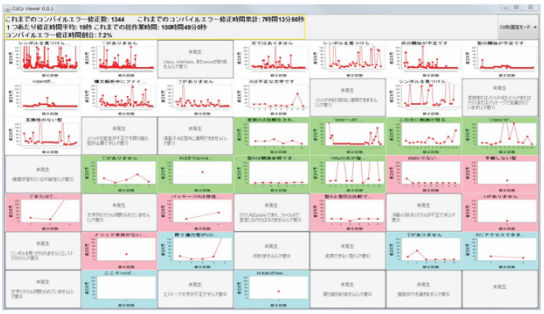

図４．CocoViewer メインウインドウ

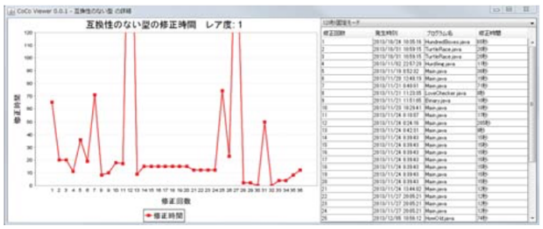

図５．修正詳細ウインドウ

文科科系の大学生約 100 名に対し講義時間内に評価実験を行った．実験は 12 週目の講義時に行われた．まず筆者が修正時間推移グラフの読み取り方，実験の概要について説明した．被験者は講義時間内にワークシートにそって，自分自身の１２週分のコンパイルエラー状況が示された CocoViewer を操作し，自身のコンパイルエラー修正記録を振り返ってアンケートに回答した，

主要な結果の一部として，受講者が理解した，コンパイルエラー修正に関して感じていることと実際の差異を示したデータを図６に示す．経験したコンパイルエラーの数，種類の数，そして修正時間が，システムを使って実際のデータを観察するまでに想像していたよりも長いか，短いかを聞き，回答を得た結果である．

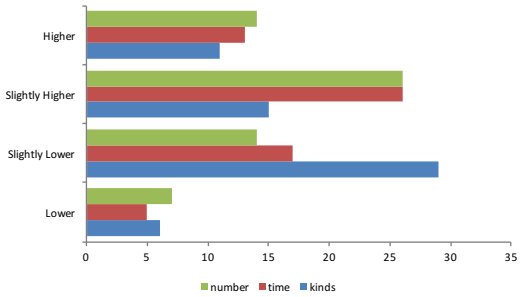

図６．コンパイルエラー修正記録について，想像していた程度と実際の程度の差異はどの程度でしたか？

被験者のうち，65%についてコンパイルエラー発生数が想像よりも多く，64%についてコンパイルエラー修正時間が想像よりも長く，57%について修正したコンパイルエラーの種類が想像よりも少ないと答えた．

本システムを使うことでコンパイルエラーへの恐怖感が軽減されたかを問う質問の回答結果では，被験者の 44%がコンパイルエラーへの恐怖感が軽減されたと回答した．自由記述欄の回答においては，CocoViewer を操作することによりコンパイルエラーに対する恐怖感が軽減されたという意見が得られた．

コンパイルエラーに対する恐怖感が軽減された要因としては，学習者自身がコンパイルエラーに対して抱いているイメージと実際の修正状況の差異が埋まり，学習者が想像している以上にコンパイルエラー修正できていると自覚できたことと，他の人と本システムを比較することによって自らのコンパイルエラー修正を相対的に考えられ，コンパイルエラー修正に対するイメージと修正状況の差異が埋められたことが挙げられる．

５．主な発表論文等

〔雑誌論文〕（計　４　件）

(1) Yoshiaki Matsuzawa, Motoki Hirao, and Sanshiro Sakai: ″Compile Error Collection

Viewer: Visualisation of Compile Error Correction History for Self-assessment in Programming Education", International Journal of Engineering Education, Vol. 32, No. 3(A), pp.1117-1127, 2016.査読付
(2) 野沢光太郎, 松澤芳昭, 酒井三四郎:"一貫性・明瞭性診断による静的 UML モデリング学習支援システムの設計と評価", 情報処理学会論文誌, Vol.55, No.5, pp.1471-1484, 2014.査読付
(3) 石川裕季子, 松澤芳昭, 酒井三四郎:"オブジェクト指向言語におけるポリモーフィズムの概念を理解するためのワークベンチの試作", 教育システム情報学会誌, Vol.31, No.2, pp.208-213, 2014.査読付
(4) 松澤芳昭, 保井元, 杉浦学, 酒井三四郎:"ビジュアル-Java 相互変換によるシームレスな言語移行を指向したプログラミング学習環境の提案と評価", 情報処理学会論文誌, No.55, No.1, pp.57-71, 2014.査読付

