JSiSE Research Report
vol.37,no.4(2022-11)

「ものグラミング」手法のプログラミング教育への応用

大野 浩之 *1, 松浦 智之 *2, 森 祥寛 *1
*1 金沢大学 学術メディア創成センター, *2 ユニバーサル・シェル・プログラミング研究所

Application of the “Monogramming” Method to Programming Education

Hiroyuki Ohno*1, Tomoyuki Matsuura*2, Yoshihiro Mori*1
*1 Emerging Media Initiative, Kanazawa University, *2 Universal Shell Programming Laboratory Ltd.

“Monogramming” is a method for efficiently developing IoT devices by combining shell scripts and small-scale microcontrollers in a POSIX environment, which the authors have proposed and are working to promote. This method has been developed based on the assumption that engineers in the field will use this, but recently we are also aware of its application to programming education. In this paper, we describe our findings in applying the “Monogramming” method to programming education, as well as future developments.

キーワード: シェルスクリプト, IoT 教育, ものグラミング

1. はじめに

「ものグラミング」とは大野が発案し普及啓発に努めている，POSIX 環境下のシェルスクリプトを活用した IoT 機器を開発するための手法である．「ものグラミング」はいくつかの異なるアプローチがあるが，これらのうち「ものグラミング 2」は POSIX 機と小規模なマイクロコントローラを組み合わせて効率よく IoT 機器を開発できる．これまで「ものグラミング」は当該分野の技術者を念頭に置いて展開してきたが，最近ではプログラミング教育への展開も意識するようになった．本報告では「ものグラミング」手法を外観した後，プログラミング教育へ応用する際に得た知見，今後の教育現場展開について述べ，技術面において派生した手法，今後の展開についても述べる．

2. 背景

近年，情報技術はさまざまな分野で活用され，それらの技術を持つ人材が必要とされている．2010 年代初頭には ICT などが，2010 年代半ばからはデータサイエンスや DX，Society5.0 などの言葉で表わされる情報技術を元にした社会の変革が求められてきた．例えば，第 5 期科学技術基本計画（2016 年度から 2020 年度）(1) には，データドリブンな社会を目指すことが明記されている．データサイエンスの広まりからは，数理・データサイエンス教育の必要が求められ，文部科学省では，高等教育における「数理・データサイエンス・AI 教育認定制度 (2)」が実施されている．

また，2010 年代初頭頃から電子工作を中心としたメイカーによる新たなものづくりの潮流は，新たな文化であるとしてクリス・アンダーソンによって名付けられた「メイカームーブメント (3)」が広がった．メイカームーブメントの広がりは，さまざまなマイクロコントローラ（以下マイコン）やセンサやアクチュエータ類を取り扱える電子回路が安価に入手できるようになったこと，3D プリンタやレーザーカッターなどの工作機器の利用が簡単にできるようになったことなどを背景にしている．これはこれからの情報技術が，ソフトウェアだけでなく，限定的ながらもハードウェアの取扱いも必要となることを示している．

2020 年代に入り，工業化社会（Society3.0）から情報化社会（Society4.0）への移行は完了し，社会全体にどのようなデータをどのように存在させるのか，そして存在するさまざまなデータをどのように活用するかが重要となっている．

このような流れの中で小規模な IoT デバイスを実装する際にどのようなマイコンを用い，どのようにしてインターネット上のクラウドサービスと連携させ，その際のセキュリティをどのように確保し，どのような言語で記述す

ればよいかについてはさまざまな方法を検討する余地があった．今回報告する「ものグラミング」はこの問題に対するわれわれの回答である．

3. 関連研究

「ものグラミング」の基本的な考え方の一つである「POSIX 環境下のシェルスクリプトの活用」は，POSIX 中心主義というプログラミング指針 (4) が元になっている．POSIX 中心主義とは，ソフトウェアの持続可能性を向上させるための指針であり，一つの企業・団体等が権利を所有するソフトウェアへの依存を極力なくすことによって自ソフトウェアの寿命を延ばそうとする戦術をとる．具体的には，国際規格である POSIX (5) の仕様書に記載されているもののみに依存することを原則とし，POSIX の互換性・持続性の高さで自ソフトウェアの互換性・自足性を担保する．「ものグラミング」が主たる言語としてシェルスクリプトを採用しているのも，それが POSIX 文書に記載されているからである．

