オブジェクト指向習得のためのビジュアルプログラミング言語

久乗 翔大[1]　鷲見 亮太[1]　角田 雅照[1]

概要：本研究では，オブジェクト指向プログラミングの習得に最適化したビジュアルプログラミング言語の作成を試みた．システムにユーザの操作ログを収集する機能を追加すれば，ビジュアルプログラミング言語をユーザがどのように扱っているかを分析することができる．

1. はじめに

学校教育において，「教育の情報化の推進」というテーマでプログラミング教育の必修化が決まっている．小学校においては 2020 年から，中学校では 2021 年から，高校では 2022 年から全面実施となる新学習指導要領でプログラミングに関する内容が補充されることも決まっている．さらに小学校では Scratch というビジュアルプログラミング言語（以下 VPL）の開発環境を利用したプログラミングの授業が行われることも決定している[1]. VPL とはプログラムをテキストで記述するのではなく，視覚的なオブジェクトでプログラミングするプログラミング言語を指す.

そこで本研究では，プログラミング教育で利用されることを前提とした VPL を開発する. 主にターゲットとするユーザは，プログラミングを初めて学ぶ生徒やプログラミング経験の少ない教員であるが，プログラミング経験の豊富な生徒や教員も利用しやすい VPL とする. 対象ユーザにとって理解しやすく，ある程度規模の大きなソフトウェアの開発も可能な VPL とする.

VPL は既存のものも多数存在するが（[2]など），既存のものでオブジェクト指向を前提としたものは我々の知る限りほとんどない．現在ソフトウェア開発で主流である言語（例えば Java など）は多くの場合，オブジェクト指向に基づいている．従ってプログラミングを発展的に学習していくならば，オブジェクト指向の習得は避けて通ることができない．VPL にオブジェクト指向を導入することにより，学習者はプログラミング言語の文法の習得を避けつつ，オブジェクト指向の習得に集中することができる.

本研究で開発する VPL では, クラス間の委譲や継承関係について，視覚的なオブジェクトを利用してプログラミングできる機能を持つ. 本研究で扱う VPL では，プログラミング入門からクラスの委譲や継承を扱う部分までのプログラミングを対象とする.

また，視覚的なオブジェクトを用いてプログラミングした際に, 完成したプログラムのソースコードを Java で出力する機能を持つ. ソースコードの出力として Java を選んだ理由は，オブジェクト指向言語であり，かつ日本のソフトウェア開発で最も利用されている言語[3]であるためである．出力されたソースコードをユーザがブロックと比較することにより，Java 習得に参考となることが期待される.

さらに，このようなシステム上でユーザの操作ログを蓄積する機能を追加することにより，様々な利点が考えられる．これまで，ビジュアルプログラミング言語で作成されたプログラムを分析した研究はいくつか存在する. ただし，ビジュアルプログラミング言語をユーザがどのように扱うかを分析した研究はほとんどなく，そのような分析に操作ログが有用であると考えられる．例えば，初学者がオブジェクト指向プログラミングやビジュアルプログラミング言語でつまづきやすい点が明らかになることが期待される.

2. 作成したシステム

既存の（VPL そのものではなく）VPL 開発環境である Google Blockly でも 1 章で述べた機能を実現できる．ただし後述するような機能の実装は容易ではないため，本研究では Google Blockly で用意されている開発環境を用いずに, フルスクラッチで VPL を開発することとした.

VPL の開発にあたり，開発に使用するプログラミング言語として Java と Processing を利用した．Processing は Java ライクな言語ではあるが，グラフィックを容易に扱えるという特長がある．そこでブロックなどの視覚的表現を実現するために Processing を用いた．その他の実装については Processing と連携が行いやすい Java で行うこととした.

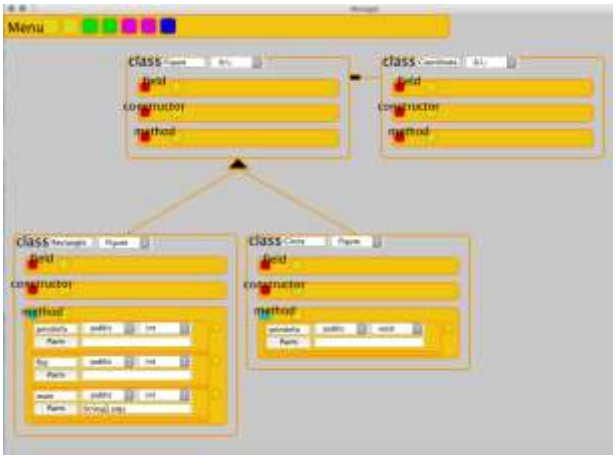
作成したシステムでは, メソッドレベルでは for, if-else, while, print, 変数初期化のブロックを用いることができる. これらの命令は順序，選択，反復と，画面出力を含んでおり，入門レベルのプログラミングには十分に対応可能である. これらを用いて作成したプログラムの例を図 1 に示す.

