信州大学審査学位論文

組込みソフトウェアエンジニアを目指す学生のための実践力教育に関する研究

2020年3月

舘　伸幸

目 次

表 目 次

図 目 次

第1章　はじめに

本章では，本論文の概要を示し，研究の背景と目的，および本研究で試行した教育モデルについて述べる．

1.1　本研究の概要

情報技術を高度に活用して社会に貢献できる人材の育成は，喫緊の課題となっている．産業界からは，情報系学生に対して「実践力」を求める声が強い．特に組込みソフトウェアは，我が国の産業構造にとって重要な技術領域であり，実践力ある人材の育成が必要とされている．

従来この種の教育は，インターンシップや一部の PBL によって実施されてきた．しかし後述するように，インターンシップは中長期の期間を必要とし，また受け入れ企業の文化に大きく依存する．一方 PBL は，参加者に一定のスキルが要求される学習法であり，初学者向きではない．現状でもその多くは大学院生向けに実施されている．

本研究では，組込みソフトウェアの特徴および求められている実践力とは何かを明らかにし，それらを網羅した新しい教材を開発した．また，企業における製品開発工程を倣った学習計画と組み合わせることで，初学者向けの演習カリキュラムを構築した．さらに，本研究で新たに提案した測定手段を使うことで，本研究のカリキュラムが，組込みソフトウェア開発者を目指す学生の実践力の向上に寄与することを示した．

1.2　本研究の特徴

本研究の特徴は，次の通りである．

1. ソフトウェア開発における実践力とは何かを明らかにした
2. 実践力を高めるために効果的な学習モデルとして 4P モデルを提唱し，演習授業で短期間に修得するためのカリキュラムを開発した

3. 教材が実践力の向上に寄与することを測定する方法を提案した

1.3　研究の背景

先に述べた社会からの期待に対し，組込みソフトウェアは次のような特徴を持つため，大学学部における教育は困難を伴っている (渡辺, 吉田, 2011).

1. 開発範囲が，アプリケーション層からデバイスドライバ層まで幅広い
2. 周辺回路や機構など，システムに関わる製品技術が求められる
3. 自動車や医療機器のように，信頼性・安全性が強く求められる

すべての国立大学（86 校），すべての公立大学（93 校），高等専門学校（57 校）において，組込みソフトウェアに関する教育がどのような状況かを調査した．また，組込みソフトウェアをカリキュラムに持つ一部の私立大学（10 校）も調査した．すべてのカリキュラム中，授業名やシラバスにて組込み技術を対象としたものから，以下に列挙した内容のものを除外し,「純粋に組込みソフトウェア開発に関する演習授業」を抽出した.

1. ソフトウェア開発が主なテーマでないもの（プログラミングを，ハードウェア理解のツールとして扱っているなど）
2. OS やプログラミング言語そのものをテーマとしたもの

分析の集計結果を表 1 に示す．調査対象としたシラバスは 2019 年時点のものである．対象学科が異なっても，内容が同じと思われるものは集約して 1 件としている．また，本研究で行っている実験授業は，集計に入れていない．表中，横軸の各項目の意味は以下に示す通りである．2 と 3 は，いわゆるプログラミングだけでなく，工学的手法による開発の演習をしているかを調べている．4〜7 の 4 項目は，実践的な組込みソフトウェア教育に必要と考えられる 4 つの項目について調べている．これら 4 項目の根拠と詳細については，後述する．表中の値およびカッコ内の値については，各項目ごとに説明する.

1. 組込みソフト演習カリキュラム数

   対象学校群中，前述の方法で抽出をした組込みソフトウェア開発に関するカリキュラムの総数.

2. 仕様分析/定義

   要求仕様に対する分析や定義を行っているカリキュラム数．カッコ内は，シラバスに言葉はあるものの，実際の作業まで言及されていないカリキュラムの数.

3. 設計

   設計を行っているカリキュラム数．カッコ内は，シラバスに言葉はあるものの，実際の作業まで言及されていないカリキュラムの数.

4. 製品技術（ハードウェア）

   組込みシステムの特徴である，製品技術（ハードウェアやソフトウェア個別の技術を組み合わせて使う技術）を扱っているカリキュラム数．カッコ内は，ハードウェア個別の扱いはあるものの，製品技術としては言及されていないカリキュラムの数.

5. 開発プロセス

   なんらかの開発プロセスに準拠して演習を進めているカリキュラムの数

6. プロジェクト/計画

   計画の立案，もしくは提示された計画に従った開発演習をしているカリキュラムの数

7. 技術者としての行動規範

   シラバスでなんらかの行動規範に言及しているカリキュラムの数

要求仕様について分析や定義を行うことを明示している演習は，国立大学・大学院では60件中3件であった．公立大学では1件，高等専門学校では19件中1件であった．これら5件はいずれも，少なくともシラバス上では仕様を演習の課題として扱ってい

大学/高専	1 組込みソフト演習 カリキュラム数	2 仕様 分析/定義	3 設計	4 製品技術 （ハードウェア）	5 開発プロセス	6 プロジェクト /計画	7 技術者として の行動規範
国立大学・大学院	60	3(+3)	4(+8)	0(+12)	3(+5)	3	1
公立大学・大学院	8	1	1	0(+1)	0	0	0
高等専門学校	19	1(+4)	5(+8)	0(+15)	6	0	7
私立大学・大学院 ※調査対象は10校	5	0	1	0(+4)	0	1	0

表 1: 組込みソフトウェアの演習授業に関する調査結果

るものの，そこに矛盾や抜け漏れがあるといった，実際の要求仕様とは若干の乖離がある可能性もある．設計や開発プロセスについても，ほぼ同様の件数であった．開発プロセスを導入していれば，要求分析や設計は必然的に行うことになるので，これらの項目に相関があることは集計結果として矛盾はない．製品技術に関しては，LED やスイッチなど，代表的な周辺ハードウェアについての個別の制御プログラミング等は，カッコ内に示した件数見られたが，製品技術としての言及は見られなかった．開発プロジェクトや計画を扱っているカリキュラムは，全調査対象中 4 件のみであり，うち 2 件は大学院生を対象とした PBL 型授業だった．全大学中，6 つの調査項目すべてを網羅したカリキュラムは 0 件だった．これらの結果から，初学者である大学学部生や高専生は，学んだ知識を実践的に活用する学習を行う機会が，通常の授業では少ないと言える．

1.4 研究の動機と方法

筆者は，企業において組込みソフトウェア開発に従事し，新入社員の育成にもたずさわってきた．そこで，新入社員に特徴的に感じられる弱みが 2 点存在した．第 1 点は，開発プロセスや各工程に関して，知識は学んでいるが実践に関する教育をほとんど受けていない点である．そのため，特に顧客要件の分析やそれに基づく設計など，上流工程において実践力を発揮することが困難だった．第 2 点は，ハードウェアなど組込みの特徴的な技術への不慣れである．前述の調査結果は，この心象を裏付けするものと感じられた．

そこで，仕様分析からテストまで，開発プロセスに則って，かつ，組込みの特徴的技術を内包した教材を導入することで，学んだ知識を実践し，あるいは逆に実践の中

で知識を学ぶ教材を開発することにした．

研究の冒頭にあたって，まず実践力とは何かを明らかにすることとした．学生に対する社会の要求として語られる「実践力」については，たとえば経済産業省では社会人基礎力，文部科学省では学士力といった表現を用いて，主にジェネリックスキルの向上を学習目標として掲げている．しかし，組込みソフトウェア開発を行う技術者の実践力を考えた場合，はたしてジェネリックスキルが最も重要と言えるだろうか．

仮に実践力をジェネリックスキルと考えた場合，それは，1.3 に示した組込みソフトウェアの特徴に，少なくとも直接反映する力とはいえない．従って，組込みソフトウェア業界に向けた大学学部レベルで修得すべき実践力の明確化と，その実践力を育成するための教育方法の具体化は，まさに高い実現可能性と確実な教育効果を意識して取り組むべき課題と考える．

そこで本研究では，最初に組込みソフトウェア技術者に求められる「実践力とはどのような能力なのか」を明らかにし，それを目指す大学学部生（以下，学生）のための効果的な教育教材の開発と，教材の有効性の測定方法の確立を目指した．本研究の目的は次の通りである．

1. 組込みソフトウェア技術者に求められる「実践力」とはなにかを明らかにする
2. 実践力を修得するための効果的な教育モデルを設計し，具体的教材を開発する
3. 開発した教材を用いて汎用的カリキュラムを作成し，組込み教育を行う大学へ展開する
4. 2 で設計した教材に関し，有効性を測定する方法を提案する

目的 1 については，社会で活躍している情報技術に携わる技術者に対し，実践力についてのインタビューを実施した．その結果得られた知見は,「最も重要な能力は，製品を作れて，保守できること」という，技術者という職務に対して当たり前のことだった．この分析結果やインタビューへの回答から,「要求に対して，主体的に，きちんと製品を作れる能力」を実践力と定義づけ，その基本を修得するための教材開発を具体的目標に掲げた．

目的2については，まず目的1の結果から「きちんと製品を作れる」という抽象概念について考察した．その結果，以下に示す4つの具体的なテーマを抽出した．

組込みシステムは，機器に組み込まれて動作する特徴から，ハードウェアなどの動作環境に関する要素技術を組み合わせ，製品に仕上げる技術を必要とする．これを「製品技術」という．当然，組込みシステムを制御する組込みソフトウェア開発には，この製品技術に対応する技術すなわち，ハードウェアを利用，制御する技術が必須である (平山, 2004)．本研究ではこのカテゴリを**プロダクト**（Product）というテーマで扱う．

次に，組込みソフトウェアは冒頭で述べた通り，信頼性や安全性を強く求められる場合が多く，そのため開発プロセスを重視する．開発プロセスの重要性については，教育の実践現場でも一定の認識が広がっている．たとえば安永らは，PBL(Project Based Learning/Problem Based Learning) 型の開発カリキュラムを進める中で，ソフトウェア開発の全体像を可視化した学習環境という対策を行っている (安永, 大場, 奥野, 伊藤, 2013)．また松澤らは，ソフトウェア開発PBLにおいて，反復プロセスを導入する試みを実施しており．開発の進め方に関する学生のポジティブな反応が紹介されている (松澤, 塩見, 秋川, 酒井, 2009)．沢田らも同様に，ソフトウェア開発実習において開発プロセスを強く意識させるカリキュラムを用いており，学生の成果発表などから「ソフトウェア開発プロジェクトで行うべき作業内容に対する理解が深まっていることが分かる」と述べている (沢田, 小林, 金子, 中道, 大久保, 山本, 2009)．その他，情報系の大学院生向けに試行した教育においても，開発プロセスを導入することが重要であることが報告されている (名古屋大学 OJL, 2015)．IT人材白書2011(IPA, 2011) では，教育機関の実践的IT教育内容のうち，卒業後に企業での実務に役立っているものについての調査結果が報告されている．そこでは,「システム開発手法や開発プロセスに関する知識経験」という回答が最も多かった (IPA, 2011)．これらのことから，仕事の進め方の基本として開発プロセスを身につけさせることは，実践力への教育として重要であると考えられる．本研究ではこのカテゴリを**プロセス**（Process）というテーマで扱う．

開発プロセスに従って仕事を進めるには，有期性ある計画（プロジェクト）として作業する考え方が必要である．事実，企業においては，製品開発は計画に基づいて

実行している．したがって，開発プロセスを学ぶ上では，最終納期だけでなく中間の工程に関しても，有期性のある作業というプロジェクト管理の考え方を学ぶ教育が必要と考えられる．プロセスやプロジェクトを教育に取り入れた例には，名古屋大学における OJL(On the Job Learning) などがあり，受講学生が有益であったと回答した報告がある (名古屋大学 OJL, 2015)(舘, 山本, 高嶋, 松原, 本田, 高田, 2014)．また，Maria Lydia らによる，PBL にプロジェクトマネジメントを組み合わせて，開発演習にマネージャやステークホルダが登場する実践的な学習の取り組みが報告されている (Fioravanti,Sena,Paschoal,Silva,Allian,Nakagawa,Souza,Isotani,Barbosa, 2018)．本研究ではこのカテゴリを**プロジェクト**（Project）というテーマで扱う．

以上，プロダクト，プロセス，プロジェクトなるテーマを挙げたが，実際にこれらの知見を駆使して仕事を進めるには，専門職としての行動規範も必要と考えられる．本研究ではこのカテゴリを**プロフェッショナリズム**（Professionalism）というテーマで扱う．

実践力には，以上 4 つのテーマに関して総合的に実力を発揮することが必要と仮定した．以下この仮説を，4 つのテーマの頭文字が P であることに因んで，4P もしくは 4P モデルと称する．

4P モデルは，情報処理推進機構（以下 IPA）がエンジニア向けに策定した組込みスキル標準（ETSS）が掲げる項目と，奇しくも近似することを見た．このことは，本モデルの方向性の正しさを示唆すると考えられる．ETSS は組込みソフトウェア開発に必要なスキルの構造を提示したものである．一方，本研究の 4P モデルは，組込みソフトウェア技術者を目指す学生のための，新しい学習モデルを示すものであり，ETSS とは内容が異なる．以下，双方の差異について，テーマごとに説明する．

1. プロダクト

   ETSS は,「技術要素」というカテゴリを提示しており，組込みシステムに必要な機能を技術要素として分類している．各要素技術は，機能としての定義となっており，その詳細は教育等の実装にゆだねられている．つまり，何をどのように学ぶかは，定義も例示もしていない．スイッチや LED といった具体的なセンサやアクチュエータは，あくまでもユーザ・インターフェースを構成する機能として扱われている．

本研究では，製品技術という学習テーマとして扱っている．製品は，それを構成するハードウェアおよびソフトウェアの技術要素が有機的に結合して動作する．特に組込みソフトウェアでは，入力も出力も実際の物理世界とインターフェースする特徴を持つ．そのため開発者には，使用する各技術要素に関して，非機能要件も含めてその性質を理解してから使うという姿勢が求められる．また，駆動プログラムの想定と実際の動作（物理的ふるまい）との関係も理解している必要がある．そこで本研究におけるこのテーマでは，単に個々の要素技術の知識にとどまらず，要素技術を組み合わせて，要求された動作をする製品に仕上げる技術を学習することを目的としている．

2. プロセス

ETSS は,「開発技術」というカテゴリで，SLCP（Software Life Cycle Process）で定義された開発プロセスを基準として，個々の工程についてのスキル項目を定義している．すなわち，開発プロセスに沿って開発を行うことは前提となっている．

本研究では，そもそも開発プロセスを知らない，あるいは開発プロセスに沿った開発の経験がない学生を対象として,「あるべき作り方」を学習するためのテーマとしている．課題を読んで，粘土細工のようにコードをこねて，それらしく動いているかのようなプログラムで完成といった，アマチュア的手法からの脱却を目的としたテーマとしている．

3. プロジェクト

ETSS は,「管理技術」というカテゴリで，開発プロジェクトおよび開発プロセスのマネジメントに関するスキルを提示している．主に，組織体におけるマネージャのスキルとしての定義が色濃い．

学生またはジュニアクラス社員に管理技術は必要か．このことについて，SNS で有識者に意見を求めた．21 件（うち，技術職 15 名，大学教員 5 名，公務員 1 名）の回答があり，全員が必要ではないという意見だった．その一方で，現役のシニア技術者からは次のような意見もあった.「マネジメントスキルは，部

分的には個々のエンジニアにも必須のスキル．必要な情報がきちんとPMに上がってこないとプロジェクト全体のマネジメントなんてできっこない」（原文のまま）．つまり，マネジメントされる側にも，一定の基本的なスキルが必要という意味である．

本研究では，マネジメントされる側の能力として，本テーマで管理技術を扱っている．たとえば，決められたスケジュールプランに従って作業を進める，現在の作業目標に対して進捗を報告するといった能力を学習することを目的としたテーマとしている．

4. プロフェッショナリズム

ETSSは,「パーソナルスキル」というカテゴリで提示しており，コミュニケーション，ネゴシエーション，リーダシップ，問題解決といったスキル分類を提示している．

本研究では，経済産業省が平成22年に発表した「大学生の「社会人観」の把握と「社会人基礎力」の認知度向上実証に関する調査」(経済産業省, 2010)から，学生に不足していると思う能力要素の上位3要素である主体性，コミュニケーション力，粘り強さについて着目した（図1）．これらはいずれも抽象的概念であるため，エンジニアを目指す学生向けに，より具体的な行動目標として次のように定義しなおした．

(a) 主体性

技術的課題を，自ら発見し，自力で調査・勉強できる

(b) コミュニケーション力

技術的課題に関して，周囲と相談できる（わかりやすく伝える，議論する）

(c) 粘り強さ

手がけた課題を，最後までやり遂げる

上記定義も，SNSにて有識者に意見を求めた．29件（技術職・元技術職27名，大学教員2名）の回答があり，全員の賛同が得られた．また17件のコメントがあり，この内容への関心の高さが伺えた．

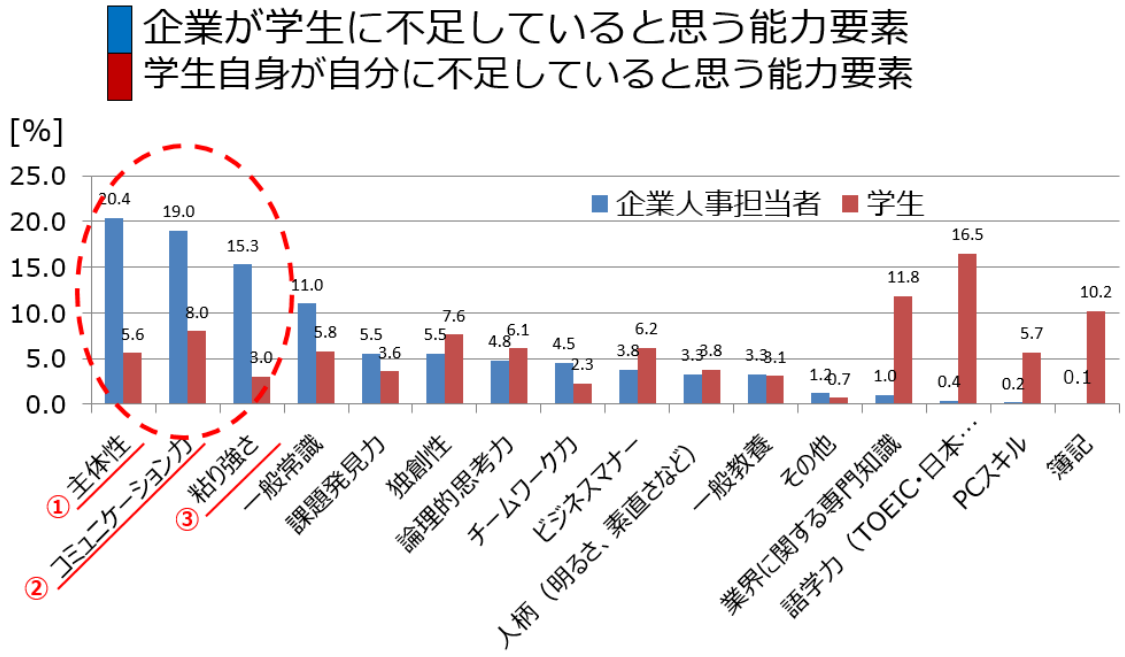


図 1: 大学生の「社会人観」の把握と「社会人基礎力」の認知度向上実証に関する調査

以上述べたような，本研究の 4P モデルに対して準拠もしくは網羅するような実践的学習は，従来はインターンシップにゆだねられることが多かった．インターンシップは，一般に数週間という長時間が必要であることと，実施先企業の文化や教育姿勢によってその教育品質に大きなばらつきが発生するといった問題点がある．

本研究の 4P モデルに基づく学習は，インターンシップという方法を用いることなく，学内の演習授業で質の高い実践力を修得することを目標としている．また，方向性において ETSS と共通することから，将来就職後に ETSS に基づくスキルアップにも，継ぎ目や障壁なく学習を継続できる可能性もあると考えられる．本研究では，この 4P モデルに基づいて，1 つの教材例を開発し実践した．詳細は第 4 章で述べる．

目的 3 については，目的 2 のプロジェクト項目を利用して，教材の開発課題の開発計画を授業進行とリンクさせることで実現した．具体的には，学生に開発のガントチャートを示しつつ，それに沿って授業を進めることで，誰にでも本教材による授業を可能とすることを狙った．

目的 4 については，新たな測定方法を試行した．従来のような学習教材や授業そのものへの学習者の主観的アンケートではなく，学習者自身の技術的能力向上を推し量

ることでの測定を目指した．具体的には，組込み産業界での実開発で使用可能な達成レベルの要求仕様書を読ませ，自力での開発の可否を問うものである．演習で学んだ知識を使って，類似の別の開発にとりかかれるかどうかを尋ね，可能という回答の場合にはその根拠について調べ，本研究の 4 つのテーマに基づいて分析を行った．本研究で開発した教材を用いた授業の前後や，別の開発演習授業との間で，明確な差異を検出できることがわかった．

1.5 4P モデルの工学的位置づけ

工学教育に関しては，工学における教育プログラムに関する検討委員会の定義がある．報告書「８大学工学部を中心とした工学における教育プログラムに関する検討」(工学における教育プログラムに関する検討委員会, 1998) では，工学教育について次のように述べられている．

「工学教育とは技術者・研究者に必要な工学におけるスキルと知識を与えることである．スキルとは「物事を正しく行うことの出来る能力」であり，また「問題と解答との間のスペースを埋めることのできるプロセスを構成する能力」である．工学に関するスキルによって技術者・研究者は専門分野の知識を駆使し，関連分野の知識を関連付け，統合し，また，その後の学習の習慣を身に付ける」

つまり，知識とそれを正しく行う（用いる）ことのできる能力を涵養することと学習の習慣を身に着けることが，工学教育とされている．本研究で開発した教材の 4P の，学んだ知識をシステムに合わせて組み合わせる製品技術，正しくものを作る開発プロセス，計画に従って作業を進めるプロジェクトは，まさにこの定義にある「知識を正しく用いる」ための基本要素と言える．また，主体的に学ぶことを包含したプロフェッショナリズムは,「学習の習慣を身に付ける」ことを目的としている．

1.6 研究成果の適用

先に示した 4P すなわち，プロセス，プロダクト，プロジェクト，プロフェッショナリズムの 4 つのテーマを組み込んだ，シンプルな演習教材を 2016 年度に開発し，2017 年度に 2 回の試行を行った．改良を経て 2018 年度に再試行し，その後実験授業として複数の大学に展開を行い，2019 年度も実施中である．表 2 に試行ならびに実験授業

表 2: 本研究で開発した教材による授業実施大学一覧

年度	大学	学年	授業時間	工夫点	備考
17	K大学	3年上期	18 x 90分	1.テキスト初版で試行	
				2.動作環境を整備	
17	K大学	3年上期	18 x 90分	1.事前学習の時間を増加	
				2.フローチャートを書かせる試行	
18	K大学	3年上期	18 x 90分	1. 詳細設計から実装への指導方法改善	効果計測を試行
				2. スイッチ処理の指導方法改善	
				3. 開発プロセスの意識づけ強化	
				4. 部品化の考え方(階層構造)を追加	
18	K大学	3年上期	18 x 90分	同上の内容で実施	
18	C大学	3年上期	18 x 90分	同上の内容で実施	効果計測を試行
18	B大学	2年下期	18 x 90分	同上の内容で実施	効果計測を試行
19	K大学	3年上期	18 x 90分	同上の内容で実施	別教員が実施
19	C大学	3年上期	18 x 90分	同上の内容で実施	効果計測を試行
19	M大学	3年上期	16 x 90分	同上の内容で実施	効果計測を試行
19	S大学	4年上期	16 x 90分	同上の内容で実施	効果計測を試行

実施大学の一覧を示す．

2017 年度は，まず K 大学の演習授業で 2 回の試行を実施した．ここでは主に教材の構成単元の時間割り振りや，説明の変更を試行してより良い指導方法を模索した．2017 年度に得られた知見をもとに，教材を改良して 2018 年度に再度 K 大学で 3 度目の試行を実施した．教材改良の詳細は 5 章で述べる．これら一連の試行にて，教材が一定の品質に達したことを受け，同年度に K 大学（試行とは別のクラス），C 大学，B 大学へと，実験授業を展開した．また 2019 年度も継続して授業を行い，2018 年度実施大学に加えて新たに M 大学と S 大学にも展開を行った．この中で，2019 年度の K 大学では，教材開発者とは別の教員によって授業運営を行い，まだ 1 例ながら問題なく授業を遂行できることを確認できた．これら実験授業では，実践力に関する効果測定を実施しており，その結果については 6 章で述べる．

また，カリキュラムとしての展開は，2019 年度に 2 校が実施済みである．2020 年度には，さらに 3 校からの要求に対応する予定である．

1.7 本論文の構造

図 2 に，本論文の構造を示す．本章ではまず，研究の背景と目的，および本研究で試行した教育モデルについて述べた．第 2 章では，組込みソフトウェア分野の特徴と教育事例を示し，本研究の位置付けを示している．第 3 章では，組込みソフトウェア技術者の実践力の定義を示し，プロダクト，プロセス，プロジェクト，プロフェッショナリズムの 4 つのテーマに基づく新しい学習モデル（すなわち 4P モデル）を提案している．第 4 章では，4P モデルに基づく教材の設計事例を述べ，続く第 5 章で試行とその結果に基づく改良事例を示している．第 6 章では，第 5 章に示した教材の効果測定について述べている．提案した評価方法は,「組込み産業界での実開発で使用可能な達成レベルの要求仕様書を読ませ，その仕事に取り組めるかを問う」というものである．最後に第 7 章で，提案のまとめと考察を述べている．

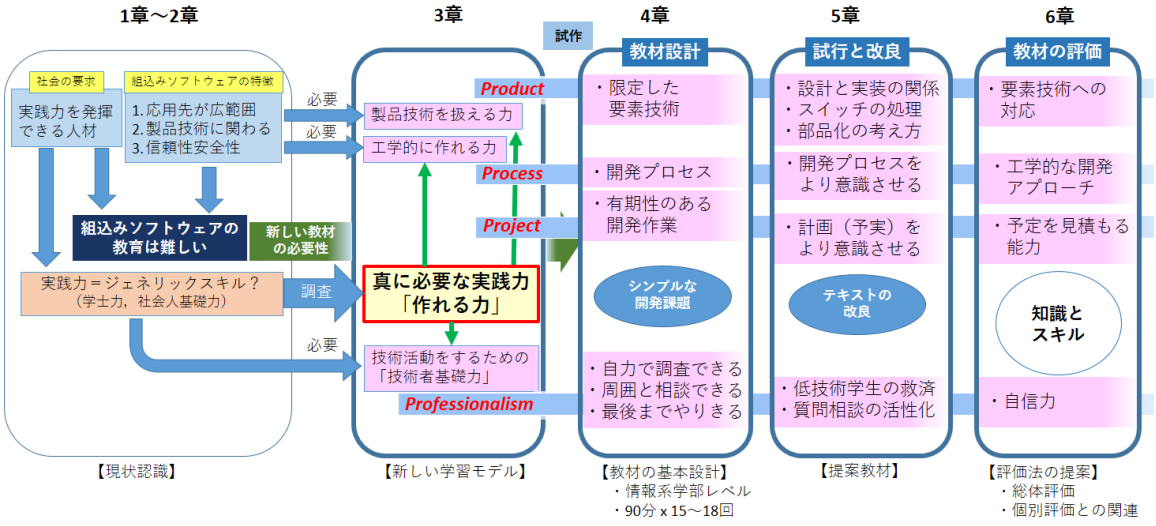


図 2: 本論文の構造

第2章　組込みソフトウェアの特徴と教育事例

本章では，研究の対象である組込みソフトウェア分野の特徴と，それを目指す学生向けの既存教育について述べる．

2.1　組込みソフトウェアの特徴

組込みソフトウェアの定義は明確でない．たとえば「機械的，電気的もしくは電子的な制御を行うハードウェア装置に搭載されるソフトウェア」(野口, 2015),「各種の機器に組み込まれてその制御を行うコンピュータシステム」(高田, 2001),「制御向けや情報処理向けなど，応用分野が多様」(ETSS, 2006) など，表現は多様である．これは，組込みソフトウェアの応用範囲が多岐にわたっていることが原因である．そのことは，たとえば情報処理推進機構（IPA）による 2017 年度の組込みソフトウェアに関する動向調査 (IPA, 2018) などに見ることができる．これによれば，組込みソフトウェア産業の主要な事業カテゴリは図 3 の通りであり，産業用から家庭用まで幅広い分野に広がっていることがわかる．またその広がりは，ここ数年ほとんど変化がない．

一方で，組込みソフトウェアの技術的特徴については，文献間で共通する内容が見られる．表 3 に，組込みソフトウェアについて述べている 5 つの文献が記載している技術的特徴をまとめたものを示す．それぞれ，時間制約やハードウェア協調など，記載してある特徴を表す代表的な言葉で分類した．参照した文献は次の通りである．

1. モデル検査を用いたフォールトアナリシス手法の提案 (野口, 2015)
2. 組込みシステム開発技術の現状と展望 (高田, 2001)
3. 組込みソフトウェア開発のための ETSS 標準ガイドブック (ETSS, 2006)
4. 組込みソフトウェアの現状 (平山, 2004)

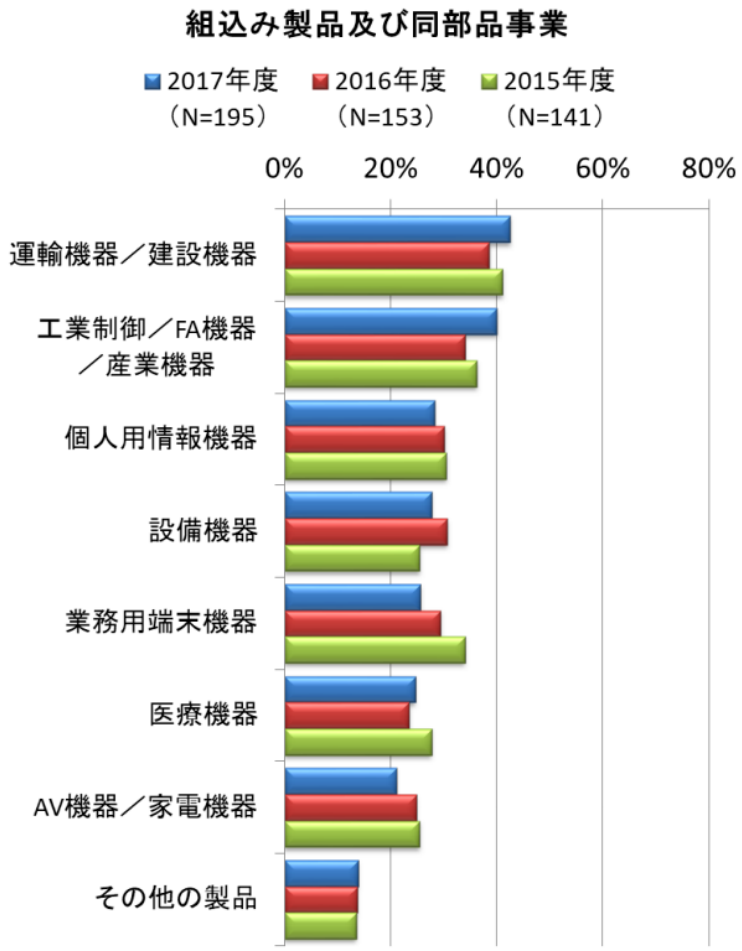


図 3: IPA2017 年度組込みソフトウェアに関する動向調査

5. 車載ソフト開発の現状 (川名, 2004)

また，分類した言葉の出現頻度を表 4 に示す．これによれば，ハードウェアとの協調動作に関する記述が最も多く，時間制約や信頼性がそれに次いでいる．それら以外の，移植性・拡張性，リソース制約，タスク制御，開発環境，非機能要件といったものも，文献での言及は少ないが組込みソフトウェアの重要な技術的特徴である．たとえばタスク制御は，出現数は少ないものの，現実には必須とも言える技術である．それは，特に小規模の組込みシステムにおいて，OS を使用しないケースも多いためである．

図 4 は，2008 年時点と 2017 年時点での，組込みシステムにおける OS の使用度合いの調査結果である (IPA, 2008)(IPA, 2017)．OS を使用していないプロジェクトは，

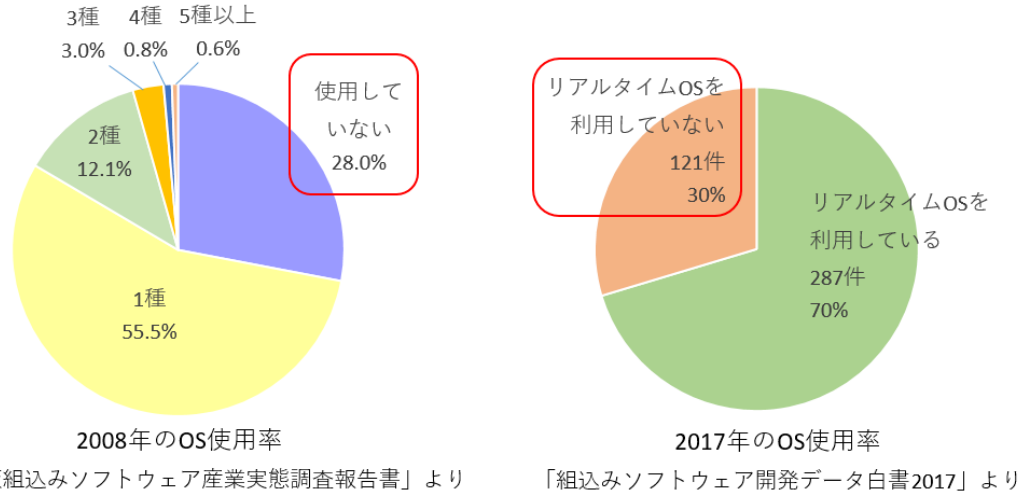


図 4: 2008 年と 2017 年の OS の使用率

2008 年には全体の 28 %，2017 年には 30 %と，ほぼ同じ割合を示している．このことから，将来的にも同程度の割合で OS を使用しない開発があり得ることが予測できる．つまりこれは，一般に OS に委ねる並行動作などのタスク制御も，組込みソフトウェアの重要な特徴として，学ぶべき技術であることの証左である．また OS を使う場合であっても，上記の特徴分類に出現しているリソース制限などの理由により，OS そのものの動作原理を熟知していることが要求される場合がある．よって，タスク制御はハードウェア協調や時間制約などに勝るとも劣らない重要な技術と考えられる．

表 4 の中には規約制約という分類項目がある．前述の参考文献で言及されているのは法規対応であるが，それ以外にもたとえば通信プロトコルや設計基準，コーディング規約などに準拠するといった，既存のルールへの対応が求められる場合も，本項目に含めるのが妥当と考えられる．

非機能要件や規約規定については，他の技術項目とは粒度の異なる項目である．しかしここでは単に，組込み技術に関する教材設計時に，考慮すべき項目という意味でキーワードとして扱っている．

以上のことから，組込みソフトウェアの教育教材には，表 4 に示したような技術的特徴に関する内容について，文献等での言及頻度に関係なく一様に学ぶことが必要と考えられる．ただし，授業時間等の制約はあるので，ある 1 つの教材にどの項目を優先して取捨するかについては，その設計方針にゆだねる事項となる．

表 3: 文献に記述している組込みソフトウェアの特徴

文献	文献に記述している組込みソフトウェアの特徴	分類
1	並行動作	タスク制御
	ソフトウェアだけでは動作が決まらない様々な非決定性	ハードウェア協調
2	専用化	移植性・拡張性
	厳しいリソース制約	リソース制約
	高い信頼性	信頼性
	リアルタイム性	時間制約
	ハードウェアに密着したプログラミング	ハードウェア協調
	開発環境とターゲットシステムの分離	開発環境
	ハードウェアとの並行開発とコデザインの可能性	開発環境
	多様なプロセッサや基本ソフトウェアへの対応	移植性・拡張性
3	自然法則を扱う	ハードウェア協調
	リアルタイム制御	時間制約
	制約要件　コスト，メモリ，その他リソース	リソース制約
	信頼性	信頼性
4	ハードウェアデバイスと連携して機能を実現する	ハードウェア協調
	システムの外部の状況をセンサなどを通して受け取る	ハードウェア協調
	決められた時間内の処理や微妙なタイミングを必要とする	時間制約
5	高信頼性への要求	信頼性
	社外と連携した開発形態	(ソフトウェア一般)
	開発規模の多様性	(ソフトウェア一般)
	納期の絶対的な厳守	(ソフトウェア一般)
	リアルタイム制御	時間制約
	法規対応	規約制約
	多様な使用環境	非機能要件

表 4: 技術的特徴の出現頻度

分類項目	出現数
ハードウェア協調	5
時間制約	4
信頼性	3
移植性・拡張性	2
リソース制約	2
開発環境	2
タスク制御	1
非機能要件	1
規約制約	1

