JSiSE Research Report
vol.36,no.5(2022-1)

課題管理機能を有する UML プログラミング環境の設計と実装

丸山凌凱[*1]，香山瑞恵[*2]，永井孝[*3]
*1 信州大学大学院, *2 信州大学, *3 ものつくり大学

Design and implementation of a UML programming environment with task management capabilities

Ryoga Maruyama[*1], Mizue Kayama[*2], Takashi Nagai[*3]
*1 Graduate School of Science & Technology, Shinshu University
*2 Shinshu University, *3 Institute of Technologists

本研究の目的は，課題管理機能を有する UML プログラミング環境の提案である．モデリングツールには教育用途に適したものが少ない．提案環境は，状態遷移図を記述するためのモデルエディタと，モデルから対象デバイス向け実行コードを自動生成するシステムにより構成される．DSL に基づく課題設計機能と記述されたモデル図の管理機能を有することで，教育利用での実用性を意識した．本稿では，提案環境の設計と実装について述べる．

キーワード: プログラミング教育，モデル駆動開発，MDD，UML

1. はじめに

近年，小学校でのプログラミング教育の必修化や中学・高等学校での情報関係科目の内容の充実など，わが国では情報教育がより一層推進されている(1)．中学校技術科の新学習指導要領解説(2)では，「D 情報の技術」における計測・制御のプログラミングによる問題を解決する学習活動として，栽培ロボットや生活サポートロボットのモデルの開発が例示されている．その際，統一モデリング言語(Unified Modeling Language，以下 UML)や構造図の利用を指導するようになっている．このことから，中学校におけるプログラミングの学習として UML のモデル図を扱うことは適しているといえる．

一般に，システム設計におけるモデル図として UML が用いられる．UML では定められた記法に従いモデル図を記述するため，認識の齟齬が少なくなる．産業界ではモデルからプログラムを自動生成する技術であるモデル駆動開発(Model Driven Development，以下 MDD)が実用化されている．

MDD を教育用途で利用する事例は少ない(3),(4)．しかし，MDD はモデル図による設計の妥当性を視覚的に理解するのに適している．本研究では，制御対象の観察とモデル図の評価を繰り返すことにより，システム開発の本質を学ぶことができると考える．本稿では，教育用途に適した UML プログラミング環境の設計と実装について述べる．提案環境の特徴は以下の三点である．

- 学習者の記述したモデル図の管理
- プログラミング初学者に適した UX
- 課題に応じたドメイン特化言語の定義

2. MDD とドメイン特化言語

MDD とは，モデルコンパイラによりモデル図からソースコードを自動生成する技術である．MDD により，設計図と実装の乖離を無くすことが可能となる．また，コード文法の知識がなくともシステム開発が行える点もメリットといえる．

ドメイン特化言語(Domain Specific Language，以

下 DSL)とは，特定の問題領域における課題解決用に特化した語彙集合である．DSL を用いることで，解決すべき課題を整理するための語彙を制限することができる．これにより学習者に与える課題の難易度を指導者等が調節することができる．

3. 既存の MDD ツール

MDD ツールの多くは商用ベンダから提供されている．本章ではそれぞれの MDD ツールの概要と，教育利用における問題点について述べる．

3.1 BrigePoint

BrigePoint[5]とは，xtUML.org により提供されるオープンソースな MDD ツールである．変換ルールの開発を行うことで，様々なプラットフォームを対象に実行コードを生成できる．モデル図の記述においても技術者が利用するに足るような多様な機能を有しており，汎用的な MDD ツールとして利用可能である．しかし，技術者が用いる高度な機能を備えるツールを，プログラミング初学者が短時間で扱えるようになることは困難だと考えられる．

3.2 astah

astah[6]とは，Change Vision 社により提供される UML モデリングツールである．astah には，ユーザ独自のプラグインを導入するための API が用意されている．例えば m2t[7]プラグインを導入することにより，モデルからソースコードを生成することが可能となる．しかし，DSL に相当する機能のプラグインが現状無いため，課題の難易度の設定を指導者等が行えない．

