ソフトウェア開発の教育のビジョンを語ろう

山崎 進

北九州市立大学でのプログラミングを中心とするソフトウェア開発の教育の今までとこれからについて語る.「自ら学ぶ力を持たせる」「個性に合わせて長所を伸ばす」「現実社会の問題解決の経験を積ませる」教育の実現と普及を通して，創意工夫にあふれた社会の形成に貢献しようとしてきた．これまでインストラクショナル・デザイン，反転授業，アクティブ・ラーニングを取り入れた授業づくりと共同研究型インターンシップに取り組んできた．成果は出始めているので，今後は拡充と普及に努めたい．

We'll talk about the past, the present and the future of our software development education including programming in University of Kitakyushu. We are contributing to establishment of creative society through realization and spread of education embodying "mastery of self learning", "providing tailored learning" and "mastery of problem solving". We've work on instructional design including flipped classroom and active learning, and collaborative research internships. We produce successful results and will enlarge and spread our education.

1 はじめに

情報系の学科においてプログラミングをはじめとするソフトウェア開発の能力を学生に習得させることは重要であることに異論はないだろう．しかし，情報系の学科出身者の中でプログラミングを苦手とする者が少なからず存在することに頭を痛めている現状がある．もちろん北九州市立大学国際環境工学部情報メディア工学科 (以下，本学科) も例外ではなかった．

この現状を変えようという問題意識に立ち，2009 年から教材設計マニュアル [3] に基づいてインストラクショナル・デザイン (ID) による自習教材づくりに取り組み始め，順次授業科目を刷新していき，2013 年度入学生から実施されるカリキュラム改定で全面的に導入した．

本発表では，まず我々のこの経験を総括することを目的とする．ソフトウェア開発とその周辺のカリキュラムを提示し，それに沿ってこれまでに公表してきた各授業科目の ID 実践事例を紹介し総括する．

次に我々のソフトウェア開発教育のビジョンを提示したい．プログラミングが苦手な学生を減らそうという思いで始まった改革ではあるが，その延長線上には情報系の学科の卒業生の多くが就業することになるシステムエンジニアをどのように育成するかという課題がある．それに対する答えとして「自ら学ぶ力を持たせる」「個性に合わせて長所を伸ばす」「現実社会の問題解決の経験を積ませる」教育が理想像だと考えている．

さらにそのビジョンを実現するにあたって今後取り組むことについて述べたい．これには普及促進のため，本発表の全体にわたって今までの成果物を紹介することも含んでいる．

以下の構成は次の通りである．第 2 章では，カリキュラムと授業科目を開発するにあたってよりどころとした ID について紹介する．第 3 章では，我々のソフトウェア開発教育のビジョンを提示する．第 4 章では，我々がビジョンに沿ってどのようなカリキュラムと授業科目を開発してきたかを紹介する．第 5 章で

A Vision of Software Development Education
Susumu Yamazaki, 北九州市立大学国際環境工学部, Faculty of Environment Engineering, University of Kitakyushu.

は，今後取り組む授業科目を紹介する．最後に第 5 章でまとめる．各章の参考文献として提示することで，今までの成果物を紹介していく．

2 インストラクショナル・デザイン (ID)

インストラクショナル・デザイン (ID) とは，学習の効果・効率・魅力を最大限に引き出すために計画的に教育や学習環境を作り上げていくことである．ソフトウェア開発において議論されているのと同様に ID においても教育プログラム開発の工程 (分析:Analysis, 設計:Design, 開発:Development, 実施:Implemetation, 評価:Evaluation) が定義されており，頭文字をとって ADDIE モデルと呼ぶ．アジャイル開発プロセスのように工程を繰り返して改善していくことが推奨されている．

筆頭著者が ID に取り組むにあたり，特に初期時点では教材設計マニュアル [3] を参考に進めた．また ID の理論的背景や研究動向を把握するために主にガニェの著書 [1] を参考にした．

3 ビジョン

我々は 2013 年に教育理念として次の 5 つをまとめた [18].

1. 主体的な学びを促進する場をつくる
2. 個性に応じて適切な学びの機会を与える
3. 現実社会の中から問題を抽出して解決する経験を積ませる
4. レビューを重視する
5. 自ら学ぶ力を習得させる

2015 年にビジョンを見直すにあたって，2, 3 ,5 が目的により近く，それ以外が手段により近いと考えた．目的に近いものを元に文言を改め優先順位を見直して定義したのが次の 3 つの教育方針である [29].

1. 自ら学ぶ力を持たせる
2. 個性に合わせて長所を伸ばす
3. 現実社会の問題解決の経験を積ませる

以下，ブログ [8] に記述したそれぞれに対する思いをそのまま引用する [18].