2.2 これまでの組込みソフトウェア教育事例

前節で挙げた組込みソフトウェアの特徴に対して，これまでどのような教育がなされ，どのように評価してきたのかについて，事例を挙げて本研究の 4P モデルに照らして説明する．

2.2.1 事例 1

伊藤らは，携帯電話用ソフトウェアをテーマとした教育を報告している (伊藤, 渡辺, 樽松, 2007)．この教材は，本研究の 4P モデルに対して次のような特徴を持つ．

・プロダクト

組込みソフトウェアと称しているが，実際は BREW と呼ばれるプラットフォーム上でのアプリケーション開発である．そのため，ハードウェアとの協調といった，製品技術的な内容は扱われていない．このことは，今後の課題としている．

・プロセス

開発プロセスは，特に言及がないものの，ウォーターフォール・モデルであることが推定できる．工程という言葉は出てこないが，工程名は意識する教材設計となっており，各工程成果物としてドキュメント作成を行わせている．

開発プロセス導入は，信頼性ではなく，属人性の排除を目的としている．

・プロジェクト

実行計画の作成と進捗管理を行っている．

・プロフェッショナリズム

メンバ間のコミュニケーションについての言及がある．一方，主体性や粘り強さについては不明である．

・評価

アンケートを使った，学生による自己評価と，教員による成果物評価を行っている．いずれも，この教材が扱っているテーマである，携帯電話のアプリケーション開発に対する能力の評価となっている．

本事例は，産学連携で行われている．そのため，企業の技術者が普通に，仕事を進める上で必要と考える作業が内蔵されており，結果的に教育内容を本研究の 4P モデルの項目にうまく分類できる．このことは，逆に 4P モデルを軸に教材設計をするこ

とで，企業のエンジニアが期待するような内容に近接する可能性を示しているとも考えられる．

2.2.2　事例 2

花野井らは，現役技術者を招へいして実際の業務に即した模擬演習の報告を行っている (花野井, 有田, 他, 2007)．この教材は，本研究の 4P モデルに対して次のような特徴を持つ．

・プロダクト

特に言及はない．

・プロセス

要求仕様からの機能仕様書や，プログラム設計など，工程名に近い語彙が記述されているものの，開発プロセスという言葉はない．

・プロジェクト

実行計画に関し，スケジュール表を活用している．また，コスト管理に力を入れている．

講師やインストラクタが，上司や顧客役となることで，実際の業務に近いロールプレイを行っている．

・プロフェッショナリズム

コミュニケーションの重要性への言及があるが，具体的に何のコミュニケーションかは不明である．また，主体性や粘り強さについての言及はない．

・評価

受講生および講師・インストラクタに対してのアンケートで行っている．内容は，主に授業への感想である．

本事例は，コスト管理や小規模な組織の擬態など，プロジェクト面で特色がある．一方で，4P モデルに照らしてみると，扱っている素材は組込みソフトウェアながら，組込みソフトウェアの特徴を網羅しているとはいいがたいと考えられる．

2.2.3　事例 3

沢田らは，修士課程の大学院生向けに，組込みソフトウェアのプロジェクト開発をテーマにした教育を実施している (沢田, 小林, 金子, 中道, 大久保, 山本, 2009)．　こ

の事例では，(1) 開発プロジェクトの経験と，(2) 検証技術の重要性の 2 つを学ぶことを教育目的としている．この教材は，本研究の 4P モデルに対して次のような特徴を持つ．

・プロダクト

モデル飛行船という物理メカニズムや，無線通信といった要素技術を用いていると見られる．

開発環境に case ツールを導入している．

・プロセス

V 字モデルや開発フェーズに基づいた先進的な進捗管理を導入している．テスト設計に取り組んでいる．

・プロジェクト

教員がマネージャ役として進捗管理を行った以外の言及はない．

・プロフェッショナリズム

チームコミュニケーションについての言及があるが，主体性や粘り強さに関する内容はない．

・評価

受講生へのアンケートによって行っており，本演習を進めるにあたっての感想と，得られた知識について調査している．

本事例は，4P モデルに基づいて分析すると，プロジェクト管理に関する言及が少なく，教育目的 (1) に対してどのように教育が行われたかは文献上からはわかりにくい．一方で教育目的 (2) に対しては，ツールの導入や先進的な開発プロセスが考慮されていることが読み取れる．

2.3 従来教育と本研究教材との差異

前節に示した 3 つの過去事例と本研究の教材について，実践力に必要なテーマとして導き出した 4P モデルと，教育評価，また授業を教員のみで行えるかについて，どのように対応しているかを表 5 に示す．

プロダクト及びプロセスの詳細項目は，表 4 にまとめた内容をそれぞれ対応するカテゴリに振り分けている．また，プロジェクトとプロフェッショナリズムの各詳細項

目は，1.4で述べた通りである．評価については，授業そのものの評価と，授業による実践力の向上評価とに分けた．表の最後に，教員だけでの実施の可否についても記した．表中，○は教材として対応しているもの，△は論文に明記されていないが対応が推定できるもの，または記述はあるものの対応が弱いもの，－は未対応のものを示す．

最初に3つの過去事例について述べる．まず，すべての過去事例において，プロダクトすなわち組込みソフトウェア特有の要素技術への対応が希薄であることがわかる．

プロセスとプロジェクトについては，3事例とも開発プロセスの導入を積極的に行っている．特に，事例1に見られるように，教材設計に実務経験者が参画すると，開発業務に必要な作業（＝開発プロセス）を，経験則として取り入れる結果となりやすい．しかし一方でこのことは，必要なスキル項目の設計に実務者の文化的背景が影響しやすいという問題が考えられる．このことは，事例2との対比でもわかる．どちらも実務者が教材作成に参画しているが，それぞれの所属する企業規模や事業内容による差異が大きい．

うまくスキル項目を網羅できていたとしても，それはあくまで結果論である．網羅することやそれぞれのスキル項目に求めるレベルを考慮して設計したものとは言えない．これはコンピュータプログラムに例えれば，経験豊富な者が記述したものが結果的に構造化していたということに近い．結果，必要な内容の漏れがあっても気づきにくい．このことから，同様の実践力教育を設計しようとする者にとって，産学連携の名の下に実務担当者と協力しさえすれば良い教材ができるといった誤解を招く可能性が危惧される．

事例3は，実務担当者が参加していないケースである．プロセス設計については，学術的に深堀した緻密な内容となっているが，V字モデルにおいて実装に対して単体テストが対応していたり，仕様書と設計書が区別されていなかったりと，やや変則的であった．またプロジェクトという実際にプロセスで設計した作業構造を運用する内容について，一度の中間報告のみが報告されており，スケジュール運用上重要な予実管理については不明である．

4Pモデルでプロフェッショナリズムと呼称しているジェネリックスキルに関しては，どの事例もコミュニケーションのみが報告されており，やや弱さのある印象が感じられた．

評価については，例示したすべての事例では教材そのものの評価のみを行っており，その測定手法は主にアンケートによる．ただ事例 1 に関しては，中間生成物も含めて成果物について，受講者自身での自己評価に加え教員やインストラクタらによる評価も行うことで，一種のクロスチェックを行う工夫がなされている．技術的には大変有意義と考えられるが，同時に教員側の必要工数が増大していることが危惧される．

以上，従来の教育教材を，本研究で提案する 4P モデルと評価方法の視点に立脚して分析することで，それぞれの強みや弱みが浮かび上がることを示した．このことは，逆に 4P モデルを使って設計を行うことで，漏れを減らし意図した目的の教材を設計できることの証左と言える．

また，教育評価は，前に挙げた 3 つの事例以外でも，基本的に「実施した演習授業そのもの」に対する評価が行われている．技術職は，技術者個人の保有スキルに関わりなく，基本的に未知なる問題への解決が求められる職業である．その意味において，教材そのものへの感想だけでなく，教材を使って学んだことによってどのように受講者が進化したかを測定する方法が必要と考えられる．またその方法は，できるだけ簡便な方法であることが，教員の負担面からも望ましい．

授業の実施については，事例 1 と事例 2 が企業担当者を交えており，事例 3 および本研究は教員のみで実施している．企業担当者の参加は，授業により具体性が出るといったメリットもあるが，一方で 1.3 で述べたインターンシップの問題点同様，教育品質が参加企業や企業担当者に依存してしまうことが考えられる．

最後に，本研究で試作した教材について表 5 の最右桁に対応を示す．16 コマ程度の時間で，組込みソフトウェアのすべての技術的特徴を網羅し，なおかつ授業評価だけでなく実践力評価にも対応している．また授業に関しては，企業担当者を交えることなく，教員のみで実施可能である．本研究は，4P モデルを提案することで，誰にでも効率よく教材を設計できること，その教材で学んだことにより実践力が向上したかを測定することの実現を目的としている．

表 5: 教育事例と 4P 及び評価との対応

４Pと評価，実施		詳細項目	事例１	事例２	事例３	**本研究**
４P	プロダクト	ハードウェア協調	ー	ー	△	○
		時間制約	ー	ー	△	○
		リソース制約	ー	ー	ー	○
		タスク制御	△	○	△	○
		非機能要件	ー	ー	△	○
		規約制約	ー	ー	ー	○
	プロセス	信頼性 (開発プロセスに準拠する)	○	○	○	○
		開発環境	ー	ー	ー	○
	プロジェクト	スケジュールプラン に従って進める	○	○	○	○
		作業目標に対して 進捗を報告する	ー	ー	○	○
	プロフェッショナリズム	技術的課題を，自ら発見し 自力で調査・勉強できる	ー	ー	ー	○
		技術的課題に関して，周囲 と相談できる	○	○	○	○
		手がけた課題を，最後まで やり遂げる	ー	ー	ー	○
評価	授業評価		○	○	○	○
	実践力評価		−	ー	ー	○
実施	教員だけで実施の可否		ー	ー	○	○

第3章　実践力の研究

本章では，組込みソフトウェア技術者に求められる「実践力」についての研究について述べる．

3.1　関連研究

国内において，実践力という言葉を扱った文献は，少なくとも 12500 件発見することができる．このうち，ソフトウェアに関するものは 1040 件であり，うち 1010 件が，教育に関するものである (Google スカラー, 2018)．この中で，実践力そのものを定義した例は存在しないが，山本らが実践力を高める教育への取り組みを行い，その効果測定に取り組んだ事例がある (山本, 小林, 宮地, 奥野, 粂野, 櫻井, 海上, 春名, 井上, 2015)．

ここで山本らは，ブルームの認知過程次元（表 6）の 3 以上を目標とする教育内容が実践力の養成に相当するとしている．またその具体的教育内容は，レポート作成や演習であるとし，それらを受講した学生について行動特性の測定を行っている．結果，演習などの内容の差異に関係なく，前述の教育によって受講者の行動特性の向上を報告しており，これを以て実践力の向上があったとしている．

長谷川らは「知識やスキルのある人材でも，コミュニケーション力や問題解決力などのコンピテンシーが伴わないと知識やスキルを発揮し，社会や組織に貢献することはできない」と述べている (長谷川, 櫻井, 湯浦, 2014)．また駒谷らは,「主体的に物事を考え行動するスキルが身に付き，高度な実践力を身につけた」(駒谷, 田中, 北川, 2009)，など，実践力を扱っている文献では，基礎力を引き合いに出しその向上を論じているものが多い．

この背景には，高等教育の質保障という言葉のもと，ジェネリックスキルと呼ばれる汎用的な能力の育成が叫ばれるようになったことと関連している可能性が考えられる．ジェネリックスキルの代表例は，社会人基礎力（経済産業省）と学士力（文部科

表 6: ブルーム・タキソノミーの認知過程次元

1	記憶する	長期記憶からの回復
2	理解する	説明から意味を構築
3	適用する	手順の遂行・活用
4	分析する	構成要素の分析と関係性
5	評価する	基準や標準に基づいた判断
6	想像する	新しい構造の構築

学省）である．前者では「前に踏み出す力」,「考え抜く力」,「チームで働く力」の 3 つの能力を定義しており，さらに細目に分かれている．

また，日本経済団体連合会（以下，経団連）による産業界へのアンケート (経団連, 2011) において,「大学生の採用にあたって重視する素質・態度，知識・能力」という調査項目では，主体性，コミュニケーション能力といったジェネリックスキルが上位という結果になっている．このことは，経済産業省による「大学生の社会人観の把握と社会人基礎力の認知度向上実証に関する調査」(経済産業省, 2010) でも同様の結果となっており，ジェネリックスキル重視の風潮の背景になっている可能性がある．

3.2　他分野の先行研究

実践力教育としてしばしば用いられる手法の 1 つに PBL(Problem Based Learning/Project Based Learning) がある．これは 1969 年にカナダの McMaster 大学で Howard Barrows が，また 1980 年に Harvard 大学で New Pathway が，医学教育で実施した教育に起源を持つ．このことからも予想できる通り，実際に PBL や実践力教育に関する文献は医療や看護分野に多数先行研究が存在する．たとえばその中に，永田らが行った新人看護師の看護実践上の困難の分析がある (永田, 2005)．この中で永田らは，新人看護師が直面する困難として 18 のカテゴリを示し，さらにそれらを 6 つのコアカテゴリに分類している（表 7）．

この中で，コアカテゴリにおける援助技術を開発に，ケア提供を開発技術に，患者を顧客にと読み替えると，ソフトウェア開発現場で新人等が直面する問題とたいへんよく一致するように思われる（表 8）．表中，1〜3 が職務技術に関するものである．

表 7: 新人看護師の看護実践上の困難の説明概念

	カテゴリ	コアカテゴリ	
1	知識・経験不足による診療時援助技術実施困難	1	専門知識・経験不足による援助技術実施困難
2	知識・経験不足による生活援助技術実施困難		
3	知識・経験不足・過重業務による患者状態判断困難		
4	不慣れな業務体制・ケア進行過程による円滑な業務遂行困難		
5	知識・経験不足で自信がないことによる技術実施時の困惑・とまどい		
6	知識・経験不足で自信がないことによる技術実施への不安・恐怖感		
7	知識不足と不慣れによりケア・業務が思い通りに進行しないもどかしさ		
8	知識不足・経験不足により事故発生予測ができないことによる危険の誘発	2	知識・経験不足で予測ができないことによる危険の誘発
9	インシデント・アクシデント体験による精神的動揺		
10	ケア実施過程での自己の未熟さの自覚による否定的自己評価	3	ケア提供の未熟さによる自己への否定的評価との直面
11	ケア実施過程での先輩看護師との関りで感じる自己への否定的評価		
12	患者・看護師関係形成過程でのとまどい・緊張	4	多様な患者との人間関係形成過程での緊張
13	先輩看護師の看護実践や新人指導への疑問・とまどい・不安	5	職場の人間関係形成過程・サポート体制へのとまどい・緊張
14	先輩看護師との人間関係成立過程での緊張		
15	ケア実施過程での自己成長の自覚による自己効力感	6	ケア効果の確認、問題解決行動の実施による自己効力感の獲得
16	他者の肯定的評価による自己効力感		
17	自己の振り返りによる問題解決行動		
18	先輩看護師の助言による問題解決行動		

中でも特に１は最も多くの困難カテゴリを内包しており，業務を実践する上で影響の大きいコアカテゴリであることがうかがえる．また，コアカテゴリ１と２においては，（専門）知識と経験の不足が要因となっており，これがカテゴリ全体の半分を占めている．これらの結果は，前章の仮説の補強につながると考えられる．

3.3 ソフトウェア技術者の実践力

前述の行動特性やジェネリックスキル（以下，基礎力）が，社会で活躍するにあたって重要な能力であることは，文献を参照するまでもなく疑問の余地はない．実践力という能力の少なくとも一部であろうことは容易に予想できる．一方で，ソフトウェア開発を行う技術者の実践力を考えた場合，はたしてこのジェネリックスキルが最も重要かどうかという疑問は残る．基礎力を向上させれば，実践力ある技術者になると言えるだろうか．

齋藤らは，コミュニケーションという基礎力を重視した技術者教育を行った結果として,「事前に開発工程を学習し，１人でソフトウェア開発を完遂させる技術を養った後に教育を実施する事が，コミュニケーションスキルを高める教育をより効果的にで

表 8: 新人看護師の困難と開発者の困難の対比

	看護の場合	開発の場合
1	専門知識･経験不足による 援助技術実施困難	専門知識･経験不足による **開発の**実施困難
2	知識･経験不足で予測できないことによる 危険の誘発	知識･経験不足で予測できないことによる 危険**(リスク)**の誘発
3	ケア提供の未熟さによる自己への 否定的評価との直面	**開発技術**の未熟さによる自己への 否定的評価との直面
4	多用な患者との人間関係形成過程での緊張	多用な**顧客**との人間関係形成過程での緊張
5	職場の人間関係形成過程. サポート体制への とまどい･緊張	職場の人間関係形成過程. サポート体制への とまどい･緊張
6	ケア効果の確認, 問題解決行動の実施による 自己効力感の獲得	**開発成果**の確認, 問題解決行動の実施による 自己効力感の獲得

きることがわかった」と述べている (斎藤, 久野, 2013).

また，情報処理推進機構が実践的な教育課程を修了した卒業生に対して Web を用いたアンケート調査 (IPA, 2011) によれば，入社後特に役に立っている実践的な教育内容として次の 5 つが上位に挙げられている．

1. システム開発手法やプロセスに関する講義
2. 情報システム開発における各種ドキュメント作成
3. チームによる情報システムの開発（実装工程の経験）
4. 情報システム開発におけるスケジュール管理の実践
5. システム設計に関する講義

一般に実践力向上には PBL などの演習が効果的とされているが，それのみならず講義が 1 位と 5 位に入っていることは興味深い．いずれにしても，3 項目目にチーム開発という基礎力に近い項目が入っているものの，ほとんどは開発技術の手法に関するものである．

前章で述べた経団連のアンケート (経団連, 2011) では，基礎力重視の結果の次に大学教育に期待するものという調査結果も報告されている．それによれば，技術系・理科系学生に期待するものの 1 位は「専門分野の知識を身につける」であり，2 位は「論

表 9: 基礎力と開発スキルの組み合わせ

		基礎力	
		弱い	強い
開発スキル	弱い	×	×
	強い	○	◎

理的思考力や課題解決能力を身につける」である．つまり，技術系・理科系学生の採用において，広義での技術力は優先度が低いのではなくむしろ「あたりまえの能力」と考えられている可能性が高い．

ではソフトウェア技術者（を目指す者）にとって，基礎力と技術力のどちらがより実践力として重要と言えるだろうか．簡単な思考実験として，基礎力の強弱と，実際にソフトウェアを作る能力の強弱との関係を想定した場合,（表 9）のような判断となることは明白である．なぜなら，基礎力の強弱に関係なく，社会においてソフトウェアを作れない人は技術者と認められないからである．

以上のことから，基礎力以前に，まずはソフトウェア開発に関する基本的技術力を獲得しておくことが実践力として重要であるという仮説を導くことができる．

3.4 インタビューによる検証

前節で導いた仮説すなわち，ソフトウェア開発技術者の実践力は，一般に言われる基礎力よりも開発技術そのものの中にあるのではないかという点について，実際に社会で実践力を発揮しているひとへ，実践力とはどのような力と思うかというインタビューすることで検証を試みた．

一般的にこのような調査は不特定多数に対してアンケート形式で実施することが多い．しかし今回，あえてインタビューとしたのは，本人が社会で活躍している必要性があるためである．そもそも実践力を発揮していない人からは，有効な回答が得られないと考えられる．インタビューは，ソフトウェア開発者または開発経験者で，なおかつ SNS 等で情報を発信しつづけているなど社会で活躍している人を対象に，面談またはチャットを用いて実施した．回答者は 26 名であり，専門技術領域は IT 系から

表 10: インタビュー対象者の分布 (単位:名)

年齢層	30 代	40 代	50 代	60 代
男性	1	7	8	8
女性		1	1	

組込みまで各自異なるものの，次のような共通の性質を持っている．回答者の年齢分布は表 10 の通りである．

1. ソフトウェア開発に携わった，または今も携わっている
2. 社会で，積極的に情報を発信するなど活躍している
3. うち 24 名は，マネジメントの経験がある

3.5　インタビュー結果とその考察

インタビュー結果の分析として，テキストマイニングを行った．ツールには，ユーザーローカル社のオンライン・マイニング・ツール (ユーザーローカルテキストマイニングツール, 2007-2019) を使用した．また，マイニングを行うテキストは，事前に漢字表記とひらがな表記の統一，語尾の「思う」「思います」の削除，質問テーマなので当然出現頻度が高くなる「実践」という単語の削除を手作業で行った．また，課題は問題と同意ととらえ，問題に統一した．

テキストマイニングした結果を図 5 に示す．これは，インタビュー結果のテキストから，すべての品詞についてその出現頻度を示したものであり，文字が大きいほど，出現頻が高いことを示す．この結果からは,「できる」という動詞が際立って出現頻度が高いことがわかる．

図 6 は，動詞のみに着目した出現頻度についてグラフ化したものである．上位 12 位までを示しているが，やはり「できる」という動詞の出現頻度が圧倒的に高い．ちなみに図中のスコアは，テキスト中でその単語がどれだけ特徴的であるかを示している．通常はその単語の出現回数が多いほどスコアが高くなるが,「言う」や「思う」な

ど，どのような種類の文書にも現れやすいような単語についてはスコアが低めになる(マイニング結果のスコアについて,).

次に，図 7 に名詞の出現頻度を示す．出現頻度では「仕事」が最多であるが，一般的な名詞であるのでスコアは低くなっている．スコア，出現頻度ともに高いのは,「計画」「実行」「エンジニア」「能力」，そして「解決」であることがわかる．インタビュー中，これらの単語は，どれも技術活動に直結する内容で語られている．また，コミュニケーションや主体性といった基礎力に関する言葉は，少なくとも上位には登場していない．

表 11 は，本章冒頭で示した，最も出現頻度の高かった「できる」という動詞が，どのような名詞と共に使われているかを分析したものである．1 文中に動詞と名詞のペアが出現した数を，共起回数として示している．最も多い「問題」は，図 7 に示すように名詞の出現頻度としても高い．しかし,「問題」という単語はマイニングツールによって一般的な言葉と判断されているために，図 7 ではスコアは低くなっている．そこで実際のインタビューの回答中で,「問題」という単語がどのように使われているかを調査した．その結果,“ ソフトウェア開発における ”「問題点の抽出を行える」「問題を発見・分析・解決できる」「問題を洞察し問題の本質を把握する」といった具体的文意で使われており，決して一般的な表現の数ではなく，開発行為を意味する有意な検出結果であると考えられる.「解決」や「達成」という「問題」と対になる単語も高頻度で出ており，以上のことから「実際に開発を行える」ことを実践力として重視している傾向があると考えられる．また，計画とマイルストーンはほぼ同質の言葉であり，これらを合わせると 13 件となり「問題」につぐ出現頻度となる．問題の解決には，計画の立案や遂行「できる」能力を重視していると考えられる．一方で,「できる」と共起している名詞にも，上位には基礎力に分類されるものは現れなかった．

3.6 実践力についてのまとめ

今回の調査分析は，ソフトウェア開発者にとっての実践力の実体は開発技術領域にあること，計画を立て，それを遂行し，問題を解決できるすなわちプロダクトを作れることが，実践力として求められているという可能性を示したと考えられる．

ソフトウェア開発において，計画立案の基本が開発プロセスであり，遂行の手段が

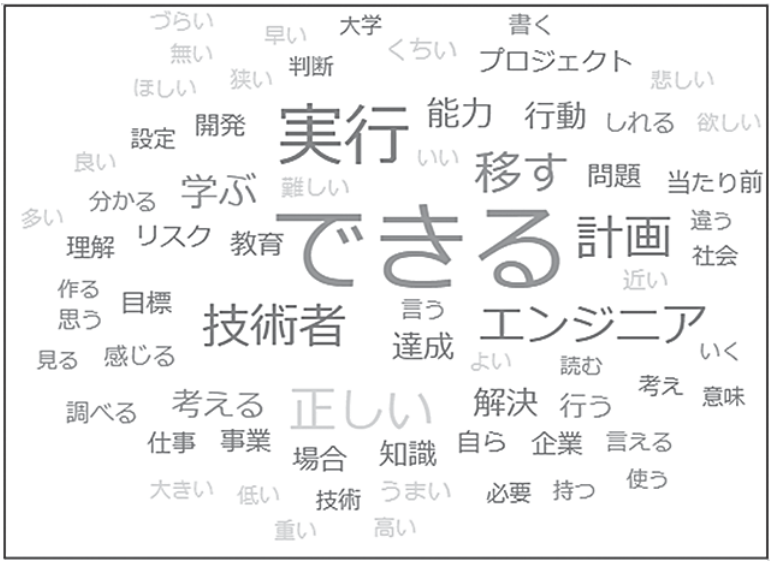


図 5: テキストマイニング結果の図示

開発プロジェクトである．プロセスが解るとプロジェクトが解り，プロダクトの作り方が解る．これら一連の作業を明示的に学び，練習することで，より効率的に実践力を向上させることができるのではないかと考えられる．

動詞	スコア	出現頻度
できる	7.10	88
考える	1.02	20
言う	0.15	15
思う	0.13	15
行う	0.55	11
学ぶ	1.66	9
使う	0.09	7
書く	0.17	7
感じる	0.29	7
いく	0.08	7
しれる	0.33	7
見る	0.02	6

図 6: 動詞の出現頻度

名詞	スコア	出現頻度
仕事	1.00	22
計画	7.07	20
問題	1.73	19
実行	11.98	17
行動	2.75	14
能力	3.46	14
場合	1.54	13
エンジニア	6.43	11
達成	2.94	11
解決	3.16	11
知識	2.04	10
開発	1.28	9

図 7: 名詞の出現頻度

表 11: 「できる」と共起している名詞

動詞	名詞	共起回数
できる	問題	16
できる	達成	12
できる	正しい	10
できる	解決	10
できる	仕事	9
できる	行動	7
できる	設定	7
できる	計画	7
できる	説明	6
できる	マイルストーン	6
できる	理論	5
できる	行う	5

第4章　教材の設計

前章までで，組込みソフトウェア技術者に求められる実践力を明らかにし，それを修得するための 4P モデルを示した．本研究では，4P モデルを実際の教材として開発し授業を試行している．本章では，開発した教材の設計について述べる．

4.1　教材と教育方法

4.1.1　教材の基本的考え方

教材は，表 5 に示した 4P の詳細項目を網羅することが必要である．一方で，授業時間や受講学生の能力範囲を考慮することもまた重要である．今回の設計ではまず，授業時間は 90 分 x16 時限を想定した．この時間で 4P の詳細項目を網羅しつつ完遂可能な開発案件として，次のような外形条件を仮定した．なお，入出力装置の条件に示した「1 種類で複数個」とは，装置の数は複数あっても，それぞれの制御方法は同一ということを意味する．

- プログラミング言語は C 言語
- プログラム規模は数百ライン程度
- 入力装置は 1 種類で複数個
- 出力装置も 1 種類で複数個
- リアルタイム制御を必要とする

対象学生は情報系学部生 3 年生を想定しているが，受講条件は C 言語を学習済みであることのみとしたので，特に学年等には依存しない．

4.1.2　教材のテーマ

開発課題について概要を説明する．学習に使用する開発課題は前述の条件を満たすものとして，キッチンタイマのソフトウェア開発とした．図 8 に，開発課題の装置のイメージを示す．本装置は，入力装置としてタクトスイッチを 2 個（1 種類を 2 個），表示出力として LED を 3 個（1 種類を 3 個）備えている．

入力について，SW0 のスイッチはタイマ機能のスタートとストップを行う．SW1 のスイッチは，タイマの時間を選択するもので，押すごとに 30 秒計測か 60 秒計測かをトグルに切り替えることができる．SW0，SW1 ともに，マイコンの入力ポートに接続している．

出力は，LED0 が電源投入状態を表示し，1 秒周期で点滅をする．LED1 は，先に説明した SW1 によるタイマ時間の選択状態を表示する．消灯していれば 30 秒計測，点灯していれば 60 秒計測を示す．LED3 はタイマの動作中を表示するもので，カウント中は 5 秒間に 2 回，0.25 秒ずつ点滅するという特殊なパターンで表示を行う．また，タイマのカウント終了時には，15 秒間 0.5 秒周期で点滅する．すべての LED は，マイコンの出力ポートに接続している．

このキッチンタイマの制御プログラムは C 言語で記述を行い，そのプログラム規模は 200 ライン程度を想定している．小規模ながら，先に表 5 に示した，組込みソフトウェアの技術的特性をすべて網羅している．対応の詳細は 4.2 で述べる．

4.2　4P モデルの実装方略

開発課題（キッチンタイマの制御ソフトウェア開発）が内包する要素技術と，表 5 に示した 4P モデルの詳細項目との対応について，その構成テーマであるプロダクト，プロセス，プロジェクト，プロフェッショナリズムの順に示す．

4.2.1　プロダクト

1. ハードウェア協調

   入力 1 種類，出力 2 種類のハードウェアを扱う．それぞれ動作の概要は前節の説明の通りである．

   (a) 入力：タクトスイッチ 2 個（SW0，SW1）

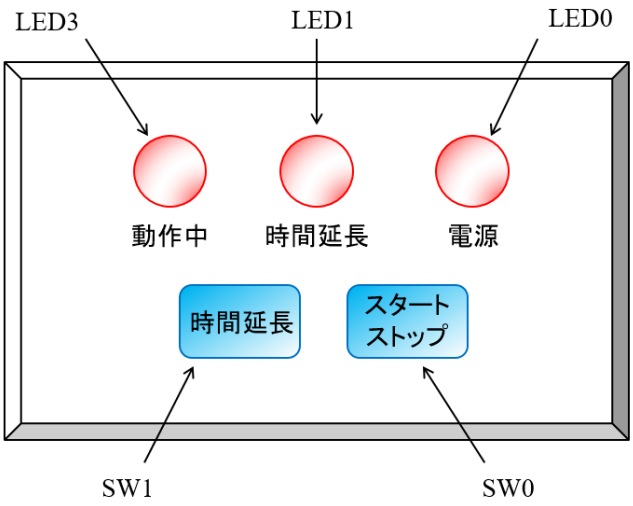


図 8: 開発課題の装置のイメージ

(b) 出力：LED0（点滅）

(c) 出力：LED1（点灯，消灯）

(d) 出力：LED3（消灯，点滅パターン 2 種類）

2. 時間制約

次の 5 つの時間的要件を与える．

(a) 単純点滅

(b) 特殊パターン点滅

(c) 動作時間の誤差指定

(d) 動作時間の切り替え

(e) 操作感（リアルタイム処理）

単純点滅とは，1 秒や 0.5 秒の周期でデューティ 50%の点滅を意味する．プログラムの状態や，人による操作に影響を受けては要件を満たせないので，特殊パターン点滅と同等に精密な設計を要求する．

動作時間の「誤差指定」は，大学の演習課題では要求されることが少ないが，実際の開発では必須の項目である． 動作時間の切り替えとは，開発課題で説

明したカウント時間を30秒と60秒で切り替える要件を意味する．　操作感は，たとえばスイッチを押してもすぐに動作しないとか，安易なチャタリング対策としてスイッチを離したときを動作トリガとする設計をしたときの違和感を意味する．これはリアルタイム処理の問題であると同時に，非機能要件の問題としても学生を指導することができる．

3. リソース制約

複数の時間軸が必要な要件に対して，使用できるタイマリソースは1つだけとなっている．たとえば，電源投入状態を表示するLED0は1秒周期で点滅するが，同時にタイマ動作時にはLED3が別の時間周期で点滅する．あるいは，スイッチの読み取りにタイマを利用したい場合もある．しかし，動作環境のマイコンで使用できるタイマは1つだけに制限している．このような状況は，実際の開発でもきわめて一般的である．

また，学生によってはスイッチ入力を割り込みとして使いたいと要望する場合がある．しかし動作環境ではスイッチによる割り込みはハードウェア的に使用できず，これも入力割り込みを使いたい学生にとってはリソース制限として働く．

4. タスク制御

異なる点滅パターンのLEDの出力タスクや，2つのスイッチ入力タスクなどを，同時に動作させる必要がある．一方でOSは使用しないので，タスク制御の原理的な動作を理解し解決する必要がある．

5. 非機能要件

スイッチのチャタリングへの対応を非機能要件として用意した．また，操作感という暗黙の要件も非機能要件として指導する．

6. 規約制約

コーディング規約を課している．ただし，対象としている大学2年生〜3年生は，C言語そのものに不慣れな学生も多い．そのため，コーディング規約は規

約を体験することに目的をおき，最低限の項目とした．内容は次のようなものである．

(a) インデントを適切に付ける

(b) 数値は，define や enum で定義して使う

(c) 二項演算子の両端はスペースを入れる

(d) 変数名は，意味のある名前にする

(e) 条件文で，実行文が 1 行だけの場合でも，でくくる

コーディング規約は，規約制約を学ぶということ以外に，プログラムコードの体裁のばらつきが減って，教員が指導しやすくなるという副次効果も期待できる．

4.2.2 プロセス

1. 信頼性

この項目に関しては，開発プロセスについて知り，それを遵守して演習を進めるよう教材を設計した．また，信頼性確保のためには各工程での施策が重要である．本教材では，1. 敢えてあいまいな仕様書を分析することで，最上流工程の重要性を学ぶ，2. 仕様からいきなりコーディングでなく，設計を行う　の 2 点を取り入れている．概要は次の通りである．

(a) V 字モデルに準拠したプロセスで進める．図 9 に本教材での例を示す．

古来，品質を重視する日本の組込みソフトウェア開発では，工程ごとに品質をチェックしやすいウォーターフォール・モデルが採用されてきた．V 字モデルはさらに，設計工程に対応した品質保証を明示的に表現できるよう改良したもので，現在でも多くの開発現場で利用されている．各工程の責務がわかりやすい利点があると判断してこのモデルを採用することとした．

(b) 仕様分析では，実際の業務同様に，漏れや矛盾のある仕様書を分析する．

(c) 設計を行う．設計では部品化の考え方を学ぶ．部品化しないと，そもそも単体テストという工程が成立しない．

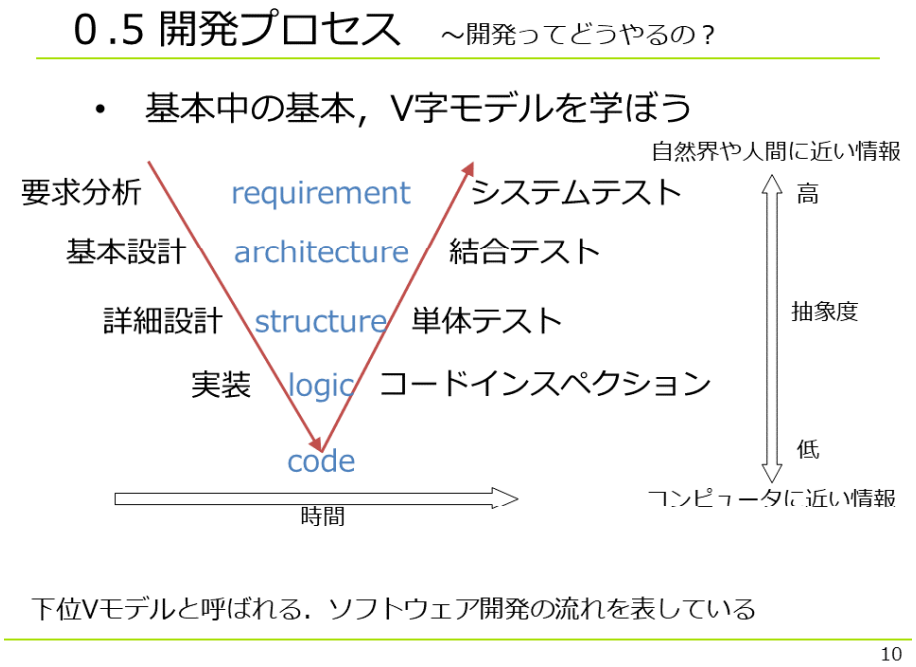


図 9: 本教材における V 字モデル

(d) 階層化構造を学ぶ．組込みソフトウェアでは，基本的に全階層を扱うので，設計を行う上で不可欠の知識と言える．

(e) 開発工程を通じ，漏れをなくすことの重要性を学ぶ

2. 開発環境構築

ほとんどのプロセスモデルにおいてこの工程は明記されないが，実際に開発を行う上では必須の工程である．一般に大学の演習では，開発環境はあらかじめ整備されていることが多い．一方，プロの開発現場では担当者が自ら構築することがあたりまえである．この工程には，純粋に開発環境を構築することに加えて，開発で必要になる未知の技術について，あらかじめ調査したり実験したりする作業も含めている．

(a) 開発環境は学生が自分で構築する（教師が準備しない）．

(b) 構築した開発環境を使って，使用する製品技術について事前学習する

4.2.3 プロジェクト

PMBOK（Project Management Body of Knowledge）(Project Management Institute, 2018) ではプロジェクトについて,「独自のプロダクト，サービス，所産を創造するために実施する有期性のある業務」と定義している．本教材では，特に有期性という点に焦点を当て，スケジュールプランに沿って作業を進めるということを体感学習させるようにした．

