母語によるプログラミング教育

大岩 元[†1]　　中鉢欣秀[†2]

小学生からプログラミング教育を全ての市民に行なうことが先進国で始まっている．この教育の目的は，プログラミング言語の使い方を覚えることではなく，小さな命令を組合せて大きな仕事ができることを体験することにある．それには，学習者が自分の考えたプログラムの意味が自分で理解できる，母語である日本語を用いたプログラミングが有効である．この教育経験に基づいて，文法を可能な限り簡単にした日本語プログラミング言語「敷島」を構想した．当面の目的は小学生の国語と算数の授業を支援できることである。

Programming Education by Mother Language

Hajime OHIWA[†1]　　Yoshihide CHUBACHI[†2]

Programming education for all kids has been started in developed countries, including Japan. This is not for learning programming language, but for fostering the capability of doing works by properly combining small operations. For this purpose, It was found that writing a program in mother language was effective, because written program can be read by the learner and its meaning can be understood by the learner without mastering new language. We are designing a programming language in Japanese, namely, "Shikishima" for supporting elementary school subjects of arithmetic and Japanese.

１．はじめに

これまで日本では、プログラミングは専門家が行なうもので，一般人が学ぶ必要はないと考えられてきた．この考えは，パソコンソフトが提供されるようになり，Basic でプログラムを書かなくてもパソコンが使えるようになってから一般化した．この点は欧米でも同じであったが，大学進学者については，プログラミングを中心とする情報教育が行なわれてきた．このことは，UNESCO が主として途上国に対して行った情報教育に関する提言として記載されている［１］.

英国や米国では，大学で情報専門学科に進学する学生に対しては，日本の大学の専門学科で教えられる内容乃至，それ以上の高度な内容が，中等教育段階で教えられる仕組がすでに１９９０年代から確立している．例えば，米国では大学の初年次の授業を高校で受講して，その単位が大学入学後に認められる Advance Program の制度が広く行なわれているが，プログラミングに関しては，オブジェクト指向のプログラミングではなく，オブジェクト指向概念が教えられている．英国式の教育を行っているオーストラリアでは，専門学科に進学する学生は，プログラム設計に関しては構造化設計を中等教育で習得済として，オブジェクト指向設計から教育が始まるそうだ．専門学科でも Java の記述法しか教えられていない日本の大学やＩＴ企業の大部分の教育内容とは大きな違いがある．

しかし，それにもかかわらずアマゾン，グーグル，フェイスブックなどの先端企業では，高度なＩＴ技術者の獲得が社運を決める状況にあり，ＩＴ技術者の能力が国の経済を左右する状況が続いている．また，高度に発達したソフトウェア製品を使いこなすには，利用者側にも論理思考が要求され，その育成が問題となる．最近の OECD による生徒や成人に対する学力評価（PISA と PIAAC）が求める問題解決能力は，こうした状況を反映したものであろう．

こうした状況から，欧米諸国ではでは国の方針として，小学生からプログラミング教育を行なうことが決定された[2]．これを受けて，日本でも，２０１４年５月の閣議決定で初等・中等教育でプログラミング教育を行なうことが明示された．ここで行なわれるプログラミング教育は、UNESCO が提案しているプログラミング教育[1]の入門段階で示されているように,自分がコンピュータに行なわせたい仕事をプログラムとして実現することを目的とするものであって，プログラミング言語の書き方を教えるものではない.

このような目的のために我々が開発するプログラミング言語「敷島」は,一般人がそれを学べば容易にプログラムが作れるようになることを目指す.このため．文法を可能な限り簡単化することを目的とする。また、書かれたプログラムが日本人の母語である日本語として読めることを最重要開発課題とする．具体的には，教育用言語として成功し、実用言語としても使われるようになった Pascal のような言語，を完全な日本語として読めるようにすることが開発目標である.

