特集　情報システム

研究ノート

プログラミング学習に向けた ビデオからの手順とリソース抽出に基づく教育手法

營田茂生*・山口崇志*・狩野達哉*・
岸本頼紀*・布広永示*

要旨：システム設計、プログラミング、プロジェクト管理などの情報システム開発の基本的なスキルは、一般的な問題解決の基本と非常に似ている。本論文では現実問題のモデル化と、システム設計技術に基づく実用的な問題解決のための教育フレームワークと、動画教材に関するフレームワークの適用を提案した。このフレームワークは、Resource Flow Diagram（RFD）を使用して、問題解決の手順とリソースの理解をサポートする。RFDは統合モデリング言語（UML）のシーケンス図に基づく手続きと、リソース管理のための提案された視覚化手法である。UML は単一の図でリソースの使用や占有を定義することができない一方で、RFD は手続きフローと必要なリソースを直感的に表現できるように設計されている。本実験では、料理の調理動画から調理手順を理解するための教育フレームワークを提案した。

キーワード：e-Learning，システム設計，要件定義，関数型プログラミング，動画教材による学習

Education Framework Based on Flow and Resource Extraction for Programming Learning

Shigeo TSUKUTA*, Takashi YAMAGUCHI*, Tatsuya KANO*,
Yorinori KISHIMOTO* and Eiji NUNOHIRO*

Abstract: The fundamental skills for information system development such as system designing, programming and project management are very similar to the fundamentals of general problem-solving. In this paper, we proposed an education framework for practical problem-solving based on system designing technologies and an application of proposed framework on video training materials in order to train the skills of modeling and understanding from ambiguous matter in practical problem via non-verbalized video training material. Our framework uses Resource Flow Diagram（RFD）to support the understanding of procedure and resources on problem-solving. RFD is our proposed visualization method for procedure and resource management based on Sequence Diagram in Unified Modeling Language（UML）. RFD is designed for the intuitive representation of the procedure flow and required resources since UML could not define them with single diagram. In this experiment, proposed education framework was applied for the understanding of cooking procedure from the cooking exhibition videos on the demonstrative lecture.

Keywords: e-Learning, System design, Requirement definition, Functional programming, Education by video

*東京情報大学　総合情報学部
Faculty of Informatics, Tokyo University of Information Sciences

2018年５月15日受付
2018年９月18日受理

1．はじめに

近年、ネットワーク化された情報化社会の進展により、高度な知識とスキルの必要性が高まっているため、高度に熟練した人材教育が非常に重要である。システム設計、プログラミング、プロジェクト管理などの情報システム開発の基本的なスキルは、一般的な問題解決の基本に非常に似ている。これに伴い、情報学のための小学校教育の方法と応用が日本で議論されている。

本稿では、動画から現実問題のモデルを抽出し、問題解決するためのスキルを訓練するため、システム設計技術に基づく実用的な問題解決のための教育フレームワークと、動画教材に関するフレームワークの適用を提案した。提案したフレームワークは、Resource Flow Diagram（RFD）を使用して、問題解決の手続きとリソース遷移の理解をサポートする。ここで言うリソースとは素材や道具等、ある問題解決の手続きにおいて必要とされる資源やインフラストラクチャーなどを指し、RFDはUML[1][2][5][6]のシーケンス図に基づく手続きと資源管理のための可視化手法である。UMLはソフトウェア開発において設計を図示するための記述形式を定義したもので、手続きやモジュールの依存性等を表現することができる。一方で、ソフトウェアの実装であるプログラミングの学習においては、手続きの読み取りに加え、手続きによるデータの変化を捉えることが重要であり、UMLではこれらデータの変化を表現するのが難しいことから、RFDではUMLのシーケンス図を拡張しリソースであるデータの変化や保持が必要な期間等を明示できたるようにした。

実際の応用では、調理を課題とした模擬講義を実施した。調理はソフトウェアプログラムと同様にレシピに記載された手順に従い素材を変化させることから、プログラミングに必要とされる手順を明確化する能力を育成できる。さらに、プログラム内のデータに相当する素材が明確にモノとして変化を観て触ることができることから、初学者にとってより直観的な理解が得られると期待できる。ここでは調理レシピ動画を、調理手順を理解し必要となるリソースやインフラストラクチャーを抽出するための教材として提供し、調理手順のRFDを作成する演習により試行錯誤のし易さ、学生の躓くポイントや提案する教育モデルの課題を検証するための模擬講義を行った。調理レシピ動画では素材と主要な道具と調理手順が示される。実際の調理では、調理台やガスレンジなどのインフラストラクチャーの競合を回避するリソース管理となるが、実際に調理する場面で必要となるものの、動画作成時と調理者では異なる可能性が大きいインフラストラクチャーについて、調理レシピ動画では明示されていない。また、加工した素材を入れておくボウルなど、レシピに記載されない道具なども必要となる場合があるが、それらも明示されない。学習者は実際の生活の中で起こりうる他の問題と同じように、インフラストラクチャーの競合や二次的に使用する道具などが明示されない調理レシピ動画から、調理手順とリソースを明確にする。提案方法は、学習後に実技で試行錯誤した結果を確認ができることから、試行錯誤の結果得られた知識がより定着し易い学習手法である。近年のプログラミングやシステム開発においては必要な知識の範囲が広く、このような実技による確認を伴った演習が難しい。これらを筆記テスト等により実技を伴わない方法で学習する場合、明記されていない手順やリソースへの知識が不足しているのか、主たる学習範囲の知識が記憶できない、あるいは記憶違いであったのかの判別が難しく実践的な知識として定着しにくい。

2．提案手法

2．1　動画からのモデリング教育

実世界における一般的な調理では3つの要素に分けることができる。第一の要素は選別と成型、調味、加熱、盛り付けなどの行為である。第二の要素は肉や野菜などの素材、調味料などの材料である。第三の要素は包丁、まな板、鍋やフライパン、電子レンジといったツールやインフラストラクチャーである。これらは入出力を定義しプログラムライブラリによって人間の意図する処理を行うよう、コンピュータに指示を与えることと同様である。本研究では、情報システムの設計、開発、プログラミングに適用できる問題解決スキルを向上させるために、これらの課題の解決方法を動画から抽出することに焦点を当てる。