オブジェクト指向プログラミングを容易に行うために, クラスのテンプレートとなるブロックを作成した．このブロックにはフィールド，コンストラクタ，メソッドが記述でき，さらに，それぞれの内部に含まれるブロックを表示したり隠蔽したりすることができる．図 2(a)にそれぞれの要素を表示した場合と隠蔽した場合を示す.

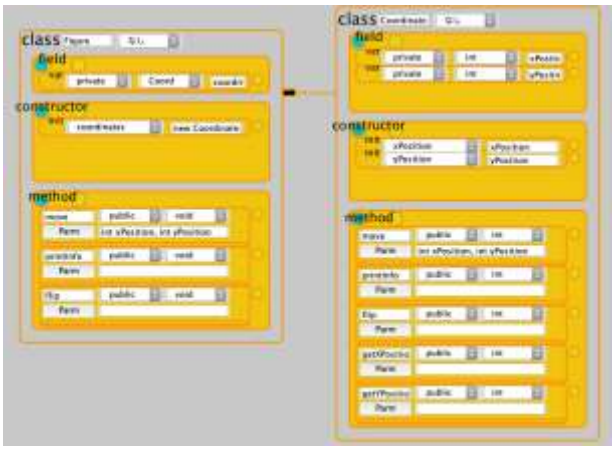
クラス自体は，継承，移譲を表現する機能を持っている. さらに，抽象クラスを定義することができる．図 2 に委譲と継承を表したブロックを示す．このように，作成したシステムでは, クラス間の関係を VPL 上で視覚的に表現することができる.

1 近畿大学
Kindai University

図 1　メソッドレベルのブロックを用いた
プログラムの例



(a) 継承の視覚表現
（メソッド内部を隠蔽した場合と展開した場合）



(b) 移譲の視覚表現
図 2　継承，委譲の視覚表現の例

作成したシステム上で，ブロックから実際の Java ソースコードに即座に変換することができる．この変換後のソースコードを eclipse などの統合開発環境やテキストエディタなどにコピーアンドペーストすることにより，プログラムを実行する．

3. システムの評価

作成したシステムの比較対象として，単純に Java の予約語をブロックに置き換えた場合を想定し（以降単純ブロック化と呼ぶ），下記の項目を実現できるか比較した．

(a) 継承，委譲，抽象クラスを実装できるか
(b) クラス間の関係を視覚的に表現できるか
(c) 少ないブロック数でクラスを実装できるか
(d) メソッド内部のブロックの展開，隠蔽機能があるか

表 1 単純ブロック化との機能比較

機能	(a)	(b)	(c)	(d)
従来研究	○	×	×	×
本システム	○	○	○	○

比較結果を表 1 に示す．表に示すように，単純ブロック化では，項目(a)を除いて実現することができず，本システムのほうがオブジェクト指向プログラミングの学習により適しているといえる．

作成したシステムが，実際のプログラミングの学習にどの程度適用可能かを評価した．具体的には，ある大学の 2 年生前期に実施される，オブジェクト指向プログラミングの演習で用いられる課題（継承関係を用いて，各種図形の座標を大きさ，直径などを定義したクラス）が，作成したシステムにより表現可能かどうか確かめた. その結果, VPL のブロックを組み合わせることにより課題を実装できることを確認した．

4. おわりに

作成したシステムにより，VPL を用いたオブジェクト指向プログラミングを，従来よりも容易に行うことができるといえる．また，オブジェクト指向の入門レベルのプログラムを作成可能であることを確かめた．

今後のユーザの操作ログに関する活用方法について述べる．提案システムでは，1 章で述べたように，ブロックをソースコードとして出力可能である．そのため，例えばあるデザインパターンに基づくプログラミングの課題を，VPL を使用する場合と使用しない場合とで被験者に取り組んでもらい，それぞれの場合のソースコードのスナップショットを一定時間ごとに取得することができる．このスナップショットを比較することにより，初学者がオブジェクト指向プログラミングを習得する際の，VPL の長所と短所が明確になることが期待される．

参考文献

[1] 文部科学省 教育の情報化の推進 http://www.mext.go.jp/a_menu/shotou/zyouhou/detail/1416408.htm
[2] 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, vol.55, no.1, pp.57-71 (2014).
[3] 情報処理推進機構 社会基盤センター: ソフトウェア開発データ白書 2018-2019, 情報処理推進機構 (2019).