Webブラウザ上で動作する「ある」プログラミング学習環境の実装に関する考察

中西 渉[1,a)]

概要：多くのプログラミング学習環境が，Web ブラウザ上で動作する Web アプリケーションとして実装されている。筆者自身もある意味独自のプログラミング言語について，そのような環境の開発を手掛けてきた。しかし専門的な教育を受けてない者による一人プロジェクトであるため，とりあえず動くものは出来ているものの，それが良い実装になっているか，あるいは常識を踏まえた開発ができているかどうかについて，まったく判断ができない。そこで本稿では筆者が行なった実装の概要ならびに現時点で判明している問題点についてのまとめを行う。これを議論の俎上に乗せることができれば，今後同様の開発を行う者への反面教師としても役立つものと考えている。

キーワード：プログラミング教育，教材開発

1. はじめに

2003 年度から実施された学習指導要領 [18] により，普通教科「情報」が必履修となった。当時は科目が「情報 A」「情報 B」「情報 C」からの選択で，内容にプログラミングを含む「情報 B」が実施される学校は少なかった。2013 年度からの学習指導要領 [20] では共通教科「情報」は「社会と情報」「情報の科学」の 2 科目からの選択となったが，やはりプログラミングが含まれる「情報の科学」が履修されることは少なかった。2023 年度からの学習指導要領 [21] で必履修科目が「情報 I」に一本化されることによって，ようやくすべての高校生がプログラミングを含む内容を履修することになり，そのことは広く報道された通りである。高校の情報科に関わる教員の間では，プログラミング言語として何を用いるか，あるいはどのような環境で学習をするかといったことが話題になることも多い。

しかしそれとは別に，プログラミング教育についての研究は早くから行われている。情報処理学会では 2006 年度に「教育用プログラミング言語に関するワークショップ 2006」が実施され，十数種類の言語や処理系が取り上げられた [17]。筆者はこのワークショップで PEN[23] の存在を知り，翌年度から授業で使い始めた。

PEN で用いられた言語は大学入試センター試験の「情報関係基礎」という科目のプログラミング問題のために作られた疑似言語 DNCL[15] に，変数宣言などを追加した xDNCL である。なお，DNCL は大学入学共通テストが実施されるのに合わせて改訂されている [16]。さらに 2025 年度の大学入学共通テストで情報が導入されることに対応して大学入試センターが公開した試作問題・サン

[1] 名古屋高等学校
[a)] watayan@meigaku.ac.jp

```
a ← 2022 - 1964
もし a > 17 ならば
｜「おいおい」を表示する
を実行する
```

(a) 旧 DNCL

```
a = 2022 - 1964
もし a > 17 ならば：
└ 表示する("おいおい")
```

(b) 新 DNCL

図 1: 新旧 DNCL の比較

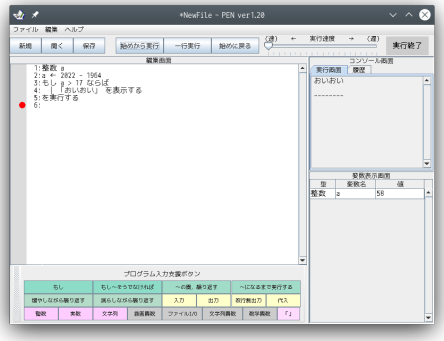

図 2: PEN の実行画面

プル問題では，Python のようにインデントで構文のブロックを指定する新しい DNCL が用いられている [14]。以下では区別のために「旧 DNCL」「新 DNCL」と呼ぶことにする[*1]。簡単なプログラムを新旧 DNCL で書くと**図 1** のようになる。

筆者はこれまでに DNCL の学習環境をいくつか開発してきた。しかし専門教育を受けていない者の一人プロジェクトであるため，とりあえず動くというレベルのものでしかない。本稿では開発の経緯や過程を明らかにすることによって，それにどのような問題があるかといった批評を仰ぎたい。そうすることで筆者のプロジェクトは改善できるし，同じような境遇で開発を行う者にとっては反面教師として有効なものになると考えている。

2. 開発に至る経緯

2.1 PenFlowchart の開発

前述したように，筆者は 2006 年度から PEN を用いて授業を行ってきた。PEN の実行画面を**図 2** に示す。それ以前は Yabasic[5] を用いていたが，