また，「ものグラミング」においては，ホスト―デバイス間のデータ送受信路としてしばしば調歩同期方シリアル通信（USB によるエミュレーションも含む）や MQTT (6) を採用する．その理由は，これらを UNIX ホストから見た場合，キャラクタ型デバイスファイルとして扱えたり，UNIX のパイプライン経由で接続可能なコマンドとしてデータ交換ができ，既存の UNIX コマンドとの親和性がよいためである．UART（シリアル通信用回路）のみを持つ単純なデバイスには前者で，TCP/IP スタックを搭載してネットワークを介して接続できるデバイスには後者での通信を採用する．

「ものグラミング」が採用しているシェルスクリプトは，その必要性が注目されており，2020 年初頭には，MIT が「The Missing Semester of Your CS Education (7)」という形でレクチャーが YouTube で公開され，講義資料はボランティアのサポートを受けて各国語に翻訳されている．オンライン公開された．ここでは「コンピュータに代表される計算システムは，手作業を自動化するようにつくられているが，多くの学習者は，繰り返し作業を手作業で行ったり，IoT 機器の操作やデータ処理などでコマンドラインインターフェースのツール，特にシェルを十分に活用できていない.」と述べている．

4. 「ものグラミング」

現在「ものグラミング」と呼ぶ手法は，2012 年に発売された Raspberry Pi (9) を発売開始直後から積極的に利用する過程で自然発生的に発案され，その後いくつかの形態に分化して現在に至る「ものづくり」のためのプログラミング方式である．「ものグラミング」は「ものづくり」と「プログラミング」を合わせた大野らによる造語で，英語では Monogramming と表記する．「ものグラミング」の発案当時から変遷を図示すると図 1 のようになる．

以下にこの図を構成するさまざまな「ものグラミング」の特徴をまとめる．

4.1 ものグラミング 1

Raspberry Pi の OS である Raspbian (現在の Raspberry Pi OS) では 2012 年当時から gpio コマンド (10) が利用可能だった．gpio コマンドは Raspberry Pi が提供している GPIO ピンを個別に操作できるコマンドで，このコマンドを使うことで Raspbian 上のシェルからそれぞれの GPIO ピンをデジタル入力またはデジタル出力用途に割り当てることが可能だった．特段のデバイスドライバを開発したり `ioctl()` システムコールを伴うようなシステムプログラミングをせずとも POSIX 系 OS を搭載したコンピュータ（以下 POSIX 機）の上で一般権限のユーザがデジタル I/O をコマンドを介して実行できるのは画期的な特徴であった．以下に gpio コマンドを使って LED を 2 秒周期で点滅せるコマンドラインを示す．

```
$ while [ 1 ]; do gpio -g write 4 1;
sleep 1; gpio -g write 4 0; sleep 1; done
```

しかし，Raspberry Pi が搭載していない機能は当然ながら利用できず何らかのハードウェアの追加が必要だった．たとえば A/D 変換器を搭載していないためアナログデータを取り込むコマンドは用意されていない．