3.3 clooca

clooca[8]とは，Technical Rockstars 社により提供される MDD ツールである．メタモデルとテンプレートの機能を利用し，モデルからソースコードを生成できる．メタモデルにより DSL を定義できるため，教育利用に適したモデリング環境といえる．我々はこれまでに，clooca と独自に開発したコンパイルサーバを組み合わせた S-clooca という UML プログラミング環境を 2012 年より運用してきた[9]．2018 年からは中学校技術科の正規授業へ適用されている．この中学校での運用を通して，いくつかの問題が明らかになった．例えば，中学校の場合は共通の課題をクラスの全生徒が取り組む．指導者はこれを効率的に評価したい．しかし，指導者は学習者のモデル図を確認できない．また，図の書き直しがしにくい等，プログラミング初学者に対して提供する UX が限定的であった．

4. 提案環境：SRPS

これまでの運用で明らかになった問題点を踏まえて，本研究で提案する環境の特徴は以下の 3 点である．

- 学習者の提出したモデル図の管理
- プログラミング初学者に適した UX
- 課題に応じた DSL の定義

本研究ではこれらを実現するために，新たな UML プログラミング環境：SRPS を開発した．

一般的な MDD ツールでは，UML のクラス図と状態遷移図を記述することにより，実行可能なコードを

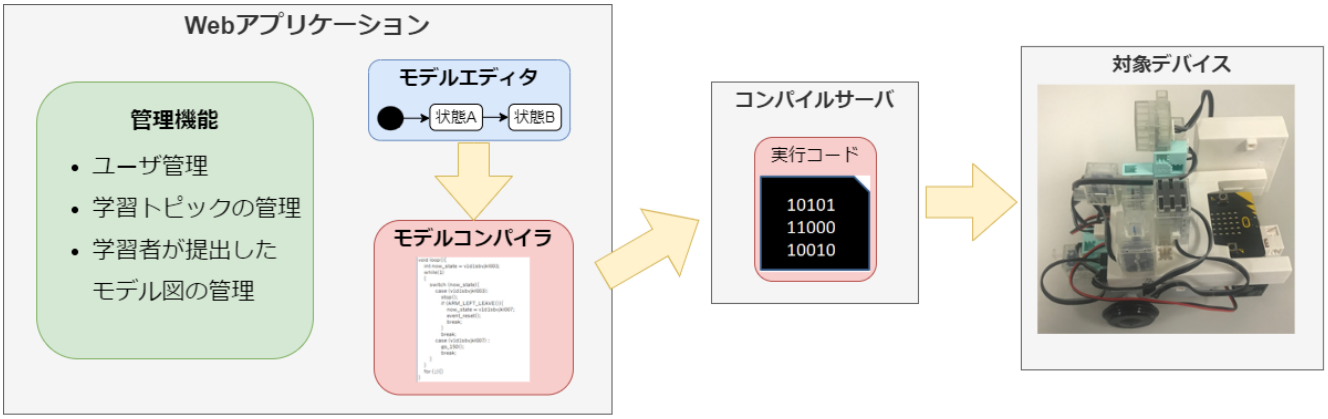


図 1 提案環境の構成

生成する．本研究では，このうち状態遷移図に着目した．扱う図の種類を限定し，学習項目を明確化することで初学者の理解を促進する．状態遷移図では，状態(□)と遷移(→)を用いてシステムの振る舞いを表現する．状態には“動作”を，遷移には“イベント”を記述する．モデル図を構成する要素は，簡単な図形のみであるため初学者による教育利用に適している．

提案環境の構成を図 1 に示す. SRPS は Web アプリケーション，コンパイルサーバ，操作対象となるデバイスで構成される. このうち Web アプリケーションを新規に設計・開発した．ここでは，学習者向けの UX を工夫し，指導者向けの学習トピック作成・提出されたモデル図の管理機能を充実させた．コンパイルサーバは S-clooca と同様のものを利用する．対象デバイスは micro:bit[10]で制御可能なものとした．