以下母語によるプログラミングが入門教育で有効であること，言語技術教育でないプログラミング教育がどのようなものであるべきか，そこでは日本語プログラミングが有効であり、それが国際化にも逆行しないこと説明する．そ

†1 慶應義塾大学
Keio University
†2 産業技術大学院大学
Advanced Institute of Industrial Technology

の後，学習者の学習負荷が最小になることを目指した日本語プログラミング言語「敷島」の設計思想を示し、その最初の実現として、小学校における国語と算数の教育への応用可能性について考察する。

２．論理思考は母語で行なう

慶応大学湘南藤沢キャンパス（ＳＦＣ）では，１９９０年の開学以来，外国語教育，保健体育とともに３つだけの必習科目の１つとして情報科目が８単位，全学生に義務づけられてきた．情報科目の内容はプログラミングを中心とするものであるが，全員が自分の目的を実現できるプログラミング能力を獲得出来たわけではなかった．

そうした状況の中で,「論理思考トプログラミング」の授業ではプログラミング言語を日本語にすることで，クラス全員が挿入ソートを、フローチャート作成による設計から始めて作れるようになったことが杉浦らによって報告されている［3]．彼らの授業では，週９０分２コマの授業を１３週にわたって行っていたが，その前半に「ことだま on Squeak」[4]による日本語プログラミング教育を行ったことによって，目的とするプログラムを作れたものと推測される. 授業の後半は,「ことだま」で教えられた内容を Java で繰り返すことによって，実用言語においても，前半の授業で獲得された論理構築能力が,生かされたことが報告されている．受講学生による評価を次に示す.

プログラミングには（中略）論理に基づき筋道だったプログラミングをする以外の方法はない。その意味でプログラミング自体に関する知識や表現の使用を多く求めない「ことだま on Squeak」は、思考訓練の面では大いに有効であったと思うのである。(中略) プログラミングに関する文法や表現を詳しく知らない段階では、「ことだま on Squeak」を使うことで、すでに用意された表現を使用しながら、より本質的な論理思考の訓練に専念できたと感じている。(中略) 読解が出来ない状態で作文など無理であるように、表現や文法を知らない段階で Java によるプログラミングは大変なハードワークであろう。私は初期段階において「ことだま on Squeak」を通して「順次」「分岐」「繰り返し」「変数」などの基礎的な思考方法を重点的に学ぶことができた。そこで得た思考能力が、他者が作成したプログラムを読解し理解する際に活用されたと思う。

（総合政策学部２年男子）

従来ＳＦＣでは，授業全体を実用言語の Java（やC）で行ってきていたのであるが，９０年代後半以後は，いわゆる学力低下のせいか，自分の目的を実現できるプログラミング能力を獲得する学生が減少して，プログラミングが習得出来るのは入学前からその能力を持っている学生だけに，ほぼなってしまっていたようである．

実用言語を使って初心者にプログラミングを教えると，

（１）「新しい言語を理解すること」
（２）「それを使って仕組を作ること」

という２つの初めての作業を同時に行なわなければならない．殆どの受講者は，新しく学ぶ言語の文法通り，プログラムを書く段階で挫折してしまう．最近の日本の教育は正しいことを覚える教育に終始して，試行錯誤して何かを作り出すような教育が行なわれなくなってきたため，文法エラーが頻繁に起こることだけで挫折してしまい，授業は文法通り書くことの訓練で終ってしまう場合が多い．

実用言語は実用目的を達成することを目的とした，専門家を対象として言語であるので，実用プログラム作成を効率的に行なう工夫が多く成されている．このため，初心者にとって意味の理解できない作業を多く強いられることになってしまう．

もう一つの問題は，実用言語を将来使う可能性である．一般教育の受講者で，将来プログラマーになる人は少数であろう．今後はさらに，高度な技術者だけが生き残る時代になってきたために，プログラマー自体の人数が減っていくことが予想される．従って，現在の実用言語を学習者が使う可能性は少ない[5].