布広ら[3]が検討した初期の研究では、問題解決方法の抽出は文章からであり、動画からの抽出とは

異なる。動画コンテンツは現実の問題に似ているため、実践的な問題解決のスキルを向上するための訓練として効果的である。

近年eラーンニング等で一般的に用いられている動画教材は、オンラインでの講義の役割を目指しており、実際の講義と同様、学びのテーマに沿って解説する内容となっていることが多い。しかしながら、従来のeラーンニングシステムでは学習者からの質問等へリアルタイムで応答するのが難しく、理解のし易さは動画の品質によるところが多い。その一方で、提案する教育モデルにおける動画教材の要件は、非言語情報からのリソースや手順の抽出であり、動画を使用した一般的な教育モデルと比較して直感的な理解への期待は重要ではなく、例えば調理の過程を撮影した動画等を未編集で利用することもできる。

教育の概要を図１に示す。まず、教師や教育システムが動画からの課題の目標を提示する。その後、学生は動画視聴を反復して行い、リソースやツールなどのパーツ定義、フロー定義、自己の論理検証などを行う。教師や教育システムは、この反復的な思考をサポートする。調理レシピ動画の例では、一般的に自明とされる調理方法やリソース、ツールの名称や、調理タスクについては言語的に明示されない。図２に使用する道具が言語的に明示されていない動画の例を示す。この調理レシピ動画では、合わせ調味料を作成するために使用するボウルについては、調理レシピ内でも言語的に明示されず、添付されている文章のレシピでも言語的に明示されていない。合わせ調味料を作成する際、調味料を合わせる器については、調理レシピ動画の制作者は、一般的な技能と知見を持つ調理者にとって、自明と考えたものと想定できる。このような非言語情報からのリソースや手順の抽出を行った後は、実技により自らの問題解決方法抽出の検証を行うことが望ましい。

この教育の流れにおいて最も重要な点は、問題解決方法の抽出に関する議論のため、反復的な思考をサポートする必要がある。本稿では、情報システム設計技術に基づく反復的思考のための非言語的支援ツールを検討する。

２．２　Resource Flow Diagram（RFD）

本稿では、「何が必要なのか」、「行う方法」、「何がもたらされるか」を時間的な尺度と共に簡潔な図形で表記し、これらの配置と組み合わせにより「いつ行うか」および「準備するか」を直観的に表現する手法としてResource Flow Diagram（RFD）を提案した。RFDは人間によるグループワークや情報システムのモジュール間連携のためのスケジュール管

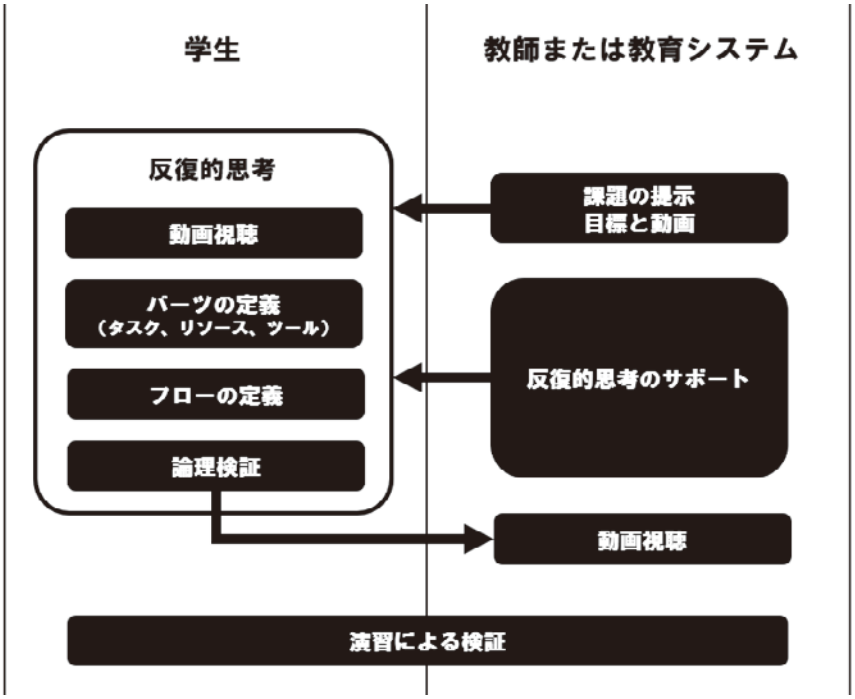

図１　動画を使った教育モデルの概要

図２　使用する道具が言語的に明示されていない動画の例

理、ソフトウェアプログラム等の振る舞いを、手続きだけでなくその手続きに必要な資源と共に可視化することができる。

RFDのオブジェクトは、図３に示すリソース、ツール、タスクで構成されている。タスクは処理や手順を示すオブジェクトでリソースによる入力と出力のデータ型定義に加え、タスクの処理を抽象化した機能型の定義を持つ。タスクは関数型プログラミング[4]の関数として捉えることができ、タスク内で必要なデータは入力として定義する必要がある。リソースは一般的な手続き型プログラミング言語の変数や定数に相当し、タスクの入出力のデータ型を定義するために使用されるオブジェクトであるが、なんらかの値を保持する関数と考えることができる。リソースには、料理中の食材の一時保管のためのテーブルや冷蔵庫、およびソフトウェアプログラムにおけるグローバル変数やクラス変数のためのメモリ領域のようなリソースを保持するための関数型もあることに留意が必要である。次にツールはタスクを入力として取る関数で、タスクの実行に必要な機能型の定義を持つ。ツールはタスクを実行する主体を表現するオブジェクトであり、例えば料理におけるコンロやナイフ、情報システムにおけるサーバモジュール、ソフトウェアプログラムにおけるコンピュータやプロセッサに相当する。

RFDの可視化はUML[5]のシーケンス図に基づいているが、RFDの各オブジェクトは図３のグラフィックオブジェクトと動画で表現されると共に、図４のようにタイムラインでのリソース管理に厳格である点でUMLのシーケンス図と異なる。UMLのシーケンス図はモジュール間でのメッセージのやり取りのタイミングを記述するのに対し、RFDはモジュール間でやり取りするリソースの型や保持期間も併せて記述する。リソースとファンクションは、各タイプに対応するアイコンによって定義される。さらに、説明はできる限り文章を排して画像または動画として提示される。RFDオブジェクトの大きさは、実施時間による横方向とリソース占有の縦方向に合わせて表現する。ただし、本実験では学生の理解し易さを考慮し、ツールにおけるリソース占有率を考慮せず単純化している。なお、リソース定義をメッセージ、リソースをリソース保持用モジュールの実行仕様、タスクを実行仕様、ツールをライフラインとすると、UMLのシーケンス図によって時間や占有率の要素を除いた振る舞いを表現することが可能である。