Web アプリケーションは，モデルエディタ，管理機能，モデルコンパイラからなる．本研究のモデルコンパイラは，モデルエディタで記述したモデル図から C++言語で記述されたソースコードを生成できる．モデル図の動作で指定する DSL に対応する関数が予め用意されている．イベントで指定する DSL が条件分岐となり，動作の関数を呼び出すようなテンプレートファイルを持つことにより，ソースコードを生成している．モデル図の記法が誤っている場合は不具合が生じる可能性があるため，記法チェック機能を使用した後のみソースコードの生成が行える．生成されたソースコードをコンパイルし，実行コードをダウンロードする．これを操作対象のデバイスに書き込むことで，モデル図に記述した動作を実際に確認できる．

5. Web アプリケーションの設計

本研究ではユーザを管理者，指導者，学習者の三つの役割に分類した．本章では学習者向けのモデルエディタ，管理者・指導者向けのユーザ管理，学習トピックの管理，学習者が提出したモデル図管理の各機能について設計した成果について述べる（図 2 参照）.

5.1 モデルエディタ

Web アプリケーションには，モデル図を描くためのエディタが必要となる．モデルエディタの画面を図 2(a)に示す．状態を遷移でつなぎ合わせ，状態には動作を指定し，遷移にはイベントを指定する．指定できる動作やイベントは課題毎に DSL によって定義されたものである．モデルエディタの機能を以下に示す.

5.1.1 UX の向上を意識した機能

このモデルエディタでは，フローティングポイント(図 2(a)右側の桃色二重丸)を選択するとインタラクティブなチュートリアルを表示することができる．基本的な使用法を学習者が自ら確認することで，指導者による使い方の説明を省力化できるようになる.

SRPS にはユーザによる誤操作に対応するために，操作履歴を保持する機能を設計した.「Undo/Redo」機能は，モデル図に対していつでも機能する．複数の授業日で一つのモデル図を記述する場合でも，操作履歴が保持されているため「Undo/Redo」機能が動作する.

モデルエディタの図要素にメモを記述できるものがある．メモの図要素により，指導者が学習者の回答を確認する時に，モデル図の設計意図などを知ることで適切な評価が行える.

5.1.2 モデル図の記法チェック機能

モデル図の記法誤りは，モデルエディタ上で確認できる．記法誤りがある箇所はダイアログで表示され，該当要素が赤くなる．記法チェックにより全ての記法誤りが修正されたモデル図のみ，モデルコンパイラの機能を実行することができる．モデル図の記法チェック機能により，不適切なソースコードが教材ロボットに送られてしまうことを事前に防ぐことに加え，学習者自身でモデル図の誤りを訂正することを支援する.

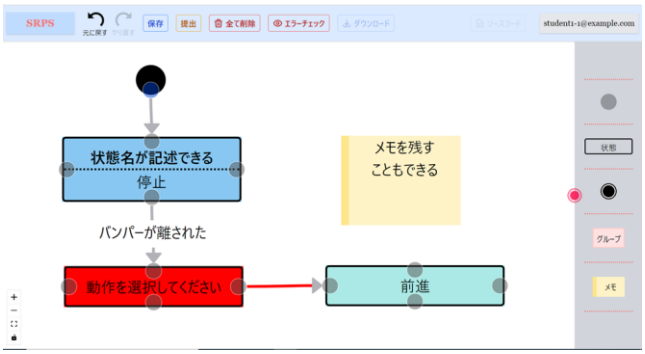
5.1.3 モデル図の保存・提出機能

記述したモデル図は保存・提出ができる．モデル図の保存では, 学習者と課題毎にモデル図の保存を行う. モデル図を記述し保存をしておくと，モデル図を記述したユーザで保存を行った課題を選択したときに，前回の続きから再開することができる.

提出機能では，指導者による「学習者が提出したモデル図の管理」(5.5 で後述)の対象となるモデル図を保存する．モデル図の提出は複数回行うことができる.