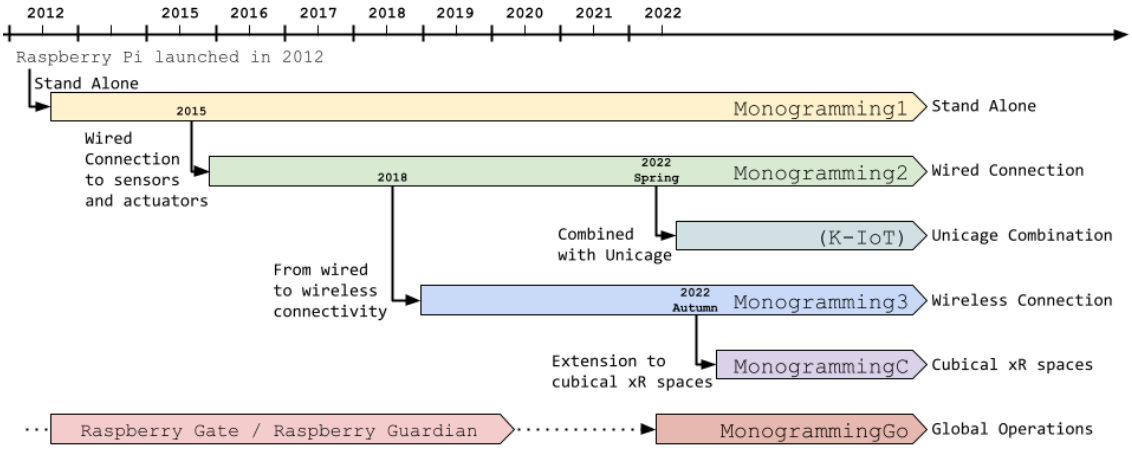

図 1 ものグラミングの発展

4.2 ものグラミング 2

2015 年頃に命名した「ものグラミング 2」は，現在に至るも主流となる方式である．「ものグラミング 1」では Raspberry Pi 単体でデジタル入出力を行ったが，「ものグラミング 2」では Arduino UNO Rev.3 (8) のような小規模なマイクロコントローラ (以下, マイコン) と Raspberry Pi のような POSIX OS を搭載した手のひらサイズかそれより小型の組み込み用途のコンピュータを使い，「選択と集中」のコンセプトをもとにそれぞれに役割を分担をさせている．具体的には，マイコン側にはネットワークを介した通信機能を持たせないかわりにマイコンが得意とするデジタル入出力やアナログ入出力に専念させ，POSIX 機はクラウド上のサービスとの通信や当該通信のセキュリティ確保に専念するかわりに実際の入出力は担当しない．この切り分けはマイコンと POSIX 機の得意とする分野を考えれば自然なことである．この場合，マイコンと POSIX 機を接続する必要があるが，有線接続による非同期シリアル通信で賄っている．この相互有線接続も「ものグラミング 2」の大きな特徴である．ここで問題となるのがマイコンとの非同期シリアル通信をコマンドでどうやって表現するかである．もちろん，目的に合致したコマンドを新たに開発することもできるがわれわれは既存の cu コマンドを用いて以下のように実現している．この例では command1 の出力をマイコンのシリアルポートに送出し，マイコンのシリアルポートを介した出力を command2 の標準入力に送り込んでいる．command1 や command2 がたとえば MQTT のクライアントである mosquitto_sub や mosquitto_pub コマンドであれば，これだけでマイコンはインターネット上のサービスと連携できることになる．

```
$ command1 | cu -l /dev/ttyACM0 | command2
```

4.3 ものグラミング 3

「ものグラミング 3」は，当初は「ものグラミング 2」との対比ために　2018 年に発案された方式である．前項で述べたように「ものグラミング 2」では，マイコンと POSIX 機を優先接続している．実際には Raspberry Pi の大きさは Arduino の大きさはそれほど変わらないので両者を同じ場所に設置すれば有線接続距離は数 cm であるし，Raspberry Pi には Wi-Fi 機能を持たせられるので，「ものグラミング 2」方式で実装した IoT 機器はワイヤレス機器にできるが，実際にはマイコンと Raspberry Pi もワイヤレス接続したいという要求があった．実際にはこの部分を無線接続としてしまうとセキュリティ上の問題が生じる．数 cm の有線接続であれば，この部分で情報を取り出すためのタッピングを発見されずに行うことは実質的には不可能であるが，この部分が無線通信であれば通信の傍受はある程度離れた場所からも可能である．傍受されても通信内容が漏洩しないようにするためには暗号化が必要になるが，これはマイコン側には大きな負担になるし，マイコン k によっては限られた処理能力から実行できない場合もある．当初の「ものグラミング 3」は，このことを明確にし理解を促すことを目的に準備したという経緯があるた

め,，それほど積極的には展開されなかった.

4.4 ものグラミング C

「ものグラミング３」は，上述のような特徴ゆえにその普及には積極的ではなかったが，2022 年になり「ものグラミング 3」から「ものグラミング C」が分岐した.「ものグラミング C」は今後一定の普及が見込めるバーチャル・リアリティヘッドセットを POSIX 機からコマンドを介して利用する方式である．この方式は特定のバーチャル・リアリティヘッドセットを対象にしているのではないが，どのようなバーチャル・リアリティヘッドセットであっても当該ヘッドセットと POSIX 機との接続は無線接続となることが予想されるので,「ものグラミング２」ではなく「ものグラミング３」を発展させて実現することとした．なお，表記上は「ものグラミング C」であるが，発声する場合は「ものグラミング・タイプ C」とする場合が多い.