プロジェクト学習はマネジメントすることに重きが置かれがちだが，大多数の学生は就職後に担当者というマネジメントされる側からスタートする．従って，まずは決められたスケジュールを遵守しながら作業を進めるということを学ぶ必要がある．本教材では，次の 2 つのしくみを取り入れた．

1. スケジュールプランに従って進める

   ガントチャートという汎用ツールを学ぶ．ガントチャートは，縦軸に工程名，横軸に授業時間をとったシンプルなものとした．教師が作成し，毎授業ごとに実績を記入したものを示すことで，基本的な予実管理すなわち，自分は全体スケジュールのどこにいるのかを把握させることを狙いとしている．図 10 に，本教材実装におけるガントチャートの例を示す．開発環境構築工程の一部に，実績を示す矢印を記入した例である．

2. 作業目標に対して進捗を報告する

   作業報告は，あらゆるプロジェクトにおいて必要不可欠なマネジメント事項である．本教材では，毎授業時間終了ごとに，ふりかえりシートで報告をするしくみとした．報告内容は，次のようなものとした．

   (a) 今日の学習目標

   (b) 今日学んだこと

   (c) 今日できたこと

   (d) 今日難しかったこと

   (e) 今日できるつもりができなかったこと

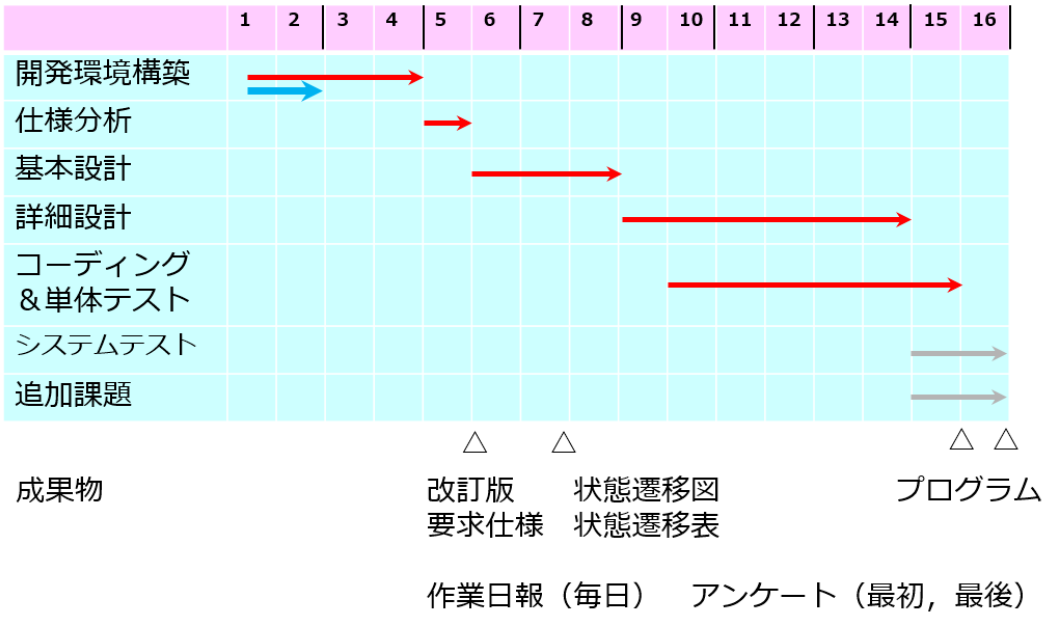


図 10: 演習授業のガントチャートの例

(f) 次にすること

(g) 所感

4.2.4　プロフェッショナリズム

1. 技術的課題を，自ら発見し自力で調査・勉強できる

   問題を主体的に解決することへの意識づけを目標とした．学生には,「自分でなんとかする，なんとかする力」というキーワードでモチベーションにつなげる．具体的方法としては，授業において教師や TA が主導的に説明することなく，学生にはインターネットや書籍での調査・勉強を積極的に推奨するという運用を行う．

2. 技術的課題に関して，周囲と相談できる

   上の項目と類似するが，周囲の人を情報源とする場合の取り組み方を学ぶことを目的とした．参加学生同士の相談を積極的に推奨し，また教師や TA への質問は，学生からする（教師や TA から声がけしない）といったシンプルなルールで行う．

仕様分析では強制的にグループワークを行わせ，他人の意見の重要性やレビューの意義を学ぶことも目指した．

3. 手がけた課題を，最後までやり遂げる

最終成果物だけでなく中間成果物の品質も重要であることを，社会ではあたりまえのこととして明示することで，作業責任の学びを狙った．また，作業が遅れている学生には教材を貸し出すことも，この項目への学びに有効と考えられる．

4.3 教材の構成

本研究で開発した教材は，1 つの開発環境（ハードウェア）と，ふたつのテキストで構成した．テキストの 1 つは，開発環境構築と，本編の演習で使う基本的技術についての練習を行う内容とした．本教材を「事前学習教材」と称する．もう 1 つのテキストは，開発演習を進めるためのものであり，これが本研究における開発教材の主体である．本教材を,「開発演習教材」と称する．

4.3.1 開発環境

開発環境のハードウェアには，Arduino Leonardo と専用シールドの組み合わせを用意した（図 11）．Arduino を選んだ理由は，Windows でも Mac でもほぼ同一の環境で開発可能であること，開発環境のインストールに手間がかからないこと，初心者でも習得が簡単であることである．

Arduino の入出力コネクタに挿して使う回路基板をシールドという．今回の教材では専用のものを作成した．搭載部品は，LED4 個，タクトスイッチ 2 個，DIP スイッチ 4 個，7 セグ LED4 桁，スピーカ 1 個，WiFi モジュール 1 個，I2C コネクタである．本稿の教材テキストでは，このうち LED3 個とタクトスイッチ 2 個のみを使用している．ただし，早期に課題を終えた学生のために，例えば 7 セグ LED を使って残り時間を表示するなどの追加課題も用意している．

ハードウェア教材の作成にあたっては，マイコン関連商品を扱うディストリビューターに，詳細設計～量産，および販売を委託した．本教材を利用するためのネックは，

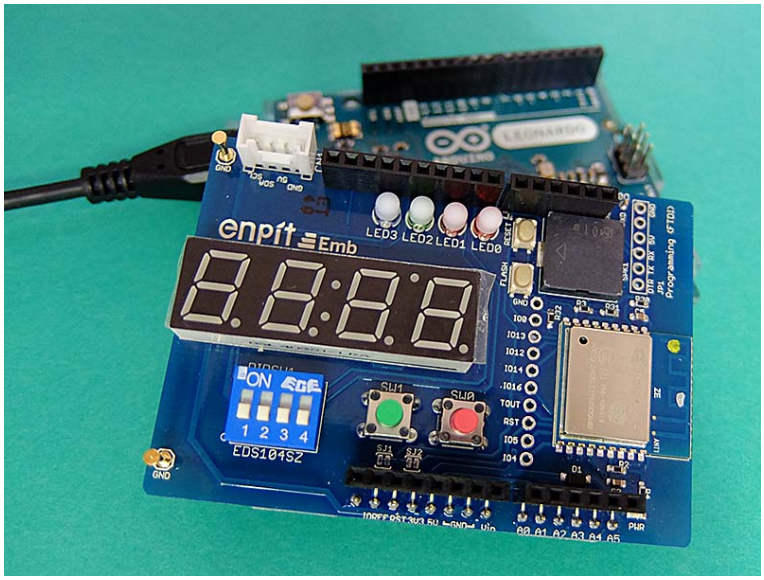


図 11: Arduino Leonardo + 専用シールド

この専用ハードウェアの入手にある．そのため，著名なディストリビューターの協力を得ることにより，その販売チャネルから誰もが利用できるようにすることを狙った．

専用シールド詳細

図 12 に，開発環境として設計した Arduino シールドを示す．本研究における開発課題だけでなく，幅広いテーマに応用できるよう工夫している．以下，各搭載デバイスについて説明する．

① は，I2C デバイスを追加するためのコネクタである．本シールドでは，⑨の 7 セグメント LED を I2C 接続としている．そのバスをそのままこのコネクタに出している．コネクタ形状は，seeed 社の Grove という実験用電子デバイスモジュールの規格に適合させることにより，同規格のセンサ類を使いやすくしている．(seeed, Grove)

②は，通常の LED である．LED0 から LED3 まで 4 個搭載している．LED0 と LED1 は赤色，LED2 は緑色，LED3 は青色と色分けしてあり，またそれぞれの明度が同じになるよう，電流制限抵抗値を設定している．本研究の開発課題では，このうち LED0 と LED1,LED3 の 3 個を使用している．

③は，ベースとなる Arduino マイコンボードの，信号及び電源端子をそのままピン

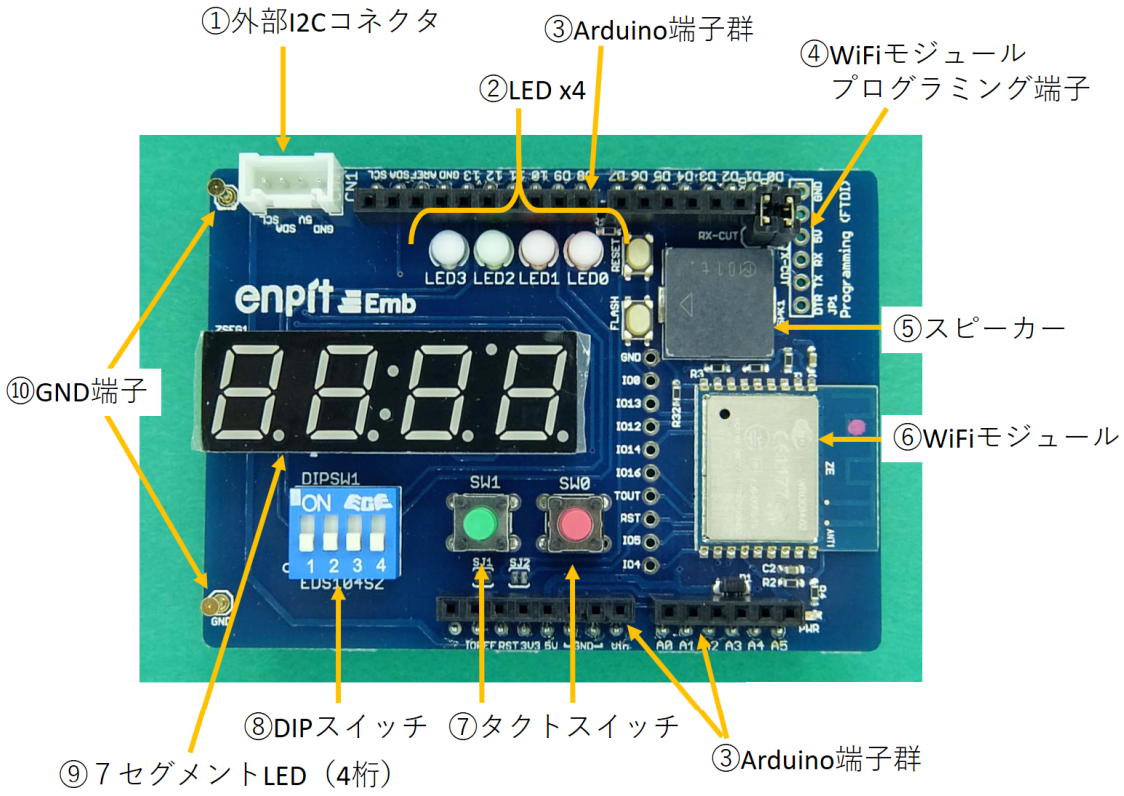


図 12: 専用シールド上の部品

ソケットで実装したものである．たとえば追加課題などで PWM による LED の明度変調など行った場合，このソケットを使ってオシロスコープで波形を観測させることが可能である．

④は，⑥の WiFi モジュールのためのプログラミング端子である．通常⑥の WiFi モジュールは，モデムとして動作するファームウェアを搭載している．しかし，場合によっては別のファームウェアを書き込みたい場合が想定され，その場合はこの端子を使ってファームウェアを書き込むことができる．通常は使用しないので，パターンのみ実装することとしている．

⑤は，表面実装型のスピーカである．追加課題などで，音を出すテーマは，本課題でやや疲弊した学生のモチベーションを回復させるのに役立つ．

⑥は，WiFi モジュールである．ESP-WROOM-02 という，本ボード開発時点でのデファクトスタンダード的な通信モジュールを使用している．これを利用することにより，容易に IoT 関連教材を構築することができる．実際に別の研究にて実施いてい

るが，詳細は本稿では省略する．

⑦は，タクトスイッチである．あえてハードウェアによるチャタリング対策は行っていない．SW0 と SW1 は，それぞれ Arduino の A0 と A1 という，通常はアナログ入力に使うポートに接続している．これは通常ポートの空きが不足したためであるが，プログラム上からはデジタル入力端子として初期化して使用することができるので，ソフトウェアの設計上特別な考慮をする必要はない．

⑧は，DIP スイッチ（スライドスイッチ）であり，ソフトウェアになんらかの状態を設定する課題を想定して 4 個搭載した．本研究の教材でも追加課題で使用しているが，本稿では省略する．

⑨は，7 セグメント LED である．通常の接続では Arduino のポート数が圧倒的に不足するため，専用の集積回路を使用して I2C バスで接続している．これを搭載した目的は，本研究の教材において 2 つある．

一つ目は仕様分析の失敗事例を学ばせることにある．本研究の教材は，仕様分析を重要視する構成としている．要求仕様では，タイマという時間計測装置ながら時間の表示は要求していない．しかし，開発環境に 7 セグメント LED という数字表示デバイスがあることで，仕様分析に失敗して時間表示があるものと思い込む学生が一定数現れる．そのことを教室で水平展開することで，仕様表記の重要性を伝えるツールとなりうるという期待があった．実際，これまでの授業試行において，2 割程度の学生が仕様分析において時間表示があるという誤解をしており，目的は達成している．

二つ目目の目的は，追加仕様として時間表示機能の実装を行わせるためであるが，単に機能追加ではなく，既存ソースの分析や必要な情報の収集を自力で行わせることを重視している．先に述べた通りこの表示機は，専用の集積回路を経由して Arduino に接続している．使用するためには，その集積回路を制御するプログラムを開発する必要がある．教材では，英文でヒントのコメントを入れたサンプルプログラムのみを提供する．学生は，それを読み解き，必要なライブラリを入手し，要求仕様に合致する制御処理を開発する必要がある．これは，単純ながら改造設計における実践的なプロセスのひな型に他ならない．これにより，限られた演習時間内で，さらに実践力の向上を図ることを狙った．

⑩は GND 端子である．波形計測では，計測プローブの GND の確保が意外なネッ

クであることが多い．そこで，長さ 15mm のやや大型のピンを 2 本実装することとした．

シールドの回路

図 14 に，シールドの全回路図を示す．Arduino には，バス電圧が 3.3V 系のものと 5V 系のものがある．そこで本シールド回路では，どちらでも使えるよう設計した．

WiFi モジュールである ESP-WROOM-02 は 3.3V で動作するので，供給電圧は 3.3V 固定としている．ESP-WROOM-02 と Arduino は UART の通信線である RXD と TXD で接続しているが，この信号電圧が Arduino のバス電圧に比例する．そこで，U3 に示す MOS-FET により，レベル変換を行っている．Arduino 側の信号は，IOREF という Arduino のバス電圧と同じ電圧が供給される端子によりプルアップを行って，信号線の安定を図っている．

また，7 セグメント LED の駆動 IC である HT16K33 は，LED もろとも 5V で動作するので 5V 固定で駆動している．ここに接続する I2C バスの電圧は Arduino のバス電圧に比例するため，U2 の PCA9515 という I2C バスリピータを使ってレベル変換を行っている．Ardiuno に接続する側のバスは，WiFi モジュール時と同様に，IOREF 電圧によってプルアップを行っている．

その他スイッチ類はすべて 10K Ω というやや小さめの値でプルダウンした．OFF 時は Lo，ON 時は High である．また Arduino 側には 100 Ω のターミネータ抵抗を入れて回路の Q を落とし，リンギング等によるラッチアップの対策を行った．

ハードウェアの故障は，授業の進行に少なくない悪影響を及ぼすので，できる限りの定石技術を投入しておくことが必要である．

図 13 に，全部品のピンアサインを示す．

ピン	機能	備考
D0,D1	ESP HW UART	
D2	N.C.	
D3	N.C	
D5	LED0	HIGHで点灯
D6	LED1	HIGHで点灯
D7	圧電スピーカ	
D8	スライドSW4	ONでHIGH OFFでLOW
D9	スライド SW3	ONでHIGH OFFでLOW
D10	LED2	HIGHで点灯
D11	LED3	HIGHで点灯
D12	スライド SW2	ONでHIGH OFFでLOW
D13	スライド SW1	ONでHIGH OFFでLOW

ピン	機能	備考
A0	SW0	通常LOW　押したらHIGH
A1	SW1	通常LOW　押したらHIGH
A2	N.C.	
A3	N.C.	
A4	N.C.	
A5	N.C.	
SCL,SDA	I2C	・7セグLED ・GROVEコネクタ
POWER （7pins）		

図 13: ピンアサイン

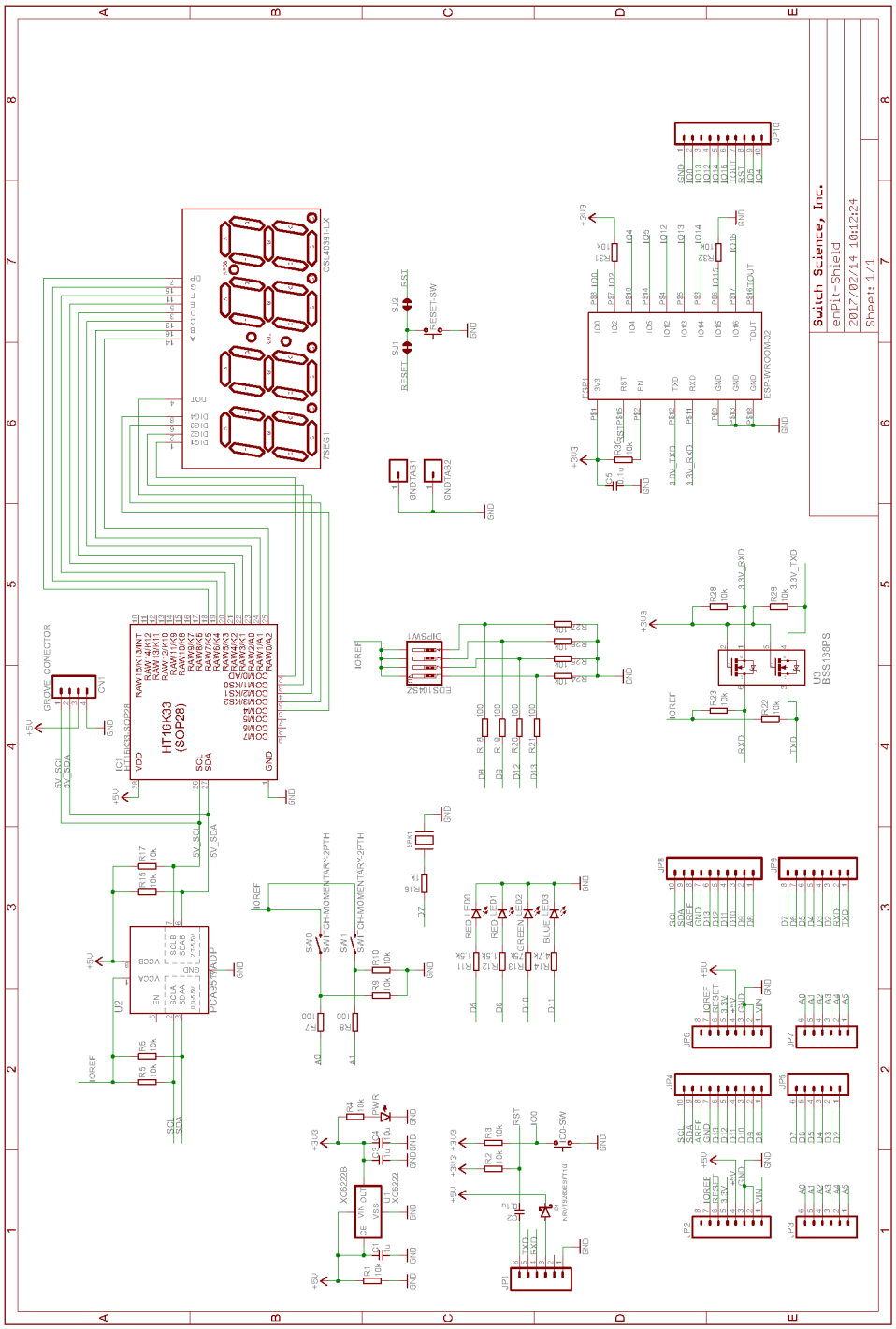


図 14: 専用シールド回路図

4.3.2　事前学習教材

開発環境の構築とその学習，またスイッチや LED といったハードウェア部品の基本的な使い方について，専用テキストを用意し，本開発演習の前に自己学習させることとした．詳細については 4.5.1 で述べる．

4.3.3　開発演習教材

本研究における，中心的教材である．信頼性や安全性を強く求められる組込みソフトウェアの特徴に基づき，開発プロセスに従って「きちんと作る」ことを学ぶことを最重要視した．そのための最初のしかけとして，一般的な授業の演習テーマと異なり，要求仕様をあえてあいまいな内容としている．これにより，要求仕様を斜め読みしてとりあえずコードを書くようなやり方が通用しないことを体験させると同時に，要求仕様を分析する，そして設計するといった，開発プロセスに従った作り方への導入を狙っている．この開発プロセスに従った作業を成立させることにより，各工程に有期性を持たせることができ，プロジェクトの一端を体験学習させることも狙った．結果，学生らによる自由記述の学習記録には，仕様分析や設計の重要性についての言及が多く現れ，それらには演習実施日のプロセス内容が反映しており，一定の理解が得られたと考えられる．その他演習結果の詳細については 5 で述べる．

4.4　授業の設計

本研究における，前述の教材を使った教育の基本設計について述べる．この設計には，教材設計マニュアル (鈴木, 2002) を参考にした．

4.4.1　教育方法

従来の，演習を伴う教育の取り組みに特徴的なことは，ほとんどが PBL として実施している点である．PBL は課題解決型のチーム学習であり，その学習効果については，これまで参照した文献以外にも多くの肯定的報告がある．一方で PBL は，参加するための最低限のスキルが要求される学習法でもある．

無手勝流でも課題解決に至ることも可能なため，今回のような基本の型を学習するというような目的には使いにくい．また，チームでの学習であるため，最低限のスキルに満たない人はうまく学べない場合がある．これについてはたとえば河西らは実験

結果から,「知識が乏しく自己学習習慣も身に付いていない学生の場合，能動的な学習や参加が求められる PBL のような授業にうまく対応できず，苦手意識や劣等感を抱きやすい」という仮説を述べている (河西, 丸山, 2010)．また松浦は，PBL の効果を述べながらも，前提知識の教育などの授業設計が重要である旨述べている (松浦, 2007)．

そこで本研究では，あえて PBL の形態とせず，講義＋演習という形で，開発の基本の型を学ぶところを志向することとした．

4.4.2 学習体制

教員 1 名に学生数十名の演習授業を想定した．また，今回の試行では，学生 10 名あたり 1 名程度の TA を配置した．開発環境は 1 人に 1 台で，個別学習である．周囲の学生同士相談することや，TA または教師に質問することは，積極的に推奨としている．また，インターネット等による調査も許可している．これらは，プロフェッショナリズムの行動目標を涵養することを目的としている．

4.4.3 前提条件

学習対象者は，情報技術系の大学学部生 3 年生としている．しかし，次に述べる前提条件を満たしていれば，特に制限を設ける必要はない．授業受講の前提条件は次の通りとした．

1. C 言語によるプログラミングについて学習済みであり，文法を知っていて，100 行程度のプログラムを自力で書くことができること．

2. 開発環境（Arduino (www.arduino.cc, 2019)）の使い方，UML (OMG, 2019) によるモデリング，組込み分野に特化したプログラム開発の経験については，必要な条件としない．

4.4.4 行動目標

教育モデルで定義した 4 つのキーワードごとに，次のような行動目標を定めた．

1. プロダクト

   次の 5 つの製品技術を学習し獲得する．カッコ内は対応する表 4 の詳細項目である．

(a) LED の制御方法（ハードウェア協調）

(b) スイッチの特性と使い方（ハードウェア協調，非機能要件）

(c) タイマの使い方（リソース制約，タスク制御）

(d) 複数のタスクを同時実行させる方法（タスク制御）

(e) 規約を遵守する（規約制約）

2. プロセス

(a) プログラムを作る基本的な手順としての開発工程（仕様分析，基本設計，詳細設計，実装，テスト）に従って作業を進められる．

(b) 開発環境を自力構築し，使用する製品技術について事前自主学習できる．

3. プロジェクト

(a) 開発工程毎に締め切りを意識して作業を進められる．

(b) 作業報告できる

4. プロフェッショナリズム

(a) 技術的課題を，自ら発見し自力で調査・勉強できる

(b) 技術的課題に関して，周囲と相談できる

(c) 手がけた課題を，最後までやり遂げる

4.4.5　評価条件

学生は，1 人に 1 つの開発環境（PC ＋マイコンボード）を占有できる．インターネットへのアクセスや専門書の参照は完全に許可される．他の学生との相談，教師・TA への質問は推奨される．ただし，教師・TA から直接解答は与えられない．

4.4.6　合格基準

各種図表などの中間成果物（設計成果物）と，最終プログラム及びその機能チェック項目（表 13）に対する網羅度を総合的に判断する．中間成果物は，教師によって自

由に設定可能である．以下に中間成果物の例を，また表 12 に中間成果物および最終成果物に関する評価のルーブリック例を示す．

- 事前学習の進捗結果

  事前学習問題には 6 つの演習問題と，追加問題（アドバンス）があり，それらをどこまでできたか．

- 仕様分析結果（読解網羅度）

  課題の仕様を分析する際，文面をラインマーカで塗りつぶしながら読んで読み漏らしをなくす方法を指導する．詳細は 4.5.3 で説明する．この作業の実施結果の提出を求め評価する．

- 仕様問題抽出数

  課題の仕様に含まれる問題点の抽出数で評価する．抽出内容は，上記の塗りつぶした仕様書へのメモ書き数で判断する．詳細な指導方法は 4.5.3 で説明する．

- モデル図

  基本設計では，状態遷移モデルを使用する．この際の作図内容を評価する．

- プログラム完成度

  表 13 に照らして，No.12 まで動作確認できたものを完成とした．さらに No.13 のチャタリング対策までできていればより高く評価する．

4.4.7 指導方略

本教材は，4.3.3 に述べたように，プロセスに則って開発を進めることを重要視している．そこで，授業全体を大きく「開発環境構築と基本要素技術の学習」「仕様分析」「基本設計」「詳細設計・実装・単体テスト」「システムテスト」の 5 つの開発工程に区切って，順を追って進める形とした．プロセスモデルには，最も基本的なＶ字モデルを採用した．工程ごとの指導方略は，4.5 にて実際の教材例を交えて説明する．

表 12: 評価ルーブリックの例

	5	4	3	2	1
事前学習教材	追加問題 完了	演習 6 完了	演習 5 完了	演習 4 完了	演習 4 未満
仕様読解 網羅度	完全	ー	漏れあり	ー	未実施
仕様の問題 抽出数	5件以上	ー	5件未満	ー	1件以下
状態遷移図	完全	ー	一部不足	ー	なし
プログラム 完成度	チャタリング 対策完	完成	最終 デバッグ中	部品は ほぼ完成	部品の 一部完成

4.5　教材の初期仕様

本研究で開発した教材に関して，具体例と個々の指導方略を述べる．教材は実際の開発プロセスに準拠して，開発環境（ハードウェアおよびテキスト），仕様分析，基本設計，詳細設計・実装・単体テスト，システムテストの各工程に明確に分割してあり，順に説明する．

4.5.1　事前学習教材

開発環境の整備と，開発演習で使用する基本技術の事前学習を目的としている．一般に学校での演習授業では，演習環境（開発環境）は事前に教員側によって準備されていることが多い．一方で社会での開発では，開発環境の構築と学習は重要な作業工程であり，担当者に自己解決が求められる．また，本研究の学習対象者である情報技術系の大学学部 3 年生は，その多くがスイッチや LED といったハードウェア部品のふるまいや使い方について経験が少ない．

そこで本教材では，開発環境や使用するハードウェアの学習に関する専用テキストを用意し，自己学習させることとした．授業冒頭でこのような自主的学習を行うことで，教えてもらうといった受動的学習から脱却し，自分でなんとかするという，主体性の萌芽を狙っている．これは，本研究の教育モデルにおけるプロフェッショナリズム修得の項目にも相当している．

テキストは全 54 ページで，まず開発環境のセットアップの方法から解説してある．事前学習教材では，まず冒頭で開発環境のセットアップを行わせる．以降は，開発教材のテーマに準じた技術解説と，それに対応した 6 つの演習問題を設けた．これら演習問題は，本教材での開発課題で使用する製品技術に対応しており，事前の学習だけでなく開発演習途中での参考書の役割も狙っている．各演習の内容は次の通りである．

1. 演習 1　サンプルコードをもとに LED の点滅を行う
2. 演習 2　入出力定義のヘッダーファイルを完成させる
3. 演習 3　2 つの LED を異なるタイミング（同期）で点灯させる．シリアルモニタでデバッグする
4. 演習 4　スイッチ入力プログラムを作成する
5. 演習 5　チャタリングを考慮したスイッチ入力プログラムを作る（サンプルあり）
6. 演習 6　タイマ割込みによる LED 点滅プログラムを作る

4.5.2 開発演習教材

本研究で開発した教材の中心的テキストであり，学生はこれに従って課題のソフトウェアを開発することで，行動目標への到達を目指す．以下，工程ごとに教材の設計内容を，指導方略とともに述べる．

4.5.3 仕様分析

仕様分析は開発工程の最上流であり，ここでの不具合は製品の品質に大きな影響を及ぼす．たとえば飯泉らの調査では，組込みソフトウェア開発で作りこまれた不具合のおよそ 40%が要求仕様に関わるものとされている (飯泉, 田所, 大立, 平島, 井上, 2008)．従って，実践力という観点において重要な学習課題であるが，一般に大学の演習授業における課題では，わかりやすく完成されており，学習項目として扱われないことが多い．

本教材では，あえてあいまいな記述や矛盾を含んだ要求仕様を意図的に使用することで，仕様分析の重要性を学ばせることを目的としている．学生に最初に提示する仕様を図 16 に示す．

仕様分析の指導方略は次の通りとした．

1. 製品のイメージ図の作図
2. 詳細解析
3. 数名でのレビュー
4. 全体での質疑

以下，各実施項目の詳細を述べる．

1. 製品のイメージ図の作図

最初に，学生に製品のイメージ図を書かせる．これは当初，仕様に含まれるランプやスイッチなどのアイテムとそれらの役割の理解を促すことだけを目的としていた．しかし，本教材の運用過程で，対象学生群に「仕様に記述されている外形である 3 個の LED と 2 個のスイッチを網羅できない」だけでなく,「仕様にはない数値表示器などを描く」といった現象が，少なからず出現することが発見された．対象学生群にもよるが，これまでの過程では，全体の 1/4〜1/2 の学生が時間表示器を描いた．おそらくは，タイマという名称や，教材ボードに搭載された 7 セグメント LED の印象によるものと思われる．

本教材の仕様は図 16 の通り，スライド 2 ページ程度の簡易なものである．従って，容易に内容をトレースしながら，想定できる外形仕様を説明し，学生に回答の正解度を認識させることができる．たったこれだけの文書でさえ読めていないことがあるという事実は，次に設けている詳細解析への導入として大変効果的に作用し，演習作業への取り組みが向上する効果が得られる．この点は本教材を用いる教育方法の工夫点の 1 つとなる．

2. 詳細解析

次に行う詳細解析では，各人で仕様書を再度詳細に読むよう指示する．この際，ラインマーカなどで記述を塗りつぶしながら読むことを求める．そして，疑問に感じたことなどを記入させる．この方法は「塗りつぶしチェック」と称し，高信頼性機器を開発している企業で実際に使われている方法である．

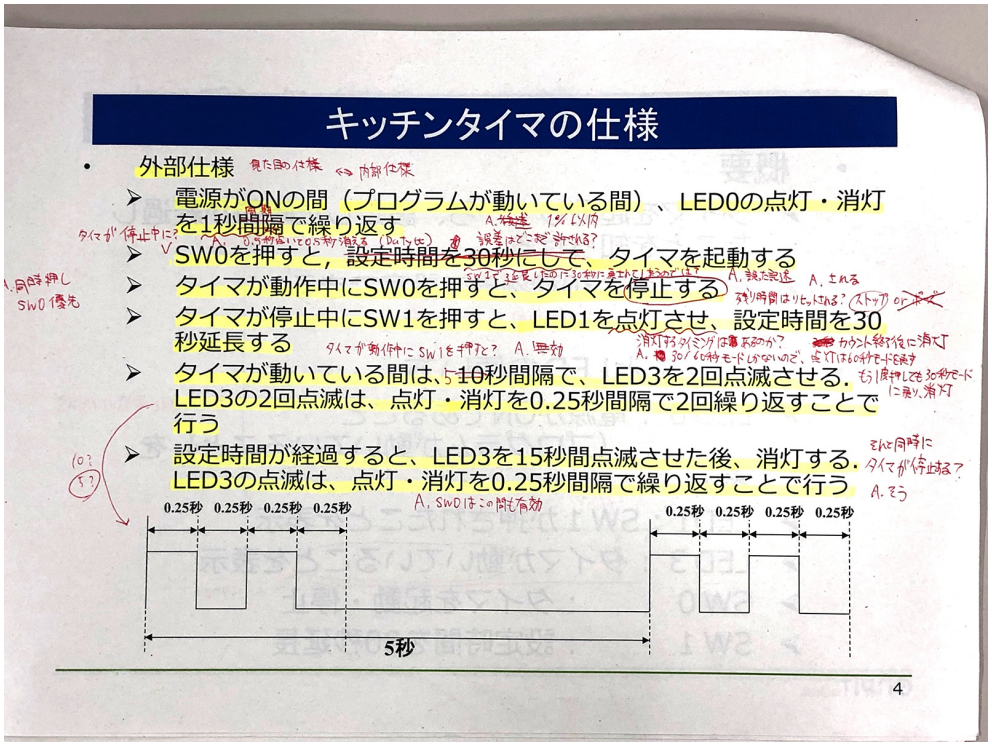


図 15: 塗りつぶしチェックで仕様分析を実施した例

初期仕様の不備について，たとえば図 16 の外部仕様の 4 項目目には「タイマが停止中に SW1 を押すと，LED1 を点灯させ」とある．しかし，タイマが動作中に SW1 を押した場合の動作については記載がない．また，外部仕様の 1 項目目には，LED0 の点灯消灯を 1 秒間隔で繰り返すとある．これは，点灯 1 秒消灯 1 秒なのか，あるいは 1 秒周期で 0.5 秒点灯 0.5 秒消灯なのか，わかりにくい．

図 15 に，実際に学生が仕様書の塗りつぶしチェックと疑問点記入を行った例を示す．

3. 数名でのレビュー

次に学生には，隣同士のペアで不明点や疑問点について話し合わせる．さらに，複数ペアでの 4〜6 名で話し合わせる．これにより学生は，自分が気づかなかったことに他人が気づいて，より多くの不明点や疑問点を抽出できるという，レビューの効果を実体験することを期待している．

4. 全体での質疑

最後に，グループごとに順に疑問点を発表させ，それに対して教師が顧客役となって回答を出す．学生はそれをメモして手元の仕様書を完成させる．単に与えられた仕様に従ってプログラムコードを書くのではなく，顧客と協力して仕様書を完成させるという行為は，実践力として意義が大きいと考える．最終的に修正した要求仕様内容の外部仕様を図 17 に示す．学生はこれに至った時点で，最初に提示された要求メモのまま開発を始めていたら大変なことになっていたことを体験学習する．

キッチンタイマの仕様

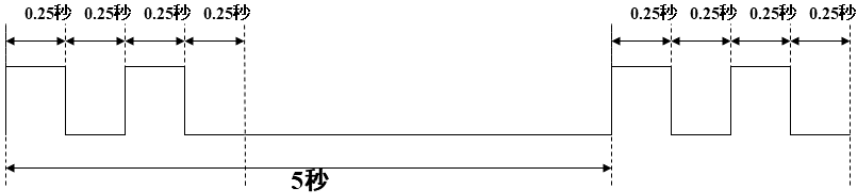
- 概要
  - タイマを起動してから、設定した時間が経過したことを知らせる
  - 設定時間は30秒刻みで設定できる

- スイッチとLEDの使い方
  - LED０：電源がONであること（プログラムが動いていること）を表示
  - LED1：SW１が押されたことを表示
  - LED３：タイマが動いていることを表示
  - SW０　　：タイマを起動・停止
  - SW１　　：設定時間を30秒延長