5.2 ユーザ管理

SRPS のユーザはメールアドレス，パスワード，グループ(例えば, 学校でのクラスに相当), 役割により区



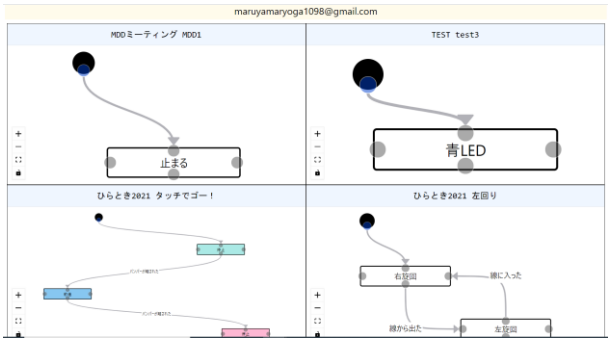
(a) モデルエディタ画面

(b) ユーザ管理画面

(c) DSL 作成画面

(d) 学習トピック割り当て画面

(e) 回答の確認画面



(f) 一覧表示画面

図 2 SRPS 画面群

別される．役割は Web アプリケーションの機能や権限に関係する．ユーザ管理画面を図 2(b)に示す．SRPS では管理者は admin，指導者は teacher，学習者は student として管理している．

teacher は複数のグループに対応し，そのグループに所属する student の管理，「学習トピックの作成」，「学習トピックの割り当て」，「回答の確認」の機能が使える．student は一つのグループに対応し，モデル図の記述や提出の機能が使える．また，admin は全ユーザに対して teacher と同じ機能が使える．

teacher の場合は自分が管理するグループを登録した後，そのグループに所属する student を登録していく．同じグループを管理する teacher を登録することも可能である．登録は一人ずつ追加する方法と，csv ファイルにより一括で追加する方法がある．一人ずつ追加する場合はメールアドレス，パスワード，グループ，役割を指定する．一括で追加する場合は，student のみに対して利用できる．また，admin の場合は全ての役割のユーザを登録できる．

5.3 学習トピックの管理

5.3.1 学習トピックと学習課題の管理

学習トピックは複数の課題により構成される．特定のグループに与える課題の集合名であり，例えば，「交通事故を減らせ」等の学習テーマ名や「サマーキャンプ」等のイベントの名前等が相当する．DSL 作成画面を図 2(c)に示す．課題に名前を付け，その課題で使用する動作やイベントを選択し，名前を付けることで

DSL を作成する．図 2(c)の例では，レース課題においては，赤外線センサの黒色検知を「ゴールに着いた」，タッチセンサのバンパーが押されたというイベントを「スタートボタンが押された」という語彙で利用する．一方，障害物検知課題においては，赤外線センサは利用しない．タッチセンサのバンパーが押されたというイベントを「障害物に触れた」という語彙で利用する．必要な課題を作成後，SRPS に登録することで学習トピックが作成される．

学習トピックは同じ学校の teacher グループ間での共有や．特定の teacher に共有したい場合がある．SRPS では柔軟に学習トピックの公開・非公開が設定できる．公開に設定した場合は全ての teacher が利用可能な学習トピックとなる．特定の teacher に公開したい場合は，学習トピックの設定ファイル(JSON 形式)を共有する．なお，admin の学習トピックは必ず公開の設定となる．

5.3.2 学習のトピックの割り当て

学習トピックに対し，割り当てるグループを指定することでそのグループに属する student が課題を利用できるようになる．学習トピック割り当て画面を図 2(d)に示す．割り当てが行える学習トピックは，「自分が作成した課題」，「公開されている課題」，「管理者が作成した課題」の三種類である．

5.4 学習者が提出したモデル図の管理

admin と teacher は student が提出したモデル図(以下，提出図)の確認を行える．提出図の確認画面を図 2(e)に示す．SRPS では，提出図を課題毎またはユーザ毎に管理する．提出図の履歴を保持しており，各提出図の参照と編集ができる．提出図の確認方法は，一覧表示と個別表示がある．一覧表示の画面例を図 2(f)に示す．特定の課題について，グループ全体の提出図を一度に確認したい場合や，特定の student の提出図を一度に確認したい場合は一覧表示が適している．一方，特定の student の特定の課題の提出履歴を遡りたい場合や一つの提出図を詳しく確認したい場合は個別表示が適している．提出図の確認では，提出履歴を参照しない場合は最新の提出図が確認対象となる．