4.5 ものグラミング Go

これまで述べた 4 種類の「ものグラミング」が接続形態の提案であるのに対し,「ものグラミング Go」管理方式の提案である.「ものグラミング」では POSIX 機単体（ものグラミング１）か，選択と集中のコンセプトに基づいたマイコンと POSIX 機から構成されており，インターネット上のクラウドサービスと通信を行う場合だけでなく，会社や学校などのインターネットとの直接通信ができないネットワーク上であっても通信上のセキュリティを確保するためのセキュリティゲートウェイが必要である．通常は POSIX 機がその機能を担うが,「ものグラミング」方式の基づく IoT 機器が複数存在する場合，セキュリティゲートウェイのポリシーを一括して管理したり，遠隔地に設置した「ものグラミング」方式に基づく IoT 機器と相互連携する場合場合などには，VPN のような安全な仮想通信路を提供する必要がある．このような用途に適した IoT 機器向けのセキュリティゲートウェイとそのポリシーの管理方式の一つに大野らが開発を続けている Raspberry Gate (11) と Raspberry Guardian (12) がある．Raspberry Gate と Raspberry Guardian を「ものグラミング」方式で運用する IoT 機器群のために転用したのが「ものグラミング Go (Global Operations)」である.

4.6 K-IoT

これまで取り上げた 5 つが「ものグラミング」の主要な５つであるが，これとは別に「ものグラミング 2」から分岐する形で生まれたのが「K-IoT」である.

K-IoT とは,「ものグラミング」の考え方に基づく教育を，松浦が属する企業（USP 研究所）の社内講習として展開するにあたっての名称である．5. 節で記す活動は教育機関に向けたものであり，コンピュータリテラシ教育を主眼としている．一方 K-IoT は，営利企業の従業員が業務案件をこなせるようになるための実践教育を主眼としており，実際の業務で取り扱う具体的な製品の取り扱い，また，業務レベル品質での開発者を養成するための講義・実習内容を取り入れている.

K-IoT のもう一つの特徴は，USP 研究所が推進している業務システム開発フレームワークであるユニケージ開発手法 (13) と強く結びついた講習内容としている点である．ユニケージ開発手法は，UNIX 哲学 (14) の考え方をベースにし，業務システム開発の生産効率や費用対効果を上げることを主眼にまとめられたフレームワークであるが，そこで確立されたマネジメント手法やプログラミング技法，ソフトウェアライブラリ（UNIX コマンド）等を IoT 分野に応用し,「ものグラミング」の考え方との融合あるいは連携を図っている.

K-IoT の講習会を通し，受講者達は「ものグラミング」の考え方への理解が進んだ．その結果，市販されている電子機器が持つ汎用的な機能を UNIX コマンドとして作り置けば開発がより簡単にできるという発想を持つに至っており，新たなコマンドの開発が進んでいる.

5. プログラミング教育への展開

5.1 「ものグラミング」の演習型講義への展開

メイカームーブメントは，電子回路を利用した電子工作を中心としたものづくりへの挑戦をしやすい環境を作った．一方で，これまでマイコンボードを用いた電子工作を完成させるために必須だったプログラミングが，インターネットへの接続機能などの付加により複雑さが増しており，この知識や技術の習得へのコストが挑戦への躊躇要因

となっている．これを解決するために提供する方法が「ものグラミング」であり，われわれは POSIX 環境下での「ものグラミング」によるプログラミング教育を作成し，演習型の講義として実施した．

このプログラミング教育の目的は，以下の 3 点に集約している．

1. センサやアクチュエータなどの簡単な操作方法の習得
2.「ものグラミング」の考え方を踏まえた 1 行のシェルスクリプトを用いて，センサから取得したデータをクラウドへの送信したり，クラウドに送信されたデータをパソコンやスマートホンで受信したりする流れを MQTT を使って実現する方法の習得
3. 同様に，MQTT を使って，パソコンやスマートホンからクラウドを介してアクチュエータや表示器を制御する方法の習得