タスクの手順は、スイムレーン[6]のツールを使用したシーケンス図のように書かれている。リソースとタスクはツールのレーンに配置され、タスクは手順に対応する線で接続される。例えば図４では、ピリ辛きゅうりという料理のレシピと調理手順をRFDで記述している。ピリ辛きゅうりはラー油入りのピリ辛ダレにきゅうりを漬け込んだ漬物である。めんつゆを加えて旨味を足している点がレシピ

の特徴である。図４のように料理の手順を記述する場合、料理に必要な機材や容器をツールとし、各種素材をリソースとする。タスクの定義では素材を変化させる処理と考え、例えばきゅうりと加工済みきゅうりのように元の素材とその加工済み素材を別のリソース型として定義することに注意する。ツールの機能型と配置されたリソースおよびタスクの機能型は一致する必要があり、リソースとタスクの接続ではリソース型はお互いに一致する必要がある。

このようにRFDによってモデリングすることにより、「何が必要なのか」、「行う方法」、「何がもたらされるか」を時間的な尺度と共に簡潔な図形で表記し、これらの配置と組み合わせにより「いつ行うか」および「準備するか」をリソース型や機能型を直観的な図形で視覚化することで、ユーザはこの図形が一致するよう組み合わせるなど、直観的な操作によって問題解決のための試行錯誤をすることができる。

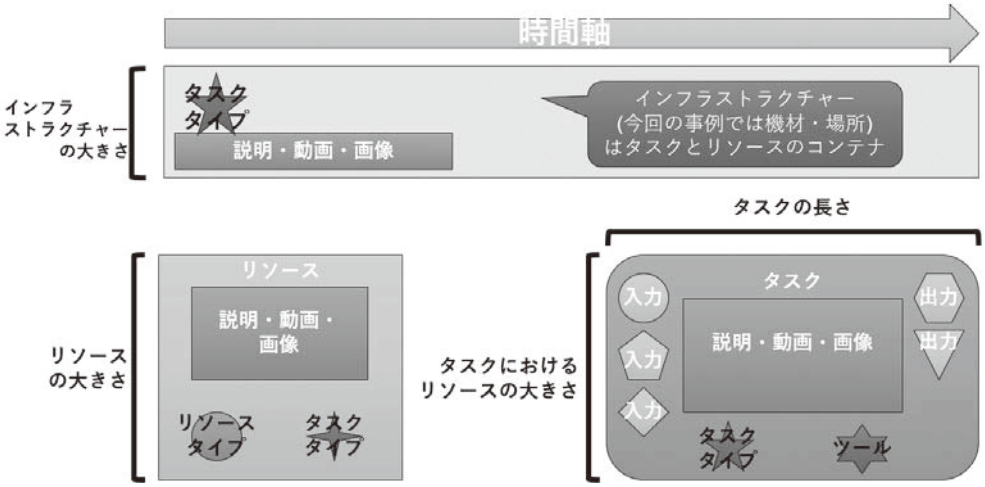

図３　RFDのオブジェクト

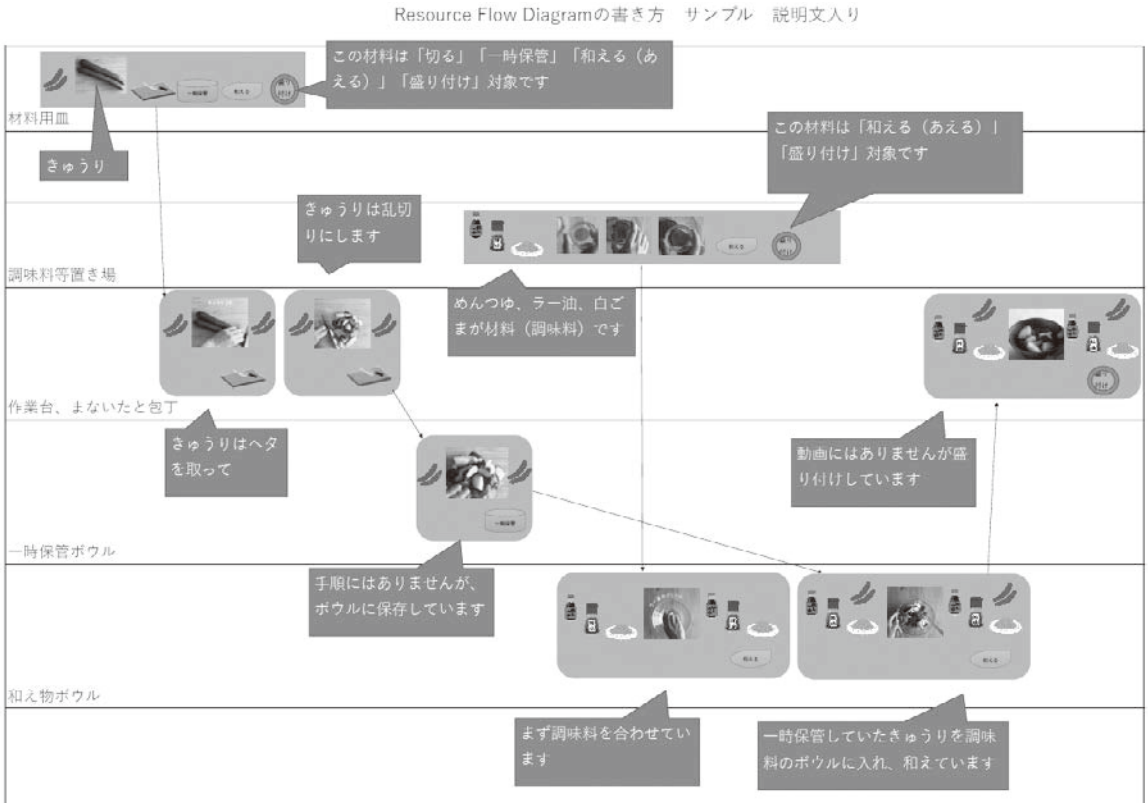

図４　RFDの例

3．実　　験

3．1　実験方法

基礎実験として、試行錯誤のし易さ、学生の躓くポイントや提案する教育モデルの課題を検証するために、演習に提案された教育フレームワークを適用した。この実験では、調理レシピ動画をモデリング対象として使用する。料理は作業の順序性が厳格であり、リソースの競合／非競合と時間の管理が必要である。「何が必要なのか」、「行う方法」、「何がもたらされるか」を時間的な尺度と共に簡潔な図形で表記し、これらの配置と組み合わせにより「いつ行うか」および「準備するか」を明確化する。