一方で，論理思考が必要とされる人数はこれから増大していくことが予想される．情報技術の応用分野がますます広がるからである．コンピュータの導入で，社会生活で使う情報システムが複雑な作業を行なうようになり，使い方を丸暗記するのでは対応できなくなって，論理的に考えて使う必要性が高まることが予想される．こうした論理思考能力を育成する手段として，プログラムを作ってみる体験は大変有効である．

論理思考教育を行なおうとすると，プログラミング活動の中で，アルゴリズム構築が一番役に立つ部分である．従来のプログラミング授業では，ここに到達する前に，文法理解で終ってしまっていたために，社会に出て役に立つ教育が行なわれてこなかったのである．

アルゴリズム構築の教育においては，従来は疑似コードが使われ，それは母語としての日本語が用いられてきた．疑似コードは実行できないので，それを実用言語に翻訳しなければならない．すると，文法エラーを退治できたとしても，続いて実行時エラーを退治する必要がある．それが終ると，予期したようには動作するとは限らない．これを行なおうとすると，慣れない実用言語を正確に解読しなければならない．これが初心者には難しいため，あてずっぽうでプログラムをいじり出して，試行錯誤で動かそうとする．これでは論理思考の教育にはならない．

疑似コードとして日本語が使われてきたのは，教育用に

限らない．大手金融機関では日本語の疑似コードを１対１で COBOL 文に対応するように作ってきた．そして，この対応を維持したまま，巨大な基幹システムを維持してきたのである．疑似コードをコボル文に翻訳する人間を「プログラマー」と呼ぶのが，業界の慣習であるらしい．単なる「コーダー」を「プログラマー」と呼ぶことから，「プログラマー」の社会的な地位が日本では低いことがうなずける．

疑似コードの日本語プログラムがそのまま実行できれば，これを実用言語に翻訳する必要がなくなる．実際，「言霊」はそのような目的で開発されたプログラミング言語である[6]．それが科学教育用にアラン・ケイによって開発された Squeak 上に搭載されたことで，テキスト入力が不要となり，タイルを貼ってプログラムを作ることになったため，文法エラーが起こらなくなった．これによって，学習者はアルゴリズムの構築体験に集中できることになったのである．

もう一つ，杉浦らの授業が成功してクラス全員が挿入ソートのプログラムが作れたのは，最初に選択ソートのアルゴリズムを，手作業で体験したことが大きく影響している．この体験をフローチャートで表現することを学んだ後，それを「ことだま」のプログラムとして組み立てて，実行させた．その後で，挿入ソートの実行過程を見せて，そのアルゴリズムを理解させた後，それをフローチャートに表現させ，「ことだま」のプログラムとして完成させる，というのが授業の流れであった．

タイルを貼る作業でプログラムを作って文法エラーから解放されても，意図通りには動くとは限らない．自分が書いたプログラムを読んで，そこに書かれたアルゴリズムがなぜ意図通りに動かないかを考えなければならない．この部分が論理思考教育として重要な部分である．ここで，日本語プログラミングが威力を発揮する．書かれたプログラムの意味が日本語で記述されていれば，それを読み解くことでなぜ意図通り動作しないかが理解できる．正確に日本語文を読む能力が要求されるのである．

従来の国語教育では，このように論理的な日本語を読み解く訓練はほとんど行なわれていない．文を読むことに関して，コンピュータに較べてはるかにインテリジェントな人間が読むことしか想定していないから，国語の授業ではプログラミングで要求される正確な読解は想定されていないのである．かつては数学の教育で，このような正確な日本語を読み書きする訓練が行なわれてきたが，近年はそのような手間のかかる教育があまり行なわれなくなってきた．公理系から始まる幾何の証明が，数学教育から消えてしまった影響が大きい．

３．UNESCO が勧める情報教育

2002 年に発表されたユネスコの勧める情報教育[1]では，情報教育を次の４つの段階に分けて論じている．

１．ICT Literacy
２．Application of ICT in Subject Areas
３．Infusing ICT across the Curriculum
４．ICT Specialization