「ものグラミング」によるプログラミングを学ぶ講義では，シェルスクリプトを使用した演習を行うことが前提となる．そのため，学習者が講義中に使用するノートパソコンと，そのノートパソコン内に POSIX に準拠した UNIX 系 OS が使用可能な環境の構築を前提となる．学習者が準備したノートパソコンが，WindowsOS であれば WSL（Windows Subsystem for Linux）をインストールさせて POSIX 環境を確保し，macOS もしくは Linux であればそれ自体が POSIX 環境であるので最初から利用可能なターミナルを使用させた．macOS のターミナルを使用する場合は，Homebrew (15) をインストールさせた．一方，WSL 上では Ubuntu を導入するのでコマンド管理には apt コマンドを用いることとし，apt-get コマンドは非推奨とした．シェルには sh, csh，bash，zsh などがあるが，対話的利用では bash を，シェルスクリプトでは sh を用いることを原則とした．bash を使用する場合は，必要に応じて，最近の macOS で採用されている zsh への読み替え方法などを提供する．ここまでの対応で，学習者は自身が持つパソコンで POSIX 環境下の CLI 端末を使用でき，パソコンの OS の種類によらず，シェル上の操作やシェルスクリプトプログラミングなどの作業を同じように実施可能になる．

講義ではその後，CLI による操作方法，コマンドライン入力とは何か，コマンド入力の方法，記号の読み方と入力方法などの説明を行う．コマンドライン入力は，UNIX コマンドによる処理がファイルの処理であることから説明する．これは前述の UNIX 哲学に基づく処理であること，その中のシェルスクリプトのコマンド群は C 言語などによるプログラミングの結果の活用に他ならないこと，シェルスクリプトのコマンドを使って，データ処理を行う場合，その多くは，データファイル内に書かれたプレーンテキストを扱うことになり，直観的な作業が可能になることを説明する．そして，1 コマンド毎の結果を確認しながら，コマンドとコマンドを「｜（パイプ）」で繋ぐことで，段階を追った作業が可能となり，最終的にある特定のファイル処理を行うスクリプトが完成する．これは扱うデータの発生源や種類，規模が異なっても，同じような作業となる．ここまで行うことで，環境構築が完了する．

5.2 「ものグラミング」を応用した教材開発

「ものグラミング」をテーマとした数々の講義を実施する活動と並行して，これら講義で用いる教材としても役立てられる様々なソフトウェアやハードウェアの開発も行っている．

2019 年から，われわれは「タイミング管理コマンド」(16) と称する UNIX コマンド群を作成している．これは，UNIX の標準入出力を流れるデータのタイミングを自在に制御することで，データの値のみならず，それらの到来時刻にも情報的価値を持たせようとするものである．音楽データの通信規格である MIDI メッセージは，演奏における楽器の種類や音程等を値で表現している一方，リズムについてはデータの送出タイミングで表現しているが，これと同様の考え方である．タイミング管理コマンドの作成によって，「ものグラミング」におけるデバイスからの到来データにタイムスタンプを付けたり，あるいは付加したタイムスタンプに基づくタイミングに従って出力値をデバイスに送信することが容易になった．

タイミング管理コマンドによって，センサデバイスから報告される値の時刻を計測すること，それに基づくア

クチュエータデバイスへの出力値計算を，「ものグラミング」の考え方に則って実装することが容易になったため，それを応用した倒立振子ロボットの開発にも成功した (17) (18)．この倒立振子は三つのセンサ（加速度・ジャイロ・ロータリーエンコード）と同軸二輪のモータを搭載しており，それらセンサから逐次報告される計測値にタイムスタンプを付加し，独自開発したカルマンフィルタコマンドでノイズを除去した後，同様に開発した線形二次ガウシアン制御や PID 制御コマンドに通して制御計算を行い，モータへの出力値を計算・送出している．これは，フィードバック制御になっており，データのループは UNIX の無名パイプおよび名前付きパイプという従来から存在する仕組みのみで実装されている．

