モデル検査向け教育用ビジュアルプログラミング言語の開発

山下 誠治[†1]　角田 雅照[†1]　横川 智教[†2]

概要：モデル検査を大学などの講義で取り上げることを容易にするために，Google Blockly を用いたビジュアルプログラミング言語を開発した．これにより受講者の文法誤りが減少し，文法誤りに対する教員側の質問対応時間を軽減することができると期待される．

1. はじめに

近年，ソフトウェアの品質を高めるための手段として，モデル検査が着目されている．モデル検査とは，システムの状態を網羅的に探索することにより，システムが求められる性質を満たしているかどうかを判定する方法である．モデル検査方法として，特に SPIN（Simple Promela Interpreter）が注目されている．SPIN は Promela で記述されたプログラムを実行することにより，プログラム（主に分散システム）の性質を検証する．

SPIN を始めとするモデル検査を普及させるためには，大学などの講義で取り上げて，学生がモデル検査に触れる機会を増やすことが重要であると考えられる．ただし，講義などで取り上げる場合，実習形式以外では一人の教員で教えることが多く，多数の学生が書いたプログラムの文法誤りを対応することは時間的に困難である．

そこで，大学などの講義において，モデル検査を学生が容易に試すことができるように，ビジュアルプログラミング言語によりモデル検査のプログラムを記述することを提案している[3]（本稿で紹介するシステムは文献[3]で提案したプロトタイプを拡張したものである）．ビジュアルプログラミング言語とは，Scratch[1]に代表される，ブロックなどを組み合わせてプログラムを作成する言語を指す．これにより，少なくとも文法誤りは軽減されることが期待され，講義においてモデル検査を取り上げやすくなる．

なお本稿では，実務者の利用時の負荷を軽減することを目的としていない．実務者がモデル検査の入門としてビジュアルプログラミング言語を使うことは可能である．ただし，実務でのモデル検査の適用の難しさは，文法の理解や文法誤りの除去にあるのではなく，モデル構築にある．そのため，ビジュアルプログラミング言語による実務者の負荷軽減はあまり期待できない．

2. Google Blockly

ビジュアルプログラミング言語を実現するために，本稿では Google Blackly[2]を用いた．これは Google が提供するビジュアルプログラミング言語の開発環境である．Blockly を用いたデモプログラムのひとつとして，Code Editor が公開されている．本稿ではこのデモプログラムをベースとしている．

Code Editor では，ブロックでプログラムを記述することができ，それを Javascript などの別の言語に変換することができる．Code Editor のスクリーンショットを図 1 に示す．図のように機能ごとにブロックが分類されており，ブロックをドラッグ・アンド・ドロップすることによりプログラムを作成する．ここで「Javascript」などのタブをクリックすると，その言語で記述されたコードが表示される．これらのブロックの動作や，ブロックからコードへの変換規則は Javascript で記述されている．これらのコードを修正することにより，新たなブロックを作成したり，変換対象の言語を追加したりすることが比較的容易となっている．

ブロックにはテキストボックス，リストボックス，チェックボックス，ラベル（文字列）を設定することができる．また，他のブロックを内部にはめられるかどうかを設定できるとともに，ブロックを他のブロックにはめられるかどうかを設定することもできる．これらの設定によりブロックの外観が変化する．変数や関数のブロックも定義されており，例えば新たな関数を定義すると，その関数を呼び出すブロックが自動的に追加される．関数の定義時には引数の指定も可能である．変数などの定義時に名前の重複を防ぐ機能も実装されている．さらに，ツールチップの表示や，ブロックの文字列などの多言語対応も可能である．

Blockly は Web ブラウザ上で動作するため，Web サーバを設置し，サーバ側でモデル検査を行ってクライアントに結果を返す仕組みを構築することも可能である．本稿で示すシステムでは，クライアント側（学生のコンピュータ）の Web ブラウザ上でモデル検査のためのプログラムを作成し，サーバ側でそれを検査できるようにしている．

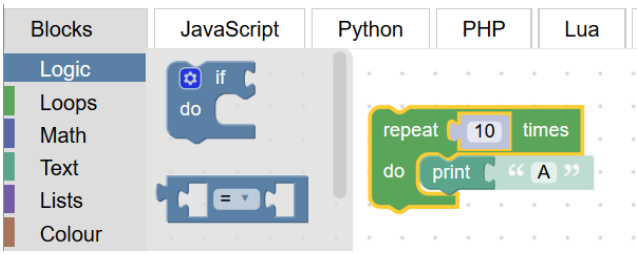

図 1　Code Editor のスクリーンショット

†1 近畿大学
Kindai University
†2 岡山県立大学
Okayama Prefectural University

表 1　ビジュアルプログラミング言語に含まれるブロック

ステートメント	詳細
if	else 使用可
assert	
LTL	名称 定義不可
二項演算子	implication, equivalence, until など
単項演算子	globally, finally, next など
loop	break, timeout, skip 使用可
変数の型	int, bit, bool, byte, short, mtype
#define	
length of	
channel 定義	バッファ 指定可
send	
receive	evaluate 使用可
process	active 指定可，initialize 使用可
run	
goto	ラベル 使用可
atomic	