6. Web アプリケーションの実装

6.1 フロントエンド

開発には，JavaScript 用ライブラリである React のフレームワーク Next.js[11]を用いた．Web アプリケーションは Vercel[12]によりホスティングした．開発言語には TypeScript を使用した．図を記述するためのライ

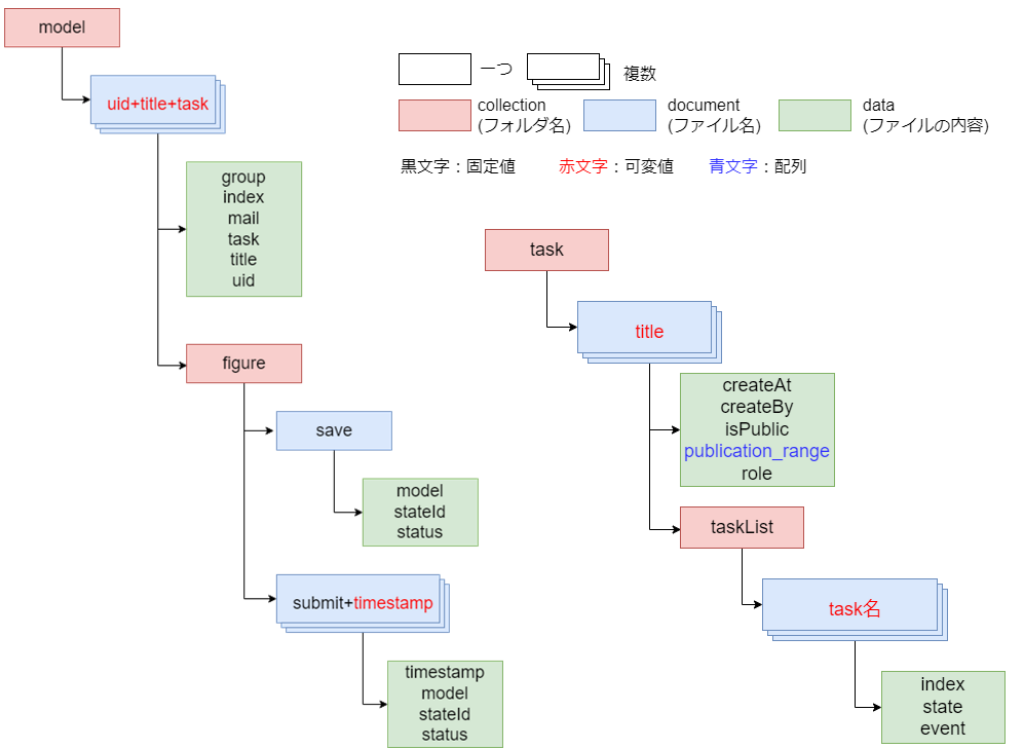


図 3 データモデル

ブラリとして React Flow[13]を利用した.

6.2 バックエンド

SRPS では，モデル図から C++のソースコードを生成する．このコードを，我々が独自開発したコンパイルサーバ[14]に POST メソッドのリクエストで送る. そして，コンパイルサーバのレスポンスとして対象デバイス用の実行コードをユーザ端末にダウンロードする.

Web アプリケーションの各種機能は Firebase[15]を利用することにより実現した．認証は Firebase Authentication により行い，データベースには Cloud Firestore を利用した. 認証機能は Node.js のフレームワークである Express により作成した REST API を Cloud Functions にデプロイすることで提供する．モデル図と課題を管理するデータモデルは図 3 に示す構造となっている．ここでは，提供されている学習トピックと学習者が保存, 提出したモデル図が格納される.

7. 評価

SRPS の機能評価とユーザビリティを予備評価するために，2 種のユースケースを設けた.