17

キッチンタイマの仕様

- 外部仕様
  - 電源がONの間（プログラムが動いている間）、LED0の点灯・消灯を1秒間隔で繰り返す
  - SW0を押すと，設定時間を30秒にして、タイマを起動する
  - タイマが動作中にSW0を押すと、タイマを停止する
  - タイマが停止中にSW1を押すと、LED1を点灯させ、設定時間を30秒延長する
  - タイマが動いている間は、5秒間隔で、LED3を2回点滅させる．LED3の2回点滅は、点灯・消灯を0.25秒間隔で2回繰り返すことで行う
  - 設定時間が経過すると、LED3を15秒間点滅させた後、消灯する．LED3の点滅は、点灯・消灯を0.25秒間隔で繰り返すことで行う



18

図 16: あいまいな要求仕様

4.5.4　基本設計

基本設計では，組込みソフトウェアの 3 つの技術要素に関し,「機能」に対する「ふるまい」と「構造」について学ばせることを目的としている．基本設計での記法には，UML(OMG, 2019) を使用した．基本設計の指導方略は次の通りとした．

1. 状態遷移図の学習
2. キッチンタイマの状態遷移図による設計
3. 状態遷移表の作表

以下，各実施項目の詳細を述べる．

1. 状態遷移図の学習

最初にシステムのふるまいを理解させるために，状態遷移図を書かせる．これは，組込みソフトウェアの設計において状態遷移を扱うことが最も一般的であることによる．そのことはたとえば，組込みソフトウェア技術者の教育を目的とした NPO である SESSAME(組込みソフトウェア管理者・技術者育成研究会, 2000) が定期的に状態マシン図セミナを開催していたり，JASA(組込みシステム技術協会, 1986-) においても専門のワーキンググループにて状態遷移設計の研究を行ったりしていることからも明らかである．ほとんどの学生が書き方を知らないので，信号機や音楽プレーヤなど身近な例をつかって「状態」について演習を行う．

2. キッチンタイマの状態遷移図による設計

次に UML における状態遷移図の書式について説明する．最低限必要な説明内容は次の通りである．

1. 開始から初期状態に遷移する
2. 状態には名前がある
3. 状態の中には，/entry /do /exit　という内部イベントを記述できる
4. ある状態から別の状態へは，イベントで遷移する

5. イベントにもアクションを記述できる

そして，課題の仕様について順を追って穴埋め式に状態遷移図を記述させる．たとえば，最初に図 18 の状態まで例示し，右の状態の四角の中や,「？」で示したイベントのとび先について考えさせる．最初のあいまいな仕様では，キッチンタイマの動作終了後の動作について記述されていない．これは仕様分析でも学生が見落とすことがある．しかし，状態遷移図を図 19 まで書き進むと，学生は右下の状態からの移動先が不明であることに気づく．このことから，状態遷移図を書くことで仕様の不備を発見できることも学べる．

3. 状態遷移表の作表

状態遷移図が書けたら，それをもとに状態遷移表を作成させ，状態遷移図の記述漏れを防ぐことも学ぶ．図 20 に，状態遷移表の書き方を例示しているテキストを示す．状態遷移表は書き方の法則が平易であるため，一か所例示するだけでほとんどの学生は書き方を理解する．

初期試行時のテキストでは，次にユースケース図やコンテキスト図を説明し，静的モデルを導き出すまでを講義する構成を採っていた．この段階で学生は，どのようなソフトウェア部品（以下，部品）を作れば，目的のソフトウェアを作成できるか，すなわちプログラムの構造を理解することができると予想していたためだった．

しかし実際には，ここでうまく理解できない学生が多く見られた．そこで次年度はユースケース図やコンテキスト図を用いず，ソフトウェアの階層構造図で全体の世界観を説明するよう改めてみた．その結果，基本設計で混乱する学生は減少した．詳細については試行結果の節で説明する．

4.5.5 詳細設計・実装・単体テスト

詳細設計では，組込みソフトウェアの特徴であるリアルタイム性と同時実行を学ばせることを主たる目的としている．これには，シーケンス図とフロー図を組み合わせた，シーケンス・フロー図という表記法を考案して使用した．組み込みソフトウェアの技術的特徴の 1 つは，厳密な時間制約があることである．これは，組み込みソフトウェアがハードウェアを使用して物理現象を処理する必要があるためである．しかも，時間制約に関連する技術は組込みソフトウェア技術の中で生産性に大きく影響を及ぼ

すことが知られている (IPA, 2017)．つまり，組込みソフトウェアエンジニアにとって，時間制約に対応する技術は非常に重要であると言える．そこで，本研究における教材の中でも，特に注力することとした．

基本設計の指導方略は次の通りとした．

1. シーケンス・フロー図による時間依存設計
2. ルールに従った実装
3. 単体テストの方法

以下，各実施項目の詳細を述べる．

1. シーケンス・フロー図による時間依存設計

この問題を授業で扱いにくい原因の 1 つは，初心者でも設計の考え方を理解できる，よい表記法がないことだと考えた．たとえば UML には，タイミングに関する表記法として，シーケンス図とタイミング図がある．しかしシーケンス図は，そもそも各オブジェクトが並行または並列に動作することが前提となっている．したがって，どのように並行動作をさせるかという原理部分の設計には不向きである．また，組込みソフトウェアで重要な情報授受のタイミングや，それに伴う処理内容まで表すことは得意ではない．

一方でタイミング図は，ハードウェアのそれ同様に，とても詳細な時間関係を表記することができる．しかしタイミングと処理との関係を表現することには不向きである．これらに代わって，オブジェクト間のタイミングと，各オブジェクトの処理内容を同時に表記して理解させる平易な方法が必要だった．

そこで，従来は通信プロトコルの表記に使うこともあったシーケンス図を，マイコン内の処理フローに応用した表記法を考案し，実際に授業で使用した．この表記法をシーケンス・フロー図と呼称する．

シーケンス・フロー図の書式は，基本的には通信で用いるシーケンス図と同様である．図 21 に基本形を示す．ここでは，部品 A と部品 B の 2 つのソフトウェア部品間の相互動作を表している．部品ごとには，動作フローを記述する．書式はフローチャートに準ずるが，基本的に逐次処理のみとする．反復や分岐を書くと，時間推移が直観

的に読み取りにくくなるためである．部品間の通信は，基本的には通知または要求と応答に 2 種類とし，部品のフロー軸から相手のフロー軸への横矢印で記述する．たとえば図 21 の場合，処理 10 の終了と同時に部品 B に要求を出し，それをトリガとして部品 B で処理 20 が行われる．また，処理 11 の終了による要求で処理 21 が起動し，その終了からの応答を待って処理 12 が起動している．このように，シーケンス・フロー図は，複数の部品間の同期動作をシンプルに表記することができる．

シーケンス・フロー図による指導方略

今回の課題における明示的な同時実行処理は，図 17 における LED0 の点滅動作と LED3 の点滅動作である．双方とも，リアルタイムな制御と同時実行する性質を持つ．

最初に，LED0 の点滅について，動作フローを図 22 のように例示する．次に，LED3 の点滅について，動作フローを考えさせる．特殊な点滅パターンのために一定数の学生に困惑が見られるが，ペアもしくはグループでの相談を導入すると解決しやすい．結果は，図 23 のようになる．

次にこれら LED0 と LED3 それぞれの動作フローを並べ，同時に動作させるにはどうしたら良いかを考えさせる．通常，ほとんどの学生が良い案を考えつけない．ここで，2 つのフローをむりやり合体させた案を図 24 のように例示する．この動作フローについて，学生は次のような特徴を容易に理解する．彼らは，動作はするが使えない処理というものの存在を知る．

- LED0 と LED3 は同時動作できそう
- かなり処理が複雑
- 点滅動作以外の処理を入れるのは困難

ここで，1 つの解決策として，LED0 と LED3 の 2 つの部品をマネジメントする別の部品を使ってみる案を説明する．図 25 に例を示す．この場合，LED 制御と命名されている部分が，マネジメントをする部品に相当する．LED 制御は，必要なタイミングに応じて LED0 と LED3 に点灯もしくは消灯の要求を出す．

図 25 では，一件処理が整理されたかのように見えるものの，今度はマネジメント部品が多忙であることについて，学生は容易に理解できる．そこで，タイマというマ

イコン特有の機能について解説を行い，タイマを使った場合のシーケンス・フローを図 26 を例示する．これらは図 25 の LED 制御部品をタイマに置き換えたものに相当するので，学生は容易に理解する．

そして，タイマ 1 つで，LED0 と LED3 両方を駆動する方法を考えさせる．この場合も，ペアまたはグループによる討議が有効なことが多い．導き出される回答は，図 27 または図 28 のようなものとなる．図 27 は，タイマ部品が LED0 と LED3 ぞれぞれの点滅タイミングを知っている場合であり，図 28 は LED0 と LED3 それぞれの部品自身が，自分の点滅タイミングを知っている場合である．どちらも正解なので，学生には自分で選んだ方法で，その後実装させる．

最後に，シーケンス・フロー図による詳細設計から，どのようにプログラムに実装するかを例示する．図 29 のように，特定の部品として最も簡単な LED0 の具体的な動作モデルを例示する．ここでは次の 2 点を重要ポイントとして教示する．

- 部品には初期化が要る
- 横軸の矢印（要求）を受け取るところを，関数の入り口と見なせる．名前を付ければ，それを関数名にできる

2. ルールに従った実装

企業における実装は，品質確保を目的として，なんらかのコーディング・ルールに準拠して実施する．通常は数百項目に及ぶことも少なくないが，本教材では体験することだけを目的とし，次の 5 項目を指示している．

1. インデントを適切に付ける
2. 数値は，define や enum で定義して使う
3. 二項演算子の両端はスペースを入れる
4. 変数名は，意味のある名前にする
5. 条件文で，実行文が 1 行だけの場合でも，{} でくくる

3. 単体テスト

本教材では，テスト内容よりも，いかに単体部品を動かして動作確認するかという，実作業方法の学習に重点を置いている．具体的には，LED0 に関しては部品を呼び出す方法を例示し，LED3 やスイッチの部品については，テスト方法を自分で考えさせ，実行させる．

4.5.6 システムテスト

システムテストでは，最後に検証をするということを学ぶことを目的としている．本来であれば単体テストの次に結合テストがあるが，課題のプログラムは小規模のため，本教材では省略することとし，ここではシステムテストのみを扱う．また，授業時間上の制約から，異常系動作については省略し，正常系のみの簡易チェックとしている．指導方略は次の通りである．

1. 表 13 のチェック項目に従ってセルフチェックさせる
2. 隣席など別の学生，または TA のチェックを受けさせる
3. 教員による合否判定を受けさせる

最後のチェック項目であるチャタリング対策は，都合よくチャタリングを起こすのが難しいため，実際に動作で確認するのは困難である．そこで，本項目に関しては，処理方法を説明させ，実際のコードを確認することで実施する．

4.6 教材設計のまとめ

設計した教材と，組込み技術の特徴やプロセス，プロジェクトツールとの関係を，図 30 に示す．4P の各項目ごとに，本章で述べた学習項目を設定している．それらは，組込み技術の代表的特徴を網羅している．また，開発プロセスには V 字モデルを用い，プロジェクトの計画についてガントチャートを使った．

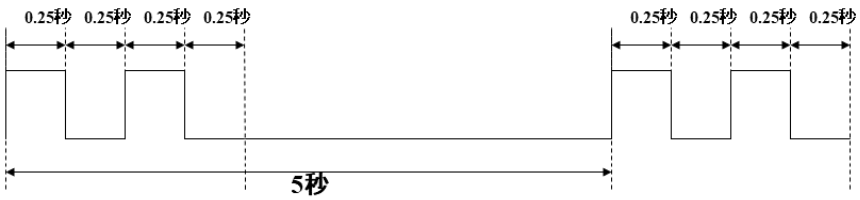
キッチンタイマの改訂仕様

- 外部仕様
  - 電源がONの間（プログラムが動いている間）、LED0の点灯・消灯を1秒周期で繰り返す．（0.5秒点灯，0.5秒消灯）
  - 電源をONにしたあとの起動状態を停止状態と称する．
  - タイマ計測時間の初期値は，30秒とする．
  - SW0を押すと，~~設定時間を30秒にして、~~タイマを起動する
  - タイマ動作中は，次ページのようにLED３を点滅させる
  - タイマが動作中にSW0を押すと、停止状態に戻る．
  - タイマが停止中にSW1を押すと、LED1を点灯させ、設定時間を30秒延長する．
  - 停止中以外にSW1を押しても無視する
  - 停止状態かつLED1点灯中にSW1を押すと，LED1を消灯し，設定時間を30秒に戻す．
  - 設定時間が経過するとアラーム状態となり、LED3を15秒間点滅させた後、消灯する．LED3の点滅は、点灯・消灯を0.25秒間隔（0.25秒点灯，0.25秒消灯）で繰り返すことで行う

27

キッチンタイマの改訂仕様

- 外部仕様
  - アラーム中にSW0を押すと，停止状態に戻る．
  - タイマ動作中やアラーム中から停止状態に戻るときは，LED３の点滅は停止してLED3を消灯する．また，時間延長設定は取り消し，LED1を消灯する．また，タイマカウントは０に戻す．
  - タイマが動いている間は、下図のように，5秒間隔で、LED3を2回点滅させる．LED3の2回点滅は、点灯・消灯を0.25秒間隔で2回繰り返すことで行う
  - 各時間の誤差は１％以内とする．



28

図 17: 修正した要求仕様

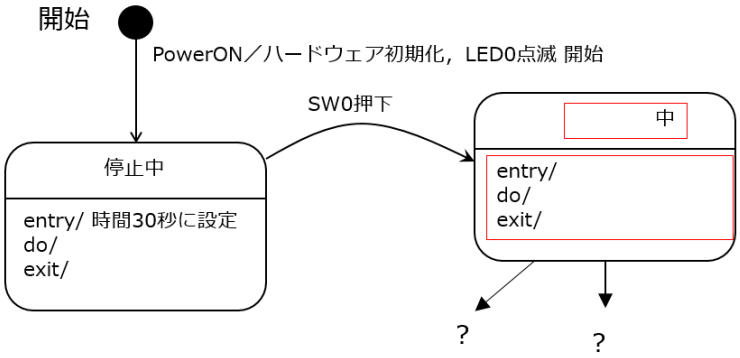


図 18: 状態遷移図 1

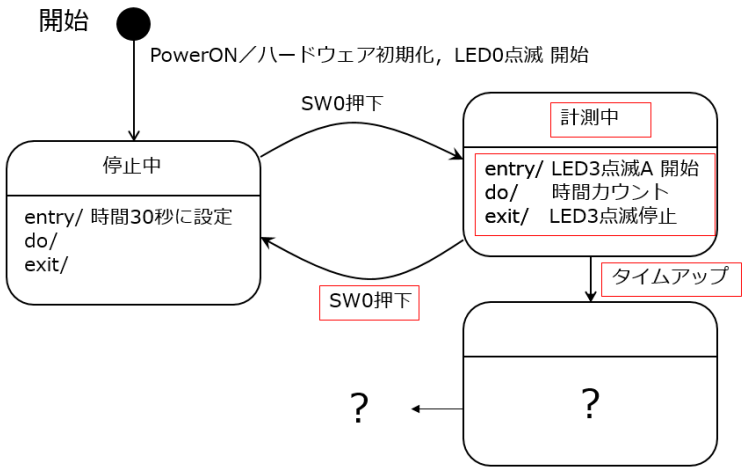


図 19: 状態遷移図 2

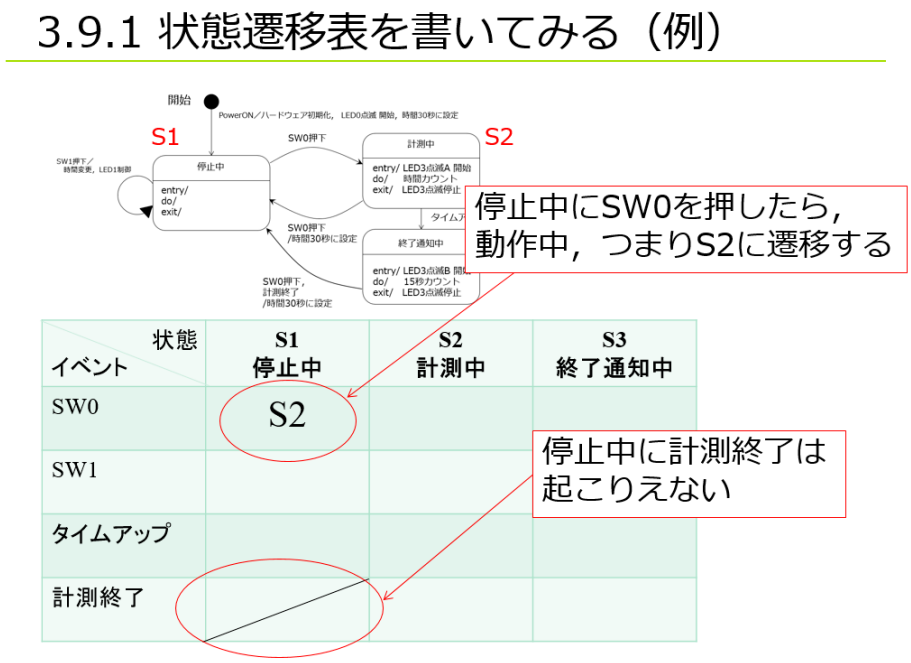


図 20: 状態遷移表の例示

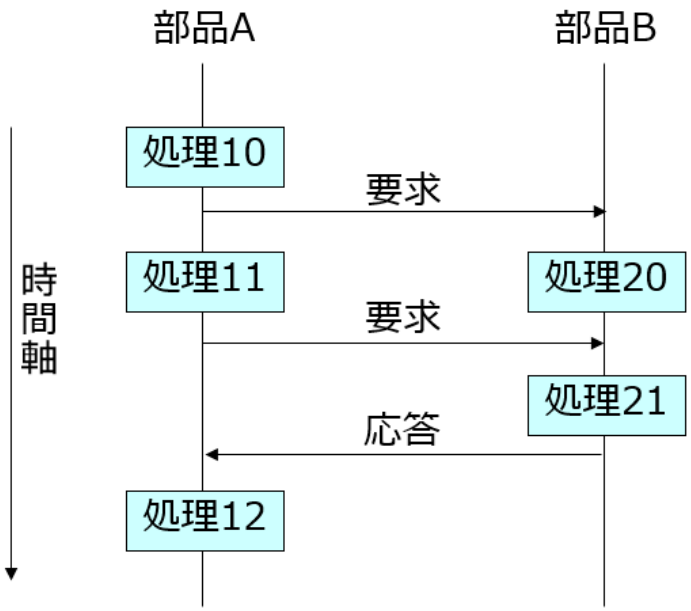


図 21: シーケンス・フロー図の基本形

図 22: LED0 の動作フロー

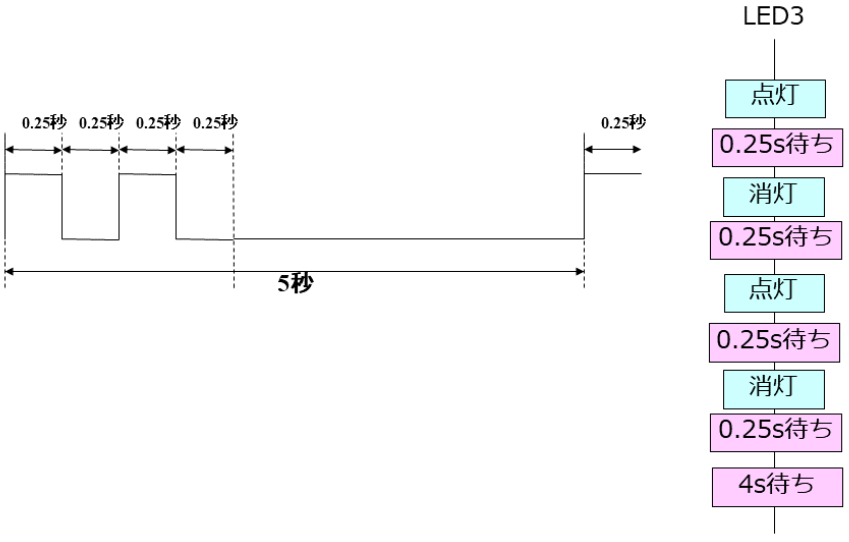


図 23: LED3 の動作フロー

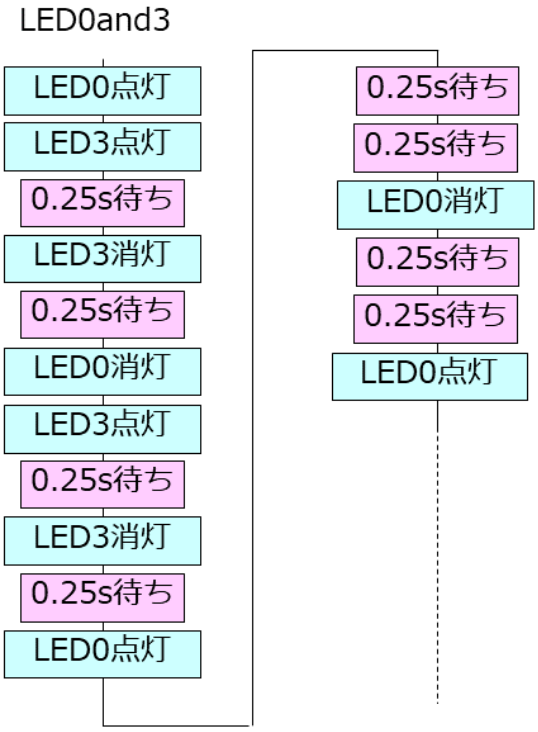


図 24: LED0 と LED3 の同時動作フロー

別の制御用部品を作ってみる

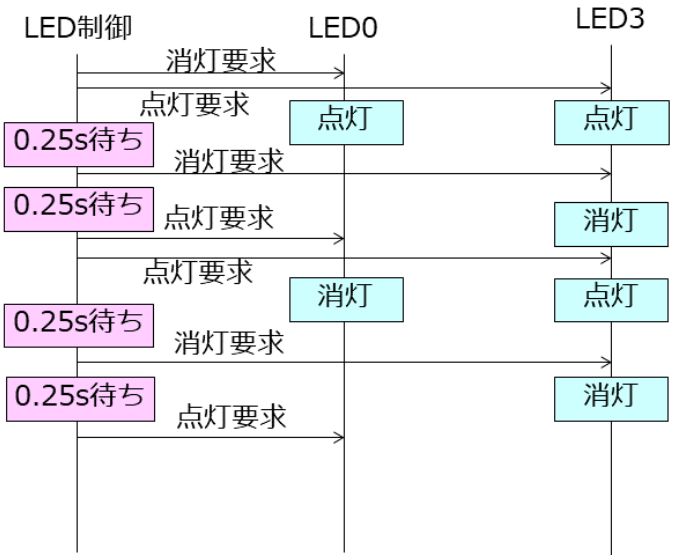


図 25: マネージャ部品を投入した場合

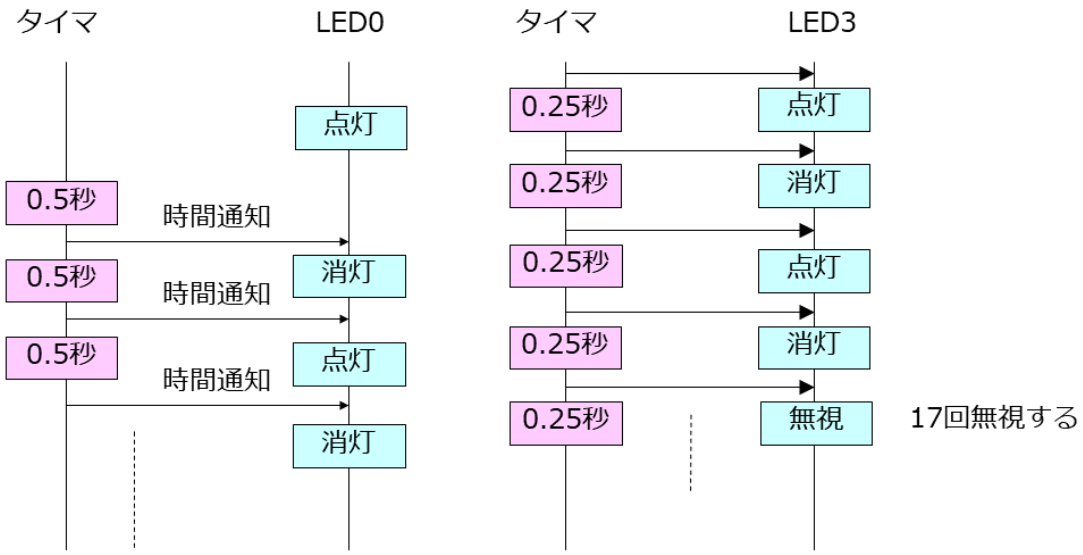


図 26: タイマによる駆動シーケンス・フロー図

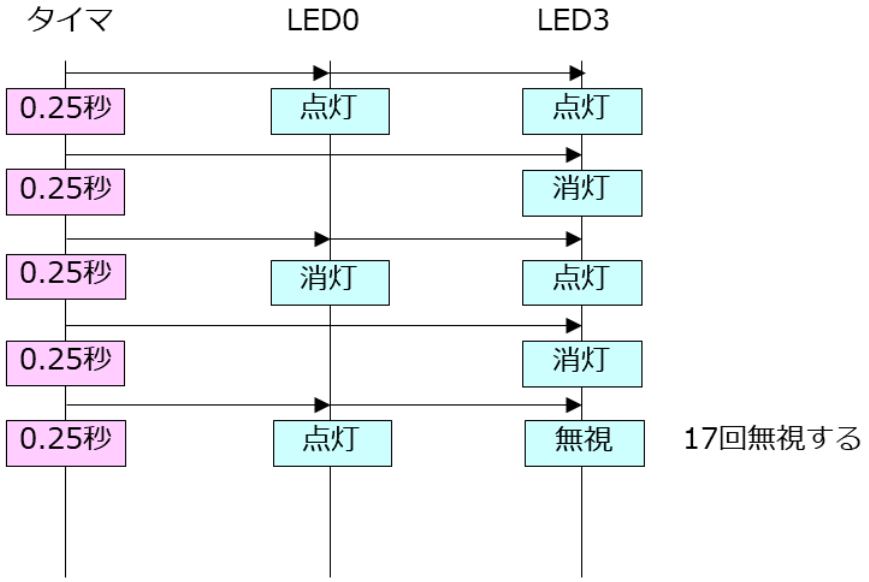


図 27: タイマによる同時駆動のシーケンス・フロー図　その 1

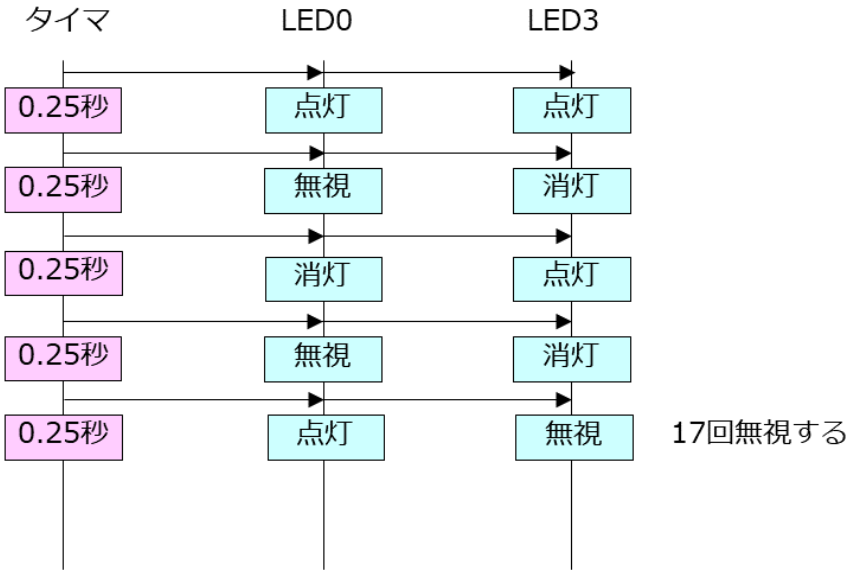


図 28: タイマによる同時駆動のシーケンス・フロー図　その 2

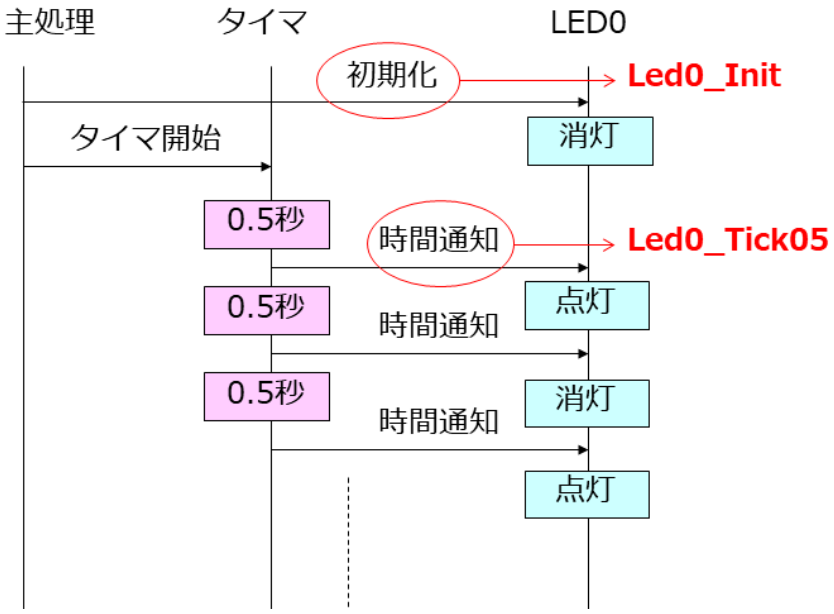


図 29: シーケンス・フロー図による部品動作モデル

表 13: 機能チェック項目

No.	チェック内容
1	LED0 が 1 秒周期で点滅する
2	LED3 が動作中パターンで点滅する
3	LED3 が 0.25 秒で 15 秒間点滅する
4	SW0 でタイマがスタートする
5	SW1 で LED1 の点灯と延長の設定を制御できる
6	LED0 と LED3 が同時点滅する
7	カウントアップで LED3 が終了の点滅をする
8	停止中以外に SW1 を受け付けない
9	カウント中に SW0 で停止できる
10	終了点滅中に SW0 で停止できる
11	停止状態に戻ったら時間延長をクリアする
12	時間を正確に測れる（誤差 1 秒以内）
13	スイッチのチャタリング対策ができている

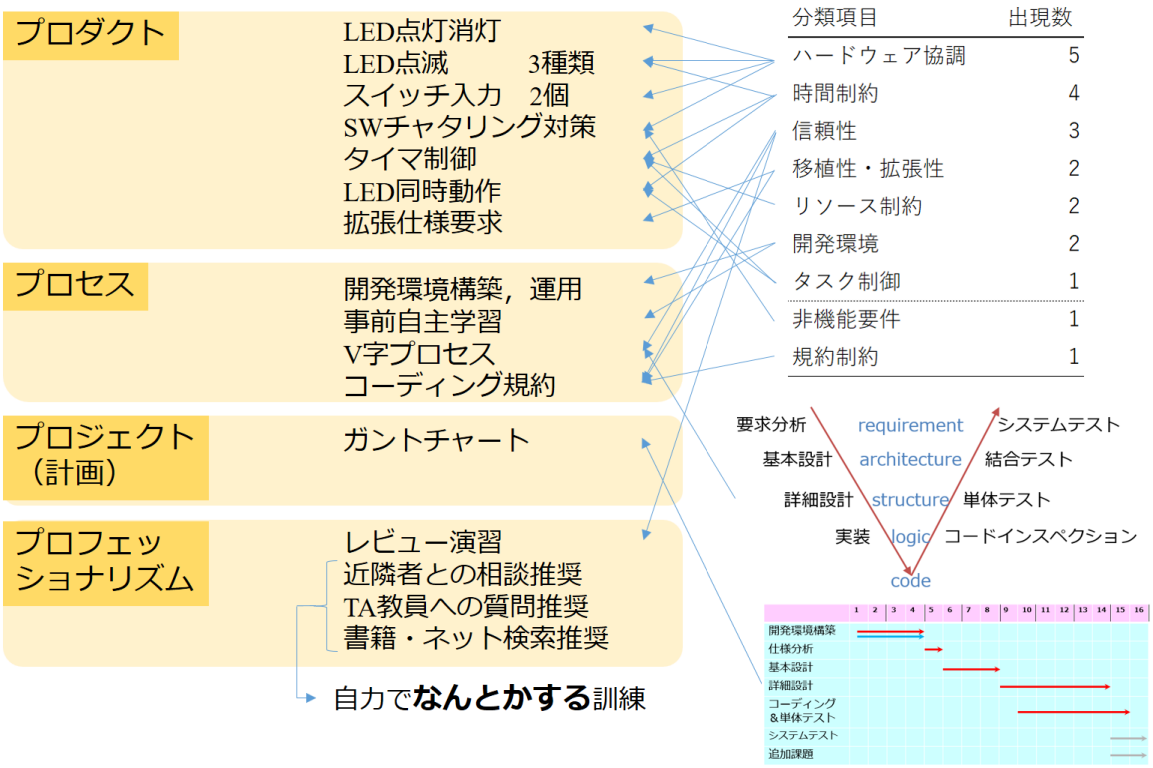


図 30: 4P モデル教材と技術の対応

第5章　提案教材による教育試行

4P モデルにもとづいて設計した教材について，実際に演習授業で使用することにより改良を進めることとした．まず 2017 年度に 2 回，2018 年度に 1 回の，合計 3 回の試行を実施した．次に試行による改良を行った教材で，7 回の実証実験授業（以下，授業）を実施している．本章では，試行に伴う学生の反応と，それに基づく教材改良の内容について説明する．

5.1　授業実施の概要

5.1.1　実施大学

本研究の教材による授業は，2017 年度から試行を開始し，2019 年度上期終了時点において 5 大学で 10 回，本研究の教材を使用した授業を実施している．実施大学の一覧を以下に示す．

- C 大学：私立大学．情報工学科．受講生は 3 年生上期
- B 大学：私立大学．情報工学科．受講生は 2 年生下期
- K 大学：公立大学．情報システム工学科．受講生は 3 年生上期
- M 大学：国立大学．情報学科．受講生は 3 年生上期
- S 大学：国立大学．電子情報システム工学科．受講生は 4 年生上期．

5.1.2　実験の時間的推移

表 14 に，これまでの授業実施の概要を示す．実施大学中，K 大学ではおよそ 60 名から 80 名の受講学生を 2 班に分け，順番に受講させている．ここではそれぞれの班を仮に A 班 B 班と呼称して区別している．2 つの班の学生は無作為に選ばれており，スキル分布に差異はないと考えられる．この受講生群の特徴を活かし，まずは K 大学で教材の試行と改良を行うこととした．改良目標は，4.4.4 で定義した行動目標の

表 14: 授業実施の概要

年度	大学	学年	授業時間	人数	実施内容	
17	K大学(A班)	3年上期	18 x 90分	30	1.テキスト初版で試行	試行 1
			暫定対策			
17	K大学(B班)	3年上期	18 x 90分	31	1.事前学習の時間を増加 2.フローチャートを書かせる試行	試行 2
			教材の改訂			
18	K大学(A班)	3年上期	18 x 90分	43	1. 詳細設計から実装への指導方法改善 2. スイッチ処理の指導方法改善 3. 開発プロセスの意識づけ強化 4. 部品化の考え方(階層構造)を追加	試行 3
			授業展開			
18	K大学(B班)	3年上期	18 x 90分	40	同上の内容で実施	実証実験授業
18	C大学	3年上期	18 x 90分	8	同上の内容で実施	
18	B大学	2年下期	18 x 90分	14	同上の内容で実施	
19	K大学	3年上期	18 x 90分	71	同上の内容で実施(別教員による試行)	
19	C大学	3年上期	18 x 90分	6	同上の内容で実施	
19	M大学	3年上期	16 x 90分	18	同上の内容で実施	
19	S大学	4年上期	16 x 90分	4	同上の内容で実施	

達成である．次に，改良した教材を他大学で使用することにより，教材の実証実験を行うこととした．

実施の推移について，時間を追って説明する．2017 年度にはまず K 大学（A 班）にて初回の試行授業「試行 1」を実施した．ここで検出した問題点に対し，暫定的に運用面での対策をして，同 K 大学（B 班）にて「試行 2」を実施した．2018 年度には，2017 年度に実施した試行で検出された問題点について，大きく 4 項目の教材内容の改訂を行い，試行 3 を K 大学（A 班）にて実施した．試行 3 において良好な結果が得られたため，これを本研究の教材の第一版とした．以降 2018 年度に K 大学（B 班），C 大学，B 大学にて授業を行った．