[*1] 新 DNCL を DNCL2 と呼ぶ人もいるが，大学入試センター関係者の発言では DNCL と呼称されているようである。

```
整数 a
a ← 2022 - 1964
もし a > 17 ならば
｜「おいおい」を表示する
を実行する
```

(a) 正しいプログラム

```
整数 a
a ← 2022 - 1964
もし a > 17 ならば
「おいおい」を表示する
```

(b) 誤ったプログラム

図 3: 多く見受けられた誤り

PEN に変更することにより，日本語をベースとした擬似言語であることや，画面下部の「入力支援ボタン」によって構文の雛形が入力されることなどにより，学習者の負担は小さくなったと考えている。

しかし，エラーを引き起こす生徒も少なくなかった。たとえば「入力支援ボタン」を使えば**図 3a** のように正しいプログラムが作れるのだが，わざわざ `ENDIF` に相当する「を実行する」を削除して**図 3b** のようにしてしまうのだ。聞いてみると「～を表示するを実行する」よりも「～を表示する」の方が日本語として自然だからということらしい。

筆者はフローチャートのようなものからプログラムが生成できればそのようなエラーは防げると考えて，PenFlowchart を開発した [10]。その実行画面を**図 4** に示す。これはフローチャートを編集することで，それに相当する PEN のプログラムを生成するものである。PEN のプログラムを編集した後「フローチャート」ボタンを押すことで，逆にプログラムからフローチャートを生成することもできる。

実は PenFlowchart は旧 DNCL 版だけでなく，BASIC や JavaScript，C++のコードを生成するものも作ったのだが，あまり使い物にはならなかった [11]。フローチャートからそれぞれのコードを生成することは簡単にできる。部品の連結に関する情報を構文規則に従ってテキスト化すればいいだけだからだ[*2]。しかし逆にコードからフローチャートを作るには言語ごとの構文解析が必要になる。あまり人が使わないであろう他言語版のためにそれを開発するのは重荷であるし，実行環境

[*2] おそらく Flowgorithm[4] もそのようにやっていると想像する。

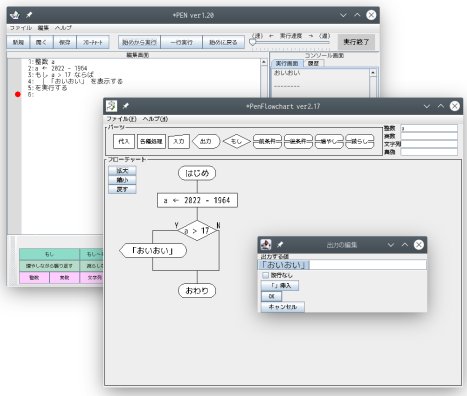

図 4: PenFlowchart の実行画面

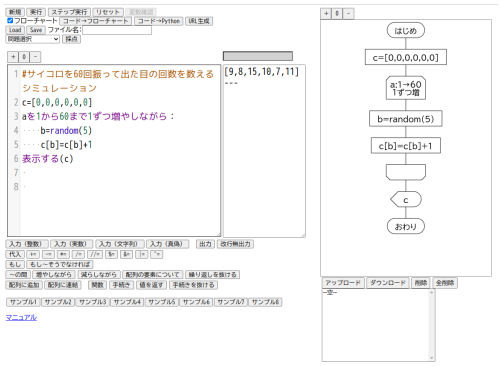

図 5: PyPEN の実行画面

につなげるのも大変だと考えたので，開発をそれ以上進める気にならなかったのである。

DNCL については PEN がプログラム実行のため javacc や jjtree で生成したパーサで構文木が作られていたので，それをフローチャートにするだけで済んだのであった。一応 BASIC と JavaScript の簡単な構文については，DNCL の構文定義ファイルを書き換えて作ったパーサを作りフローチャートの生成や（BASIC に限って）プログラムの実行まではできるようにしたが，それらの言語のすべての構文に対応することは難しい。C++については最初から諦めている。

2.2 WaPEN の開発

しばらくは PenFlowchart を使っていたが，PEN も PenFlowchart も Java アプリケーションであるから，（Java のランタイムも含めて）インストールが必要になる。学校によっては情報教室のパソコンにソフトウェアを自由にインストールできないこともあり，それが理由で使えないという話も聞くことがあった。