7.1 モデルエディタとモデルコンパイラ

■小学生向けワークショップでの機能評価

SRPS を利用して，小学校 5 年生 9 名・6 年生 11 名を対象としたワークショップを実施した．ここでの学習トピックは「チョロ Q」とし，2 つの DC モータとタッチセンサ, 赤外線センサを主として利用した(課題用の DSL は省略).

参加者は簡単な利用説明のみでモデルエディタの使い方を理解した．マウスによる図要素の選択やエラーチェック機能を用い，参加者全員が与えた全課題をクリアしていた．このことから，本環境は小学生が利用可能な UX を提供していたことが示唆された.

■大学生対象の使用感調査実験

大学生 4 名・大学院生 5 名を対象に，モデルエディタの UX に関する質問紙調査(5 段階評価[1:最も使いやすい，5:最も使いにくい]と自由記述)を実施した．これらの被験者は全員，S-clooca の利用経験を有する．ここでの評価項目は以下の 3 点と S-clooca との比較とした.

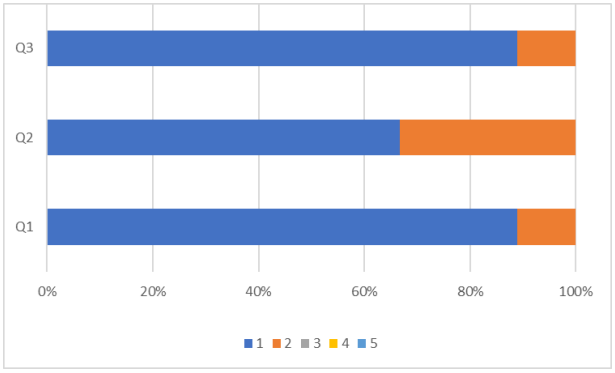


図 4 モデルエディタの UX に関する調査結果

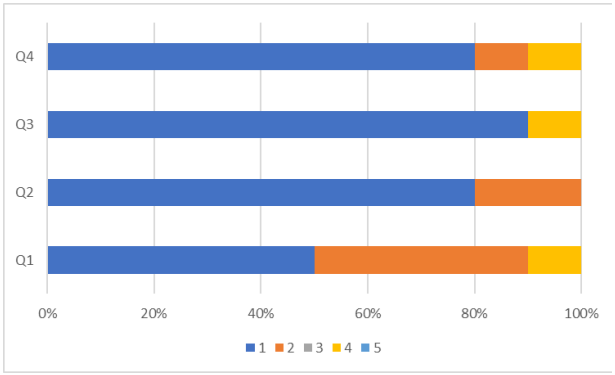


図 5 管理機能質問紙調査の結果

Q1：ドラッグ＆ドロップでの操作の使いやすさ

Q2：イベントや動作の選択画面の使いやすさ

Q3：全図要素の削除，Undo，Redo などの SRPS の機能の使いやすさ

これら 3 項目に対する回答を図 4 に示す．各項目の平均値は Q1=1.11，Q2=1.33，Q3=1.11 であり，全てで 1.5 以下であった．この結果からこの予備調査では SRPS のユーザビリティは高い評価を得たと考える.

S-clooca との比較の自由記述では，全ての回答で SRPS が優位に評価された．具体的には，「状態・遷移の記述がしやすい」(4 名)，「チュートリアル機能があるのが良い」(2 名)ことが評価された.

これらの結果から，SRPS のモデルエディタの UX は良い評価を得たと考える.

7.2 管理機能

大学生 4 名・大学院生 6 名を対象に，SRPS の管理機能に関する質問紙調査(5 段階評価[1:最も使いやすい，5:最も使いにくい])を実施した．ここでの評価項目は以下の 4 点とした.