2018 年度までは，教材開発者が自ら授業を行っていたが，2019 年度には K 大学（A 班および B 班）において，「別教員による授業の実施」を行った．結果，一定の成果を以て授業を完遂できた．詳細については 5.9.4 にて述べる．2019 年度には，さらに C 大学，M 大学および S 大学においても授業を実施し，完遂した．2019 年度下期には

表 15: 事前学習教材実施の詳細

実施授業	大学	人数	開始	終了	コマ数	総時間(分)
試行 1	K大学	30	2017/4/25	2017/5/9	3	250
試行 2	K大学	31	2017/6/20	2017/7/4	4	340
試行 3	K大学	43	2018/4/24	2018/5/8	4	340

B 大学（およそ 40 名），また 2020 年度上期には K 大学において他教員による授業の実証の 2 回目を実施する予定である．その他，2019 年度同様，C 大学，M 大学，S 大学，B 大学でも，継続的に実施の予定である．これら大学においても順次，教材開発者とは別の教員への移管を予定している．

5.2　事前学習教材の試行と改良

まず，4.5.1 で述べた事前学習教材の試行結果と改良について述べる．これは表 14 における試行 1，試行 2 と試行 3 で行った．詳細を表 15 に示す．

当初試行 1 では，暫定的に 3 時限（270 分）を割り当てた．うち 20 分で事前ガイダンスを実施しているので，実質は 250 分である．その結果，6 項目ある演習問題のうち，最後の演習 6 まで終了したのは 30 名中 4 名で，大半の学生が演習 4 の途中か終了という状態だった．

そこで，試行 2 では 1 時限 90 分を増やして 360 分（ガイダンス 20 分＋学習時間 340 分）を割り当て，進捗状況を計測した．結果を図 31 に示す．横軸は人数で，縦軸は学習中の演習課題である．上に行くほど学習が進んでいることを示す．31 名中 14 名（45%）の学生が，2 日間 340 分で全演習を終えている．

また，演習 5 の途中～演習 6 の途中であった学生は 10 名であった．彼らに，完遂しなかった原因について聞き取り調査をした結果，全員がプログラミングに対する不慣れのため，初歩的なプログラムの記述や，それらに起因するトラブルシューティングで，無駄な時間を費やしていたことがわかった．したがって，このような層を救済すれば，31 名中 14 名+10 名の 24 名，77%が所定時間で終了できたことが予想された．

問題を抱え込んでしまって解決できないという問題は，4P モデルの 1 つとして 4.2.4

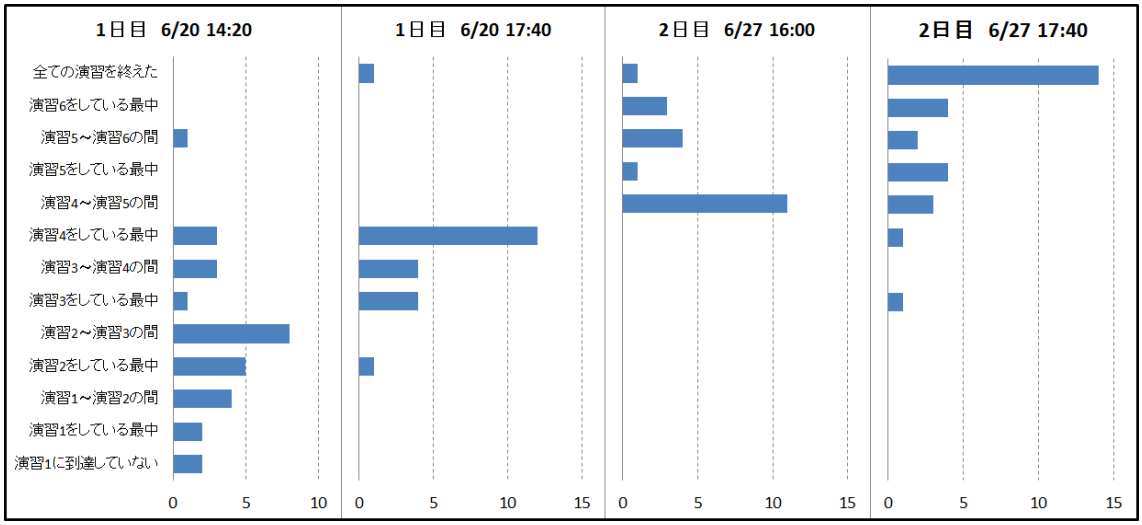


図 31: 開発した教材の実施状況

の (2) で定義した「技術的課題に関して，周囲と相談できる」に，まさに相当する．そこで試行 3 では，試行 1 と試行 2 以上に，積極的に「周囲との相談をする」「TA や教員に質問する」ことを強く推奨した．具体的には，15 分程度の間隔で相談や質問を推奨する声掛けをすることと，TA と教員の教室巡回頻度を増やした．ただし，TA や教員から学生に対して状況を尋ねることはしなかった．あくまでも，学生が自主的に問題解決に当たることを実践力として重視しているためである．

その結果，試行 1 と試行 2 ではそれぞれ 1～2 件であった学生からの質問が，試行 3 では 10 件に増え，隣通しの相談も活性化が観察できた．

図 32 に，試行 3 における事前学習教材の達成度を示す．規定の演習 6，または追加のアドバンス課題までの終了者は 82%であった．試行 2 における「周囲との相談の推奨」「TA 教員への質問」の推奨による完遂予想値 77%を上回っており，施策の効果と考えられる．

図 33 は，試行 3 とまったく同じ運用で，同年度の K 大学 B 班で授業を実施したときの結果である．演習完遂者は 79.1%など，試行 3 と同様の結果が得られた．

このことから，本研究の事前学習教材は，340 分程度の時間と「周囲との相談の推奨」「TA 教員への質問」の推奨により，実用性を発揮すると考えられた．

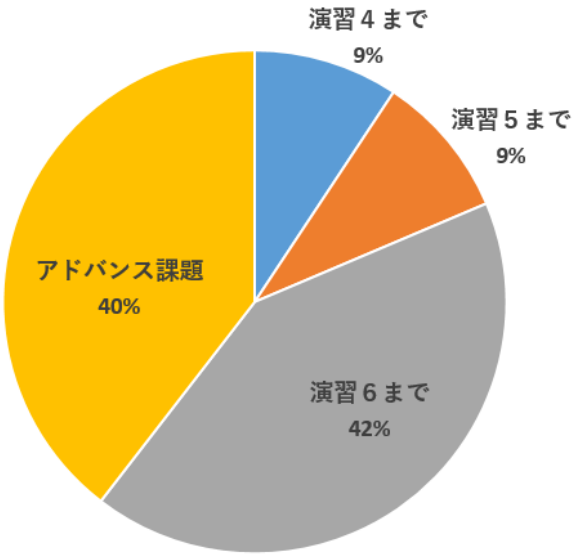


図 32: 試行 3 での事前学習教材の達成度

5.3　開発演習の試行結果

開発演習の結果を，表 16 に示す．評価項目は，表 12 のプログラム完成度に準拠し，レベル 4 以上を完成としている．試行 1 では，プログラムを完成できたのは 30 名中 5 名（16.7%）であった．最終デバッグ中が 6 名（20.0%），ソフトウェア部品はほぼ完成し，組み立て直前であったのが 8 名（26.7%）であった．これら 14 名（46.7%）は，さらに作業時間があれば完成に至った可能性はある．

一方で，必要な部品を作りきれなかったのが 11 名と，全体の 36.7%に上った．この 11 名中 9 名は，LED3 という特殊なパターンでのランプ点滅処理の作成時点から，先に進めることができていなかった．試行 1 での進捗の悪さは，授業観察からプログラミングへの不慣れによるところが大きいと思われた．そこで試行 2 では，処理手順をフローチャートもしくはアクティビティ図という形で記述させ，よく考えさせてからコーディングさせるという方法を取った．試行 2 の演習結果は表 16 の試行 2 列の通りである．完成までいった学生は 29 名中 15 名（51.7%）であり，部品完成まで行った者と最終デバッグ中だった者を含めると 27 名（93.1%）が，ひととおり設計に従ったコードを記述できていた．

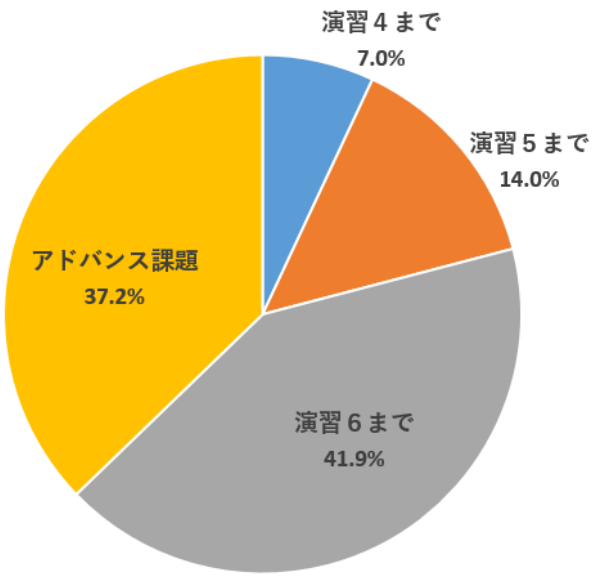


図 33: 実証実験授業における 2018 年度 K 大学 B 班の事前学習教材の達成度

表 16: 開発演習の達成度

状況	試行 1	試行 2
部品の一部のみ完成	11 36.7%	2 6.9%
部品はほぼ完成	8 26.7%	4 13.8%
最終デバッグ中	6 20.0%	8 27.6%
完成	5 16.7%	15 51.7%

5.4 開発演習の実施詳細

授業の経過について，学生による学習記録を表 17 に示す．これは各回の終わりに，その日学んだこととわかりにくかったことについての質問紙調査をまとめたものである．考察中の数値は，同様の意見の件数である．以下，回収した意見について試行 1 と試行 2 について，それぞれ考察を述べる．

5.4.1 試行 1 実施詳細

1 日目は，組込みシステムとはなにかと，開発プロセスの重要性について講義を行い，仕様分析の講義と演習を実施した．これに対して，学んだこととして仕様分析を

表 17: 試行1と試行2の学習内容アンケート結果

		試行1				試行2			
日	講義・演習内容	学んだこと		難しかったこと		学んだこと		難しかったこと	
1	組込みシステム	組込みシステム	7						
	開発プロセス	開発プロセス	6						
	要件分析	要件分析の大切さ，手法	24			要件の分析の大切さ	9		
		設計の大切さ	1						
				仕様書文章の読解	10			仕様書文章の読解	6
	基本設計			状態遷移図，表の作成	10	状態遷移図，表の作成	7	状態遷移図，表の作成	4
				状態の分け方	1				
	開発環境					開発環境の使い方	14	開発環境の演習	4
2	開発環境について自己学習					開発環境について自己学習			
3	開発環境(一部の学生)	開発環境の使い方	5			開発環境の使い方	6		
	基本設計	設計図，構造	24	用語(ミリ，チャタリング)	3	フローチャートの書き方	14	フローチャートの書き方	7
		実装方法	9	例図の読み方	1			状態遷移図との関係	1
		技術Tips	7	ハードウェアの内容	2				
	詳細設計			シーケンス図の書き方	11	シーケンス図の書き方	1	具体的処理の方法	2
								プログラミング方法	2
4	詳細設計	設計と実装の関係	20	設計と実装の関係	7				
		タイマの使い方	5			タイマの使い方	12	タイマの使い方	1
		スイッチの制御方法	8	スイッチの制御方法	3			スイッチの制御方法	2
		LEDの点滅制御	8			LEDの点滅制御	8		
				同時動作	3	同時動作	10	同時動作	2
				状態遷移図やシーケンス図	3				
	実装			プログラミングの仕方	9			プログラミングの仕方	5
5	詳細設計	設計と実装の関係	4	設計と実装の関係	2				
				同時動作	3	同時動作	3		
		スイッチの制御方法	8	スイッチの制御方法	8	スイッチの制御方法	3	スイッチの制御方法	4
				タイマの使い方	2				
				LEDの点滅制御	1				
	実装			プログラミングのしかた	4	プログラミングのしかた	5	プログラミングのしかた	2
6	詳細設計	スイッチの制御方法	3	スイッチの制御方法	8			スイッチの制御方法	2
		タイマの使い方	2	タイマの使い方	2			タイマの使い方	1
		設計と実装の関係	3	設計と実装の関係	3	設計と実装の関係	1		
				LEDの点滅制御	2	LEDの点滅制御	1		
						同時処理	2	同時処理	1
	実装			プログラミングのしかた	1				
	テスト	開発環境の使い方	3			開発環境の使い方	1		

挙げた意見が回答者29名中24件（82%）あり，この内容が印象深かったことがうかがえる．一方で状態遷移図や状態遷移表が難しかったという意見が10件（33%）あった．

次に困難点として多かったのが仕様書文章の読解で，9件と30%を占めた．簡易的とはいえ技術文書であり，しかもハードウェアに関連した内容を含むので，読み慣れていないという側面も否定はできない．一方で，文章力とプログラミング能力に関連があるという研究もあり (大場, 伊藤, 下郡, 薦田, 2018)，見落とせない結果の可能性が考えられるが，本研究では深追いしていない．

2日目は開発環境に関する自己学習を実施している．3日目は設計から実装までの講義と演習を実施した．学んだこととして設計図やソフトウェアの構造を挙げた意見が25件中24件（96%）あり，学習目的はよく理解されていることが推察できる．

難しかったこととしてシーケンス・フロー図によるタイミング設計という意見が11件（44%）あった．これには，単に図の書き方に慣れないというものと,（時間と部品との関係を）イメージしにくいというものとがあった．

一般的な情報技術系の大学学部の演習で，タイミング設計を扱うことは少ない一方，

関連業界における組込み系開発では必須技術であり，教育と社会の乖離の 1 つと言える．したがって，書き方も考え方も，さらにわかりやすく学習できるよう改良する必要があると考えられた．

4 日目は設計内容をもとに，実際にコーディングしてソフトウェア部品を作り始めた．学んだ内容については，実際のソフトウェア部品である LED やスイッチ，タイマといった項目に分散したきらいはあるが，おおむね設計結果の利用と部品化というテーマは伝わっている．

一方で，設計結果を実際にどのようにコードに変換するかということでつまずく学生が多くいた．プログラミングの仕方そのものを挙げたものが 29 件中 9 件（31%），設計と実装の関係を挙げたものが 7 件（24%）あった．重複もあるものの，計 16 件（55%）の困難報告であり，半数の学生がここでつまずいたことがうかがえる．

内容を見てみると.「（サンプルの）スイッチ処理関数がなにをしているのかわからなかった」「部品に必要なことをどこまでプログラムにすればよいかわからなかった」「ループ文の中でどのようにすれば LED の 3 つの動きを同時に操作できるか分からなかった」など，プログラミングに不慣れなことが要因と思われる意見が多かった．これはそのまま 5 日目 6 日目にも影響を及ぼし，先に挙げたように完成に至った学生が参加者 30 名中 5 名（16.7%）という結果に帰結している．

5.4.2 試行 2 実施詳細

組込みシステムの概要についての講義を一部削減したものの，仕様分析にかける時間は変更していない．しかし，1 日目最終時限から開発環境に関する演習をしたことから，印象の中心はそちらになってしまっている．学生は一般に，プログラミング作業に気をとらわれがちなので，分析や設計といった工学的取り組みは，日を改めるなどメリハリを付けた授業設計が望ましいと考えられる．

3 日目にフローチャート作成指導を実施した．今回は，図 34 のように，基本設計で作成した状態遷移図に対応させる形で，各状態の処理フローを考えて結合させるという方略をとった．その結果，学習記録には学んだこととしてフローチャートが挙がった一方で，難しかった点にもフローチャートの書き方に関するものが 14 件中 10 件（71%）挙がった．実際に，書き始めた当初はほとんどの学生がまともにフローチャー

トを書けていなかった．たとえば処理に下からフローを入れる，条件分岐でないのに分岐しているといった具合で，処理手順以前の問題が多く見られた．

書き方については，受講学生全員に対して，例示して簡単な指導を行うことで改善がみられた．しかし次には，処理手順そのものの問題，たとえば脱出できないループであったり，重複する処理を書いていたりといった問題が多数みられることとなった．これに対して，やむをえず個別指導を行ったが，これには学生の書いたフローを瞬時に解読して問題点を指摘するスキルが求められた．つまりは指導教員のスキルに強く依存することとなり，教材設計として好ましくない．次回改版時の要改善ポイントと考えられた．

4日目以降は，各自プログラミング作業を進めた．4日目に17件中5件（29%）と目立ったプログラミングに関する困難は，5日目には13件中2件（15%）となりその後消えている．授業観察によれば，自力で書き方を調べたり，他学生に相談したりして解決できていた．

一方で，スイッチの制御方法に関する問題は，最終日である6日目になっても一定の割合で残っていた．アンケート回収率が4件と極端に低い中2件（50%）の意見があり，実際に授業観察でも完成に至らなかった学生の大半が，スイッチ制御で困っていた．このことは，先に示した開発環境の演習の進捗に現れた結果と一致する．スイッチ入力の設計には，もう少し時間をかける，解説を充実させるなどの改善が必要と考えられた．

5.5　行動目標に対する試行1と試行2の結果

開発工程に従う組込みソフトウェア開発演習について，教材の概要と試行結果を述べた．試行1および試行2を通し，4.4.4で定義した行動目標に対しては，次のような結果を得た．それぞれ項目冒頭の記号について，[◎] は目論見通りだったもの，[○] は，ほぼ目論見通りながらさらに改善したほうがよいもの，[×] は要改善のものである．

1. ［◎］目標とした5つのプロダクト項目に対し，「Arduino開発環境の使い方」「LEDの制御方法」は，全員が達成できた．

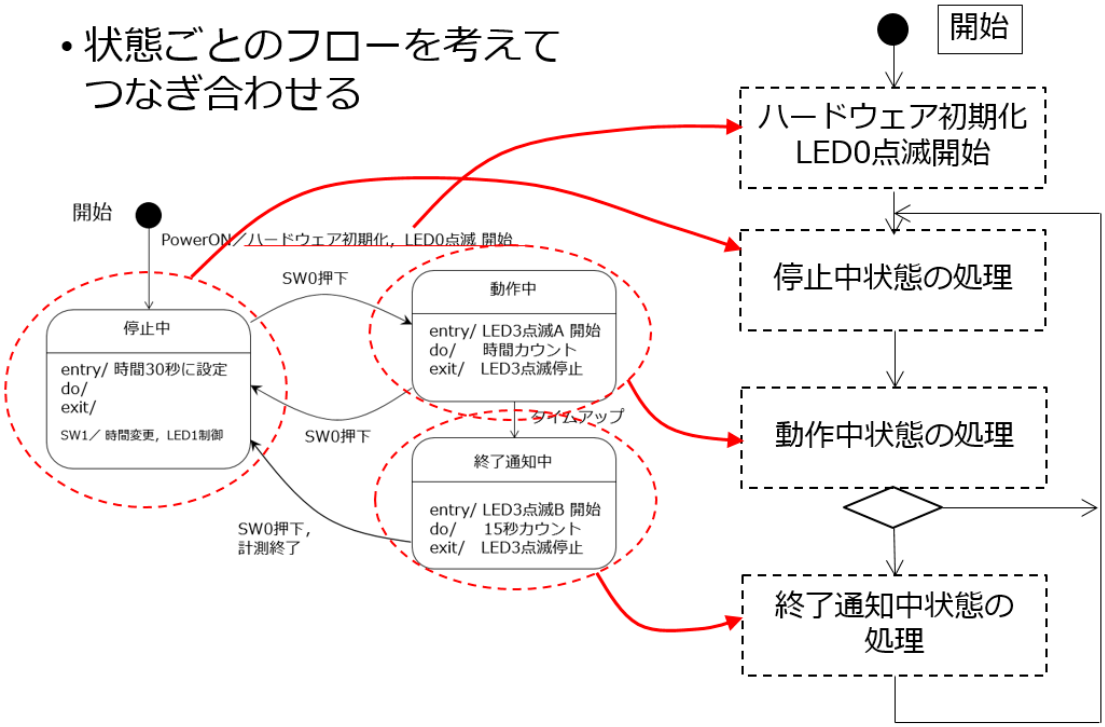


図 34: 状態遷移図と処理フローの対応

2. ［○］プロダクト項目中,「タイマの使い方」「複数のタスクを同時実行させる方法」は，基本設計後に処理フローを考えさせることで，理解に達していたと考えられる．ただし，詳細設計から実際のプログラムコードを書くところに関し困難度が高かった．学習アンケート（表 17）では，特に試行 1 で 4 日目の詳細設計から実装への工程で 9 名がプログラミングの仕方が難しかったと述べている．授業観察上も多くの学生が混乱していた．これに関し，試行 2 では，フローを書かせて個別に添削するという方法で多少の改善が見られたが，前述したとおり教師のスキル依存な方法であり，なんらかの根本的改善が必要と考えられる．

3. ［×］プロダクト項目の「スイッチの特性と使い方」は，完成に至らなかった学生の主要因となっており，これに関する教材の改善が必要と考えられる．

4. ［○］プロセスについては，学習記録にきちんとその日のプロセス内容は反映しており，一定の理解が得られたと考えられる．一方で，工程名の出現は少な

く，もう少し印象付ける工夫が必要と考えられる．

5. ［○］プロジェクトについては，演習ごとにこまめに作業時間設定を行ったが，学習記録に納期や期限といった言葉が現れていないことから，理解を得られたとは言い難い．ガントチャートなどでもっと明示的にスケジュールを示し，意識づけすることが必要と考えられる．

6. ［◎］プロフェッショナリズムとして挙げた,「不明点を自力で調査できる」「周囲の学生と相談できる」は，授業観察上はよくできていた．このことは，今回の評価条件である，自力での調査を目的としてインターネットを使うことができること，周囲の学生と相談可能とするといった施策が有効であったためと考えられる．

   また,「最後までやりきる」については，特に試行 1 ではプログラム完成に至らなかった学生全体の 83.3%と多かったものの，試行 1 試行 2 とも最後まで完成を目指して作業に集中できていた．これは，ゴールとそれに至るプロセスを明瞭に示した本教材の構造の寄与もあることが考えられる．

   また，早期に課題を完成した学生には追加課題を与えており，取り組んだ学生は全員が最後まで緊張を持って取り組むことができていた．

5.5.1　上記以外の問題

基本設計でふるまいの状態遷移図を書かせたとき，たとえば「タイマ動作中状態」での do（実行し続ける項目）に，LED3 の点滅を書く学生がおよそ半数いた．図 35 にその例を示す．これは教材の開発テーマであるキッチンタイマにおける，アプリケーション層の状態モデルのうち，タイマのカウント中ステートの部分を設計した例である．カウント中には LED3 が仕様に定められたパターンで点滅するのだが，多くの学生がその点滅動作をこのステートの do セクションに記述していた．本来，LED の点滅は，もっとデバイスに近い下位レイヤ，一般にデバイスドライバと呼ばれる部分で担当するのが妥当である．そして，アプリケーションのカウント中ステートでは，entry セクションでデバイスドライバに対して点滅開始を指示するのがよい.（プログラムがシンプルになるため）

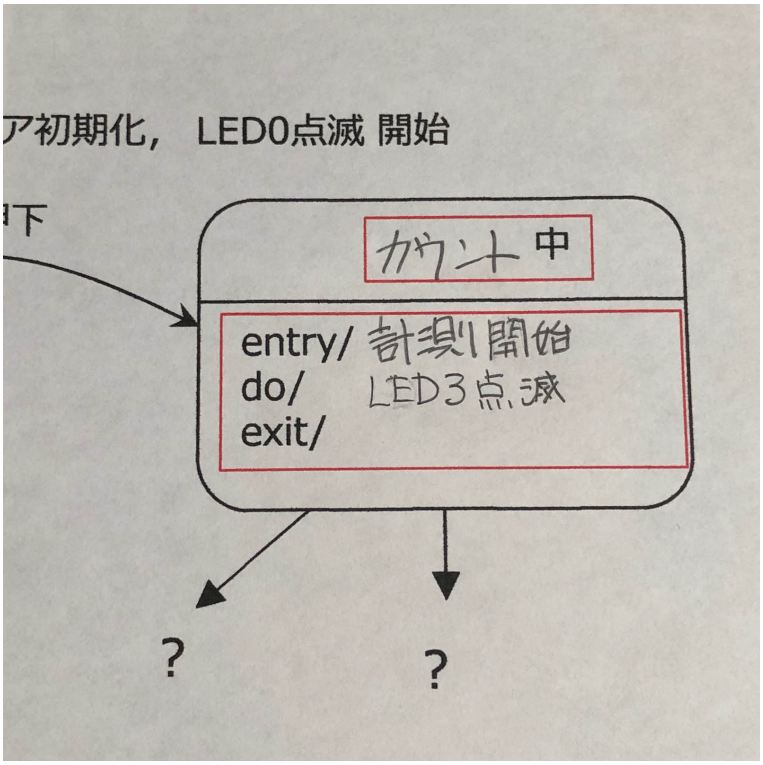


図 35: 状態遷移図の記述例

確かに，仕様の記述を平面的に解釈するならば，タイマのカウント中に行う仕事として，このステートの do セクションに LED3 の点滅を記述してしまうのは理解できる．しかし，この設計で実装すれば，アプリケーションが LED というものの駆動方法を知っており，なおかつアプリケーションからいちいちタイミングに応じて駆動処理をするコードになってしまう．これは推測ではなく，1980 年代の組込みソフトウェア（当時は一般にマイコンソフトウェアと呼称していた）では，そのような記述のプログラムが多数存在し，品質低下の問題となっていた．

このような思考になる理由として，学生の多くがプログラミング演習を経験してはいるものの，アプリケーションからデバイス制御まで幅広く網羅する例を学んだ経験は少ないことが推測される．また，工学的に言えば，結合度や凝集度といった尺度への理解の問題でもあり，それ以前に抽象度の問題でもある．

この奥深い問題を平易に理解に至らしめるために，1.6 に述べた「部品化の考え方（階層構造）」についての解説を教材に追加することとした．

5.5.2　改善すべき問題点のまとめ

以上述べた考察から，次の項目について教材の改善が必要と考えられる．

1. 詳細設計の結果を，どうすれば実装できるかの指導方法
2. スイッチ処理の指導方法
3. 開発プロセスやプロジェクトへの意識づけ
4. 階層構造に基づいた部品化の考え方

5.6　試行 3 の取り組みと教材の最終仕様

2017 年度の試行結果をもとに教材の改善を実施し，試行 3 にて改善の度合いを観測した．改善項目は 5.5.2 に示した 4 項目であり，以下に各項目に関する詳細を述べる．

5.6.1　詳細設計の結果を，どうすれば実装できるかの指導方法

試行 1 と試行 2 の取り組みで，図 29 のような LED0 部品の詳細設計までは概ね理解が進むことが分かっている．これは，少し複雑な LED3 部品でも同様である．多くの学生は，設計結果からどのようにプログラムコードを記述するかを理解できていない様子だった．

そこで，図 36 のように，最も単純な部品である図 29 における LED0 部品の実装と，そのテスト用ドライバの作り方を例示し，実際に動作させて試行させる内容を追加した．そして次に，少し複雑な LED3 について，同様に今度は自力での設計開発を促すようにしてみた．LED0 部品の例示により，詳細設計と実装との関係を理解することに期待し，LED3 部品の実装でそのことを確かめると同時に，学生の自信につなげる狙いとした．

5.6.2　スイッチ処理の指導方法

実際のところ，スイッチの処理はプロでも一定の困難度を有する．特に今回の教材のように押しボタンスイッチの場合，押された，離されたという変化点を読み取り，さらには押されたことを上位処理に知らせたかどうかまで管理する必要があるためで

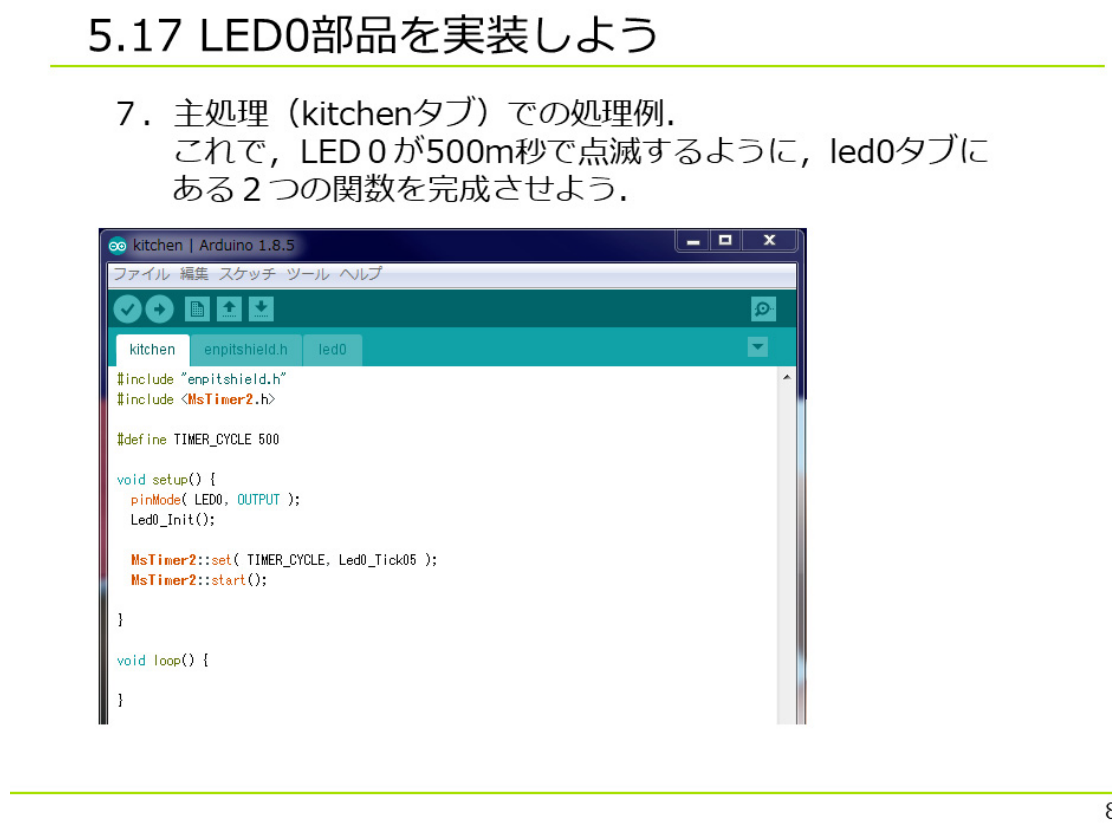


図 36: 実装方法の例示のテキストページ

ある．そこで，4 種類の学習補助を用意することにした．学生は任意に選択することができ，またどの方法を使ったかは，自己申告とした．

1. 自力で独自の方法を考える
2. 例示されたスイッチ処理の状態モデルをもとに作る（図 37）
3. 例示されてスイッチの処理フローをもとに作る (図 38）
4. 事前学習課題の処理をそのまま真似する

5.6.3 開発プロセスやプロジェクトの意識づけ

開発プロセスについては，V 字モデルと各工程の意味を明示し，授業冒頭ごとに示すことで，現在自分たちが何の作業をしているのかを認識させるようにした．なおか

- 状態遷移で考えてみる方法
- それぞれの状態のとき，IsSw()は何を返せばよいか
- このモデルを駆動するのはどうするか
  （SW押した　は，どうやって検出するか）

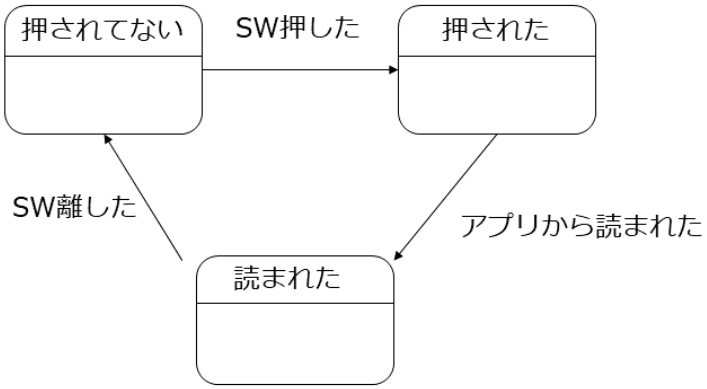


図 37: スイッチ処理の状態モデルの例示

つ「工程を移る際に，プロセスモデルにおける位置を再再度説明する」よう教材を改訂した．図 39 に，授業冒頭での V 字モデルの説明ページを示す．また，図 40 に，基本設計工程に入る時の説明ページを示す．

プロジェクトに関しては，授業全体のガントチャートを明示し，なおかつ毎授業冒頭に進捗実績を説明した．図 10 に，全 16 コマ授業の場合のガントチャート例を示す．各授業冒頭には，これに進捗実績を記入したものを表示して，受講学生に現在の立ち位置の認識を促した．

5.6.4　階層構造に基づいた部品化の考え方

5.5.1 に書いたような,「アプリケーションに相当するふるまいの記述と，物理デバイスの制御といった処理をないまぜにしてしまう」問題は，組込みソフトウェア開発業界として 1980 年代に経験済みである．当時の組込みソフトウェア（当時の一般的呼称はマイコンソフトウェア）は，ふるまいの処理の中にいきなり制御レジスタの記述があるといったことが定常的に行われ，ソフトウェアの品質に悪影響を及ぼしていた．この問題を解決した技術の 1 つが，ソフトウェアの階層化の考え方の導入であった．

今回の問題も，先に述べた平面的な設計行動から，同様のことが起こっていること

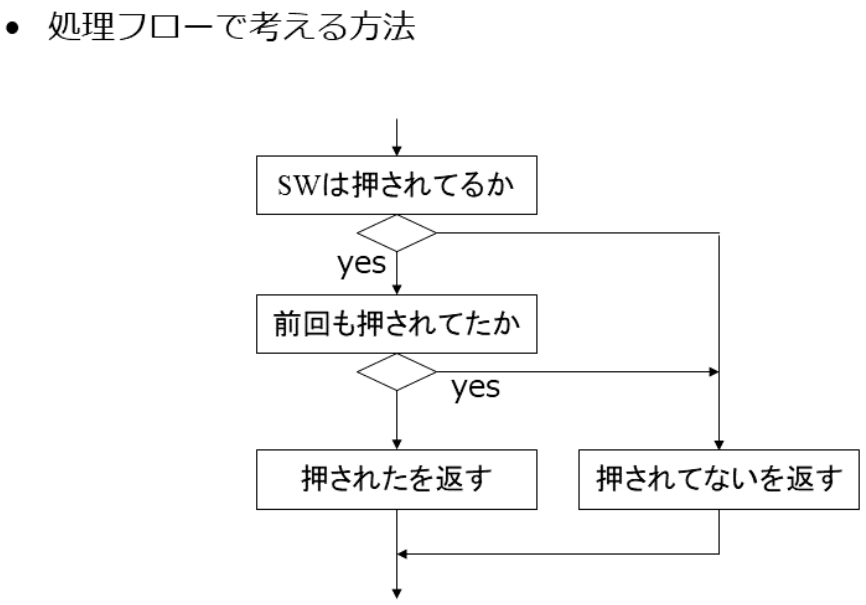


図 38: スイッチ処理のフローの例示

が推測できた．そこでテキストに階層構造の説明や，入出力（外部装置との接続）からデバイスドライバ部品を導き出す方法の図を追加し，解説することとした．

追加した図の 1 つは，図 41 のようなソフトウェアの階層構造と，各階層の役割を説明するものである．この図を用いた説明により，たとえば LED というハードウェアを「点滅させる」とか，スイッチというハードウェアの「状態を読み取る」といったことは，図 41 におけるデバイスドライバ層で受け持つ仕事であることを説明できる．そして，システムとしてのふるまいは，アプリケーション層の仕事であり，ハードウェアリソースに関わる要求は，直接アクセスするのではなく，デバイスドライバに命令するという構造を理解させることを狙った．しかし，この階層図だけでは，概念的理解にとどまってしまうことが危惧された．具体的には，必要なデバイスドライバをどのように仕様から抽出するのかわからないという問題である．

そこで，図 42 のように，システムの入出力から必要なデバイスドライバを必然的に導き出す説明を追加した．図 42 ではすべて内容を埋めたものを示しているが，授業では空白のフレームを示して内容を考えさせる．このように，外部接続を起点にプログラム構造を構築していくという方法は，小規模な組込みソフトウェアの設計にと