表１に実証実験のタイムテーブルを示す。この実証実験では評価のため、演習のための準備と実際にRFDを作成する演習を90分間連続して行う。この実証実験の準備として、RFDの書き方と使い方について説明する。実証実験に参加した学生は３年生と４年生で構成されている。学生の大半は、システム開発とプログラミングの知識があり、毎日料理をしていない。

実験のため、我々は調理レシピ動画と演習の目標について書かれたドキュメントをWebで準備した。調理レシピ動画では、時系列にタスクが表示され、タスクごとに使用するインフラストラクチャーやリソース、ツールが明示される。

この演習の目標は、３種類の調理レシピ動画で示された調理方法の組み合わせから、与えられた３種

表１　実証実験のタイムテーブル

開始時間	実施内容	実施内容
18：15	導入	演習の目的と全体的な流れを説明する
18：18	講義	タスク、リソース、インフラストラクチャー等、RFDの記述方法について
18：28	演習１	３つの調理レシピ動画を見て、それらとは異なるレシピのRFDを作成する
18：58	演習１の解説	教師側が事前作成した回答例に基づき、調理レシピ動画から抽出したポイントの説明
19：03	演習２	３つの調理レシピ動画を見て、それらとは異なるレシピのRFDを作成する
19：33	アンケート	アンケート

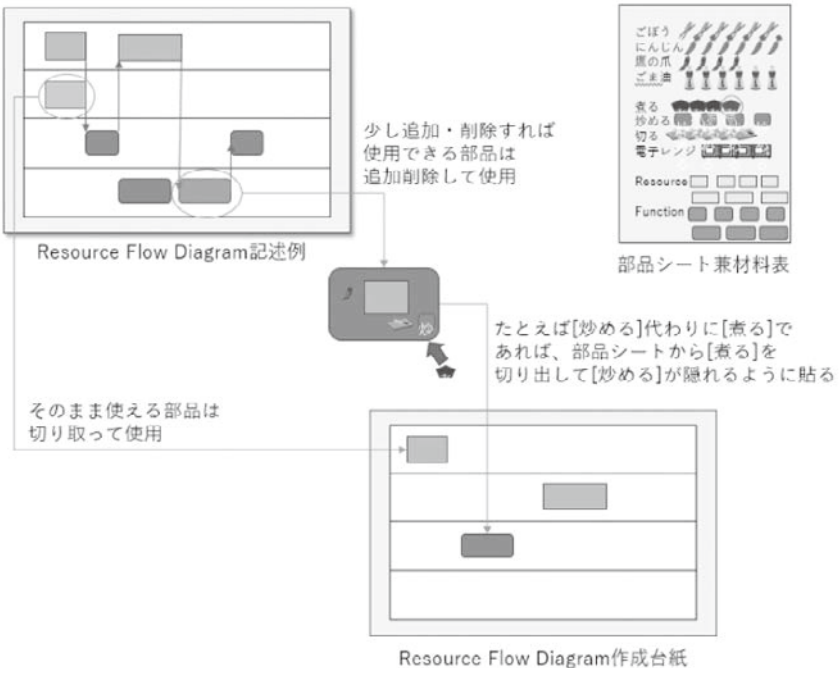

図５　演習時のRFD記述方法の概要

類の料理の調理手順の一部を構成要素とする異なる料理のRFDを作成することである。調理レシピ動画の調理手順の説明については、RFDが使用される。学生は問題を解決するために、Web上に準備した調理動画と課題と、印刷して配布した３種類の料理ごとのRFDを参照する。図４は、この演習問題のRFD記述方法の概要である。RFDはタスク、リソース、インフラストラクチャーの個々の部分と、それらを組み合わされた回答として、RFDに使用することができる。学生は３種類の料理ごとのRFDと部品シートから解答RFDとして使用できるタスク、リソースなどを切り出し、解答用シートに貼り付けて回答を作成する。

演習１ではごぼうの肉巻きの手順作成を出題した。肉巻きは日本の家庭料理である。ごぼうの肉巻きの手順を考えるために、以下の３つの調理レシピ動画と調理レシピのRFDを提示した。きんぴらはごぼうなどを使った和風炒め物である。学生は、これらの動画から肉巻きの一般的な手順とごぼうの前処理を知ることができる。

- ●アスパラガスの肉巻き
- ●オクラの肉巻き
- ●ごぼうと人参のきんぴら

演習１の後、問題解決のためのRFDの応用を理解するために、演習１の解答例RFDを用い、問題解決のための思考プロセスを説明した。

演習１におけるRFDを記述するためのパーツの定義は下記の通りである。

（１）タスク

- ・ごぼうは長さ1/4に切り、さらに十字に４分する（1/16になる）。
- ・ごぼうを器に入れ水にさらす。
- ・ごぼうの水を切る。
- ・ごぼうを器に入れラップし電子レンジで加熱する。
- ・豚バラ肉は半分に切る。
- ・豚バラ肉で1/16にカットしたうちの４本を巻く。
- ・バットに豚バラ肉でごぼうを巻いたものを並べる。
- ・茶こしで片栗粉を振る。
- ・フライパンにごま油をひく。
- ・フライパンを加熱し、豚バラ肉でごぼうを巻いたものを並べ加熱する。
- ・例示した他の肉巻きと同じ調味料を順次投入する。
- ・調味料を煮詰める。
- ・皿に盛りつける。

（２）リソース

- ・ごぼう
- ・水
- ・ラップ
- ・豚バラ肉
- ・片栗粉
- ・ごま油
- ・調味料（しょうゆ、酒、みりん、はちみつ）

（３）ツール

- ・材料を置くための皿
- ・肉トレイ
- ・調味料等置き場
- ・作業台
- ・まないた
- ・包丁
- ・電子レンジ
- ・水を張ったボウル
- ・水切り
- ・茶こし
- ・IHクッキングヒーター
- ・フライパン

ここで示したタスクを元にすると、演習１のごぼうの肉巻きのフローは図６のように考えることができる。一方でRFDでは図７のように考えることができる。RFDは一般的なフロー図と異なりリソースの依存関係を考慮しながら手続きの並列性等への気づきを促すことが期待できる。学生はフローを定義しRFDのスイムレーンに配置することで、視覚的に自らが記述した論理を検証することが可能である。与えられた出題内容に対して問題が解決できているか、過不足がないかは、動画を繰り返し視聴することで反復的に確認が可能である。