〔学会発表〕(計 13 件)

(1) 加藤優哉, 松澤芳昭, 酒井三四郎:"初学者の協調プログラミングにおけるインタラクションの分析", 情報教育シンポジウム(SSS2015), pp.177-184, 2015, 鳥取
(2) 田中良樹, 松澤芳昭, 酒井三四郎:"ビジュアルプログラミング対応版プロセス観察システムの開発と学習効果分析", 情報教育シンポジウム(SSS2015), pp.125-132, 2015, 鳥取
(3) 松澤芳昭, 坂本一憲, 大畑貴史, 筧捷彦:"プログラミング教育のための多言語間プログラミング言語翻訳システム", 情報教育シンポジウム(SSS2015), pp.223-230, 2015, 鳥取
(4) Yuya Kato, Yoshiaki Matsuzawa, Sanshiro Sakai:"Promoting collaborative programing for introductory programing courses through an individual work branch and real-time sharing approach", IFIP 2015, (presented at Vilinus, Lithuania, July 3rd, 2015)
(5) Motoki Hirao, Yoshiaki Matsuzawa, Sanshiro Sakai:"Compile Error Collection Viewer: Visualization of Learning Curve for Compile Error Correction", IFIP 2015, (presented at Vilinus, Lithuania, July 1st, 2015)
(6) Takashi Ohata, Yoshiaki Matsuzawa, Sanshiro Sakai:"MeRV: A Scaffold to Promote Creating 2D Map of Method Call Structure in Block-based Programming Language", IFIP 2015, (presented at Vilinus, Lithuania, July 1st, 2015)
(7) Yoshiaki Matsuzawa, Takashi Ohata, Manabu Sugiura, Sanshiro Sakai:"Language Migration in non-CS Introductory Programming through Mutual Language Translation Environment", SIGCSE 2015, pp.185-190, 2015. (presented at Kansas City, USA, March, 5, 2015)
(8) 平尾元紀, 松澤芳昭, 酒井三四郎:"Compile Error Collection Viewer:修正履歴分析によるコンパイルエラー学習支援システム", 情報処理学会 情報教育シンポジウム(SSS)2014, pp.151-158, 2014.
(9) 加藤優哉, 松澤芳昭, 酒井三四郎:"独立同期モデルによる初学者向け協調プログラミング支援システムの開発", 情報処理学会 情報教育シンポジウム(SSS)2014, pp.27-34, 2014.
(10) 大畑貴史, 松澤芳昭, 桐山伸也, 酒井三四郎:"BlockEditor Hinoki:ビジュアル-Java 相互変換技術を利用したオブジェクト指向プログラミング教育の提案", 情報処理学会 情報教育シンポジウム(SSS)2014, pp.35-42, 2014.
(11) Yoshiaki Matsuzawa, Yukiko Ishikawa, Sanshiro Sakai: "enPoly: Workbench for Understanding Polymorphism in Strong Typed Object-Oriented Language", International Conference on Computers in Education (ICCE) 2013, pp.319-328, 2013. (presented at Bali, Indonesia, November 20, 2013)
(12) Yoshiaki Matsuzawa, Kotaro Nozawa, Yasuhiro Noguchi, Sanshiro Sakai: "Multiplicity Doctor: A Diagnosis Tool for Clarity and Consistency of the Multiplicity in a Static UML Model", 10th IFIP World Conference on Computers in Education (WCCE 2013), pp.54-64, 2013. (presented at Torun, Poland, July 4, 2013)
(13) Yoshiaki Matsuzawa, Ken Okada, Sanshiro Sakai.: "Programming Process Visualizer: A Proposal of the Tool for Students to Observe Their Programming Process", Innovation and Technology in Computer Science Education (ITiCSE '13), pp.46-51, 2013. (presented at Canterbury, UK, July 1, 2013)

〔その他〕
ホームページ等
http://www.matsuzawalab.info

６．研究組織
(1)研究代表者
松澤 芳昭 (MATSUZAWA, Yoshiaki)
青山学院大学・社会情報学部・助教(2015 年度-現在)
静岡大学・情報学部・講師 (2014 年度)
静岡大学・情報学部・助教 (2013 年度)
研究者番号：40517017