１．は，日本でも広く行なわれている，「情報手段」（指導要領の用語）の活用である．２．の，各科目への「情報手段」の活用は，今始まろうとしている．しかし，外国で行なわれているような，周到な教師教育と，教育用のシステム開発が行なわれていないために，導入しても負の側面が目立って，こうした教育への現場教員の支持が日本では広く得られていない．高校の校長会からは，「普通教科 情報」は必履修から選択科目にして欲しい」という要望が出る状況である．欧米では現在進行中の，３．の「カリキュラムの中に溶け込む」ような状況にはない．

特に問題なのが，４．ICT Specialization である．UNESCO が推奨している教育内容が，情報技術の専門家とともに，高等教育に進学する学生のための内容であるとしていることが，日本では全く認識されていない．実際，欧米の大学卒のビジネスマンはみな，プログラミングができるのが当然で，できなければ大学に進学しなかった人であると見なされるそうである．

ユネスコの提案では，ICT Specialization は日常生活の問題をアルゴリズムとして解く能力を身につけることを目的としている．そして，その最初の内容である Introduction to Programming では，課題を解くアルゴリズムを設計(Design)すること，その設計をプログラムとして実行可能な表現に変換すること，最後にプログラムを実生活で利用できるようにする所までを体験させるよう，推奨している．このことは，プログラミングの入門段階から，プログラムが使用される環境の中で，何を実現するかという仕様を考え，それを実現するアルゴリズムを構築した上で，それをプログラムとして表現して，実際に使用して評価する所まで体験させること意味する．このように，プログラミング全体を体験させるような教育は，大学の専門教育でも，企業の技術者教育でも，殆ど行なわれていない．プログラム作成の作業手順を教えるだけで，アルゴリズムを考える教育が軽視されている．

４．プログラミングにおける日本語の有効性

日本語の語順は目的語の後に動詞が来るため，複雑な処理を記述する場合に，記述者は目的語のスタックだけを脳内に用意しておけば自分が実行したい動作をイメージできる，逆の語順の英語の場合は，動詞のスタックも必要となり，脳の負荷が大きくなる．このことから，米国人が APL, FORTH, Post Script など，日本語の語順の言語を作ってきたが，母語の英語と語順が異なるため普及しなかった．日本語はコンピュータとのインターフェースがすぐれた言語

なのである．

FORTH を輸入販売していた片桐 明が, FORTH を日本語化するだけで日本語プログラミング言語になったことから, これを本格化的な日本語プログラミング言語 Mind として商品化した[7]. この言語は, まず教育用として使われた所, Logo に較べて格段に学習効果が上がることが確認された（西之園晴夫：私信, 1995）.

Mind は単に教育用言語であるだけでなく, ソフトウェア開発言語として発展し，例えば「ぐるなび」の全文検索エンジンは，Mind の発展形の MindSearchII を使って開発が行なわれ，そのシステムは２００４年５月から６年間使われ続けた[7].

Mind の特長として, 修得がＣ言語の約３分の１の時間で実務アプリケーションが組めるようになるという．また，可読性が高く，再利用がし易いので，開発期間が短縮できるようだ[7].

実は，数学も我々は英語の語順で記述されている．

$$g(x) = d/dx\ f(x)$$

という数式は，"The function g of x is equal to the derivatives of the function f of x." という英語の文を略記したものである.「x の関数 g は x の関数 f の導関数に等しい.」という日本語表現とは全く違う語順であることが分る．数学の記述を変えることは今では不可能であるが，プログラミングは，母語を元にしたものの方が使い易い．日本人のために，日本語の語順のプログラミング言語を作る必要がある．