演習２では、親子丼の調理手順作成を出題した。親子丼は鶏肉と卵を使ったどんぶりである。親子丼の手順を考えるため、演習１と同様に３つの調理レシピ動画と調理レシピのRFDを提示した。学生はこれらの動画から卵と鶏の調理方法を知ることができる。

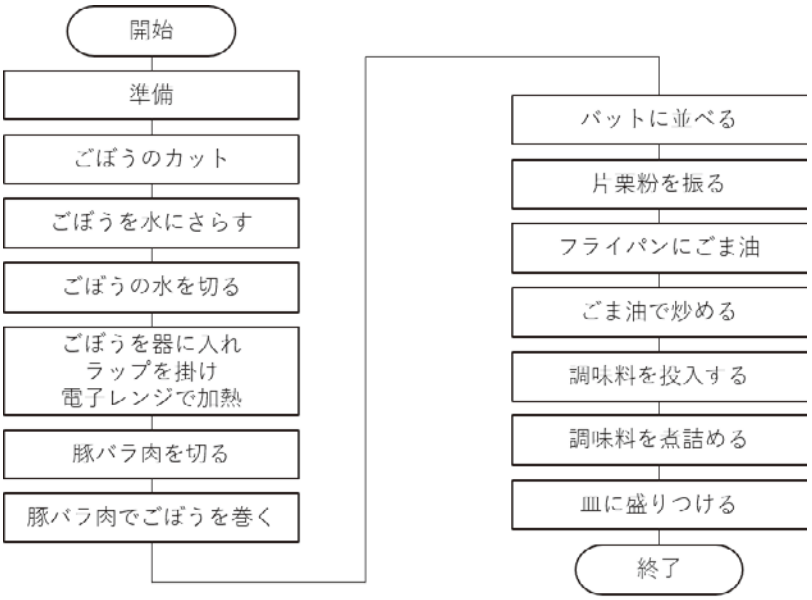

図６　ごぼうの肉巻きのフロー図の例

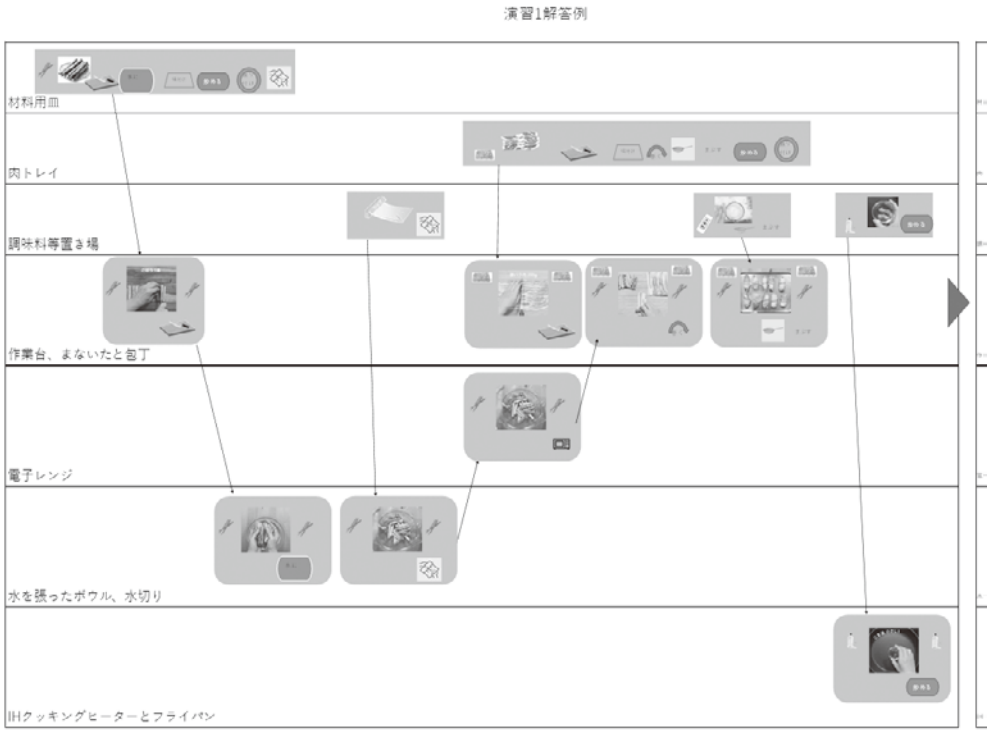

図７　演習１解答例として提示したごぼうの肉巻きのRFDの例

●ねぎと鶏もも肉のスープ
●高野豆腐の卵とじ
●お惣菜とんかつでかつ丼

演習２の後、学生は親子丼の調理レシピ動画を解説なしで視聴し、自分のRFDを確認する。演習２の後のアンケートの結果から、提案の有効性を検討する。

３.２　演習実験の結果

学生の理解と傾向を検証するために、演習１と２のRFDを評価し、学生を分類した。表２に得点と評価の結果を示す。採点に当たっては100点満点とし、設問に対する正解RFDとして記載が必要なタスクとインフラストラクチャー、リソース、フローとしてのタスクの記載順を全て列挙し、100で除算して１項目の得点とした。