また，タイミング管理コマンドを応用してバーサライタを製作することにも成功した (19)．バーサライタとは，LED 等の点光源を直線状に並べ，それを回転または平行移動させている間に各点光源を高速に明滅させることで，残像として文字や図形を浮かび上がらせる装置であるが，人間の目の特性上，実用性を持たせるには 1/100 秒（10 ミリ秒）オーダーで明滅させられる性能が求められる．しかし，文献 (18) での検証から，Raspberry Pi 3B+ 程度の安価な PC，かつそこで非リアルタイムな通常の Linux カーネルを用いていても，タイミング管理コマンドによって UNIX のパイプからバーサライター列分（30 バイト程度）のデータが 10 ミリ秒オーダー精度で送出可能であることが確かめられたため，実際に製作し，文字のビットマップデータを送りながら装置を振るとその文字が浮かび上がることを確認した．

これらの成果は，「ものグラミング」学習促進のための教材としても活用する計画である．

6. 金沢大学内外での普及啓発活動

6.1 金沢大学などでの教育活動

われわれは，前節（5.1）で示した「ものグラミング」の普及啓発のための演習型講義として，金沢大学にて「シェルスクリプトを用いた「ものグラミング」演習（1 単位全 8 回)」と「シェルスクリプトを用いた「ものグラミングと大規模データ処理」演習（2 単位集中講義)」いう演習型講義を実施している．また，石川県内の高等教育機関が連携して組織している大学コンソーシアム石川で開講されているいしかわシティカレッジ (20) では，「クラウド時代のものグラミング概論（2 単位全 16 回)」を実施している．実施時期などについては，図 2 にタイムラインを記した．特に，大学コンソーシアム石川いしかわシティカレッジ開講講義では，金沢大学以外の学生や石川県内外の市民まで履修できる講義である．

6.2 海外での教育活動

「ものグラミング」手法の普及啓発を意図した教育活動はすでに海外でも実施している．以下では地域別にこれまでの経緯を述べる．

6.2.1 インド

インドでは 2018 年にハイデラバード州のハイデラバード工科大学において，「ものグラミング１」をベースにしたワークショップを開催した．当時はまだ「ものグラミング２」手法が確立していない時期だったため，コマンドラインやシェルスクリプトを活用して Raspberry Pi 単体でデジタル入出力を行う方法（すなわち「ものグラミング１」）を実践して見せた．同大学からは「いつでも再来訪歓迎」との連絡を受けている．

6.2.2 カナダ

カナダ最東端のノバスコシア州の州都ハリファクスには同州最大となる州立ダルハウジー大学がある．大野はこの大学のコンピュータ工学部に 2018 年にサバティカル研修で 5 ヶ月ほど滞在した．滞在中，訪問先の研究室で「ものグラミング１」手法を学部および大学院生に伝えるとともに，まだ完成はしていなかった「ものグラミング２」手法についても議論しその内容を発展させた．2018 年 9 月に帰国した後も，同大学の同学部には 2018 年に 1 度, 2019 年に 2 度, 2020 年に 1 度訪問して「ものグラミングワークショップ」や関連レクチャーを開催している．滞在先の研究室とはほぼ毎週開催されている定例研究ミーティングに継続して参加しており (21)，こちらの現状を常に更新して対面でのワークショップを開催したい旨を伝えているので，いつでもワークショップを対面で開催できる準備は整っている．先方では「ものグラミング」の考え方を取り

