ビジュアルプログラミング環境 AT による Wi-Fi を用いた外部ハードウェア制御機能の開発

Development of the Functionalities for Controlling External Hardware Using Wi-Fi from a Visual Programming Environment: AT

上島　駿[*1], 國宗　永佳[*2], 新村　正明[*3]
Shun KAMIJIMA[*1], Hisayoshi KUNIMUNE[*2], Masaaki NIIMURA[*3]
[*1]信州大学大学院総合理工学研究科
[*1]Graduate School of Science and Technology, Shinshu University
[*2]千葉工業大学情報科学部
[*2]Faculty of Information and Computer Science, Chiba Institute of Technology
[*3]信州大学学術研究院工学系
[*3]Institute of Engineering, Shinshu University
Email: kamijima@seclab.shinshu-u.ac.jp

あらまし：筆者らはビジュアルプログラミング環境 AT を開発している．AT では，変数の値の変化と出力の値を画面内に表示することで，プログラムの動作を示していた．本研究では，プログラムの動作を実世界に反映し，直感的な理解を促すことを目的として，外部ハードウェア制御機能を開発している．これまで開発してきた制御機能は，USB による有線通信を行うものであった．本研究では有線による制約を解消するために Wi-Fi を用いた外部ハードウェア制御機能の実装について述べる．
キーワード：プログラミング教育，マイコン，ビジュアルプログラミング環境

1. はじめに

筆者らは，ビジュアルプログラミング環境 AT を開発している [(1)]．AT は，各プログラミング言語に特有な文法規則の知識なしに，アルゴリズム的思考法（目的を達するための処理を分解・整理する思考法）を学習するための学習・授業支援システムである．また，使用可能なプログラム構成要素（ブロック）・機能の制限，課題の作成・提示，解答の分析といった授業支援機能を持ち，PC 端末だけでなくタブレット端末でも動作する．

AT ではプログラムの動作を，変数の値の変化や出力される値を数値や文字列として画面内に表示するという形で学習者に提示しており，その内容は比較的抽象度が高い．これに対し，LED を点灯させる，車の模型を操作するというように，外界の物を動かす，あるいは，明暗や温度を計測するなど，外界の情報を取得し画面に反映する形でのプログラム動作の提示は抽象度が低いといえる．本研究では，プログラムの動作をコンピュータ外部のハードウェアにも反映させ，また，外部のハードウェアから外界の情報を取得し，プログラムの変数の値などに反映することによって，学習者のより直感的なプログラムの動作理解を促すことを目的として，AT による外部ハードウェア制御機能を開発している [(2)]．

これまで開発してきた制御機能は USB による有線通信で Arduino [(3)] を制御するもので，USB 通信のためのプログラムである通信ヘルパアプリケーションを別途起動する必要がある．さらに，開発した通信ヘルパアプリケーションは iOS や Android といったタブレット端末には対応しておらず，対応させるには環境ごとに別途プログラムを開発する必要がある．本研究では，これらの問題を解消するため，Wi-Fi を用いた制御機能を開発する．本稿では，AT と外部ハードウェア制御機能の概要について述べる．

2. AT の概要

AT は Web アプリケーションとして実装されたビジュアルプログラミング環境であり，Web ブラウザ上で利用できるプログラム作成画面と，サーバ側で提供する授業支援機能からなる．授業支援機能では，使用可能なブロックの制限やプログラムの編集・実行機能の制限，課題の作成・提示，解答の採点・分析が可能である．

AT のプログラム作成画面は，エディタ部と情報提示部からなる．エディタ部では，ブロックを組み合わせることでプログラムを作成する．情報提示部では，作成したプログラムの実行・停止，または提出や保存といった操作を行い，プログラムの変数の値や出力などの情報が提示される．AT はプログラムの実行を，ブロック 1 個または任意の複数個ごとに中断しつつ行うステップ実行機能を有する．また，ステップ実行中は，動作しているブロックを強調表示させ，その際の各変数の値や出力を表示させることで，プログラムの動作状況を学習者に提示する．

AT では，ステップ実行とステップ実行中の変数の値や出力の表示を実現するために，JS-Interpreter [(4)] というブラウザのインタプリタ上で動作し，ステップ実行機能を持つ JavaScript インタプリタを利用している．よって，AT におけるプログラムの実行は，ブロックの組み合わせから JavaScript のプログラムを生成し，ブラウザ上の JS-Interpreter で実行する形で実装されている．