５．日本語は非論理的で国際性がないか？

日本語でプログラムを作ることを提案すると，国際化時代に逆行すると言われる．しかし，この問題は，必要なら作ったプログラムを読者が読める言語に翻訳することにすれば解決できる．人工知能研究として機械翻訳が研究されたが，人間は膨大な常識を持って言語を使っているため，この扱いが難しくて実用化に至らなかった．しかし，形式的に意味が規定されているプログラミング言語の場合にはこの問題が起こらない．従って機械翻訳の研究成果を利用すれば，プログラミング言語間の翻訳は十分可能であることが予想される．実用言語で書かれたプログラムを自然言語に翻訳することは，プログラミング教育にとどまらず，使われる可能性がある．ソフトウェア開発の検収に日本語への翻訳が必要であったことは, その可能性を示している．

「日本語は論理的でない」ということから，プログラムのような論理的な事象の記述には適さないと考える人も多い．しかし，これも思いこみに過ぎない．どんな言語を使っても，使用者が論理的でなければ，表現された文章は論理的でない．高等教育に至るまで日本語だけで国を維持している日本語が論理性に欠けることは考えられない．科学ジャーナリストの松尾義之はノーベル賞講演を行なうまで外国に出たことのない増川俊英教授は日本語で物理学を研究してきたことを例にあげて、日本語が日本における科学研究の進歩に大きな影響を与えていることの主張している [8].

最近米国人のロジャー・パルパーズが，日本語は少数の語彙と単純な文法で豊かな表現ができることから，英語より世界語としての可能性が高いという指摘をしている [9]. 少数の語彙と単純な文法はプログラミング言語の特徴であり，日本語は，プログラミング言語と同じような特性を持っていることになる．

６．算譜言語「敷島」の設計方針

パルパーズの日本語に関する指摘を受けて，日本語プログラミング言語も，文法を可能な限り単純化することを試みることにした．プログラミング言語として必要最小限なものは，処理対象とそれに対する処理を記述する機能である．また，用語はできる限り日本語，特に大和ことばを使うことにする．かつてプログラミング用語の日本語化が検討されたが, 現在使われているのは,「処理系」位であろう．この復活を試みる. 手はじめに,「プログラム」には「算譜」を使うことにする.

「算譜」を記述する日本語には，新たな文字として「カッコ」が必要になる．かつて，明治維新に際して，日本人は西欧の文書習慣に合わせて, 句読点と引用符を導入した．今回，文書がコンピュータによって解読されて，複雑な処理が行なわれることから，それに必要な記号として「カッコ」を導入することにする．何種類かの「カッコ」を用いることで,「算譜」はプログラムとして読める文章になるはずである．

処理対象のオブジェクトは「物」と呼ぶことにする.「物」はスタック上で処理をして, 結果をスタックトップに置く. スタックを「(積み上げ) 棚」と呼ぶことにしよう.「物」は名前で表わし,「物」を記憶装置から取り出して棚に置く操作と棚の上からの「物」を記憶装置にしまう操作で、全ての処理を行なう.「基本物」として数と文字を用意し，これを元に「複合物（レコード構造)」によって複雑な「物」を構成していく．また，配列として,「数列」と「文字列」を用意する.「複合物」の定義と合わせて，その「物」に対する操作を定義できるようにする.

言語名としては「敷島」を用いることとした．当面の開発目標は，小学校における算数と国語の授業支援である．また，この活動を通して，コンピュータが情報を処理する仕組を理解させたい．このための「しきしま１号」では，実数と整数の区別は設けず,「数」とすることにする．整数の除算の結果としては、商と剰余からなる「商・余り」という「物」を定義する．文字列に対しては，同一性を調べ

る演算，長さを調べる演算，繋げたり，分けたりする演算を設ける．

数式は数値を与える「物」として扱う．また，数式に名前をつけることもできるようにする．学校教材を作る過程でこれらの「物事」を精密化する．

命令文は，「{物（達)} を {操作する}」という形式に統一する．制御構造としては，「逐次構造」，「条件分岐構造」，「繰り返し構造」を用意する．また，「名前」，「物」と付随する「操作」を規定する宣言の機能を設ける．「物」を「操作する」メソッドは再帰呼び出しを可能にする．