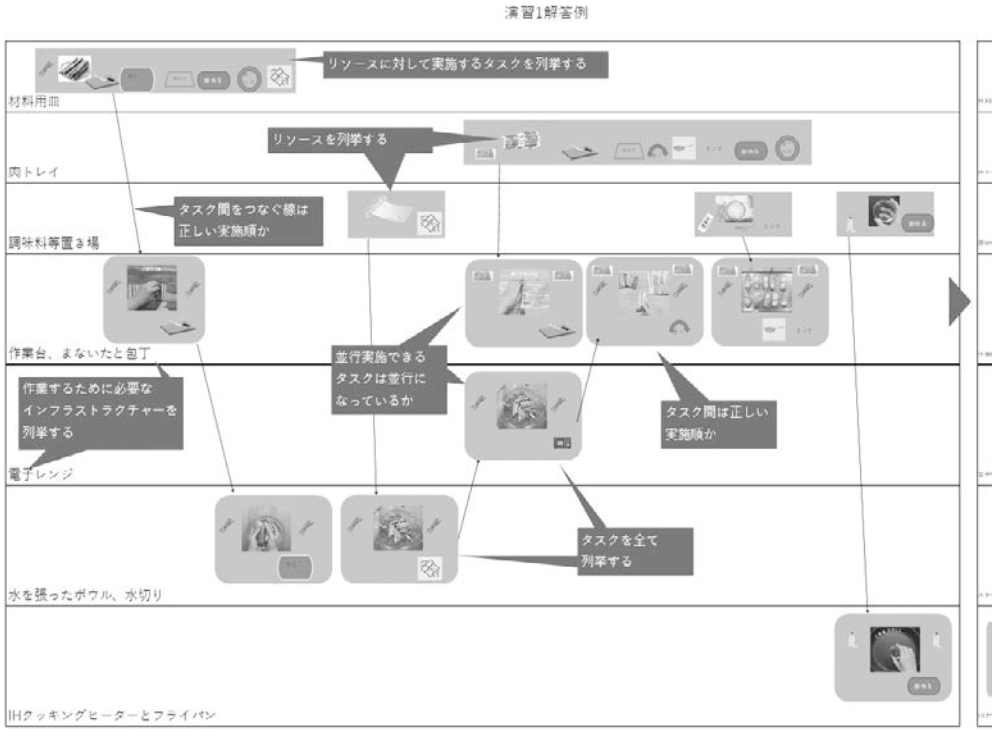

図８　採点基準

図８は図７の演習１解答例前半に採点基準を書き入れた図である。

今回の実証実験では、演習１ではRFDを学んでから作成までの時間が少ないことを考慮し、演習１と２の合計得点または演習１と２の間の得点増分に基づいて分類する。評価Aの合計得点は120より高い者とした。評価Bは合計得点50点以上または増分30点以上、評価Cは評価Bに及ばない者である。

この分類では、評価Aは２人、評価Bは９人、評価Cは７人となった。評価Cは、この演習方法では問題解決を理解することができない学生、または動機が少ない学生と考える。一方、評価Bの学生は、

表２　演習１、演習２の結果

学生	演習1	演習2	合計	増分	評価
学生1	20	70	90	50	B
学生2	5	20	25	15	C
学生3	10	80	90	70	B
学生4	0	15	15	15	C
学生5	50	75	125	25	A
学生6	50	55	105	5	B
学生7	10	10	20	0	C
学生8	5	25	30	20	C
学生9	50	60	110	10	B
学生10	10	50	60	40	B
学生11	25	60	85	35	B
学生12	50	50	100	0	B
学生13	10	10	20	0	C
学生14	0	30	30	30	B
学生15	15	15	30	0	C
学生16	70	80	150	20	A
学生17	10	80	90	70	B
学生18	5	15	20	10	C

Q.A 今回の演習を通した自身の能力向上について、それぞれの問に対して当てはまる項目を一つ選択してください。
　選択肢　｛1:まったく思わなかった｜2:あまりそう思わなかった｜3:どちらとも言えない｜4:少しそう思った｜5:とてもそう思った｝

Q.A-1 [ある事象の手順を読み取る能力が高まった]
Q.A-2 [ある事象に必要な要素（リソースやインフラストラクチャー）を読み取る能力が高まった]
Q.A-3 [似た事象の解決方法を組み合わせて新しい問題へ応用する能力が高まった]

Q.B 演習で用いたリソースフロー図について、それぞれの問に対して当てはまる項目を一つ選択してください。
　選択肢　｛1:まったく思わなかった｜2:あまりそう思わなかった｜3:どちらとも言えない｜4:少しそう思った｜5:とてもそう思った｝

Q.B 1 [手順の流れを明確にするのに役立った]
Q.B-2 [リソース種類を明確にするのに役立った]
Q.B-3 [リソースの存在期間を明確にするのに役立った]
Q.B-4 [インフラストラクチャーの稼働状況を明確にするのにのに役立った]
Q.B-5 [手順の定義が難しかった]
Q.B 5 [リソースの定義が難しかった]
Q.B-6 [インフラストラクチャーの定義が難しかった]
Q.B-7 [プログラミングに応用できると思う]
Q.B-8 [サーバーシステム等の設計に応用できると思う]

Q.C あなたについて
Q.C-1 大学での単位取得状況を教えてください [プログラミングに関する科目] ／選択肢　｛未履修｜単位未修得｜単位取得済　｝
Q.C 2 大学での単位取得状況を教えてください [通信ネットワークに関する科目] ／選択肢　｛未履修｜単位未修得｜単位取得済　｝
Q.C-3 大学での単位取得状況を教えてください [システム設計に関する科目] ／選択肢　｛未履修｜単位未修得｜単位取得済　｝
Q.C-4 あなたの経験について教えてください。[自作のソフトウェアを作ったことがある] ／選択肢　｛あり｜なし｝
Q.C-5 あなたの経験について教えてください。[サーバシステムを構築したことがある] ／選択肢　｛あり｜なし｝
Q.C-6 あなたの経験について教えてください。[アルバイトをしたことがある] ／選択肢　｛あり｜なし｝
Q.C 7 あなたの経験について教えてください。[アルバイト以外の就業経験がある] ／選択肢　｛あり｜なし｝
Q.C-8 あなたの経験について教えてください。[プログラマ等のシステム開発のアルバイトをしたことがある] ／選択肢　｛あり｜なし｝
Q.C-9 自身で調理する頻度を教えてください。　／選択肢　｛自分では料理をしない｜週に1~2日程度｜週に3~5日程度｜毎日｝

図9　アンケート一覧

学生12を除き演習２において演習１より高い得点を得た。この結果はRFDの直観性を示すものと考える。

３.３　アンケート

演習２の後、いくつかの匿名のアンケートから17の有効な回答が得られる。アンケート項目を図９に示す。質問票は、Q.AからQ.Dまで質問グループで構成されている。Q.Aは演習を通し問題解決方法の読み取り能力、応用について質問している。Q.B.はRFDによる能力向上について、Q.Cは学生の履修状況などについての質問である。

Q.Dは演習の感想について自由記述とした。

３.４　能力向上についてのアンケート

図10は、自分の能力向上のための質問票Q.Aの結果である。Q.Aアンケートの回答オプションは、「とてもそう思った」「少しそう思う」「どちらとも言えない」「あまりそう思わなかった」「まったく思わなかった」である。回答は、「とてもそう思った」の