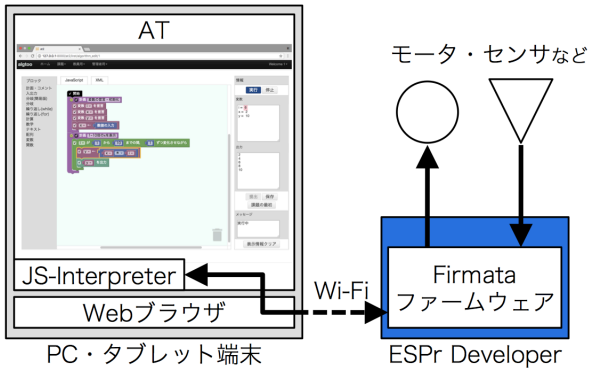

図 1 AT による外部ハードウェア制御

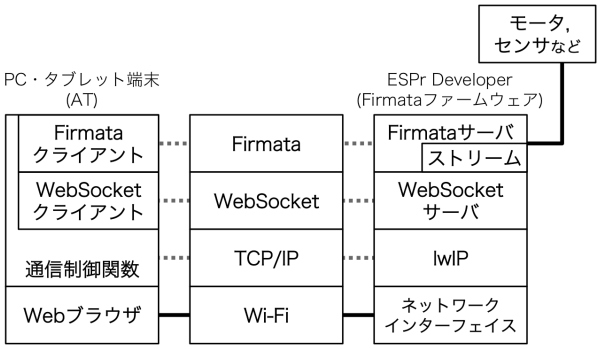

図 2 プロトコルと機能の対応

3.　外部ハードウェア制御機能の実装

本研究では AT でセンサやモータなどの機器を制御し，画面外にプログラムの動作を具体的に提示するため，外部ハードウェア（マイコンボード）を制御するプログラムを作成・実行する機能を実装している．AT が提供しているステップ実行中のブロックの強調表示や，変数の値や出力の表示と，マイコンボードに接続した機器の制御を両立しつつ，以前の実装での通信ヘルパアプリケーションの問題を解決するために，次の要件を設定した．(1) プログラムの実行は AT で行う，(2) ブラウザ上で動作する AT から直接外部ハードウェアを制御する，(3) 外部ハードウェアから取得した情報を AT に反映する，(4)AT の動作する Web ブラウザと同一の PC・タブレット端末上で別途動作するプログラムを必要としない．これらの要件を満たすために，図 1 に示す構成で実装を行った．これまで開発してきた機能では，制御するマイコンボードは Arduino であったが，Wi-Fi 通信を行うために ESPr Developer [5] に変更している．ESPr Developer は Wi-Fi モジュールを備えた非常に安価なマイコンボードであり，Arduino のようにセンサやモータといった機器を接続して利用できる．また，Arduino と同様に ArduinoIDE を用いたファームウェアの開発も可能である．

Web ブラウザ上の AT と ESPr Developer が Wi-Fi を介して直接データの送受信を行うため，WebSocket [6] を用いる．また，送受信するデータは Firmata プロトコル [7] に従う．Firmata は，PC 端末とマイコンで通信を行うためによく用いられるプロトコルで，制御命令や取得した情報のやり取りが可能となる．既存の Firmata ファームウェアには，WebSocket に対応したものが無かったため，ESPr Developer の Wi-Fi 通信に対応したファームウェアである StandardFirmataWiFi [8] を元に新たに開発した．また，JS-Interpreter には，WebSocket 通信を行い，Firmata のデータをやり取りする通信制御関数を追加した．各プロトコルと機能の対応は図 2 のようになる．図 2 中のストリームとは，Firmata サーバが下の階層のプロトコルの通信に対応するためのものであり，同図中の lwIP [9] とは TCP/IP スタックである．

AT の外部ハードウェアを制御するブロックは，これまでに開発していた機能の Arduino を制御するブロックを流用した．このブロックを使用すると，AT では，外部ハードウェアの制御命令を含むプログラムを生成する．JS-Interpreter では制御命令の部分を読み込むと，通信制御関数を呼び出し，制御命令を外部ハードウェア(ESPr Developer)へ送信する．その後，ESPr Developer の Firmata ファームウェアで制御命令が実行され，接続された機器の制御や情報の取得が行われる．取得した情報は，反対の経路で AT に転送され，画面上に反映される．

4.　おわりに

本稿ではビジュアルプログラミング環境 AT において，マイコンボードを制御するプログラムを作成・実行する機能の開発について述べた．現時点では，基本的な制御の実行と情報の取得について実装を行った．今後は，実際の教育現場での使用を目指し，機能の改良を行う．

謝辞　本研究は JSPS 科研費 15K01023, 26350284, 16H03074 の助成を受けたものです．

参考文献

(1) 國宗永佳, 大浦真暉, 香山瑞恵, 新村正明: “授業支援機能を有するビジュアルプログラミング環境 AT の開発”, 教育システム情報学会第 39 回全国大会講演論文集, I1-09, pp.17-18 （2014）
(2) 國宗永佳, 河野直, 新村正明: “ビジュアルプログラミング環境 AT における Arduino 制御機能の開発”, 教育システム情報学会第 41 回全国大会講演論文集, B4-1, pp.287-288 （2016）
(3) “Arduino”, https://www.arduino.cc/
(4) N. Fraser:“JS-Interpreter”, https://github.com/NeilFraser/JS-Interpreter/
(5) “ESPr Developer”, https://www.switch-science.com/catalog/2500/
(6) “WebSocket”, https://triple-underscore.github.io/RFC6455-ja.html
(7) “Firmata”, https://github.com/firmata/protocol
(8) “StandardFirmataWiFi”, https://github.com/firmata/arduino/tree/master/examples/StandardFirmataWiFi
(9) “lwIP”, https://savannah.nongnu.org/projects/lwip/
(文献(3)~(9)は 2017 年 5 月 17 日に閲覧した)