そこで Web アプリケーションでプログラミング学習環境が作れればいいと考え，WaPEN（Web-aided PEN）を開発した [12]。筆者にとっては，JavaScript である程度まとまったソフトウェアを開発したのはこれが初めてであった。

2.3 PyPEN の開発

2019 年の夏，友人から「WaPEN を Python っぽく改造したものを作ってみないか」と提案された。確かに Python は最近流行りでもあるし，文部科学省から出された教員研修用教材 [19] でもかなり Python 推しであるようにも思われた。そこで試しに作ってみたのが PyPEN[13] である。提案を受けてからある程度動くものができるまでおよそ 1 週間であった。PyPEN の実行画面を**図 5** に示す。

その後，大学入試センターが 2025 年度大学入学共通テストの試作問題やサンプル問題を公開したのだが，そのプログラミングの問題で使われた新 DNCL は旧 DNCL を Python っぽくしたものであった。誤解されることが多いので断っておくが，PyPEN は上述したように友人からの提案を受けて作ったのであって，新 DNCL がこのような言語であるということを知って作ったものではない。

3. 関連研究

DNCL の Web ブラウザ上の学習環境は多数作られている。たとえば Bit Arrow は Web ブラウザ上でドリトルを初めとする複数の言語が学習できる環境であるが，そこには「どんくり」[22] という DNCL の学習環境も含まれている。XTetra[8] は旧 DNCL の処理系であるが，同じ作者による新 DNCL の処理系「つちのこ」[7] も公開されている。

このような環境があるのだから，情報 I の学習は新 DNCL でいいのではないかと考えることもで

きそうだが，筆者はそれには否定的である。たしかにコードのブロックがインデントで表現されたり，インデントの前の行末にコロンがついていたり，Python でいうところの `print()` が新 DNCL では「表示する ()」という形式になっていたりして，新 DNCL が Python を意識して作られていることは明白である。しかしこれは決して Python そのものではない。ブラウザの外に何かできるわけでもないし，豊富なライブラリが使えるわけでもない。プログラミングのためのプログラミングという不毛なものになってしまう。

実際，大学入試センター試験問題調査官の水野氏は第 14 回全国高等学校情報教育研究会全国大会で「少し懸念していることは，DNCL だけでプログラミングの授業が行われることがないように，ということです。やはり授業では，教科書で使われている実用的なプログラム言語を使っていただきたい」と語っている [9]。

4. 開発過程

ここでは筆者が行った開発の過程と，そこで考えた問題点について述べる。

4.1 構文解析

PEN から PenFlowchart を作るときに，対象となるプログラミング言語のコードから構文を取り出す必要があることがわかった。PenFlowchart では若干の構文の拡張をした以外はそのまま PEN のしくみを使っている。

WaPEN を開発するにあたって，そのためのスキャナやパーサを自分で書こうとしたが，[6] を読み進めるうちに考えが変わっていった。

確かにそういった部分から自分で作れば，もっとも自由が効くものができるだろう。しかし筆者の本職は教育であり，ここで作ろうとしているのはそのためのツールなのであるから，可能な限り早く解決したい。そこでパーサジェネレータにあたるものがないかを探したところ，Bison の JavaScript 版というべき Jison[2] があることを知り，使うことにした。使い方は Bison に酷似しており，実際これを使うために筆者が参考にした書籍は [3] であった。

4.2 PyPEN の構文解析

PyPEN を作るにあたって，インデントによるブロックの表現をどのように Jison で書けばいいのかがわからなかった。そこで邪道とは思いつつ，新 DNCL のコードを旧 DNCL（に近いもの）に変換するプリプロセッサ的な処理を書いて，その出力を Jison で作ったパーサに読ませている[*3]。それに対応して，エラー表示の行数をだますためのパッチを，Jison が生成したパーサのプログラムにあてるという，歪な対応もしている。

明らかに二度手間なことをしているのは確実なのだが，解決に至る方向がわかっていない。前小節の話を掘り返すことになるがパーサを自分で書くべきなのか，Jison をもっとうまく使えば解決できることなのか，教示がいただければ幸いである。