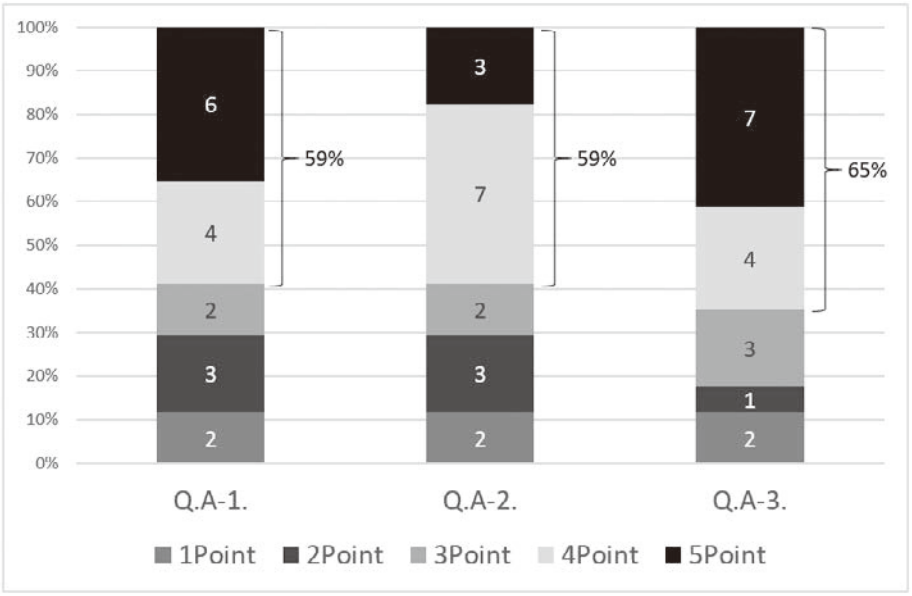

図10　能力向上についてのアンケート

場合は５、「まったく思わなかった」の場合は１と５段階で評価した。

Q.A－1の回答では「とてもそう思った」「少しそう思う」の割合は59％、Q.A－2の回答では「とてもそう思った」「少しそう思う」の割合は59％、Q.A－3の回答では「とてもそう思った」「少しそう思う」の割合は65％となった。この演習を通して半分以上の学生が自分の能力向上を感じた。この結果は、問題解決のための支援ツールとしてのRFDの有効性を示している。

３．５　RFDについての設問

図11はRFDに関する質問票Q.Bの結果を示す。回答はQ.Aと同じように５段階評価されるが、Q.B－5～Q.B－7の回答値の１～５を５～１に反転した。これはRFDについて否定的な質問をしたため、他の設問に合わせ最も高評価が５になるように調整したためである。

RFD表現能力に関するQ.B－1とQ.B－2は「とてもそう思った」「少しそう思った」が71％、Q.B－3は65％、Q.B－4は59％であり評価が高い。この結果は、RFDの直感的な表現能力の高さを示している。

Q.B－5～Q.B－7では、ほとんどの学生はリソースやツールなどのRFDパーツの定義が難しいと評価した。RFDパーツ定義は、「何が必要なのか」、「行う方法」、「何がもたらされるか」を時間的な尺度と共に簡潔な図形で表記し、これらの配置と組み合わせにより「いつ行うか」および「準備するか」を直観的に表現する。RFDパーツの定義の難しさは、部分問題や手順の粒度決定の難しさに起因していると考えられる。問題解決における粒度決定の問題は、調理であればどこまで手順を分解して詳細に記述するか、プログラミングであればどこまでサブルーチンやクラスを細かくモジュール化するかといった問題で、利用者や実行環境により最適な粒度が変わることから、特に経験の少ない被験者にとってこれらを決定するのが難しい。この問題解決における粒度決定問題に関しては、布広ら[3]の研究において試みられているように、段階的に問題を詳細化するような学習補助の仕組みにより別途トレーニングが必要であると考える。一方で、あらかじめ定義済みのRFDパーツを提示することで粒度決定の難しさを排除し、RFDは手順やリソースの関係を試行錯誤する優れた訓練ツールとして利用が可能である。

３．６　学生のプロフィールについてのアンケート

図12はQ.C.で学生プロフィールについてである。多くのアンケートでは、システム開発やグループワークに関する学生の実験について「あり」または「なし」で回答する形式とした。

図13は、QC－9の結果を「自分で調理しない」「週に１～２日」「１週間に３～５日」「毎日」で構成する自己の調理の頻度を求めた。

多くの学生はシステム開発とプログラミングに関する講義を受けている。一方で、彼らはプログラミングの仕事についての経験は無い。加えて、アンケートの結果によれば被験者の調理の頻度は全体的に低い。したがって、すでにプログラミングに関す