日本語プログラミング言語開発の問題点は，BNF で講文定義をしてテキストで記述されたソースコードを講文解析すると，自然な日本語でプログラムを記述できるようにするのが難しい．そこで，具象文法としての「日本語表現」と，抽象文法としての「言語設計」を分離する．

まず，抽象文法を定義して抽象構文木を評価できるようにする．次に，具象文法としての日本語表現を策定して，抽象講文木を自然な日本語として読み下せるように工夫する．更に，プログラム入力のための構造エディタを用意して，入力作業の効率化をはかる．

抽象文法を分離することで，具象文法をいくらでも定義することができる．日本語プログラムの例として、先行する「言霊」[6]のプログラムを示す．

```
※プログラム名：貝殻を書くプログラム
※作成者：I. E.（相愛大学音楽マネジメント学科２回生）
※作成日：2012.7.3

※授業に必要なライブラリを取り込む
TurtleLibrary を参照する.

※環境を初期化する
ウィンドウを初期化する.
亀を新規作成して、亀太郎と名付ける
ウィンドウに亀太郎を追加する.

整数型を新規作成して,i と名付ける.※ループ用１
整数型を新規作成して,j と名付ける.※ループ用２

整数型を新規作成して,長さと名付ける.※１回に進む距離
整数型を新規作成して,角と名付ける.※１回に曲がる角度

長さに 10 を入れる.

※四角を１０個描くためのループ
i に 1 を入れる.
i≦10 である限り {
  ※四角を描くためのループ
  j に１を入れる.
  j≦4 である限り {
    亀太郎を長さドット進める.
    亀太郎を 90 度右に回す.
    j に j+1 を入れる.
  } を繰り返す.

    亀太郎を 30 度右に回す.
    長さに長さ+10 を入れる.
  i に i+1 を入れる.
} を繰り返す.
```

７．小学校の算数，国語と「しきしま１号」

日本の算数教育は世界的にみて成功しており，日本人の計算能力の高さは万人の認めるところである．しかし，このことは日本語自体が計算に適した構造を持っている所による所が大きいのではないかとも考えられる．

日本の算数教育の問題点の一つとして考えられるのは、計算技術は教えているが，その意味についてうまく教えられていないのではないかということである．例えば，高畑勲監督の「おもひでぽろぽろ」(DVD)には，分数の割り算で除数の分子と分母を入れ替えて掛ければよいということが納得いかない少女の話が重要な主題となっている．操作の意味を考えようとしないで計算手順だけを覚える子供が評価され，意味を考えようとする子供が疎んじられる日本の（学校の）状況は、情報化時代の大きな問題である．理科離れ、数学離れが問題となっているが、その原因の一つはこの点にあるのではないかと思われる．

「敷島」の最初の実現である小学校教育用言語「しきしま１号」では，整校，小数，分数の表現と計算法全体を実行できるようにする．これによって、小学校教員が扱い慣れた教育素材がコンピュータ上でどのように実現されるかを体験できることになる．分数は，整数である分子と分母から成る「複合物（レコード構造)」として定義し，これに対する操作（メソッド）として記述された加減乗除のアルゴリズムをじっくり読みとることで，分数計算の意味を具体的に理解できるのではなかろうか．また，１／３の小数表現と分数表現の差から，デジタル化が誤差を伴う事も容易に体験できる．

こうした体験の中から，実際の授業において活用できるものを授業時間の制限の中から選んでいけば，算数の授業の中でプログラミングを使って教育することができる．算数で教えられる内容を「しきしま１号」で記述することで，算数教育全体を変革できる可能性がある．

小学校の国語教育の目標は，

国語を適切に表現し正確に理解する能力を育成し，伝え合う力を高めるとともに，思考力や想像力及び言語感覚を養い，国語に対する関心を深め国語を尊重する態度を育てる。

である．

このような「国語」教育の目標は，情報システムの開発における言語活動そのものである．UNESCO の提案する入門教育の"Introduction to Programming" は、単なるプログラミング言語教育ではなく，その目標として

Student should be able to design, program, and evaluate simple algorithms for elementary task-oriented problems.