4.3 実行の仕方

構文の解析ができた後，それをどのように実行するのが正しいのだろうか。他の処理系では JavaScript のコードに変換して実行しているという話も聞く。そうすれば高速なのは間違いないと思われるが，たとえば無限ループに陥ったときに停止することは可能なのだろうか。WaPEN や PyPEN には非常停止のために「リセット」ボタンを用意しているが，普通に JavaScript の無限ループに入ってしまうとボタンを押すこともできなくなってしまう。生徒がそのような状況に陥った場合に，ブラウザのその画面を放棄しなくてはいけなくなる（その結果，作成中のプログラムが失われる）ことを危惧している。

かといって `setTimeout` を普通に使うと速度がかなり落ちてしまうので，`postMessage` を使用することである程度の速度を得ている [1]。

プロシージャのうち，値を返すものを「関数」，返さないものを「手続き」として区別しなくてはいけない状態になっていることも不満である。Jison に入力する文法定義の中で，式だとか文だとかいっ

[*3] とりあえず動くものが 1 週間でできたのはこういう理由である。

たことをきちんと整理できていないのが原因だとは考えているのだが，まだ整理できていない。

4.4 型

PEN で用いる言語が DNCL でなく xDNCL と言っている理由の一つとして，変数の宣言を必要とすることがある。本来の DNCL では変数宣言は不要であるが，PEN では初学者への意識付けのために敢えて変数に型を設けたとのことである。

PenFlowchart，WaPEN ではそれを踏襲して変数宣言を必須としたのだが，DNCL にも Python にも変数の宣言がないのだから，PyPEN ではその必然性は感じない。そこで変数は型を持たないものであって，宣言しなくても値が代入されたときに生成されるものとした。後に PyPEN のコードを WaPEN に移植したので，今は WaPEN も同様の振る舞いになっている。

実は旧 DNCL には長さを決めずに使える配列というものが存在している。任意の添字で代入ができ，また初期値を定めておけば代入されていない値を読み出すときにはその初期値が返されるしくみである。これを作ることは別に難しくはないのだが，Python のコードにどのように変換していいのかがわからないため，躊躇している。しかし旧 DNCL の WaPEN ではその心配がないから実装してもいいのかもしれない。実際，前述した「どんくり」にはそのような配列が実装されている。

4.5 そうでなくもし

筆者が実装した DNCL 系の環境はどれも「そうでなくもし」の構文がない。これは構文解析についてはそれほど面倒なことではないのだが，フローチャートの表示が面倒だということで実装していない。

実は PenFlowchart では「そうでなくもし」を含むコードを読み込むと，「そうでなく」の中でさらに一段ネストを増やしたコードに変換するが，「そうでなくもし」のコードを手入力することはできないという中途半端な対応をしている。

実装の足かせになっている理由がフローチャートだけなのだから，解決策は

(1) フローチャートを「そうでなくもし」に対応させる
(2) フローチャートを諦める

のどちらかになるであろう。既にユーザ定義関数・手続きがあるプログラムについてはフローチャートを諦めているので，そうするのが近道ではある。筆者の個人的な考えとしてはフローチャートを廃止する方向に進みたい，せめてフローチャートからのコード生成は禁止したいと考えているのだが，実際に使用されていることも知っているので難しいところである。

4.6 Internet Explorer への対応

Web 関係では何かと目の敵にされる Internet Explorer（以下 IE）であるが，WaPEN・PyPEN の開発で筆者もそれを実感することになった。たとえば `class` さえ IE にはないし，`Number` オブジェクトも使えない。ある時期までは Babel を使って対応していたのだが，それでは追いきれないものもあるためかなり歪なコードを書いているところもある。

マイクロソフト社から IE 廃止の方向性が出されたことを，諸手をあげて歓迎する。

5. おわりに

恥を晒すように，開発の過程について述べてきた。経験者の方から見れば，素人の開発の稚拙さに驚愕する面があるだろう。そこで今後同様の開発を行う者のために，どういうところが「悪い見本」であるかを指摘していただければ幸いである。

なお，本稿で述べたプログラムはすべて筆者のサイト `https://watayan.net` で公開している。

参考文献

[1] Baron, D.: setTimeout with a shorter delay, (online), available from ⟨`https://dbaron.org/log/20100309-faster-timeouts`⟩ (accessed 2022-11-30).

[2] Carter, Z.: Jison, (online), available from ⟨`https://github.com/zaach/jison`⟩ (accessed 2022-11-30).