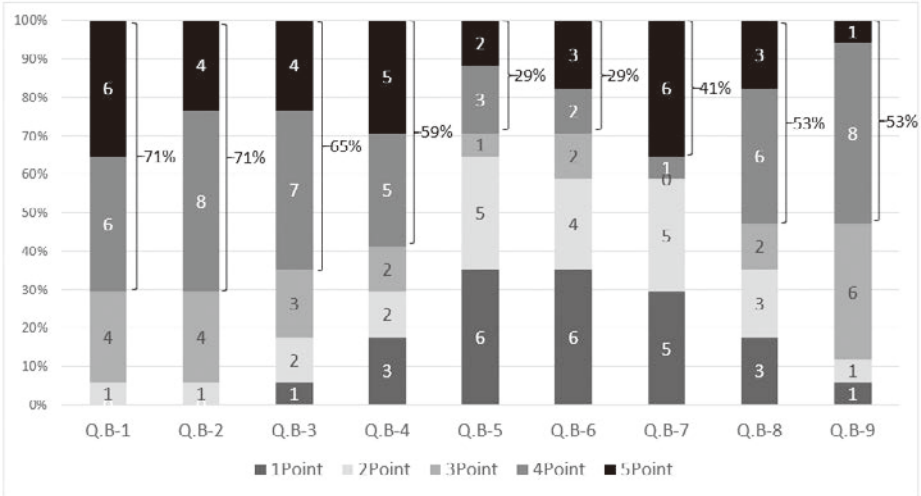

図11　RFDについてのアンケート

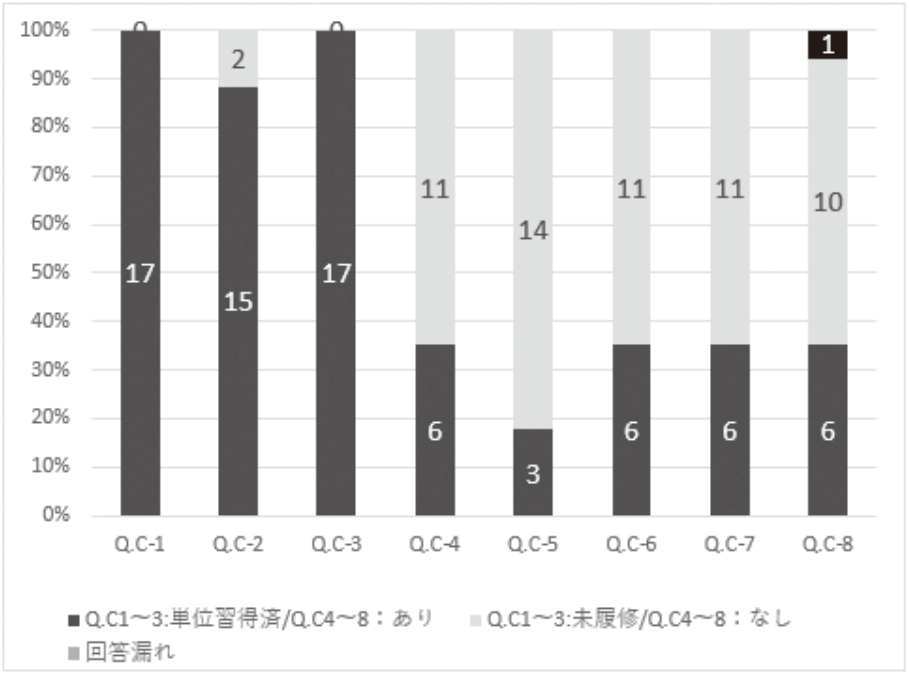

図12　学生のプロフィール

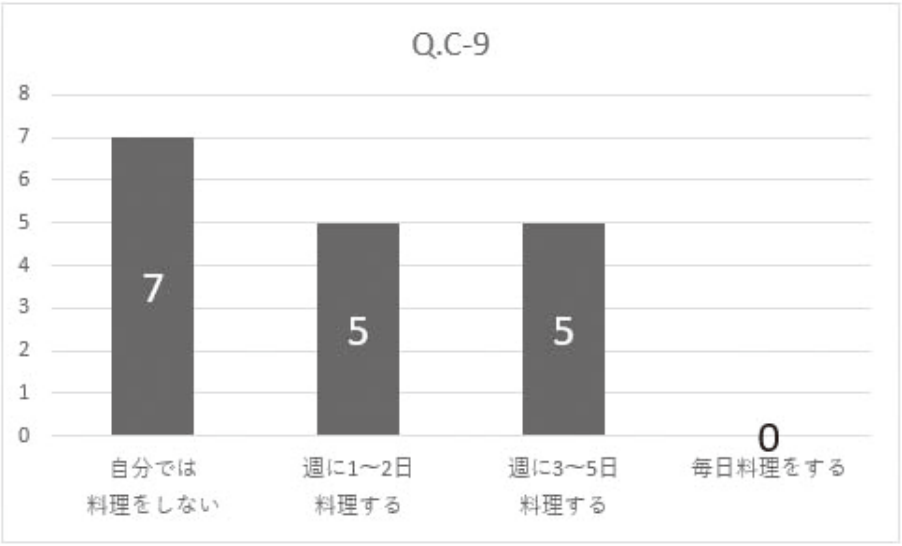

図13　料理の頻度について

る単位取得者である被験者にとって、すでに履修済であるUMLに基づくRFDは理解しやすい。一方、調理の頻度は低いため、演習の対象として基礎となる知見が少なく理解が容易ではないことを示している。

3.7　無記名の記述式コメント

アンケートQ.Dの表現では、コメント空間は肯定的なものと否定的なものに分かれていた。大多数の肯定的なコメントは、手作業による試行錯誤による問題解決と高い直感性のイベントモデリングの支援ツールとして、RFDの高い有効性を指摘している。この結果、高い直感性は小学校などの初等教育における演習での問題解決の訓練には非常に効果的である。RFDを用いた試行錯誤による問題解決型教育の実現のための情報システムの開発を計画している。

大部分の否定的なコメントは、RFDの手作業時間の長さによる欠点が指摘されている。しかし、これは今後のRFD作成支援システムにより解決できる。

３．８　アンケートの相関

各アンケートのスピアマン順位相関を求めると、Q.A－1とQ.A－3及びQ.A－1とQ.B－1が高い相関係数0.83を示す。この結果は、RFDが同様の問題の解決に関する知識の組み合わせから問題解決に有効であることを示している。

４．結　　論

本稿では、動画からのモデリングに基づいた問題解決型教育フレームワークを実現するためRFDを提案した。我々の問題解決教育のアプローチは、動画などの非言語化された複雑な問題からのイベントモデリングに焦点を当てている。本実験では、システム開発やプログラミングと類似した特性を持つ調理のためのフレームワークを検討した。この実験から、RFDの直感的な表現と情報システム設計への応用の可能性が高いという良好なアンケート結果が得られた。教育の枠組みの言語依存度が低いほど直感性が高いことを考慮すると、情報社会の問題解決のための初等教育に効果が期待されている。

また、この実験のアンケートにおける否定的結果から、RFDを用いた演習は試行錯誤を容易にするため、システムのサポートを必要とする。我々は、RFDエディタを提供することで利用者を支援する方法、あるいはブロック型玩具形式で提供するなど、これからの研究材料としたい。

【謝辞】

本研究は、文部科学省科学研究費基盤研究(C)課題番号15K01086の補助を受けた。

【引用文献】

［1］Bernardi, S., Donatelli, S. and Merseguer, J.: From UML sequence diagrams and statecharts to analysable petri net models, In Proceedings of the 3rd international workshop on Software and performance, pp. 35-45（2002）.

［2］Petre, M.: UML in practice, In Proceedings of the 2013 International Conference on Software Engineering, pp. 722-731（2013）.

［3］Nunohiro, E., Kishimoto, Y., Yamaguchi, T., Ohshiro, M. and Tsukuta, T.: Development of practice problems generating function in PPL system, In Proceedings of the 22nd International Symposium on Artificial Life and Robotics, p.64（2017）.

［4］Bird, R. S. and Wadler, P. L.: Functional Programming, Prentice Hall（1988）.

［5］OMG Unified Modeling Language（OMG UML）, Superstructure, V2.1.2,（online）〈ftp://calhau.dca.fee.unicamp.br/pub/docs/ea977/UMLSuper2.1.2.pdf〉（accessed 2017-11-14）.

［6］Miles, R. and Hamilton, K., 原隆文（訳）: 入門UML 2.0, p.56, オライリージャパン（2007）.