3. ブロックによる Promela の記述

本稿のビジュアルプログラミング言語に含まれる Promela のステートメントを表 1 に示す．本稿では，講義の数時間を使って SPIN を入門レベルで解説することを前提としている．そのため，本稿のビジュアルプログラミング言語では，Promela の一部のステートメントが含まれていない（例えば process の引数は定義できない）．

性質の検証：プログラムの性質を検証するために assert と LTL（Linear Temporal Logic）のブロックを定義した．assert では満たされるべき性質を条件として記述し，条件が偽となった場合 SPIN による検証時にエラーとなる．LTL のブロックでは，システムが満たすべき性質を線形時相論理で記述する．それぞれのブロックと生成されるコードを図 2 に示す．

名前の定義：変数や行ラベルなど，プログラムで同じものが何度か現れるものについては，最初に名前のみブロックとしてユーザが定義し，その後実際に変数を宣言することをシステムの仕様とした．図 3 に変数の定義の例を示す．

リストの作成：Promela では，チャンネルや mtype などで，カンマで区切られたリストを作成する必要がある．これについては，リストの作成のための専用のブロックを用意し，これを用いることとした．また，変数などと同様に，mtype の宣言前には，事前に名前のブロックを定義することとした．図 4 に記述例を示す．

3.1 システムの機能

本稿のシステムでは，作成されたプログラムをファイルに保存する機能，保存したファイルを読み込んでプログラムを表示する機能，および作成したプログラムをサーバ上でモデル検査する機能を実装した．作成されたプログラムを XML の形式にする機能，および XML からプログラムに戻す機能は Blockly にあらかじめ実装されており，ここではこの機能を用いた．

モデル検査では，クライアントからサーバに Promela で

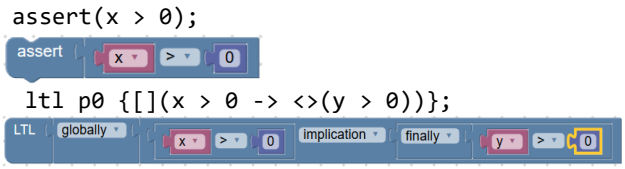

図 2　assert, LTL の記述例

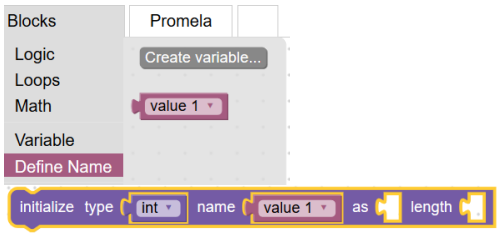

図 3　変数の定義例

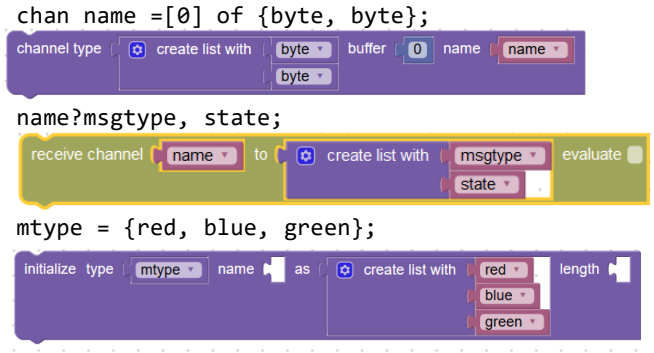

図 4　チャンネル，mtype の記述例

記述されたプログラムを渡し，サーバ上で SPIN を実行し，結果をクライアント側に返している．ブロックから変換された Promela のコードは Blockly にあらかじめ実装された機能により取得している．本稿で紹介したツールは Web サイト[4]にて公開している．

4. おわりに

ワークショップでは，モデル検査の講義でビジュアルプログラミング言語を用いる際に，どのような機能が必要であるかについて議論したい．

謝辞　本研究の一部は，日本学術振興会科学研究費補助金（基盤 C：課題番号 16K00113，基盤 A：課題番号 17H00731）による助成を受けた．

参考文献

[1] Google: Blockly, https://developers.google.com/blockly/
[2] Scratch, https://scratch.mit.edu/
[3] S. Yamashita, M. Tsunoda, and T. Yokogawa: Visual Programming Language for model-checkers Based on Google Blockly, In Proc. of International Conference on Product-Focused Software Process Improvement (Profes), pp.597-601 (2017).
[4] http://www.info.kindai.ac.jp/~tsunoda/vpl/code/