[3] Charles Donnelly, R. M. S.: Bison 入門，アスキー (1999).

[4] Cook, D.: Flowgorithm, (online), available from ⟨http://www.flowgorithm.org/⟩ (accessed 2022-11-30).

[5] Ihm, M. O.: Yabasic, Yet another Basic for Unix and Windows, (online), available from ⟨http://www.yabasic.de⟩ (accessed 2022-11-30).

[6] randy: いまどきのプログラム言語の作り方，毎日コミュニケーションズ (2005).

[7] 大門　巧：プログラミング言語つちのこ 2.0,（オンライン），入手先 ⟨https://t-daimon.jp/tsuchinoko⟩（参照 2022-10-30）.

[8] 大門　巧，大西建輔，青山　浩：XTetra の開発と授業実践による評価，情報教育シンポジウム論文集，Vol. 2021, No. 36 (2021).

[9] 水野修治：大学入学共通テスト新科目「情報 I」〜サンプル問題とそのねらい〜，河合塾「キミのミライ発見」（オンライン），入手先 ⟨https://www.wakuwaku-catch.net/kouen210801/04/⟩（参照 2022-11-30）.

[10] 中西　渉：PenFlowchart の開発，情報処理学会研究報告　コンピュータと教育 (CE), Vol. 2012–CE–113, No. 13 (2012).

[11] 中西　渉：PenFlowchart の「他」言語化，日本情報科教育学会第 1 回研究報告書，No. 5, pp. 17–20 (2013).

[12] 中西　渉：WaPEN...DNCL の Web ブラウザ上の実行環境におけるフローチャートなどの実装，情報教育シンポジウム論文集，Vol. 2018, No. 31 (2018).

[13] 中西　渉：プログラミング学習環境 PyPEN の開発，日本情報科教育学会第 13 回研究報告書，No. 5 (2019).

[14] 大学入試センター：令和 7 年度以降の試験に向けた検討について，（オンライン），入手先 ⟨https://www.dnc.ac.jp/kyotsu/shiken_jouhou/r7ikou/⟩（参照 2022-10-30）.

[15] 大学入試センター：センター試験用手順記述標準言語 (DNCL) の説明，（オンライン），入手先 ⟨https://www.dnc.ac.jp/albums/abm00004841.pdf⟩（参照 2021-10-31）.

[16] 大学入試センター：共通テスト手順記述標準言語 (DNCL) の説明，（オンライン），入手先 ⟨https://www.dnc.ac.jp/albums/abm00040701.pdf⟩（参照 2021-10-31）.

[17] 並木美太郎，辰己丈夫，兼宗　進，長　慎也，久野　靖，中野由章，西田知博：「教育用プログラミング言語に関するワークショップ 2006」の報告，情報処理学会研究報告　コンピュータと教育 (CE)，Vol. 2006–CE–085, No. 74, pp. 17–24 (2006).

[18] 文部科学省：高等学校学習指導要領（平成 11 年 3 月），（オンライン），入手先 ⟨https://www.mext.go.jp/a_menu/shotou/cs/1320144.htm⟩（参照 2022-10-30）.

[19] 文部科学省：高等学校情報科「情報 I」教員研修用教材，（オンライン），入手先 ⟨https://www.mext.go.jp/a_menu/shotou/zyouhou/detail/mext_00017.html⟩（参照 2022-11-30）.

[20] 文部科学省：平成 20・21 年改訂学習指導要領，（オンライン），入手先 ⟨https://www.mext.go.jp/a_menu/shotou/new-cs/youryou/⟩（参照 2022-10-30）.

[21] 文部科学省：平成 29・30・31 年改訂学習指導要領，（オンライン），入手先 ⟨https://www.mext.go.jp/a_menu/shotou/new-cs/1384661.htm⟩（参照 2022-10-30）.

[22] 本多佑希，兼宗　進：ブラウザ上で動作する DNCL 学習環境「どんくり」の開発，情報処理学会研究報告　コンピュータと教育 (CE), Vol. 2018–CE–147, No. 10 (2018).

[23] 中村亮太，西田知博，松浦敏雄：プログラミング入門教育用学習環境 PEN，情報処理学会研究報告　コンピュータと教育 (CE), Vol. 2005–CE–081, No. 104, pp. 65–71 (2005).