Q1：ユーザの追加しやすさ

Q2：学習トピックの作成しやすさ(DSL 定義含む)

Q3：学習トピックの割り当てやすさ

Q4：提出物の確認しやすさ

これら 4 項目に対する回答を図 5 に示す．各項目の平均値は Q1=1.7，Q2=1.2，Q3=1.3，Q4=1.4 であり，全てにおいて 2 以下であった．この結果から，Q2 に関連する課題に応じた DSL の定義，Q3 に関連する学習者に応じた課題の割り当て，Q4 に関連する学習者の記述したモデル図の管理については，良い評価を得た．Q1 に関連する機能については，登録したユーザの並び替え表示等の改良を検討する．また，特に，Q3 と Q4 で「4」と評価した被験者からは，「課題の共有，非共有が分かりづらいと感じた」等のコメントを得た．この点の改良も今後の課題となる．

8. おわりに

本稿では SRPS の設計と実装，機能とユーザビリティの評価について述べた．

今後は，モデルコンパイラの汎用化を図ることで，micro:bit 以外の制御デバイスにも対応したい．これにより，より教育現場からの多様なニーズに適応的な学習活動が提供できると考える．また，SRPS での操作履歴保持の機能を利用することで，UML プログラミングの学習データによる「ラーニングアナリティクス」や，モデル図を記述する過程を分析することで新たな知見を得る「プロセスマイニング」への応用が考えられる．蓄積されたデータから様々な解析が行えるよう，データを管理する仕組みを設計・構築していく．

謝辞

本研究は科研費基盤研究 B:16H03074 の支援を受けた．

参 考 文 献

(1) 文部科学省，“情報教育の推進”，https://www.mext.go.jp/a_menu/shotou/zyouhou/detail/1369613.htm(2021 年 11 月 21 日確認)

(2) 文部科学省，“【技術・家庭編】中学校学習指導要領(平成 29 年告示)解説”，https://www.mext.go.jp/component/a_menu/education/micro_detail/__icsFiles/afieldfile/2019/03/18/1387018_009.pdf(2021 年 11 月 21 日確認)．

(3) C. Starrett. “Teaching UML Modeling Before Programming at the High School Level”, Proc. of the 7th IEEE International Conference on Advanced Learning Technologies, 2007, pp.713-714.

(4) Seiko Akayama, Shin Kuboaki, Kenji Hisazumi, Takao Futagami, and Teruaki Kitasuka. "Development of a modeling education program for novices using model-driven development". Workshop on Embedded and Cyber-Physical Systems Education, 2012, Article 4, pp.1–8.

(5) xtUML.org : “BrigePoint”，https://xtuml.org/(2021 年 11 月 21 日確認)

(6) Change Vision : “ astah ”， https://astah.change-vision.com/ja/(2021 年 11 月 21 日確認)

(7) Change Vision : “モデル駆動開発 m2t(Model to Text) プラグイン”，https://astah.change-vision.com/ja/plugin/astahm2t.md-plugin.html(2021 年 11 月 24 日確認)

(8) Shuhei Hiya, Kenji Hisazumi, Akira Fukuda, and Tsuneo Nakanishi. "clooca : Web based tool for Domain Specific Modeling". ACM/IEEE 16th International Conference on Model Driven Engineering Languages and Systems, 2013, pp.31-35.

(9) 香山 瑞恵，小形 真平，永井 孝: “モデル駆動開発方法論に基づく UML プログラミング教育環境”，教育システム情報学会論文誌, 2019，第 36 巻, 2 号, pp.118-130.

(10) BBC，“micro:bit”，https://microbit.org/(2021 年 11 月 21 日確認)

(11) Vercel，“Next.js”，https://nextjs.org/(2021 年 11 月 21 日確認)

(12) Vercel, “Vercel”, https://vercel.com/docs(2021 年 11 月 21 日確認)

(13) webkid，“React Flow”，https://reactflow.dev/(2021 年 11 月 21 日確認)

(14) 大宅剛生，香山瑞恵，永井孝 : “コンテナ型仮想化によるモデリング教育向けコンパイルサーバでの micro:bit 対応機能の評価”, 第 46 回教育システム情報学会全国大会講演論文集，2021，pp.107-108.

(15) Google, ”firebase”， https://firebase.google.com/?hl=ja (2021 年 11 月 21 日確認)