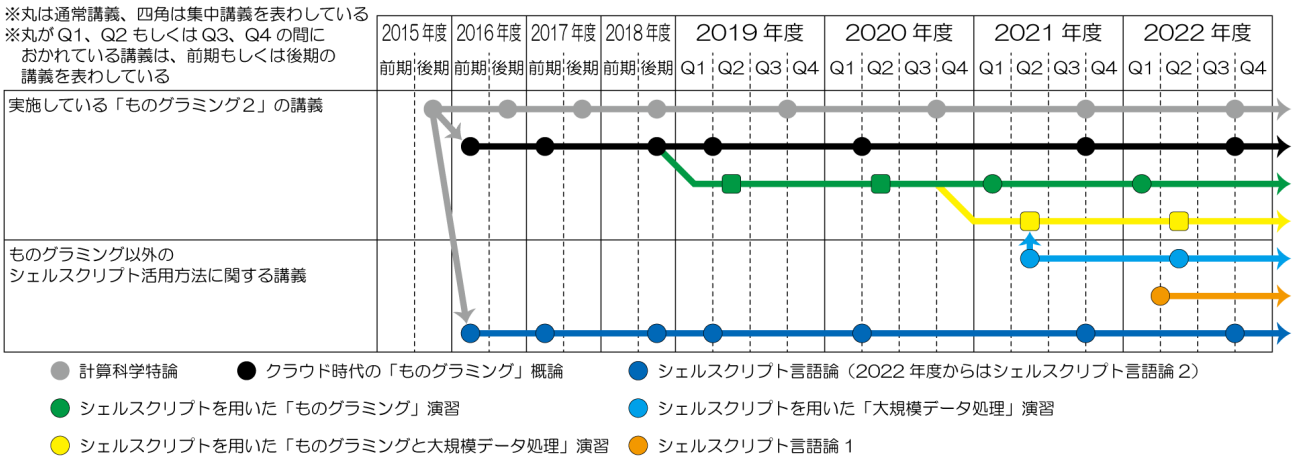

図 2 これまでの教育実践による講義の流れ

入れた学部生や修士課程の学生が卒業論文や修士論文を書き上げており，「ものグラミング」は一定の知名度を得ている．

6.2.3 ポルトガル

ポルトガルのリスボンには松浦が所属するユニバーサル・シェル・プログラミング研究所のヨーロッパ拠点がある．同拠点から紹介を受け，同拠点技術者とともに 2018 年にリスボン大学工学部を訪問し，「ものグラミング 1」について技術的説明を行い今後の連携について話し合いを行った．

6.2.4 タイ

大野と森が所属する金沢大学学術メディア創成センターには，2022 年 4 月までタイから来日して国内で修士課程と博士課程を終えて学位を取得し，その後金沢大学に就職し，われわれの研究グループに合流した教員がおり，2022 年 5 月にタイに帰国し現職（カセサート大学）に就いている．このためカセサート大学のコンピュータ工学部には「ものグラミング」を受け入れる人的環境が整っている．対面実施が望ましい実機を用いる「ものグラミング」ワークショップを，2023 年度には実施する方向で調整している．

7. 考察

「ものグラミング 1」では POSIX 機単体で IoT 機器の構築を目指し，「ものグラミング 2」や「ものグラミング 3」ではマイコンと POSIX 機を連携させている．いずれの方法でもマイコンのみで IoT 機器を構築するようなことはしない．しかし，実際の現場では ESP-8266 や ESP-32 のような 32bit マイコンと Wi-Fi 機能を組み合わせ，マイコン単体でインターネット上のクラウドサービスと連携させている事例が多い．われわれはこの方法にはいくつかのリスクがあると考えている．まず，単体のマイコンが搭載する TCP/IP や UDP/IP プロトコルスタックの品質が不明である．このためこれらのマイコンがインターネット側から Wi-Fi を経由して攻撃を受けるリスクについて評価できない．また現在の実装は IPv4 に限定されており IPv6 への対応については不明である．また対応している IPv4 についても RFC に準拠した IPv4 を完全にサポートしているのかそのサブセットなのかは不明である．仮にあるマイコンについては IPv4 の性能と安全性が確認されたとしても，他のマイコンについては別途確認しなければならない．一方，ものグラミング方式においては，マイコンは有線または無線接続で POSIX 機と通信を行う以上のことはせず，実際のインターネットとの通信は POSIX 機（たとえば Raspberry Pi）が行う．この場合 POSIX 機の通信機能は現用のサーバやルータなどのプロトコルスタックと同一である．それらはオープンソースなので安全性の確認が確実に行え，万一新たな脆弱性が見いだされた場合にはサーバやルータに対する脆弱性対策が速やかに公表されるのでそれをそのまま適用すればよい．この点は，マイコン単体ですべて行う方式に対する大きな利点である．

8. おわりに

本報告では，われわれが取り組んでいる「ものグラミング」による IoT 開発とその進展を述べ，プログラミング教

育への展開について述べた.

ものグラミング手法はそのいずれもが IoT 機器の開発を念頭に置いて開発され使われてきたが，IoT 機器開発を業務とせずとも，「シェルスクリプトを用いた電子工作向けプログラミング教育」として教育現場に投入できると考え実践してきたので本報告ではその現状について述べた．ものグラミング手法のプログラミング教育への展開は引き続き実施し，海外展開を視野に入れつつその内容を充実させたい.