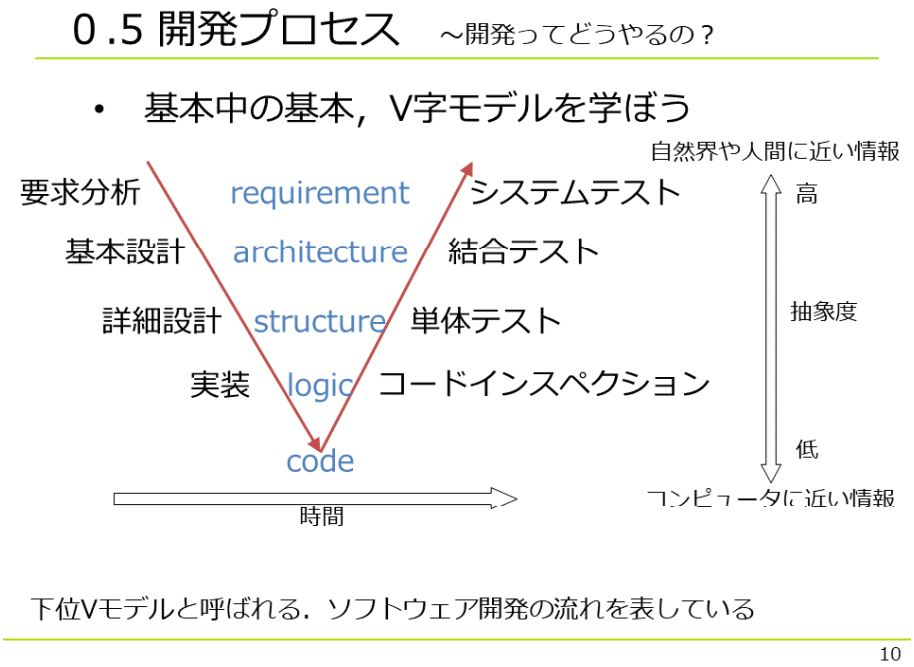


図 39: テキストにおける V モデルと工程の責務の説明ページ

ても有効に活用できる．

以上二つの施策により，学生は部品化の理解が進むと同時に，わかりやすい構造を設計することでわかりやすくシンプルなコードを生み出し，結果的に課題完遂率を向上させることを狙った．

5.7　試行 3 におけるテキスト改良結果

表 19 に，試行 3 での開発演習における学習内容のアンケート結果を示す．以下，この内容を参照しつつ，5.6 で列挙した，試行 1 と試行 2 での問題点に対する 4 つの改善策の結果について述べる．

1. 詳細設計から実装への指導方法の改善について

   難しかった項目にプログラミングを挙げたのは 3 日目，4 日目に各 4 名，5 日目に 1 名と少数であった．彼らは，そもそもプログラム言語に不慣れだったことが観察からわかっている．このことから，フローチャート等の個別指導の代わりに導入した具体的実装例の説明は効果的だったと考えられる．

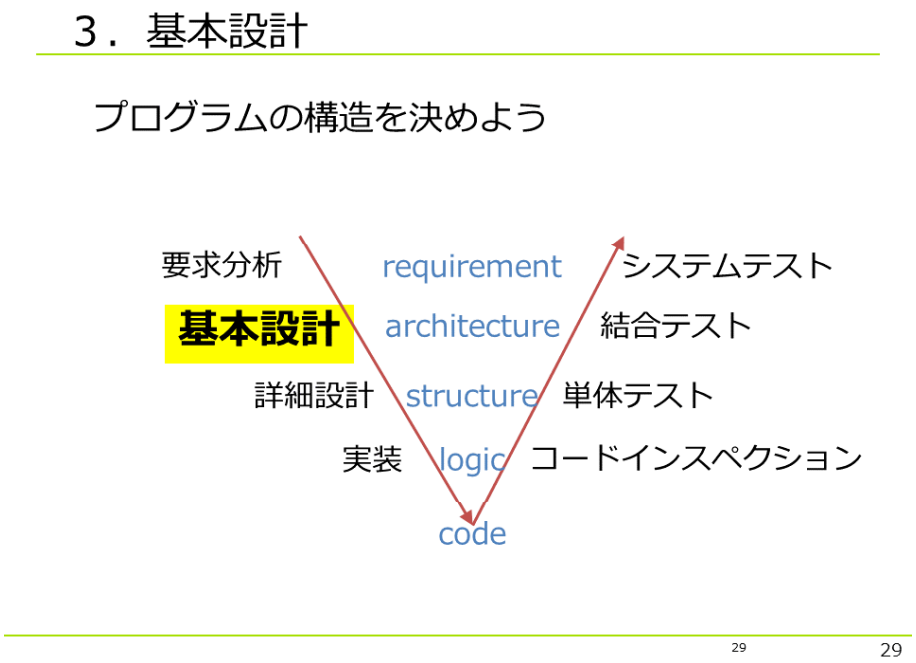


図 40: テキストにおける基本設計工程開始時の説明ページ

2. スイッチ処理の指導方法の改善について

   3 日目の難しかったことにスイッチ処理を 10 名が挙げている．内容を見ると，ヒントに頼らずに挑戦してみた結果うまくいかなかったという事情であった．4 日目には素直にヒントを利用する手段に切り替えており，難しかった点にスイッチを挙げたのは 2 名に激減しており，5 日目以降にスイッチを困難点として挙げた学生はいなかった．

   ヒントを用意するときに，自力解決の選択肢を入れておくと，安易にヒントに頼らず挑戦しようとする人が少なからず出ることがわかった．これは現象としては好ましいことと考えられる．ただし，スキルが足りない学生ほど，自力解決に見切りをつけるタイミングを見失いがちであることが，観察から感じられた．そのため，あるタイミングで方針を切り替えるよう促す指導は大切と考えられる．今回は 5 日目の冒頭に全員に対してその旨を指導している．

3. 開発プロセスやプロジェクトの意識づけについて

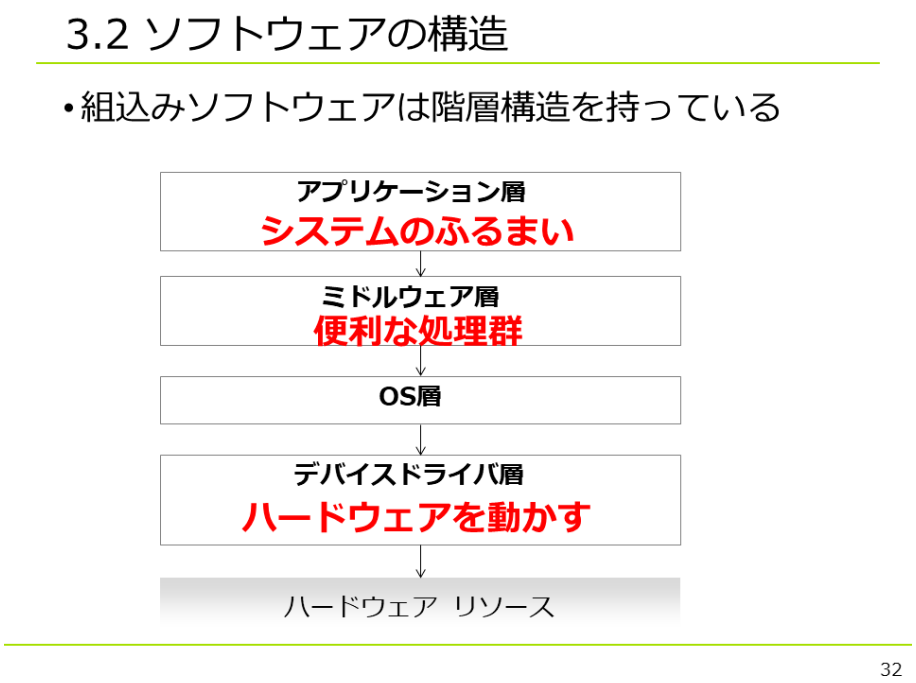


図 41: ソフトウェアの階層構造の説明ページ

人数は多くないものの，ほぼ全日程に渡って，学習内容アンケートに工程名が見られた．ただし，工程名を書いている学生はほぼ同一でもあった．これについては，さらに意識づける工夫が必要と考えられる．

4. 部品化の考え方の指導方法の改善について

状態遷移図の作成においては，学んだことに挙げた学生 14 名に対し難しかったことに挙げた学生が 11 名と，割合的にはほぼ拮抗している．しかし，観察からは，アプリケーション（ふるまい）の状態遷移図に，デバイス制御の詳細を書いた学生は減っていた．

また，仕様分析から部品化設計をして実装，テストということはほぼ全員できていた．このことは，アンケートに関し全員が部品名を使って報告していることからもわかる．

課題の完遂率について，表に示した試行 1 と試行 2 の結果に，試行 3 の結果を太字で追記したものを表 18 に示す．評価項目は，表 12 のプログラム完成度に準拠し，レベル 4 以上を完成としている．受講者 43 名中 37 名，86 %の学生が

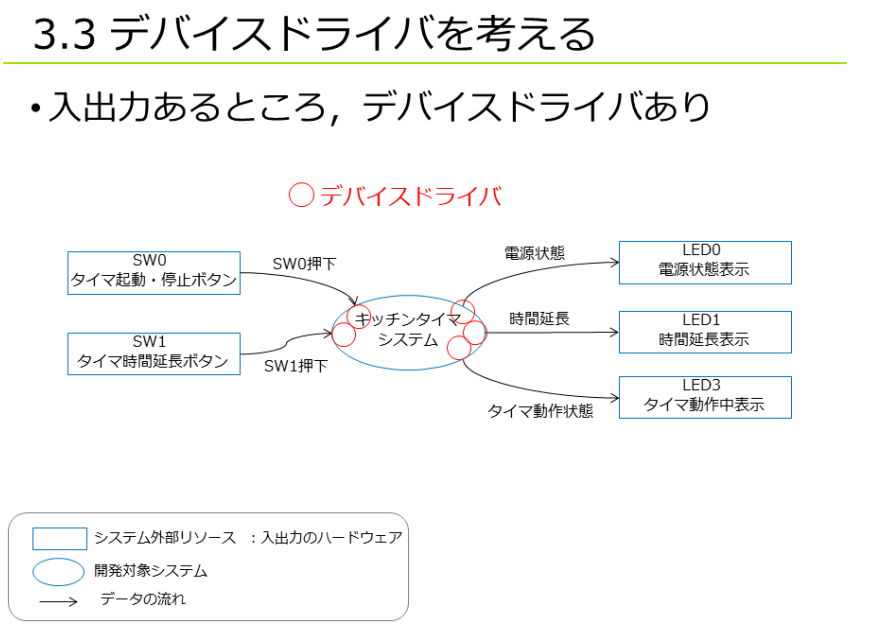


図 42: 入出力からデバイスドライバを導出する説明ページ

表 18: 開発演習の最終進捗

状況	試行 1		試行 2		試行 3	
部品の一部のみ完成	11	36.7%	2	6.9%	**1**	**2.3%**
部品はほぼ完成	8	26.7%	4	13.8%	**1**	**2.3%**
最終デバッグ中	6	20.0%	8	27.6%	**4**	**9.3%**
完成	5	16.7%	15	51.7%	**37**	**86.0%**

課題を完遂できるに至った．

5.8 行動目標に対する試行 3 の結果

2018 年度の試行に関し，4.4.4 で定義した 4P に基づく行動目標に対しては，次のような結果を得た．それぞれ，[◎] は目論見通りだったもの，[○] は，ほぼ目論見ながらさらに改善したほうがよいもの，[×] は要改善のものである．

1. ［◎］目標とした 5 つのプロダクト項目に対し，「Arduino 開発環境の使い方」「LED の制御方法」は，全員が達成できた．

2. ［◎］「タイマの使い方」「複数のタスクを同時実行させる方法」は，処理フローを考えさせて教師が個別対応するのに変えて，詳細設計と 1 対 1 で対応させた実装例をテキストに例示することで改善できた．表 17 の試行 1 のアンケート結果では，最終日である 6 日まで，難しかったこととして「設計と実装の関係」が上がっている．一方で試行 3 では，表 19 のようにその問題は消失した．ただし，2 名の学生が実装に関する質問を行ったが，単純に言語文法に関するものだった．

3. ［◎］「スイッチの特性と使い方」も，4 者択一のヒントを用意することで，学生らは自分のスキルに見合った手段で自力解決できた．表 19 でも，難しかったことにスイッチ処理が上がったのは 3 日目 4 日のみである．

4. ［○］2017 年の試行に比べると，報告に工程名の出現が増加した．しかし，開発作業が佳境に入るに伴いそれは失われて，具体的な作業内容だけになっていた．

5. ［○］プロジェクトについては，毎回ガントチャートで進捗や現在の立ち位置を説明したわりには，報告書にはキーワードとして現れなかった．一方で，授業観察からは，あと何日，あと何時間という発言が聞かれ，有期性作業という認識はある程度理解できていたと考えられる．これに関しては別途，終了後のアンケートにて実践力を測る取り組みをしている中で，ガントチャートへの認識を調査している．その結果，たとえば K 大学の 2018 年度受講生では，実践力があると判断した学生 16 名中，94%(15 名) がガントチャートへの認識を示した．また M 大学での 2019 年度の受講生で，実践力があると判断された 12 名は，全員がガントチャートを読めると回答した．他の大学の事例でも同様であった．これらのことから，一定の認識はできていると考えられる．詳細については 6 章で述べる．

6. ［◎] プロフェッショナリズムとして挙げた,「不明点を自力で調査できる」「周囲の学生と相談できる」は，2017 年度の試行から変わらず，全員最終授業まで真摯に取り組めていた．手を休める，内職をする，眠るといった行動をとった学生は皆無だった．

5.9 実証実験授業の結果

K 大学での試行を経て改良した教材を使用し，表 31 に示した大学にて，実証実験授業を実施した．以下，B 大学と M 大学での詳細，C 大学と S 大学での結果，また K 大学における第三者の教員による授業実施結果について述べる．

5.9.1 B 大学における授業実施結果

実証実験授業の実施結果の例として B 大学での結果について述べる．授業期間は 2018 年 9 月 24 日から 12 月 3 日の間の 9 日間，各日 90 分を 2 コマの合計 18 コマ 27 時間，対象は情報系大学学部 2 年生 9 名と 4 年生 3 名である．他の試行例や実証実験事例では 3 年生であったのとは異なり，2 年生と再履修の 4 年生であることから，能力的にはやや低いことが推測される．

事前学習教材は，試行 1〜3 同様に，4 コマ 360 分（内，ガイダンスに 20 分使用）を割り当てた．達成度を図 43 に示す．すべての演習を終えた 33.3%（4 名）と，演習 4 までまたはそれ未満の 66.6%（8 名）とに大きく分かれた．この結果は，K 大学における試行 3 での結果（図 32）および実証実験授業での結果（図 33）に比べて達成度が低い．

次に開発演習の達成度を，事前学習課題の達成度と対比させる形で表 20 に示す．左側が事前学習教材の達成度であり，右側が開発演習の達成度である．

事前学習教材で演習 6 またはアドバンス課題まで修了した 4 名は，全員が完成に至った．一方，演習 4 まではできた学生 4 名は完成には至らず，最終デバッグまでが 1 名，部品はほぼ完成が 2 名，部品の一部のみ完成が 1 名と分かれた．事前学習教材で演習 4 さえも未達だった学生は，全員が部品の一部のみ完成までしか至らなかった．

試行 1〜試行 3 や，その他の実証実験授業の結果から，ガイダンスおよび事前学習教材に必要な時間が 360 分程度であることは，ある程度妥当性があると考えられる．したがって，本授業での事前学習教材の達成度の低さは，受講者群の特性に原因があると考えられる．

かかる状況下にあって，事前学習教材の達成度と開発演習の達成度には，明らかに相関がみられる．先に述べたように，事前学習教材で演習 6 以上を完遂できた学生は，開発演習も完遂に至っている．このことから，事前学習教材を完遂できる程度の能力

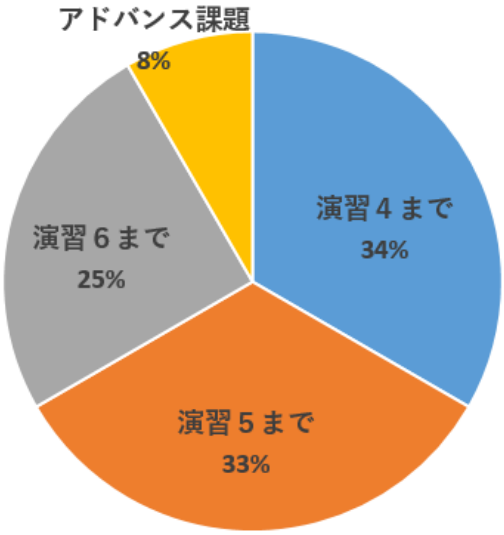


図 43: B 大学の事前学習教材と開発演習の達成度

があれば，開発演習課題は完遂可能な内容かつレベルにあると推測できる．また逆に言えば，今回作成した事前学習教材は，規定時間内での達成度によって，本研究の開発演習の受講スキルの判断に利用することもできると言える．

次に，表 21 に，開発演習における学習内容のアンケート結果を示す．この内容から，5.6 で列挙した 4 つの改善策ごとに結果を示す．

1. 詳細設計から実装への指導方法の改善について

   アンケート結果からはわかりにくいが，授業観察からは次の 2 点がわかった．事前学習教材で低成績だった学生は，そもそもプログラミング能力の低さにさえ気づいておらず，結果アンケートにキーワードとして表れていない．一方で，事前学習教材において好成績だった学生は，改善施策による例文プログラムをよく理解して，学習を進めることができていた．このため，指導方法の改善については，効果あったものと考えられる．

2. スイッチ処理の指導方法の改善について

   スイッチ処理も，前項の実装指導方法の改善と同じ傾向が見られた．最終的に完成に至った学生は，特にスイッチ処理で大きなトラブルに直面することはなかった．

3. 開発プロセスやプロジェクトの意識づけについて

   アンケート結果では，一応ほとんどの日程に工程名がキーワードとして現れている．しかし，工程名を書いているのはやはり特定の学生であり，その意味で実施結果 1 と同様にさらなる改善が必要と考えられる．

4. 部品化の考え方の指導方法の改善について

   状態遷移図の作成においては，大きな混乱はなかった．アンケートに関しても，全員が部品名を使って報告していることから，部品化の考え方は伝わっていると考えられる．

5.9.2 M 大学における授業実施結果

実証実験授業の別の例として M 大学での例について述べる．授業期間は，2019 年 6 月 18 日から 7 月 30 日の間の 7 日間，各日 90 分を 2 コマの合計 14 コマ 21 時間，対象は情報系大学学部 3 年生 18 名である．

事前学習教材の達成度を図 44 に示す．4 コマ 360 分（内，ガイダンスに 20 分使用）を割り当てた．結果，規定の演習 6 まで完了，またはさらにアドバンス課題までの終了者は 78%(14 名)，演習 5 までが 11%(2 名)，演習 4 までが 11%(2 名) であった．この分布は，K 大学で実施した試行 3 および実証実験での結果である図 32 や図 33 と近似している．

開発演習教材に関しては，受講者 17 名中 15 名が完遂し（1 名は途中リタイヤ），達成度は 88%だった．これは表 18 に示した，K 大学での試行 3 の結果に近しい結果である．

また，事前学習教材の達成度との関係は，表 22 の通りであった．B 大学同様，事前学習教材で演習 6 またはアドバンス問題までできた学生は，全員が開発演習で完成に至っている．このことは，5.9.1 で述べた仮説，すなわち「事前学習教材を完遂できる程度の能力があれば，開発演習課題は完遂可能な内容かつレベルにある」を補強する結果と考えられる．

表 23 に，M 大学における開発演習における学習内容のアンケート結果を示す．それぞれ，共通する内容が複数件あったものを取り上げている．

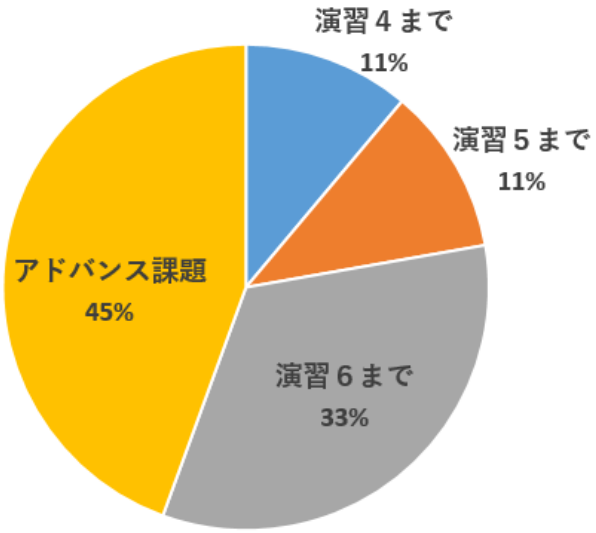


図 44: M 大学の事前学習教材の達成度

1. 仕様分析に関心が集まる

   すべての受講生が共通の意見として最も多く挙げたのが，仕様書の読み方だった．難しかったことに挙げている学生も多いが，これは関心の現れと解釈できると考えられる．

   今回の M 大学に限らずこの傾向は存在する．仕様分析という行為そのもののめずらしさによるものかも知れないが，仕様分析の重要性の学習だけでなく，開発演習の冒頭部分で授業への関心を引き付ける効果にもなっており，本教材の特徴と言える．

2. スイッチ処理で悩む学生は少なかった

   試行 1 と試行 2 で問題だった，スイッチ処理の設計の難しさは，今回ほとんど顕在化しなかった．試行 3 での対策が有効に働いていると考えられる．

3. 工程名が登場しにくい

   4 日目までは，仕様分析や詳細設計，実装という言葉がある程度登場する．しかし 5 日目以降，開発が佳境にはいると，ほとんど登場しなくなってしまった．システムテストという単語に至っては，全受講生の全レポート中，1 回しか登場していない．

こういった傾向は，すべての実施大学で見られる．V 字モデルの右側にあたるプロセスについては，単に言葉を強調するだけでは印象付けられない可能性が考えられる．しかし本研究の教材で想定している時間数で，ソフトウェアテストについて深く扱うのは困難であり，別途，V 字モデルの右側に重点を置いた教材を開発する必要があると考えられる．

表 19: 試行 3 での学習内容アンケート結果

日	講義・演習内容	学んだこと，できたこと		難しかったこと	
1	組込みシステム				
	開発プロセス	開発プロセス	2		
	要件分析	要件分析の大切さ，手法	16		
	開発環境	開発環境について自己学習			
2	開発環境	開発環境について自己学習			
	基本設計	基本設計	6	基本設計	1
		状態遷移図	14	状態遷移図	11
		階層構造	1	階層構造	1
3	基本設計	基本設計	4		
		状態遷移図	5		
	詳細設計	詳細設計	7	LED動作の制御	8
	実装	シーケンスフロー図	1	スイッチ処理	10
	単体テスト	タイマーの使い方	9	プログラミング	4
		スイッチ処理	7	状態の実装	1
		LED点滅(同時実行)	30		
		実装(コーディング)	9		
		単体テスト	5		
4	詳細設計	詳細設計	2	動作制御	10
	実装	スイッチ処理	6	プログラミング	4
	単体テスト	LED点滅(同時実行)	6	スイッチ処理	2
	システムテスト			チャタリング処理	1
		システムテスト	1		
		実装(コーディング)	4		
		単体テスト	3		
		完成	1		
5	詳細設計	詳細設計	3	動作制御	6
	実装	スイッチ処理	2	プログラミング	1
	単体テスト	LED点滅(同時実行)	2		
	システムテスト			チャタリング処理	3
		システムテスト	4	システムテスト	5
		実装(コーディング)	4	単体テスト	1
		単体テスト	4		
		完成	18		
6	詳細設計			動作制御	3
	実装	スイッチ処理	1		
	単体テスト	状態遷移の実装	2	タイマードライバ	1
	システムテスト			チャタリング処理	1
		システムテスト	2		
		完成	20		

表 20: B 大学の事前学習教材と開発演習の達成度

事前学習テキスト	人数	割合		開発演習	人数	割合
演習 4 未満	4	33.3%	→	部品の一部のみ完成	5	41.7%
演習 4 まで完了	4	33.3%	→	部品はほぼ完成	2	16.7%
演習 6 まで完了	3	25.0%	→	最終デバッグ中	1	8.3%
アドバンス課題完了	1	8.3%	→	完成	4	33.3%

表 21: 学習内容アンケート結果 2018 年度 B 大学

日	講義・演習内容	学んだこと，できたこと		難しかったこと	
1	組込みシステム 開発プロセス	組込みシステムについて 開発プロセス	1 1		
	開発環境	開発環境について自己学習			
2	開発環境	開発環境について自己学習			
3	要件分析	要件分析	6		
		仕様書の読み方	4	仕様書の不備の発見	2
	基本設計	基本設計	5		
		状態遷移図	5	状態遷移図	1
4	詳細設計 実装 単体テスト	詳細設計 実装 LEDの設計 単体テスト LEDの点滅	2 4 3 4 5	LED3の開始終了 LED3の設計と実装 LEDの同時動作同時動作 タイマ処理	2 4 2 1
5	詳細設計 実装 単体テスト	ガントチャート 同時動作 スイッチの単体テスト LED1の実装 LED3の単体テスト	1 2 1 1 1	スイッチによる制御 LED3の設計，実装 LEDの同時動作 時間カウントの実装	3 1 2 1
6	詳細設計 実装 単体テスト	LED3の点滅 LED0のサンプルコードの理解 実装 詳細設計 LED処理	1 2 1 2 2	スイッチ処理 LEDの同時動作 チャタリング対策	3 2 1
7	詳細設計 実装 単体テスト	ガントチャート スイッチ処理 詳細設計 同時動作	1 3 2 2	LEDの同時動作 スイッチ処理 アプリケーション層の設計実装	1 1 2
8	詳細設計 実装 単体テスト システムテスト	シリアルモニタでのデバッグ システムテスト スイッチ処理	1 1 3	スイッチ処理 同時動作 アプリケーション層の設計実装	2 1 1
9	詳細設計 実装 単体テスト システムテスト	完成 ほぼ完成 スイッチ処理	5 1 1		

表 22: M 大学の事前学習教材と開発演習の達成度

事前学習テキスト	人数	割合
演習 4 未満	0	0.0%
演習 4 まで完了	2	11.1%
演習 5 まで完了	2	11.1%
演習 6 まで完了	6	33.3%
アドバンス課題完了	8	44.4%

開発演習	人数	割合
リタイヤ	1	5.9%
部品の一部のみ完成	1	5.9%
部品はほぼ完成	0	0.0%
最終デバッグ中	1	5.9%
完成	15	88.2%

表 23: 学習内容アンケート結果 2018 年度 M 大学

日	講義・演習内容	学んだこと，できたこと		難しかったこと	
1	組込みシステム				
	開発プロセス				
	開発環境	開発環境について自己学習			
2	開発環境	開発環境について自己学習			
3	要件分析	仕様書の読み方	14		
		レビュー	2	仕様書の不備の発見	5
	基本設計	階層	1		
		状態遷移図	7	状態遷移図	1
4	詳細設計	詳細設計	4	状態遷移の実装	3
	実装	実装/コーディング	3	LEDの同時動作	2
	単体テスト	LEDの設計	10	LED3の設計，実装	2
		単体テスト	0		
		タイマ処理	1		
5	詳細設計	テスト	1	時間制御	1
	実装	同時動作	3	状態遷移の実装	3
	単体テスト	完成	4	LEDの同時動作	2
		追加課題	2	スイッチ処理	1
6	詳細設計	実装	2	スイッチ処理	1
	実装	状態遷移	5		
	単体テスト	チャタリング	3		
7	詳細設計	追加課題関連	6		
	実装	チャタリング	2		
	単体テスト				
	システムテスト				

5.9.3 C 大学および S 大学における授業実施結果

少人数での実施例として，C 大学での 2018 年と 2019 年，および S 大学での 2019 年に実施した結果について触れる．サンプル数が小さいため，データは偏りが大きい可能性がある．結果の一覧を表 24 に示す．

C 大学では，2018 年 2019 年ともに，事前学習教材の進捗が芳しくなかった．この点は，図 43 に示した B 大学での例に似ている．一方で，開発演習において完成に至った学生は，2018 年には 75%(8 名中 6 名)，2019 年には 66.7%(6 名中 4 名) であった．これは B 大学の結果に比べればやや良好であった．

C 大学の受講生は，ゼミ配属されたメンバであり，授業はゼミ担当の教授も同席しての 2 教員体制で行った．このため，比較的手厚く受講学生をフォローできていたことも一因と考えられる．

S 大学では，他の事例と異なり受講生は全員 4 年生の下期であった．結果，事前学習教材も開発演習も，規定時間内でそれぞれ完遂できた．

5.9.4　第三者教員による授業実施結果

最後に，K 大学での 2019 年に実施した結果について報告する．この授業は，他の実験授業と異なり，第三者の准教授が授業を実施してみた事例である．受講学生 71 名中，開発演習で完成に至ったのは 50.7%(36 名) だった．この値は，同じ K 大学で実施した試行 3 や実証実験授業の結果に対して低い．

達成度が低い原因として，担当教員の本教材への不慣れの可能性もあるものの，別の要因も考えられる．事後調査の結果，この授業ではほかの授業と大きく異なる進め方をしたことがわかった．通常は，事前学習教材をガイダンスともに 360 分実施した後，開発演習に移行させている．一方で 2019 年の K 大学での授業では，最初から開発演習を開始し，開発演習の進捗フェーズでの技術テーマが登場するごとに，その都度，事前学習教材の対応する解説や演習を実施していた．

ただ一例なので断定はできないが，このような進め方をすると，最終的な達成度が低下してしまうこともあるということを，今回の結果は示唆している．本質的原因については，本研究では深究しない．しかし，教材を水平展開するにあたって，単に解説入りのテキストだけでなく，進め方に関する指導要領も添付することが必要と考えられる．

5.10　ルーブリックによる評価

4.4.6 合格基準で示した表 12 のルーブリックによる成績評価の例について，表 25 に M 大学における 2019 年度の評価結果を示す．事前学習の結果と最終プログラムの結果に相関関係が見られるのは，5.9 で述べた通りである．このように，ルーブリックによって中間および最終成果物に関して評価を行い，成績判定等に利用することが可能と考えられる．実際には，ルーブリックのレベル値そのままでなく，評価項目ごとになんらかの重みづけをする必要があると考えられる．

5.11 提案教材による教育試行のまとめ

4 章で述べた基本設計に基づいた教材を，3 度の試行を経て改良し，現在まで 7 度の実証実験授業を実施した．実施結果からは，事前学習教材を規定時間内に完遂できる程度の能力があれば，本研究の開発演習課題は十分に完遂可能であることがわかった．そして課題を完遂した受講者は，5.8 に示した試行 3 の結果同様に，4.4.4 で定義した 4P に基づく行動目標を達成できたと考えられる．それは,「開発演習課題の完遂は，次の条件をすべてクリアしたことの証」だからである．

1. プロダクトに定義した製品技術をすべて学習して駆使できた
2. V 字モデルの開発プロセスに準拠して作業を進め，進捗を報告できた
3. プロジェクトの計画に従って作業を進め最終納期を遵守できた
4. 不明を自力で調査し，周囲や TA・教員に質問し，最後までやり遂げた

教材は，第三者の教員でも運用可能であることもわかった．ただし学生による課題の達成度を上げるには，教材の運用方法を正しく伝えるしくみが必要と考えられる．

一方で受講の前提条件は 4.4.3 の通り，C 言語によるプログラミング能力のみとしたが，受講学生に大きなばらつきがあることがわかった．そのことが，開発演習課題の達成度に大きく影響を及ぼしていた．先に述べた通り，教材の開発演習課題の達成度は，事前学習教材の達成度と密接な関係がある．これを利用して，事前学習教材を，既存の C 言語教育に関する習熟度測定に使える可能性もあると考えられる．

表 24: C 大学と S 大学の実証実験授業の達成度

C大学　2018年　3年生8名

事前学習テキスト	人数	割合
演習 4 まで完了	1	12.5%
演習 5 まで完了	3	37.5%
演習 6 まで完了	4	50.0%
アドバンス課題完了	0	0.0%

開発演習	人数	割合
部品の一部のみ完成	1	12.5%
部品はほぼ完成	0	0.0%
最終デバッグ中	1	12.5%
完成	6	75.0%

C大学　2019年　3年生　6名

事前学習テキスト	人数	割合
演習 4 まで完了	2	33.3%
演習 5 まで完了	3	50.0%
演習 6 まで完了	1	16.7%
アドバンス課題完了	0	0.0%

開発演習	人数	割合
部品の一部のみ完成	1	16.7%
部品はほぼ完成	0	0.0%
最終デバッグ中	1	16.7%
完成	4	66.7%

S大学　2019年　4年生　4名

事前学習テキスト	人数	割合
演習 4 まで完了	0	0.0%
演習 5 まで完了	0	0.0%
演習 6 まで完了	0	0.0%
アドバンス課題完了	4	100.0%

開発演習	人数	割合
部品の一部のみ完成	0	0.0%
部品はほぼ完成	0	0.0%
最終デバッグ中	0	0.0%
完成	4	100.0%

表 25: M 大学（2019 年度）のルーブリック評価例

学生	事前学習	仕様分析	モデル図	プログラム	合計
1	2	5	3	0	10
2	2	1	3	1	7
3	4	5	5	5	19
4	5	5	5	4	19
5	5	5	5	4	19
6	5	5	5	4	19
7	4	5	5	4	18
8	5	5	5	5	20
9	5	5	5	5	20
10	5	5	5	5	20
11	4	5	5	5	19
12	3	3	5	3	14
13	3	3	5	3	14
14	5	5	5	5	20
15	5	5	5	4	19
16	4	5	5	5	19
17	4	5	5	5	19
18	4	5	5	5	19

第6章　効果測定

本章では，実践力向上に向けた教材の効果測定のアプローチについて述べる．

本研究で述べたような，ソフトウェアをきちんと作る能力の向上を目的とした教育はいくつか報告されている (沢田, 小林, 金子, 中道, 大久保, 山本, 2009)(舘, 山崎, 次郎丸, 2019)．一方で効果測定は，受講後アンケートによるものが多い．この場合，測定結果は対象とする演習で行ったことに対する心象であり，それを以って別の課題に主体的に取り組めるかは疑問が残る．

アンケート以外の例としては，山本らによるコンピテンシーを使った効果測定の取り組みがある (山本, 2015)．コンピテンシー測定は，能力や資質の多元的評価を目的としている．コンピテンシーが向上すれば，その人は仕事を遂行する能力が高まったことにはなるが，実際に製品を作るべく進むことができるかどうかは未知数である．またコンピテンシー測定は，一回あたり 1 時間程度の時間を要する場合がある．

3 章にて，ソフトウェア技術者にとっての実践力は，実際に製品を作ることができる力として，社会から期待されていることを明らかにした．この力の保有度合いの変化を調べることで，ソフトウェア技術者のための実践力教育の効果を測定することができるのではないかという仮説を立てた．

そこで今回，教育内容とは別のすなわち未知の開発テーマに関して，本物に近い要求仕様書を読ませて，その開発に取り組めるかを尋ねることで，実践力の一部を測定することに取り組んだ．その結果，有意と思われる結果が得られた．実際に開発するのではなく，要求仕様書を読んだ結果のみでの測定とした理由については，6.1.2 から 6.1.3 で述べる．

6.1 効果測定の研究目的とその背景

6.1.1 研究目的

情報系技術者を目指す学生への実践力向上を目的とした教育の効果について，後述する「自信力」を主眼においた簡易で有効な測定方法を確立する．従来の，対象とする演習などの授業の課題そのものではなく，学習結果によって別の未知なる問題へ取り組めるようになったかどうかを判定できるようにする方法を試行し考察する．

6.1.2 着眼の背景

具体的な実践力向上の測定方法を考案すべく，実際の開発業務に着眼した．情報系学生が開発職に就いたとき，まず対峙せねばならないのは，要求仕様である．その時点で，自分にはできないと考えれば，それは確実にできない．逆に，一定の根拠を以ってやろうと決断できるならば，仕事に着手することはできる．一定の根拠とは，主にそれまでに学んだ工学的な知識と経験を意味する．これは知識の応用力とは異なり，応用力を発揮しようと決意できる能力と考えられる．本研究では，この力を自信力と呼称する．単なるやる気ではなく，工学的根拠に裏付けられた意志力を意味する．

そこで．実際に本物の要求仕様書を読ませて，その開発に着手できるかどうかと，その判断の根拠を学習の前後で調べれば，少なくとも製品を作ることに関する自信力の変化を測定できるのではないかと考えた．

6.1.3 自信力の重要性

企業で長年開発業務を行った経験から，次のような仮説を立てた．技術者個人の能力の高さに関係なく，新しい仕事を始める場合に十分な知識と能力があることは少ない．なぜなら，能力の向上に伴って，より難易度の高い仕事に対峙することになるためである．行うべきミッションには，未知の内容を含み新しい技術を必要とすることに挑む姿勢が少なからず求められる．従って，仕事を開始するには，工学的知識と経験に基づいて，やり遂げられると決意し開始する自信力が必要なのではないか．

これについて，SESSAME(組込みソフトウェア管理者・技術者育成研究会, 2000) のメンバに対して問いかけを実施した．11 件の回答があり，否定意見はなかった．寄せられた意見には，全面的に賛同というもののほか，次のようなものがあった．(以下，