をあげている[1].「しきしま 1 号」はこの目標を算数教育と国語教育として行なえる環境を提供することを目指す．

８．おわりに

従来のプログラミング言語は，プログラマーが遭遇した困難を解決するための機能を提供する努力を続けてきた．しかし，今後はプログラミングの専門家でない人達が，自分のしたいことを解決するためにプログラムを書くことが，社会全体で行なわれるプログラミング作業の多数を占めることになるであろう．そこで必要になることは，それぞれのプログラムを書く人が知っている、プログラムの適用領域の概念であって，プログラミングの専門家が知らないことである．プログラムを書く一般人には，自分のしたいことが何であるかをできるだけ簡潔に記述できることが有難いのであって，便利だと専門外の人間が予想して作った仕組を，苦労して習得して使うことは望ましくない．プログラミングに求められるものが変わるはずであり，その方向性を追求するために，算譜言語「敷島」を開発する．

参考文献

[1] UNESCO, Information and Communication Technology in Education
A Curriculum for Schools and Programme of Teacher Development, p.120, 2002,
http://unesdoc.unesco.org/images/0012/001295/129538e.pdf.
[2]http://news.bbc.co.uk/2/hi/programmes/click_online/9707886.stm, ;
http://jp.wsj.com/articles/SB10001424052702304730304579436021415885380, ;
http://www.itmedia.co.jp/news/articles/1406/23/news049.html .
[3] 杉浦 学，松澤芳昭，岡田 健，大岩 元：アルゴリズ構築能力育成の導入教育：実作業による概念理解に基づくアルゴリズム構築体験とその効果，情報処理学会論文誌，Vol.49, No.10, pp.3409-3427, 2008.
[4] 大岩 元（監修），松澤芳昭・杉浦 学（編著）：ことだま on Squeak で学ぶ論理思考とプログラミング，イーテキスト研究所，2008, http://crew-lab.sfc.keio.ac.jp/lectures/2011s_ronpro/data/Squeak/Text/SqueakText.zip .
[5] ケビン・メイニー：Ｃ＋＋も JAVA も子供に教えなくてもいい，Newsweek 日本版，June 24, 2014.
[6] 岡田健　秋山優：日本語プログラミング言語「言霊」の開発　－全ての人がプログラミングする社会を目指して－，2009 年度未踏人材発掘・育成事業，http://www.ipa.go.jp/files/000007129.pdf
[7] 片桐　明：日本語プログラミング言語　Mind, http://www.scripts-lab.co.jp/mind/whatsmind.html .
[8] 松尾義之：日本語の科学が世界を変える，筑摩書房，2015.
[9] ロジャー・パルパーズ著早川敦子訳：驚くべき日本語，集英社インターナショナル，2014.

著者紹介

大岩 元（正会員）ohiwa[at]sfc.keio.ac.jp

1965 年東大理学部物理学科卒，1971 年東大大学院理学系研究科博士課程修了（理学博士），東大理学部物理学科助手，豊橋技術科学大学講師，同助教授，教授，慶應義塾大学教授，帝京平成大学教授，相愛大学教授を経て現在慶應義塾大学名誉教授．研究分野は情報教育学，ソフトウェア工学，認知工学．

中鉢欣秀（正会員）yc[at]aiit.ac.jp

2006 年 10 月 慶應義塾大学政策メディア研究科後期博士課程卒．2005 年 4 月独立行政法人科学技術振興機構 PD 級研究員（長岡技術科学大学）．2006 年 4 月産業技術大学院大学情報アキテクチャ専攻准教授．研究分野はアジャイル開発手法、要求工学、ソフトウェア・アーキテクチャ設計と実装、情報技術教育．