謝辞

カナダ国ノバスコシア州立ダルハウジー大学コンピュータサイエンス学部教授の Prof. Sampalli および同教授の研究室の大学院生からは「ものグラミング」の有効性について検討する際に多くの示唆を得た．この経験が，RaspberryCom*PoTE の開発を後押しした．ここに記して感謝したい.

参 考 文 献

(1) 内閣府-科学技術政策：科学技術基本計画, https://www8.cao.go.jp/cstp/kihonkeikaku/index5.html （2022 年 10 月 5 日確認).

(2) 文部科学省：数理・データサイエンス・AI 教育, https://www.mext.go.jp/a_menu/koutou/suuri_datascience_ai/00001.htm （2022 年 10 月 5 日確認).

(3) Anderson, Chris：Makers: The New Industrial Revolution, Crown Business（2014).

(4) 松浦智之, 大野浩之, 當仲寛哲: “ソフトウェアの高い互換性と長い持続性を目指す POSIX 中心主義プログラミング”, デジタルプラクティス Vol.8, No.4, pp.352-360, (2017).

(5) What is POSIX?, The Open Group, https://collaboration.opengroup.org/external/pasc.org/plato/ （2022 年 10 月 5 日確認).

(6) MQTT - The Standard for IoT Messaging, OASIS, https://mqtt.org/ （2022 年 10 月 5 日確認).

(7) Anish, Athalye and Jon, Gjengset and Ortiz, Jose Javier Gonzalez：“The missing semester of your CS education (日本語版)”, https://missing-semester-jp.github.io/ （2022 年 10 月 5 日確認).

(8) Arduino Foundation “Arduino” https://www.arduino.cc/ （2022 年 10 月 5 日確認）

(9) Raspberry Foundation, https://www.raspberrypi.org/ （2022 年 10 月 5 日確認）

(10) Wiring Pi, “GPIO Interface library for the Raspberry Pi”, http://wiringpi.com/ （2022 年 10 月 5 日確認）

(11) 大野浩之, 鈴木裕信, 北口善明, “Raspberry Gate の設計と実装： IoT 時代に資するセキュリティゲートウェイの構築”, 電子情報通信学会技術研究報告, Vol.116, No.282, pp.21-26

(12) Shuting Hu, Hironobu Suzuki, Yoshiaki Kitaguchi, Hiroyuki Ohno, Srinivas Sampalli : “Design, Implementation and Performance Measurement of Raspberry Gate in the IoT Field”, Proceedings of the 2019 4th International Conference on Cloud Computing and Internet of Things, pp.82–89 (2019)

(13) 有限会社ユニバーサル・シェル・プログラミング研究所：“ユニケージ開発手法”, https://www.usp-lab.com/methodology.html （2022 年 10 月 5 日確認).

(14) Gancarz Mike：“UNIX という考え方: その設計思想と哲学”, 株式会社 オーム社（2001).

(15) Howell, Max：Homebrew, https://brew.sh/index_ja （2022 年 10 月 5 日確認).

(16) Timing Management Commands in POSIX, USP NCNT プロジェクト, https://github.com/NCNT/TimingCmds_in_POSIX （2022 年 10 月 5 日確認).

(17) 柳戸新一, 松浦智之, 鈴木裕信, 大野浩之: “シェルスクリプトを用いた UNIX 哲学に基づくリアルタイム制御”, ソフトウェア・シンポジウム 2021, （2021）.

(18) 松浦智之, 柳戸新一, 大野浩之: “UNIX 機における IoT 機器制御のためのタイミング管理”, ソフトウェア・シンポジウム 2021, （2021）.

(19) 松浦智之: “シェルスクリプトによる入出力のタイミング制御”, シェルスクリプトマガジン, Vol.76, pp.84-88, USP 出版, (2022).

(20) 大学コンソーシアム石川：いしかわシティカレッジについて, https://www.ucon-i.jp/newsite/city-college （2022 年 10 月 5 日確認).

(21) Dalhousie University：Facing an ”Internet of Things” future, https://www.dal.ca/faculty/computerscience/news-events/news/2019/03/12/facing_an__internet_of_things__future.html （2022 年 10 月 5 日確認).