原文のまま）

- 定量化できるかどうかは不明ですが，それはアリだと思っております
- 常にわからない問題を解決するのが，開発の仕事だと思います
- 上司視点では，簡単に「できます」という部下は心配ですが，すぐに「それは無理です」という部下の方がイヤ
- 「やってみなければわからない」と思う性質なので，依頼や注文に対してはお断りという選択肢は持っていません

回答数は少数ではあるが，回答者はいずれも開発経験を持ち技術者の育成に関心を持っている現役技術者である．彼らからの意見には，技術者教育に対する有益な知見が含まれていると考えられる．発言に共通した内容として，未知の問題に対してまずやってみようとすることに価値を見出している点が挙げられる．つまり，技術職における実践力すなわち実際に物を作れる能力に関し，着手を決意する自信力は重要とする技術者は多いと考えられる．

類似の事例として，OKR というプロジェクト管理手法では，プロジェクトを構成する各ミッションについて，自らの知識や能力に基づいて課題の困難度を見積もる，自信度という指標が用いられる．(Christina Wodtke, 2016)

6.2　関連研究

教育の効果に関して，自信を扱った文献はいくつか見つけることができる．たとえば橋本らは，特定のオフィスソフトウェア利用の訓練に関し，学習者の状態把握に自信度という要素を使っている (橋本, 佐野, 岩崎, 松永, 1997)．また，孫らは，Web 環境における学習システムに関し，学習者の心理状況の 1 つとして自信度を扱っている (孫, 甘泉,Huang,He, 程, 2005)．これらはいずれも，学習教材の内容について自信を持てたかといった測定に使われている．その他，ARCS モデルを参照した教育では，ARCS モデルが Confidence（自信）というパラメータを持つことから必然的に取り扱われているが，主に学習意欲や学習への動機付けに重点がおかれているケースが多い (王, 池田, 李, 2007)(鈴木, 根本, 合田, 2010)．

一方，Yasemin GULBAHAR らによる,「スクラッチによるプログラミング教育が問題解決スキルに与える影響について」の研究がある (Filiz Kalelioglu,Yasemin Gulbahar, 2014)．学習の前後で自信の変化測定をしており，本研究に近しい．彼らが行った研究では，プログラミング教育によって問題解決能力への自信には有意な増加がなかったとしている．このことは，自信の測定そのものに意味がないか，あるいは実施した教育が，自信を向上させるに足るものでなかったかのいずれかと推定できる．後者の場合，特定の教材による教育の効果測定として，自信を扱うことに意味がある可能性が考えられる．

6.3 測定方法の設計

6.3.1 測定対象授業

5 章で述べた本研究の教材による授業について，その実施前後で測定を行った．この教材の詳細は 4 章と 5 章に述べた通りであるが，その概要を以下に列挙する．

1. テーマはキッチンタイマプログラムの開発
2. プログラム規模は 200 ライン程度
3. 学習時間は 24 時間前後
4. 次のように 4P の学習要素を含む
   (a) プロセス
   あいまいな仕様をベースに，仕様分析，設計，実装，テストと，開発プロセスに準拠して進める
   (b) プロジェクト
   進捗にはガントチャートを提示し，プロジェクトの有期性も学ぶ
   (c) プロダクト
   同時実行やリアルタイム制御という組込み特有の要素技術を含む
   (d) プロフェッショナリズム
   不明点は，調査，周囲との相談，TA への質問など，自力解決を基本とする

4P モデルのこれら教材設計に対して，次のような質問項目を設けて測定した．

測定の質問項目	関連する 4P 項目
1. 未知の開発課題に取り組めるか	プロフェッショナリズム
2. 開発の手順を尋ねる	プロジェクト/計画
	プロセス／開発プロセス
3. 困難な工程と理由を尋ねる	プロセス／開発プロセス
	プロダクト／製品技術

測定では，演習授業の受講前後での自信力の変化を観察することを基本とした．ただし，測定結果が，本研究教材の特徴である実践力向上の作り込みによるものか，プログラミングを伴うような演習授業に共通のものかを見ることは重要である．そこで，対比群を使って比較を行うこととした．本研究の教材が 4P に基づいて主に「作り方を学ぶ」ことを目的としているのに対し，対比群の演習は 4P に相当する内容は積極的には訴求してない．主に個別技術の習得や，最終テーマの成果物を作ることを目的としている．対比は，実験授業 2 例で実施した．対比に使った授業は，あるテーマに沿ったプログラム開発演習である．詳細なシラバス等については，ここでは省略する．

結果的に，2 例とも本研究の演習授業と対比の授業とでは差異が認められた．ただしこれは，本研究の演習授業の目的をターゲットにした測定方法ゆえの差異であり，それを以て両授業の優劣を示すものではない．

6.3.2　測定手順

開発に着手できるかを調べる対象には，本物の要求仕様書を使用することが望ましい．しかし，一般的に要求仕様書は組織外秘であり，入手は現実的ではない．そこで今回は，SESSAME が中級技術者の教育向けに公開している，電気ポットの仕様書を利用することとした (SESSAME 組込みシステム教育教材, 2004)．この仕様書は SESSAME の Working Group 2 によって作成され，改版を経て第 7 版まで完成度を上げている．ただし，最終版である第 7 版は，USDM (Universal Specification Describing Manner) (清水, 2010) という特別な書式で記述されており，読解には USDM の知識

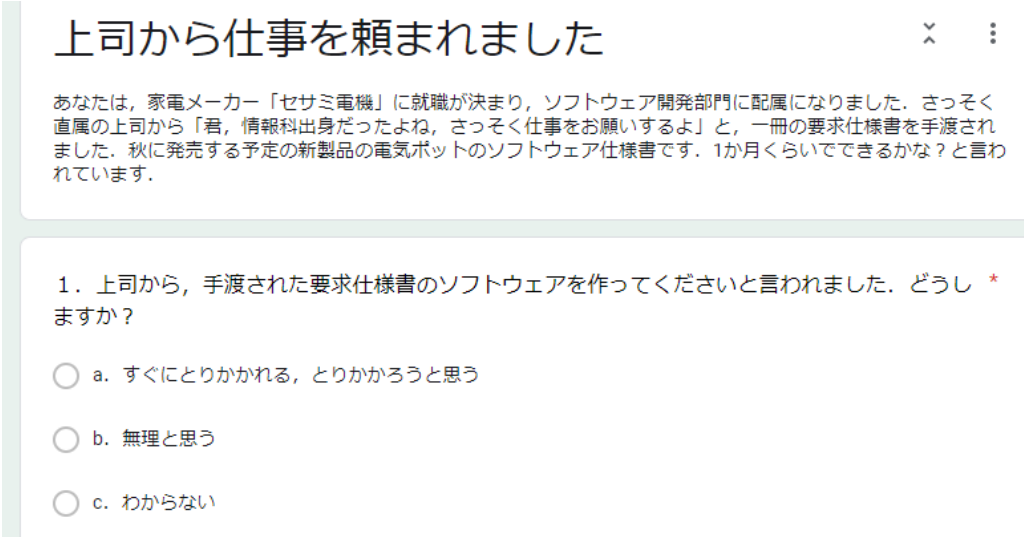


図 45: 要求仕様書を読んだ学生への質問

を必要とする．そこで本研究では，自然言語で記述された中で最も完成度の高い，第 6 版を使用することとした．この要求仕様書は，実際にこれに基づいて電気ポットの制御ソフトウェアを設計できるだけの情報量があり，また通常の業務で使用できる程度の書式で作成されている．つまり，実際に業務で使用されている本物の要求仕様書に限りなく近い．規模は全 18 ページである．

学生には，20 分程度の時間を与えてこの仕様書を読ませたのち，図 45 のようなフォームで質問を実施した．

この設問で「a．すぐにとりかかれる，とりかかろうと思う」を選択した学生には，4P に関してさらに図 46 の質問を行った．これらの質問は，開発にとりかかれるとした回答が，やる気だけなく 4P に含む工学的の知識に裏打ちされたものであることを確認することを目的としている．

「どのような順番で作業を進めるか」「一番難しそうな工程は何か」という設問は，開発プロセスの知識と理解を問うている．ただし，どの工程を難しいと感じるかは個人のスキル特性に依存するので重視しない．ポイントはその理由の妥当性である．

「ガントチャートを読めるか」は，実践的に仕事をするために必要な，プロジェクトに関する最も基本的な要素である，有期性ある計画に関する知識を問うている．「どのような製品技術を使うか」は，プロダクトに相当する設問である．組込みソフト

ウェア開発に特徴的な項目と言える.「先輩や周囲のひとに相談できるか」は，プロフェッショナリズムとして定義した，なんとかして問題を解決する能力の一端を見る目的で設問した.

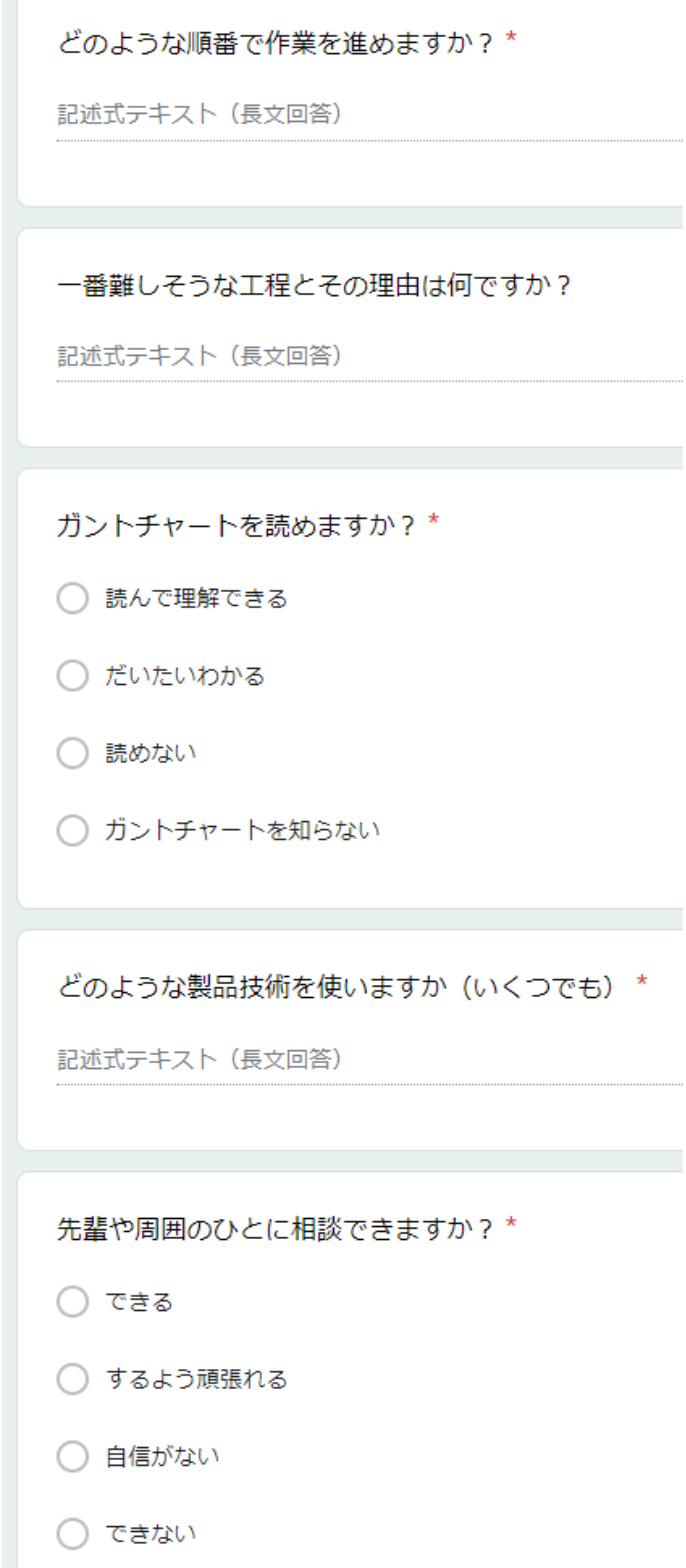


図 46: できると回答した学生への質問

図 47: 無理と回答した学生への質問

最初の設問，つまり要求仕様書を見て作れるかどうかについて，「b. 無理と思う」と回答した学生には図 47 の追加質問を行った．また「c. わからない」と回答した学生には図 48 の追加質問を行った．これらの結果については，本稿では省略する．

6.4　測定結果

本章ではまず，K 大学と B 大学の 2 つの大学で試行した測定結果例を示す．これらは，別のソフトウェア開発をテーマとした演習授業ともども測定を行い，結果の対比を行っている．測定対象と対比授業の間に大きな差異が検出されているが，これはそれぞれの教材の優劣を示すものではない．本研究の演習は，ソフトウェアの作り方の基本を学ぶことを目的に設計したものであり，今回の測定はその効果を計ることを目的としている．したがって，対象の教育効果が，対比群よりもうまく検出できているかが分析のポイントである．

次に，C 大学，M 大学，S 大学での試行結果について，授業前と授業後の変化を示す．また再度 K 大学で，別教員が授業を実施した場合の結果についても示す．

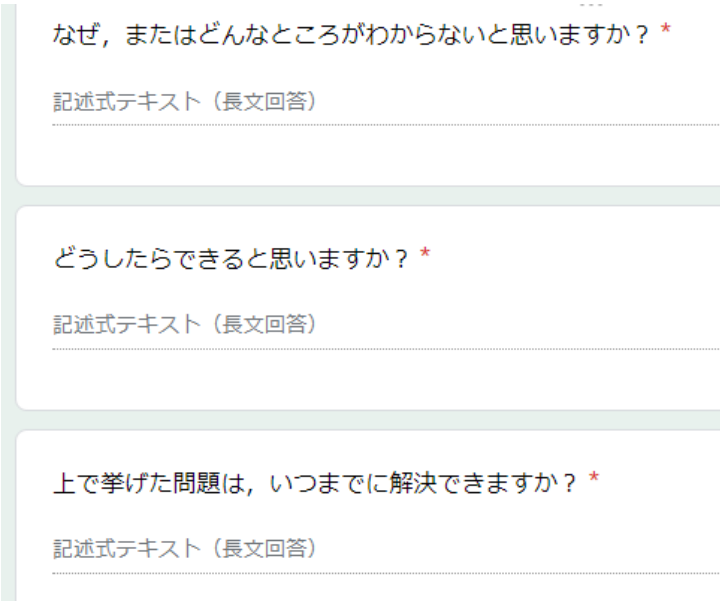


図 48: わかならないと回答した学生への質問

6.4.1 K 大学での事例

K 大学で 2018 年に実施した結果について述べる．K 大学では，情報系学部 3 年生 86 名を，43 名ずつ A と B の 2 クラスに分け，A クラスでは本研究の演習を，また B クラスでは本研究の演習ではないプログラミング演習を実施し，それぞれ修了した．

前節で示した測定手順に関し，Q1（特定の要求仕様に対して取り掛かれるか）の結果を示す．図 49 は本研究の演習を受講した A クラスについて，受講前後の調査結果である．受講前は，とりかかれる，無理，わからないは，ほぼ同数だった．受講後は,「とりかかれる」が 32%(14 人) から 56%(24 人) と増加している．ただしここでの「とりかかれる」には，根拠のない単なるやる気のみの学生も含まれているが，その分析は後で述べる．

一方で B クラスの演習授業後の調査結果を図 50 に示す．B クラスでは授業前の調査ができていないが，クラスは無作為に分けられており，A クラスの授業前調査と同等と推定している.「とりかかれる」が 42%(17 人) であり，推定演習前よりは増加しているが，A クラスの結果には及んでいない．また,「無理」という回答が A クラスの約 2 倍という結果を測定した．

この結果をクロス表にまとめたものを表 26 に示す．この自由度 2 の表から得られ

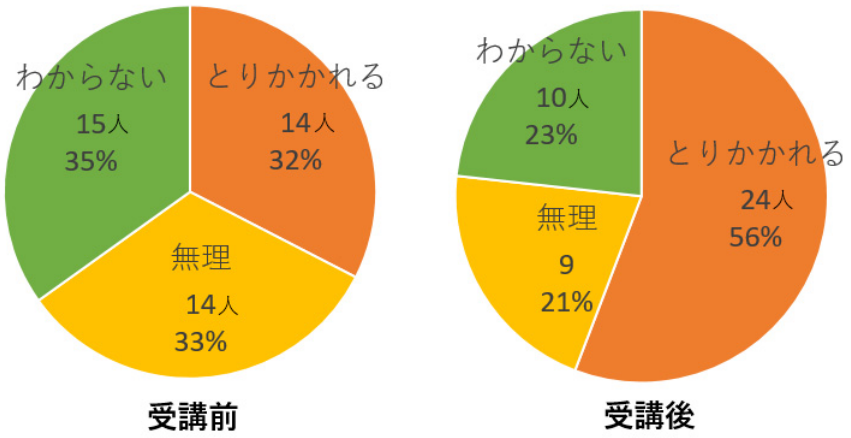


図 49: A クラス（本研究の演習）演習授業前後の結果

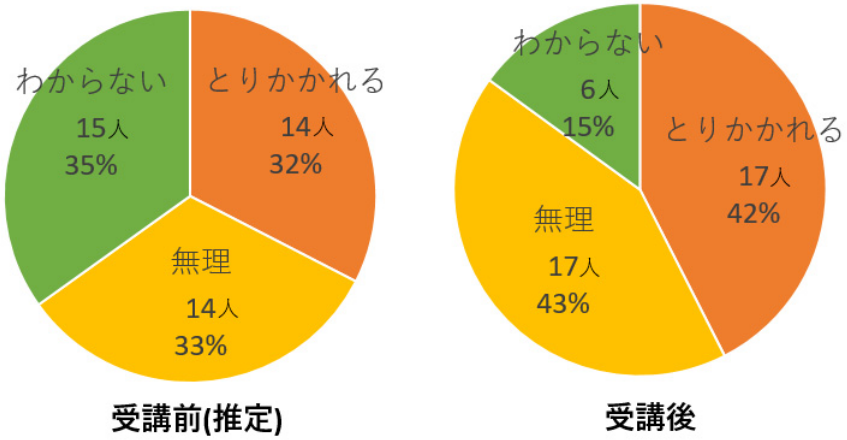


図 50: B クラス（別の演習）演習授業前後の結果

る χ 二乗値は 4.55 であり，上側累積確率は 0.10 という水準で，偏りがあることがわかった．

本項冒頭で述べたように，「とりかかれる」と回答した学生の中には，単なるやる気のみの学生も含まれていることが推定される．そこで，図 46 に示した設問のうち，「どのような順番で作業を進めますか」と「一番難しそうな工程とその理由は何ですか」についての記述内容を利用して，データの選別を行った．選別方法は，両質問ともに回答が表 27 に示したキーワードを一つでも含んだ説明をしているものを○として，そのまま「とりかかれる」群に残した．それら以外は根拠のない「やる気のみ」と判断し，「無理」群へ分類した．キーワードは，ソフトウェア開発プロセスの一般的

表 26: 本研究の演習受講者と別の演習受講者のアンケート結果クロス表（K 大学）

	できる		無理		わからない		
演習授業	観測値	期待値	観測値	期待値	観測値	期待値	観測値合計
本研究の演習	24	21.2	9	13.47	10	8.29	43
別の演習	17	19.8	17	12.5	6	7.71	40
合計	41		26		16		83

表 27: 選別のキーワード

計画に関するもの	計画　ガントチャート
要求に関するもの	要求　要件　仕様　仕様書　分析
設計に関するもの	基本　詳細　設計　部品　状態遷移　フロー
実装に関するもの	実装　コーディング
テストに関するもの	テスト　検査　単体　結合　システム
その他のもの	デバッグ

工程名を基本としているが，記述の表現にゆれがあるため，たとえば仕様については要求，要件，仕様，仕様書，分析，また実装については実装，コーディングと，表現の幅への対応を図った．また，設計については，具体的な設計手法名での説明も，設計行為を意図したこととして認めることとした．そこで，A クラス B クラスすべての回答中，設計手法として現れた「状態遷移」と「フロー（チャート）」の 2 つをキーワードに加えた．

表 28 および表 29 に，A クラス，すなわち本研究の演習受講者のうち,「とりかかれる」と回答した者の選別結果を示す.「難しそうな工程」への回答では，必ずしも工程名として回答していないものも含まれるが，開発プロセス上の作業は指しており，また主観的判断を避けるためキーワードによる抽出の結果をそのまま採用した．

次に，表 30 に，B クラスの選別結果を示す．記述内容的には開発プロセスを理解しているように思われるものもあるが，A クラスの選別同様，キーワードを使った選別のみとした．

この結果，実践力演習を受講した A クラスでは「とりかかれる」が 56%(24 人) から 37%(16 人) となり，別の演習を受講した B クラスでは「とりかかれる」が 42%(17 人) から 15%(6 人) になった．A クラスと B クラスの受講後データを選別したものを

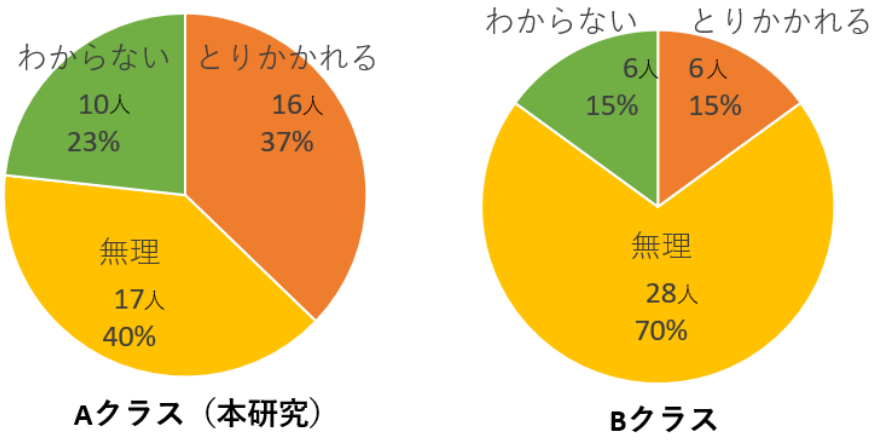


図 51: 演習受講後のデータの選別結果（K 大学）

図 51 に示す．

図 51 のデータをクロス表にまとめたものを表 31 に示す．この自由度 2 の表から得られる χ 二乗値は 8.14 であり，上側累積確率は 0.017 である．これは一般的有意水準 0.05 よりもかなり低い値での偏りがあると言える．

次に，本研究の演習受講者で,「とりかかれる」に選別できた A クラスの 16 人について，図 46 に示したアンケートの後半 3 つの問いについての回答をまとめたものを，表 32 に示す．4P のうちプロジェクトに相当する「ガントチャートを読めるか」については，1 人を除き「理解できる」「だいたいわかる」と回答した．また，プロフェッショナリズムに相当する「先輩や周囲の人に相談できるか」については，全員が「できる」または「するよう頑張れる」と回答した．一方でプロダクトに相当する，製品技術を問う設問には，ほとんどの学生が回答できていなかった．対象としているアンケートへの回答者 16 人は，全員が製品技術を駆使して，課題であるキッチンタイマのプログラムを完成させている．このことから，製品技術に関して，技術と用語の関連付けができていないことが原因と推測し，テキストに製品技術のページを追加し，製品技術とは何かについての説明を行う対策を行った．

表 28: A クラスの選別結果 1

No	どのような順番で作業を進めますか？	一番難しそうな工程とその理由は何ですか？	判定
1	まずは、問題を分析し、計画を立てる。次に設計や実装を行う。コードを完成してからテストを行い、動作を確認する。	詳細設計が一番難しいと思う。なぜなれば、この過程で様々なことを考慮しながらおこわなければならないからである。	o
2	必要な情報を要件から抜き出し、それから必要な技術を考えながら情報漏れがないか確認する。足りていることを確認できたら完成形を意識しながらパーツごとに設計、開発していく それぞれのパーツが要件を満たせたら、結合していき、微調整を加えながら全体を整える	結合テスト 正直今回の開発でも手こずったところで、作っている途中で問題が起きたときや作っている途中でもっといいやり方が見つかるのがテストのタイミングだったりするので問題がおきないように調整するのが難しいと思ったから	o
3	授業で習った順番をもとに進める。	仕様書 授業でも、難しかったところで、解釈によって意味が色々と変わってくるから。	x
4	仕様書を読んでからソフトウェア開発に着手する	デバッグで細かなところの修正 理由：修正箇所すべき場所を見つけるのに苦労しそうだから	o
5	仕様書を熟読し、実装する機能の洗い出しをする。 その後、それぞれの機能についてプログラミングをする。 そして、そのプログラミングをテストしながら、プログラムを修正していく。 最後に結合テストを行い、動作を確認する。	実装が難しそうだと思った。 なぜなら、自分がプログラミングに不得手だからだ。	o
6	仕様書を熟読してチームの人と認識を共有して設計を進める	依頼者との仕様の質疑	o
7	まず仕様書を読み込んでプログラムを組む中での疑問点がないか考える。 そのあと仕様書をもとに状態遷移図を書きプログラムを書いていく。	状態遷移図を書くのが一番難しいと思う。理由は状態遷移図がかければプログラムを書くのはそう難しくないが、状態遷移図がかけないとトラブルが起きたときにどこが間違っているのかがわからないから。	o
8	わかるものから勧めていき、わからないものはすぐ周りに聞いて解決する。	微小な部分（今回で言うチャタリング）が難題だと思う。体現しにくいから。	x
9	フローチャートを作る。 状態遷移図を作る。 システムの部品化をおこなう。 実装する。 バグが無いかを確認する。	実装の段階 うまく行かなかったときに修正する作業はどこが間違っているかを 探す必要があるので時間がかかるから。	o
10	ガントチャートの通りの順番	仕様分析 要求された仕様をどこで実装するかがまだわからないところがあるから。	o
11	今回習ったように、まずは要求分析からはじめ、基本設計、詳細設計をして実装する。	単体で書いたプログラムを結合するとき。 今回のキッチンタイマーを作る工程で9コマほど時間がかかったから。	o
12	仕様書を確認、修正 設計図の作成 部品の作成 部品を組み立てる 動作テスト	設計図の作成 それぞれの部品の動きを決めて作らなければいけなく、設計図にミスがあるとこのあとの作業で支障が出るため。	o

表 29: A クラスの選別結果 2

No	どのような順番で作業を進めますか？	一番難しそうな工程とその理由は何ですか？	判定
13	仕様書をみて自分ができるところから進めていく。	仕様書を読んで難しく感じたところ。	o
14	勉強した内容を思い出すところから始めます。	最初からつまづきそうです。 どこから手をつけていいかわからなくなると思います。	x
15	仕様書をよく読む そして、状態遷移などを考える。	デバッグ 自分で作ったプログラムだとミスが見つかりづらい	o
16	物の概要をまとめていく	設計 いろいろなことを加味しないといけないので、難しそう。	x
17	自分のできる範囲から組み立てていく。	仕様書を読んで全く理解ができない箇所 資料を読んだり上司の人教えてもらって1から学ぶ必要があるから。	x
18	まず要求分析をまとめて不明なところや疑問点などをまとめて報告したあとに細かいところなどを作っていく。	今回の実験で要求分析のところが苦手だと感じたのでそこが難しいと思います。	o
19	仕様書を読む 設計 システムテスト	設計段階でのプログラミング	o
20	仕様書を熟読し内容を紙に描き表してから上司に確認をしてもらい。わからないときは調べるなり聞くなりして作業を進める。	温度を測ること 具体的に考えつかないから	x
21	まずそのシステムに必要なものを１個ずつ作っていき最後にそれらを合わせていく	他のシステムと合わせるときに関数や変数をしっかり対応させること、プログラムのコンパイルが通っても動作するときに問題がでることがあるから。	x
22	まずはじめに仕様分析をして不足しているところがないかグループで話し合い、まとまった後に設計を行い単体テスト、システムテストを実施して完成させる。	まず1つ目が仕様分析で、不足した点があるとそれぞれで解釈違いを起こすなどの齟齬が生じるので難しそうである。 2つ目に基本設計で詰まると他のテストでもトラブルが起きそうなので難しそうである。	o
23	要求分析、基本設計、詳細設計、実装	実装、プログラミングとデバッグが難しい。	o
24	仕様書から完成系をイメージ どのような順序で設計を進めるかの決定 実行	システムのコーディング	o

表 30: B クラスの選別結果

No	どのような順番で作業を進めますか？	一番難しそうな工程とその理由は何ですか？	判定
1	仕様書を何度も読み直し、仕組みを十分に理解した上で取り掛かる。 わからないところがあればネットで検索したり、先輩や上司に聞いたりする。	温度が目標温度に達したら保温から沸騰に変えたり、その逆にも対応すること。 理由はとても複雑で条件の設定が難しそうだから。	x
2	はじめにハードウェアを設計し、そのあとにソフトウェアを構成する。	温度を制御することが難しそうだと感じた。なぜなら、加熱や保温をする際にさまざまな方式がありプログラムすることができそうにないと思ったからです。	x
3	図を描きプログラムの流れを掴んでプログラムを描き進める。	エラーの検知 条件が多く予期せぬ状態が発生しそうだから	x
4	仕様書に目を通し、構造を把握してプログラムを書き始める。 一人で問題解決できないときは、周りの人の力を借りて作業を進める。	数式を用いているため温度の制御が一番難しいと感じる。	x
5	基本動作について関数の作成を行い、条件ごとに決められた動作の呼び出しを行う	問題発生時の動作の設定 理由:問題発生とする条件の設定が難しいを思うから	x
6	まずは、要件と機能をいくつかの重要度に分類してまとめる。 次に、要件と機能を実現するための手段を考える そして、それらを元に実装を行う 最後に、実装が要件と機能を満たしているかどうか、テストし確認する。	最後に行うテストの工程。 どこに実装の誤りがあるかわからないので、気が抜けない。	o
7	まずは資料を読み、プログラムの流れを考える。その後、調べたり先輩にアドバイスを聞いたりしながら仕上げる。最後に上司に最終確認してもらう。	プログラムを作成する工程。それにかかる時間が一番長いから。	x
8	やることを明確化、効率的に作業を進めるにはどうしたらいいかを考える その後作業して全体ができたらチェックする 悪かった部分を修正していく	仕様書とまったく同じ動作にしていかなければいけないのでチェックの工程	x
9	要件定義(要求仕様書を読む), 外部設計, 内部設計, 開発	開発工程 理由:要求仕様書や設計書をもとに製品を開発しなければならないため.	o
10	仕様書を読む→わからないところを上司に聞く→仕様書の順に従って作業に取り掛かる	仕様書をきちんと理解すること ソフトウェアの知識が少ないから	o
11	何度か読んで理解していく。	すべて。	x
12	周囲の人に相談をして、過去の似たようなシステムを参考にしながら進める。	大枠を作る工程が、0から1を作らないといけないので難しいと思う。	x
13	まず、いただいた資料を一通り理解ができるまで読む。 次に、何をするかを書き出していく。 やらなければいけないことで、どうやればいいのかわからないものが出てきた場合、それをすればできるようになるかを調べる。 実際に作ってみる。	温度制御 理由： モードが3つあり、それぞれで色々な計算を行う必要があるので、想像したところ一番難しそうと感じた。	x
14	センサーの反応を確認 保温、節約、ミルクの構造体かクラスを作る ヒーターの動作確認 温度調整 条件分岐と切り替えの動作 モニターの表示 タイマーを作る	温度調整 理由：何回も試しそうだから	x
15	仕様書に書いてあるとおりに進めていく。	センサを組み込む過程 感度などの調整が必要だから。	o
16	設計書を読んだけどわからない	わからない	o
17	①仕様書をもとにUMLモデリング図を書く ②作りたいプログラムとUML図が一致しているか確認する ③プログラムを書く ④デバッグする	1番難しそうな工程はプログラムの作成。理想通りのプログラムを完成させるのに何度もデバッグや試行錯誤を繰り返す必要があるからである。	o

表 31: 選別結果のクロス表（K 大学）

	できる		無理		わからない		
演習授業	観測値	期待値	観測値	期待値	観測値	期待値	観測値合計
本研究の演習	16	11.4	17	23.31	10	8.29	43
別の演習	6	10.6	28	21.7	6	7.71	40
合計	22		45		16		83

表 32: 選別者のプロジェクト，プロダクト，プロフェッショナリズム（K 大学）
（青色太字は適切でない回答）

No	ガントチャートを読めますか？	どのような製品技術を使いますか？	先輩や周囲のひとに相談できますか？
1	読んで理解できる	**プログラミング技術、計画を立てる技術、など**	できる
2	読んで理解できる	タイマー UIを出すための 7セグメントやマトリックス表示技術 スイッチ等	するよう頑張れる
4	だいたいわかる	チャタリング対策など	できる
5	読んで理解できる	**まだどのような製品技術があるのかよくわからないが、Arduinoは使えると思う。 勉強がすすめば、他の技術も使ってみたい。**	できる
6	だいたいわかる	**わからないです**	できる
8	だいたいわかる	**わからない**	するよう頑張れる
9	**ガントチャートを知らない**	**我が道を行きます。**	できる
10	読んで理解できる	**製品技術というのが何を指すのかわかりません。**	するよう頑張れる
11	だいたいわかる	割り込み制御 ボタンのチャタリング対策	するよう頑張れる
12	わかる	**プログラミング技術 （c,python,arduino） 発想力**	するよう頑張れる
13	読んで理解できる	**やってみないとわからない。**	できる
15	だいたいわかる	**物による**	するよう頑張れる
18	だいたいわかる	熱センサやポンプ、電熱など？	するよう頑張れる
19	だいたいわかる	**自分の知っているものであれば何でも**	できる
22	読んで理解できる	**具体的に製品技術が何か調べてもわからなかったので不明です。**	するよう頑張れる
23	だいたいわかる	**プログラミング**	するよう頑張れる
24	だいたいわかる	**プログラミングスキル 状態遷移図 システムの理解**	するよう頑張れる

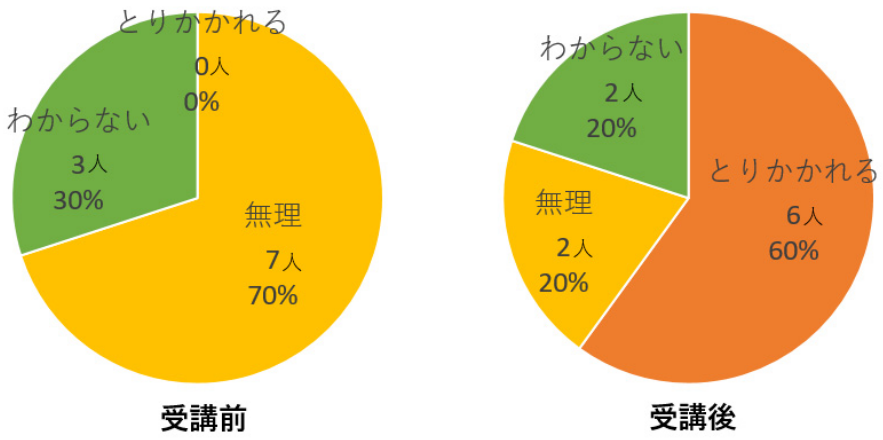


図 52: 本研究の演習受講前後の結果

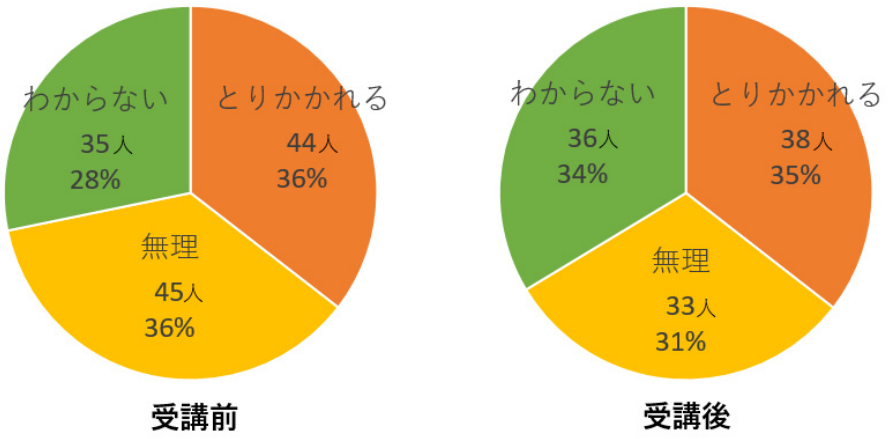


図 53: 別の演習受講前後の結果

6.4.2　B 大学での事例

B 大学では情報系学部 2 年生 10 人が本研究の演習を受講し修了した．また，それ以外の 107 人が本研究の演習ではないソフトウェア開発演習を受講し修了した．

本研究の演習受講者の結果を図 52 に示す．受講前は「とりかかれる」とした学生はいなかったが，受講後は 60 %（6 人）に増加した．

一方，別の演習授業受講者の結果を図 53 に示す．受講前後でほぼ変化がなく，しいて言えば「とりかかれる」は若干減少している．

これらの結果をクロス表にまとめたものを表 33 に示す．この自由度 2 の表から得られる χ 二乗値は 2.34 であり，上側累積確率は 0.31 という水準で，偏りがあること

表 33: 本研究の演習受講者と別の演習受講者のアンケート結果クロス表（B 大学）

	できる		無理		わからない		
演習授業	観測値	期待値	観測値	期待値	観測値	期待値	観測値合計
本研究の演習	6	3.76	2	2.99	2	3.25	10
別の演習	38	40.2	33	32.0	36	34.75	107
合計	44		35		38		117

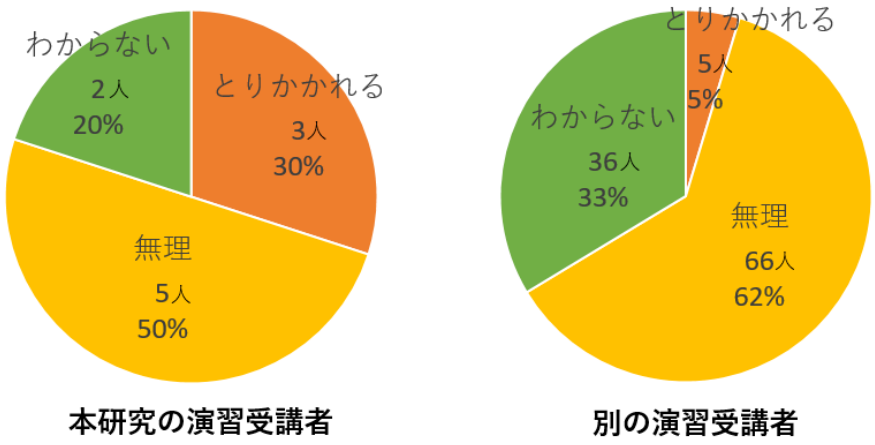


図 54: 演習受講後のデータの選別結果（B 大学）

がわかった．

6.4.1 同様に，図 46 への回答をもとに，根拠のないやる気だけの学生を「無理」群へ分類する選別を行った結果を図 54 に示す．

次に図 54 のデータをクロス表にまとめたものを表 34 に示す．この自由度 2 の表から得られる χ 二乗値は 9.30 であり，上側累積確率は 0.01 である．これは一般的有意水準 0.05 よりもかなり低い値で偏りがあると言える．

K 大学同様，本研究の演習受講者の選別者 3 人について，図 46 に示したアンケートの後半 3 つの問いについての回答をまとめたものを，表 35 に示す．ここでも，K 大学同様，製品技術という言葉への理解が進んでいないことがわかる．

6.4.3 C 大学での実施結果

C 大学では，2018 年度に 8 人，2019 年度に 6 人が受講している．それぞれについて，受講前と受講後の結果を図 55(2018 年度) と図 56(2019 年度) に示す．グラフ中の「選別削除」は,「とりかかれる」という回答ながら，表 27 を使った選別により「とり

表 34: 選別結果のクロス表（B 大学）

	できる		できない		わからない		
演習授業	観測値	期待値	観測値	期待値	観測値	期待値	観測値合計
本研究の演習	3	0.68	5	6.07	2	3.25	10
別の演習	5	7.32	66	64.9	36	34.75	107
合計	8		71		38		117

表 35: B 大学のプロジェクト，プロダクト，プロフェッショナリズムに関する回答（青色太字は適切でない回答）

ガントチャートを読めますか？	どのような製品技術を使いますか？	先輩や周囲のひとに相談できますか？
読んで理解できる	**製品技術が分からない**	するよう頑張れる
だいたいわかる	**わからない**	できる
読んで理解できる	**わからない**	するよう頑張れる

かかれる」から取り除いた人数を意味する．

2018 年度は，「とりかかれる」とした学生は，受講前 0%(0 人) であったものが受講後は 43%(3 人) に増加した．一方で 2019 年度は，受講前に「とりかかれる」と回答した学生は 67%(4 人) いたが，選別結果すなわちソフトウェア工学的根拠をもって説明できた学生は 0%(0 人) であった．受講後は選別後も 80%(4 人) が「とりかかれる」に残った．

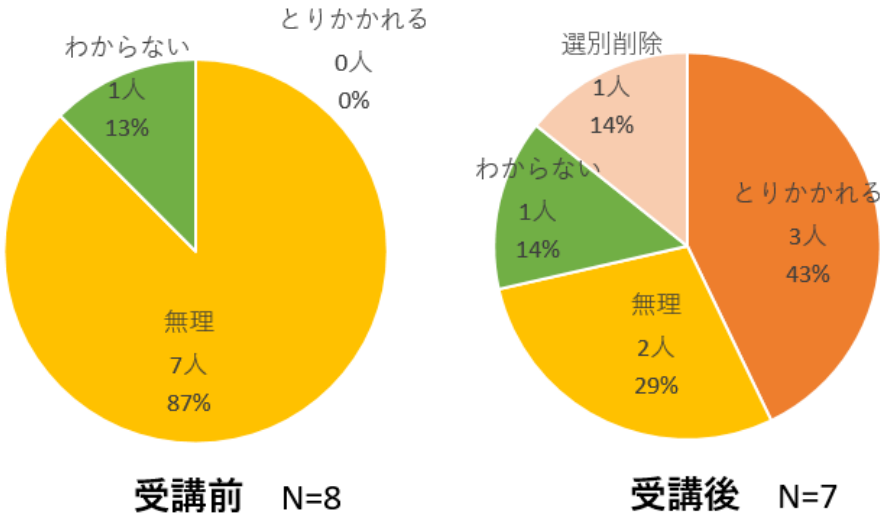


図 55: C 大学の受講前後（2018 年度）

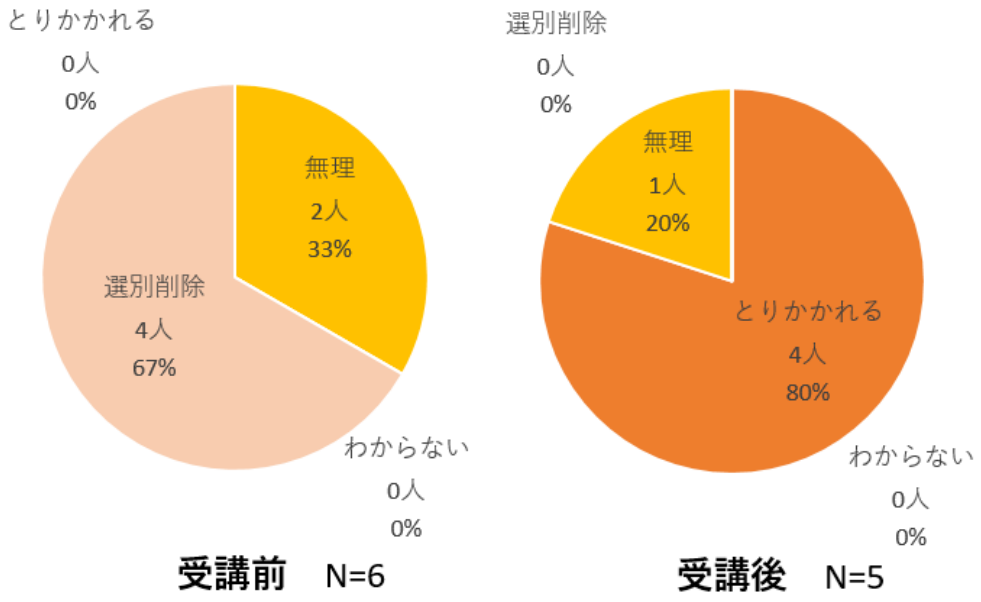


図 56: C 大学の受講前後（2019 年度）

表 36: C 大学のプロジェクト，プロダクト，プロフェッショナリズムに関する回答（青色太字は適切でない回答）

	ガントチャートを読めますか？	どのような製品技術を使いますか？	先輩や周囲のひとに相談できますか？
2018 年度	だいたいわかる	タイマー	するよう頑張れる
	だいたいわかる	水位センサ、ヒーター、ポンプ	するよう頑張れる
	だいたいわかる	ボタンなどの入出力技術、センサー類のデータを取得する技術、	できる
2019 年度	読んで理解できる	温度センサ、水位センサ、サーミスタ、ヒータ、蓋開閉感知センサ、ポンプ、	するよう頑張れる
	だいたいわかる	**C,C++、 Java、 python**	するよう頑張れる
	読んで理解できる	蓋センサ，満水センサ，水位センサのセンサ類とヒーター，スイッチ，ポンプなどのパーツ類.	できる
	だいたいわかる	水位を読み取るセンサ、水温を読み取るセンサ、タイマ、給湯口をロック・ロック解除するセンサ。	できる

選別した「とりかかれる」と回答した学生の，プロジェクト，プロダクト，プロフェッショナリズムに関する回答を，表 36 に示す．C 大学での演習では，製品技術についての説明を入れた版のテキストを使用した．K 大学 B 大学で問題であった製品技術（プロダクト）への回答は，概ね良好だった．

6.4.4　M 大学での実施結果

M 大学では，2019 年度に 18 人が受講している．受講前と受講後の結果を図 57 に示す．受講前受講後ともに,「とりかかれる」としたデータは選別を行っている．結果,「とりかかれる」人数は，受講前は 17%(3 人）から受講後 70%(12 人）に増加した．

選別した「とりかかれる」と回答した学生の，プロジェクト，プロダクト，プロフェッショナリズムに関する回答は，表 37 の通りであった．M 大学での演習では，製品技術についての説明を入れた版のテキストを使用した．

6.4.5　S 大学での実施結果

S 大学では，2019 年度に 4 人が受講している．受講前と受講後の結果を図 58 に示す.「とりかかれる」人数は，受講前は 25%(1 人) から受講後 75%(3 人) に増加した．

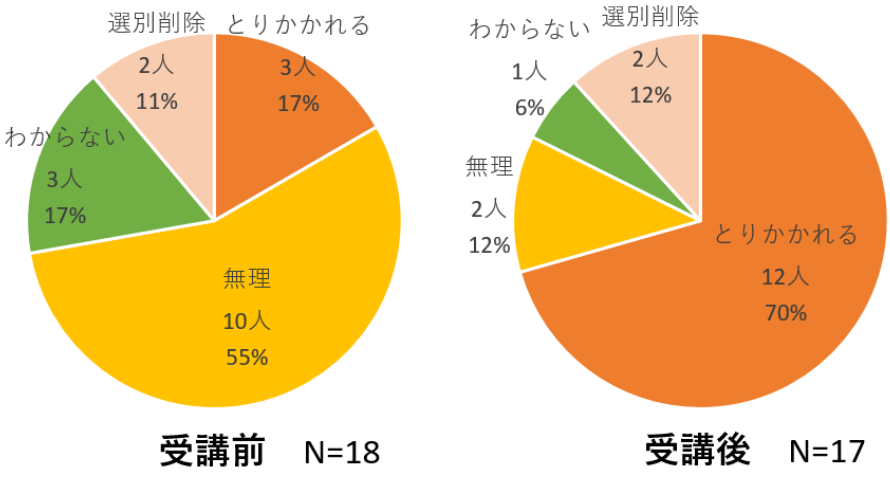


図 57: M 大学の受講前後（2019 年度）

選別した「とりかかれる」と回答した学生の，プロジェクト，プロダクト，プロフェッショナリズムに関する回答は，表 37 の通りであった．S 大学での演習では，製品技術についての説明を入れた版のテキストを使用した．

6.4.6 K 大学での実施結果

最後に K 大学における 2019 年度の結果を述べる．本演習は，別の教員が実施している点が，これまでに述べた事例と異なっている．受講者は，A クラス 34 人，B クラス 35 人だった．データを採取できた B クラスの結果を図 59 に示す．選別後の「とりかかれる」を選んだ人数は，受講前 6%(2 人) から受講後 46%(16 人) に増加している．

選別した「とりかかれる」と回答した学生の，プロジェクト，プロダクト，プロフェッショナリズムに関する回答は，表 39 の通りであった．他の授業事例と比べて，ガントチャートの認識率が目立って低い．また，製品技術に関しても，理解していない学生が多く見られた．一方で図 59 の受講後データからわかるように，「とりかかれる」とした学生 19 人中 84%(16 人）が，なんらかの工程名等を用いて開発手順を説明できており，プロセスに関する知識は一定の理解に至っていると考えられる．プロセスに関する結果と，プロジェクト（ガントチャート）やプロダクトに関する結果との違いは何か．これは，教員の授業運営の影響であろうことが，担当した教員へのインタビューでわかっている．テキストではガントチャートの説明箇所は冒頭の 1 か所だけである．通常は，毎回授業冒頭に，実績を記入したガントチャートのページを学生に

表 37: M 大学のプロジェクト，プロダクト，プロフェッショナリズムに関する回答（青色太字は適切でない回答）

ガントチャートを読めますか？	どのような製品技術を使いますか？	先輩や周囲のひとに相談できますか？
読んで理解できる	Arduino/スイッチ/水位センサ/温度センサ/ヒータ/	できる
だいたいわかる	GPIO、ADC、LCD、タイマペリフェラル、低消費電力機能	できる
読んで理解できる	チャタリング対策、割り込み処理	するよう頑張れる
読んで理解できる	押しボタン式スイッチのチャタリングを抑える技術、定期的にプログラムを実行するためのタイマー技術	するよう頑張れる
だいたいわかる	**Web系技術(ruby, AWS,mysqlなど)**	できる
だいたいわかる	スイッチの処理。	するよう頑張れる
読んで理解できる	スイッチ制御、水位センサ制御、蓋センサ制御、サーミスタ制御、タイマ制御、音を鳴らす、ヒータ制御、目標温度にして保つ技術	できる
読んで理解できる	スイッチ制御,LED制御,センサ制御,ヒータ制御	できる
だいたいわかる	スイッチの制御、LEDの制御、タイマーを使った処理	するよう頑張れる
読んで理解できる	**git, CD/CIツール, PaaS, IaaS, ほかクラウドサービス**	できる
だいたいわかる	**Arduino**	するよう頑張れる
だいたいわかる	**質問の意味がよくわからない。チャタリング防止などの話だろうか。**	するよう頑張れる

見せて，計画上の現在位置を示していた．しかし，今回の授業では，テキストに出てきた回のみの説明であった．プロダクトについても，一回のみの説明だけだった．プロセスは，テキストの構成上，工程が移るごとにＶ字モデルと現在の工程を示すページを設けており，必然的に繰り返し学べていたため，理解の定着率が高かったことが推測される．

各学校の授業構成や受講学生の保有スキルの違いにより，テキストと進捗は同期させることが困難である．そのため，プロセスと異なり，進捗のページをテキスト上に設けることはできない．対策としては，授業でのガントチャートの取り扱いについて，指導要領的に教員用テキストに記載することが考えられる．現在は，演習テキストの

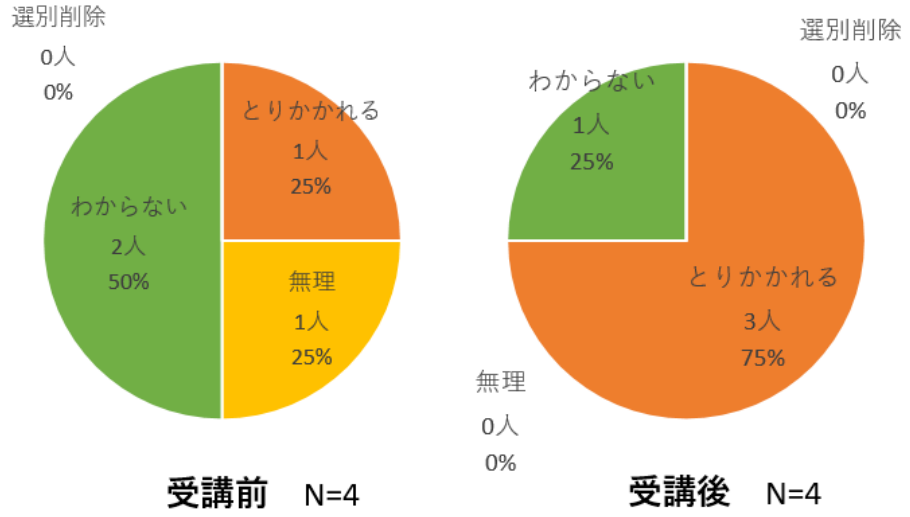


図 58: S 大学の受講前後（2019 年度）

表 38: S 大学のプロジェクト，プロダクト，プロフェッショナリズムに関する回答（青色太字は適切でない回答）

ガントチャートを読めますか？	どのような製品技術を使いますか？	先輩や周囲のひとに相談できますか？
だいたいわかる	タイマー、温度センサ、会社の既存技術	できる
読んで理解できる	**仕様書に記されたモノを実現する上で，必要最低限の機能を満たすもの.**	するよう頑張れる
だいたいわかる	タイマー制御・温度制御	するよう頑張れる

ガントチャートのページに，指導要領として記載している．

6.5 効果測定のまとめ

今回の研究では，本研究の教材による実践能力向上への寄与について，本物もしくはそれに準拠した要求仕様書を読ませて取り組みの意思を記述させることで測定することを狙った．

未知の開発に対して,「とりかかれる」と自信を表明した学生が本研究の演習受講により，少なくともこれまでの事例においては，増加することを観測できた．また 4P に関しては，まず開発プロセスに関する用語による選別を行った場合，受講後の増加

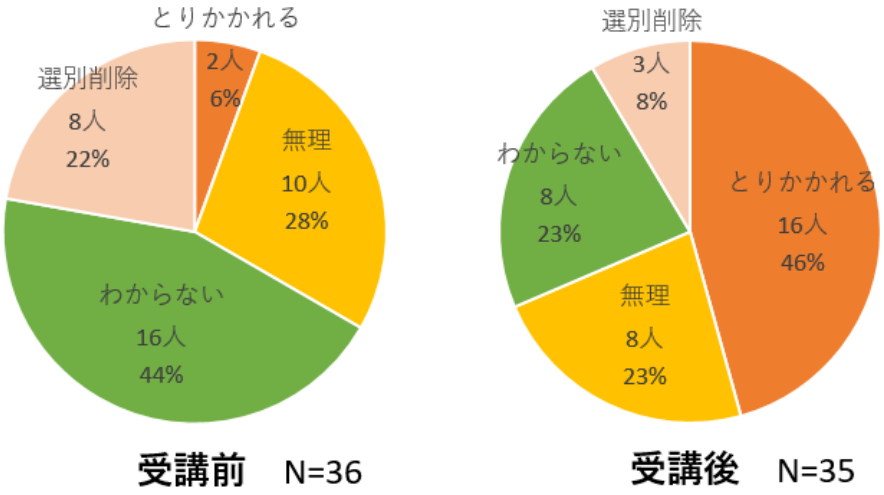


図 59: K 大学の受講前後（2019 年度）

は著しいことが測定できた．また，自信力を示した学生は，プロジェクトやプロダクト，プロフェッショナリズムに関しても，概ねポジティブな回答をしていることが測定できた．逆に，指導方法によって理解度が低下することも検出できており，今回の教育効果測定方法には，一定の意義があると考えらえる．

表 39: K 大学のプロジェクト，プロダクト，プロフェッショナリズムに関する回答（青色太字は適切でない回答）

ガントチャートを読めますか？	どのような製品技術を使いますか？	先輩や周囲のひとに相談できますか？
だいたいわかる	タイマー機能	できる
だいたいわかる	時間を図る 音を鳴らす LEDを光らせる	できる
ガントチャートを知らない	状態遷移図 クラス図 フローチャート シーケンス など	するよう頑張れる
ガントチャートを知らない	PID制御方式、温度制御テーブル方式などを使います。	するよう頑張れる
ガントチャートを知らない	プログラム上で条件を達成できるように工夫する。	できる
ガントチャートを知らない	**c 言語**	できる
ガントチャートを知らない	**まだ勉強が足りないので身についている製品技術はない。**	するよう頑張れる
読んで理解できる	**質問の意味がイマイチわかりません**	できる
読んで理解できる	**Arudino**	できる
読めない	タイマー 温度計	できる
読んで理解できる	**？**	できる
だいたいわかる	**フローチャートを使用しプログラムのながれをより視覚的にわかりやすくする**	するよう頑張れる
ガントチャートを知らない	**よくわからないのですが、一般的に知られている、使われているプログラム言語を使って作業をするよう心がけたいです。**	するよう頑張れる
だいたいわかる	タイマー、LED	できる
だいたいわかる	割り込みの仕組み、すでに存在する部分のドライバ	できる
だいたいわかる	センサー技術やスイッチのチャタリングなど	できる

第7章　研究のまとめと考察

本研究の研究目的は，次の通りであった．

1. 組込みソフトウェア技術者に求められる「実践力」とはなにかを明らかにする
2. 実践力を修得するための効果的な教育モデルを設計し，具体的教材を開発する
3. 開発した教材を用いて汎用的カリキュラムを作成し，組込み教育を行う大学へ展開する
4. 2 で設計した教材に関し，有効性を測定する方法を提案する

目的 1 の実践力については，社会で活躍する技術者も，技術に関わる教育関係者も，誰もが本研究の結論と同様の思いを持っており，それは一種暗黙の了解のような存在だった．今回，少ないながらも，技術に携わって社会で活躍している社会人の意見を集約し，発表論文について学術側にも同意を得られる形で明文化することができた．そして実践力の調査結果から，プロダクト，プロセス，プロジェクト，プロフェッショナリズムという 4 つのテーマを内包する教育モデル，4P モデルを帰納的に導いた．

目的 2 に関し，本研究では，4P モデルに準拠しつつ，16 コマ（1 コマは 90 分）前後の演習授業で学習可能な教材を開発した．まず基本コンセプトに基づいて試作教材を作成し，3 度の試行授業により抜本的改良を行い，その後の実践授業で小改良を行いつつ運用を継続している．本研究で開発した教材では，プロセスモデルに V 字モデルというウォーターフォール・モデル (Thomas E Bell. and T.A.Thayer, 1976) の変形を採用している．WF モデルのデメリットは良く知られており，今日このモデルを採用することに批判の向きもあるかも知れない．今回敢えて V 字モデルを使ったのは，仕様分析の重要性，工程の概念，全体スケジュールへのコミットメントを伝えやすいという理由による．あるいは，最悪そのような難しい概念は伝わらずとも，少なくとも組込み分野ではソフトウェア開発は工芸ではなく工学であるという認識を持たせたかったことによる．結果，受講後に未知の技術に挑む自信を得た学生の大部分

が，その根拠を工学的用語で説明できるに至っており，教材として一定の水準を得ているのではないかと思われる．

目的3については，ガントチャートによる計画とリンクする形でカリキュラム・パッケージ化ができた．実際に第三者の教員による授業の実施も問題なく行えており，2019年度は2校でカリキュラムとして実施，2020年度から3校へ教材を含むカリキュラムの配布および実施を予定している．

目的4について，6章に示した教育効果測定方法の発想と根拠は本文に述べた通りである．その特徴は以下の2点である．

1. 素材が本物の技術アイテムなので，特に本心でソフトウェア技術を目指している学生に，アンケート形式ながら訴求力がある．事務作業的な面倒くささでなく，本気で取り組んでもらえる．

2. 短時間で測定できる．たとえばPROGテスト (PROGテスト, 2011) が85分必要であるのに対し，仕様書に目を通すのに10分，アンケート記入に5分の，合計15分で実施可能である．

今回の測定では，予想通りに意図したとおりの結果が得られた．この結果が本当かどうかは，個々を中長期にトレースしない限り検証できない．しかし，長年企業の開発現場で新人を見てきた立場で敢えて主観を述べれば，あのような本物の仕様書を読んでポジティブな意見を出せるというだけで，彼らに大きく期待できることは間違いないと考えられ，そのような行動を引き出せた教材には，一定の効果があると考えられる．

本研究では，具体的な教材を実際に開発し，運用した．また今後も改良しながら運用を継続する予定である．本研究で開発した教材は，一般的な大学の授業における16コマ前後（1コマは90分）の2単位相当に授業を想定している．16コマは学生にとっては長いが，その時間数は16 × 1.5時間=24時間にすぎない．社会人の勤労時間で考えると，残業ありで2日，残業なしでも3日弱程度の時間に相当する．このような短時間で学べる内容には，自ずと限界がある．今回の教材は，本研究で掲げた4Pモデルの唯一無二の実装例ではなく，また必要な技術すべてを網羅しているわけではない．あくまでも4Pモデルのインスタンスの1つにすぎない．たとえば高信頼性を求

める開発プロセスでは，工程ごとのドキュメンテーションが欠かせないが，教材例では授業時間数を考慮して省略している．また，実際の開発では新規開発よりもむしろ派生開発が多い現状もあり，その場合は差分設計や構成管理などを行う能力も必要になる．今後，今回の教育モデルと教材例の効用を基に，別の教材の開発も進め，より実践力を身に着ける教育システムへと教材を充実させていく．

【謝辞】

本論文は筆者が信州大学大学院総合理工学研究科システム開発工学専攻博士後期課程に在籍中の研究成果をまとめたものである． 同専攻教授香山瑞恵先生には，指導教官として本研究の実施の機会を与えて戴き，その遂行にあたって終始，ご指導を戴いた．ここに深謝の意を表する．

同大学総合情報センター教授不破泰先生，電子情報システム工学科准教授橋本昌巳先生，総合理工学研究科工学専攻助教小形真平先生，ならびに中京大学工学部教授長谷川明生先生には副査としてご助言を戴くとともに本論文の細部にわたりご指導を戴いた．さらに長谷川先生には，本研究の実践の場も提供いただいた．ここに深謝の意を表する．

北九州市立大学大学院情報工学専攻准教授山崎進先生には，本研究の実践の場を提供いただいたにとどまらず，教材設計や研究手法など，仔細に渡り一方ならぬご指導を戴いた．山崎進先生なしに，本研究はありえなかった．ここに，深謝の意を表する．

中部大学情報学部教授鈴木裕利先生には，本教材実践の場をご提供いただいた．また一宮研伸大学看護学部准教授石井成郎先生ともども，研究にご助言いただいた．ここに，深謝の意を表する．

本教材で使用したキッチンタイマの仕様は，デンソー株式会社森孝夫氏の考案による．利用を快諾いただき，ここに，深謝の意を表する．

パーソナルスキルのエイリアスに関して，名古屋大学大学院情報科学研究科教授高田広章先生からプロフェッショナリズムという言葉のご助言をいただいた．ここに，深謝の意を表する．

参考文献

Christina Wodtke(2016). Introduction to OKRs, O'Reilly Media, Inc 2016
https://www.oreilly.com/library/view/introduction-to-okrs/9781491971475/ (参照 2020-01-06)

Maria Lydia Fioravanti, Bruno Sena, Leo Natan Paschoal, Laiza R. Silva, Ana P. Allian,Elisa Y. Nakagawa, Simone R. S. Souza, Seiji Isotani, Ellen F. Barbosa(2018). Integrating Project Based Learning and Project Management for Software Engineering Teaching: An Experience Report,806-811

Google スカラー (2018).
https://scholar.google.co.jp/ (参照 2020-01-06)

Filiz Kalelioglu,Yasemin Gulbahar(2014). The Effects of Teaching Programming via Scratch on Problem Solving Skills: A Discussion from Learners' Perspective,Informatics in Education,13,33-50.

Thomas E Bell. and T.A.Thayer(1976). Software requirements: Are they really a problem?,Proceedings of the 2nd international conference on Software engineering,ICSE '76,61-68

Christina Wodtke(2016). Radical Focus: Achieving Your Most Important Goals with Objectives and Key Results,Cucina Media LLC,ISBN-13 978-0996006026

プロジェクトマネジメント知識体系ガイド（PMBOK ガイド）第 6 版 (2018), Project Management Institute,ISBN-13 978-1628251906

Arduino

https://www.arduino.cc/ (参照 2020-01-06)

OMG ABOUT THE UNIFIED MODELING LANGUAGE SPECIFICATION VERSION 2.5.1

https://www.omg.org/spec/UML/About-UML/ (参照 2020-01-06)

Seeed studio

https://www.seeedstudio.com/category/Grove-c-1003.html (参照 2020-01-06)

NPO 法人 組込みソフトウェア管理者・技術者育成研究会, 組込みシステム教育教材 (2004). 組込みシステム教育教材　話題沸騰ポット要求仕様書 (GOMA-1015 型) 第 6 版

http://www.sessame.jp/workinggroup/WorkingGroup2/POT_Spec_AgreementOfRightToUse.htm (参照 2020-01-06)

NPO 法人 組込みソフトウェア管理者・技術者育成研究会

http://www.sessame.jp/ (参照 2020-01-06)

組込みシステム技術協会・状態遷移設計研究ワーキンググループ

http://www.jasa.or.jp/TOP/activity/technology/state/ (参照 2020-01-06)

伊藤篤, 渡辺裕, 樽松明他 (2007). 携帯電話用組み込みソフトウェア開発の実践的教育における産学連携の課題, 情報処理学会論文誌,Vol.48(2),846-857

飯泉紀子, 田所孝文, 大立 薫, 平島直人, 井上博史 (2008). 組み込みソフトウェア開発における品質向上への取り組み, 日本科学技術連盟 SQiP Library

https://www.juse.or.jp/sqip/workshop/report/attachs/2008/3-A-report.pdf (参照 2020-01-06)

王文涌, 池田満, 李峰茉 (2007). プログラミング 教育における動機づけ教授方法の提案と評価, 日本教育工学会論文誌,31(3),349-357

大場みち子, 伊藤恵, 下郡啓夫, 薦田憲久 (2018). 論理的な文章作成力とプログラミング力との関係の分析, 情報処理学会論文誌教育とコンピュータ (TCE),4(1),8-15

河西理恵, 丸山仁司 (2010). PBL の学習効果と学生因子の関係について, 理学療法科学,25(2),203-208

川名茂之 (2004). 車載ソフト開発の現状, 情報処理学会論文誌,45(7),713-715

経済産業省 (2006). 経済産業省, 社会人基礎力
http://www.meti.go.jp/policy/kisoryoku/ (参照 2020-01-06)

経済産業省 (2010). 大学生の「社会人観」の把握と「社会人基礎力」の認知度向上実証に関する調査
https://selectra.jp/sites/selectra.jp/files/pdf/201006daigakuseinosyakaijinkannohaakutoninntido.pdf
(参照 2020-01-06)

経団連 (2011). 産業界の求める人材像と大学教育への期待に関するアンケート結果
(社) 日本経済団体連合会教育問題委員会
http://www.keidanren.or.jp/policy/2011/005.html (参照 2020-01-06)

工学における教育プログラムに関する検討委員会 (1998). 8 大学工学部を中心とした工学における教育プログラムに関する検討
http://www.eng.hokudai.ac.jp/jeep/08-10/pdf/pamph01.pdf
(参照 2020-01-06)

駒谷昇一, 田中二郎, 北川博之 (2009). 筑波大学における高度 IT 人材育成のための実践的ソフトウェア開発専修プログラム, 工学教育,57(4),92-98

斎藤祐一郎, 久野靖 (2013). コミュニケーションスキルを重視したソフトウェア技術者教育手法の研究, 第 54 回プログラミングシンポジウム予稿集,161-170

沢田篤史, 小林隆志, 金子伸幸, 中道上, 大久保弘崇, 山本晋一郎 (2009). 飛行船制御を題材としたプロジェクト型ソフトウェア開発実習, 情報処理学会論文誌,50(11),2677-2689

清水吉男 (2010). [改訂第 2 版] [入門+実践] 要求を仕様化する技術・表現する技術 -仕様が書けていますか?, 技術評論社,ISBN-13 978-4774142579

鈴木克明, 根本淳子, 合田美子 (2010). 我が国における ARCS モデルを巡る研究動向, 教育システム情報学会第 35 回全国大会, 講演論文集,8-28

鈴木克明 (2002). 教材設計マニュアル, 北大路書房,ISBN-13 978-4762822445

孫勝国, 甘泉瑞応,Tongjun Huang,Aiguo He, 程子学 (2005). 学習者の学習順序や反応パターンに基づいた学習状態推論法を用いる Web-based 教育支援システム, 情報処理学会論文誌,46(2),327-336

高田広章 (2001). 組込みシステム開発技術の現状と展望, 情報処理学会論文誌,42(4),930-938

舘伸幸, 山崎進, 次郎丸沢 (2014). オープン・ソース・ハードウェアを活用した学び方を学ぶ組込みソフトウェア教育, 工学教育,62(4),33-38

舘伸幸, 山崎進, 香山瑞恵 (2019). 組込みソフトウェア技術者育成のための開発工程を重視したプログラミング教育, 情報処理学会論文誌,60(2),633-641

舘伸幸, 山本雅基, 高嶋博之, 松原豊, 本田晋也, 高田広章 (2014). enPiT-Emb(名古屋大学)OJL による実践的組込みシステム教育,IEICE technical report,113(498),181-186

舘 伸幸, 山本雅基, 吉田則裕, 高嶋博之, 安藤友樹, 松原豊, 本田晋也, 高田広章 (2015). OJL による実践的組込みシステム教育, コンピュータソフトウェア,32(2),79-85

独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センター (2006). 組込みソフトウェア開発のための ETSS 標準ガイドブック 2006, 日経 BP,ISBN-13 978-4822202576

独立行政法人 情報処理推進機構 ソフトウェア・エンジニアリング・センター (2008). 組込みスキル標準 ETSS2008
https://www.ipa.go.jp/files/000023848.pdf (参照 2020-01-06)

独立行政法人 情報処理推進機構 (2008). 2008 年版組込みソフトウェア産業実態調査報告書・プロジェクト責任者向け調査
https://www.ipa.go.jp/files/000004463.pdf (参照 2020-01-06)

独立行政法人 情報処理推進機構 (2011). IT 人材白書 2011, 独立行政法人情報処理推進機構 IT 人材育成本部,ISBN 978-4-905318-04-0

独立行政法人 情報処理推進機構 (2017). IT 人材白書 2017, 独立行政法人情報処理推進機構 IT 人材育成本部,ISBN 978-4-905318-50-7

独立行政法人 情報処理推進機構 (2017). 組込みソフトウェア開発データ白書 2017
https://www.ipa.go.jp/files/000062317.pdf (参照 2020-01-06)

独立行政法人 情報処理推進機構 (2018). 組込みソフトウェアに関する動向調査
https://www.ipa.go.jp/files/000065315.pdf (参照 2020-01-06)

永田美和子, 小山英子, 三木園生, 上星浩子 (2005). 新人看護師の看護実践上の困難の分析桐生短期大学紀要 第 16 号,2005,31-36

野口秀人 (2015) モデル検査を用いたフォールトアナリシス手法の提案,2015-09-24
http://hdl.handle.net/10119/12968 (参照 2019-06-15)

橋本はる美, 佐野繭美, 岩崎重剛, 松永公広 (1997). 情報処理基礎教育における学習者の状態把握 - Lotus 123 を例にして- , 情報処理学会研究報告コンピュータと教育 (CE),94,23-30

長谷川喜子, 櫻井良樹, 湯浦克彦 (2014). 実践型 IT 演習による学生の行動特性向上の評価, コンピュータと教育 (CE),22,1-10

花野井歳弘, 有田五次郎, 澤田直, 牛島和夫, 吉元健次, 牧薗幸司 (2007). 双方向型産学連携実践教育, 情報処理学会論文誌,48(2),832-845

平山雅之 (2004). 組込みソフトウェアの現状, 情報処理学会論文誌,45(7),677-681

松浦佐江子 (2007). , 実践的ソフトウェア開発実習によるソフトウェア工学教育, 情報処理学会論文誌,48(8),2578-2595

松澤芳昭, 塩見彰睦, 秋川友宏, 酒井三四郎 (2009). ソフトウェア開発の教員主導型 PBL における反復プロセスと EVM 導入の効果, 情報処理学会研究報告,2009-CE-99(9),1-8

文部科学省 (2012). 情報技術人材育成のための実践教育ネットワーク形成事業の概要 http://www.mext.go.jp/a_menu/koutou/kaikaku/itjinzai/ (参照 2020-01-06)

文部科学省 (2016). 成長分野を支える情報技術人材の育成拠点の形成 2016. http://www.mext.go.jp/a_menu/koutou/kaikaku/enpit/index.htm (参照 2020-01-06)

安永航, 大場みち子, 奥野拓, 伊藤恵, 山口琢 (2013). PBL を対象としたインフォーマルラーニング環境の構築, 情報処理学会研究報告,2013-CE-121(10),1-7

山本雅基, 小林隆志, 宮地充子, 奥野拓, 粂野文洋, 櫻井浩子, 海上智昭, 春名修介, 井上克郎 (2015). enPiT における教育効果測定の実践と評価, コンピュータソフトウェア,32(1),213-219

渡辺晴美, 吉田正廣 (2011). 大学学部における組込みソフトウェア教育事例, システム LSI 設計技術 (SLDM),2011-SLDM-149(50),1-6

テキストマイニングツール http://textmining.userlocal.jp/ (参照 2020-01-06)

マイニング結果のスコア http://social.userlocal.jp/document/textmining_faq (参照 2020-01-06)

リアセック, 河合塾 (2004). ジェネリックスキルの成長を支援するアセスメントプログラム http://www.riasec.co.jp/progtest/test/ (参照 2020-01-06)