3.1 自ら学ぶ力を持たせる

技術や社会環境は急速に進化するので，陳腐化も早くなってしまいます．そのような状況では，一旦学んだら終わりではなく，常に学び続ける姿勢を身につけることが求められます．また，整備された教材が常に用意されているとは限りません．適切な指導者もいないかもしれません．いつかは独り立ちしなければならない，それが宿命です．私たちは，教材がなく指導者がいない状態でも，自力で学び続けることができるように学生を育て上げます．

3.2 個性に合わせて長所を伸ばす

入学試験や企業の採用活動において，最初から優秀な学生を厳選して採りたいと思っている人は多いでしょう．しかし私の経験からは，一見優秀でないように見える学生であっても，適切な学びの機会を与えることで，めきめきと実力をつけて見違えるように成長することがしばしばあると主張します．大事なのは，どのような学びの機会を与えればいいか，学生の個性をよく見て個別指導することです．私たちは，学生の個性を見極め，個性に応じた学びの機会を与える指導を行います．

3.3 現実社会の問題解決の経験を積ませる

学生が社会に出るとさまざまな問題に直面します．それらの問題の多くは，1 つの模範解答があるとは限らない複雑な問題です．業務の中で大小様々な問題を解決していくことが求められます．問題解決能力を高めるには，最初は簡単な応用問題から始め，徐々に複雑で現実的な問題に取り組む訓練を積むことが大事です．また，複雑な事象の中から本質的な問題が何なのかを抽出する訓練も必要です．私たちは，そのような問題の発見と解決を繰り返し訓練する機会を学生に与えます．

4 本学科のソフトウェア開発教育の今まで

4.1 ソフトウェア開発とその周辺のカリキュラム

図 1 に本学科のソフトウェア開発とその周辺のカリキュラムを示す．このカリキュラムは本学科 2013 年度以降入学生のものである．図中のカッコ内は 2012

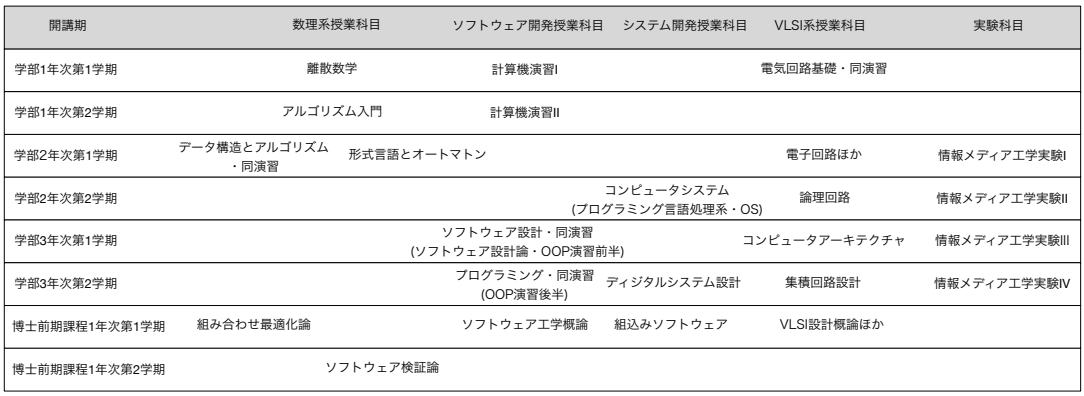

図 1　ソフトウェア開発とその周辺のカリキュラム

(本学科 2013 年度以降入学生のもの．カッコ内は 2012 年度以前入学生のカリキュラムの授業科目との対応関係)

年度以前入学生のカリキュラム (以下旧カリ) の授業科目との対応関係を示している．

ソフトウェア開発授業科目，システム開発授業科目，実験科目を中心に，大きく数理系授業科目，VLSI 系授業科目の 2 つを周辺授業科目群とした．

ソフトウェア開発授業科目，システム開発授業科目は，それぞれ講義科目と演習科目に分かれる．これらは次のように分類される．

- **ソフトウェア開発講義科目** (4.2 節):「ソフトウェア工学概論」
- **ソフトウェア開発演習科目** (4.3 節):「計算機演習 I/II」「ソフトウェア設計・同演習 (旧カリのソフトウェア設計論と，オブジェクト指向プログラミング演習 (OOP 演習) の前半) 」「プログラミング・同演習 (旧カリの OOP 演習後半)」
- **システム開発講義科目** (4.4 節):「コンピュータシステム (旧カリのプログラミング言語処理系とオペレーティングシステム (OS))」
- **システム開発演習科目** (4.5 節):「組込みソフトウェア」

また実験科目の中でとくに筆頭著者が開発した「情報メディア工学実験 III」の一部 (4.5 節) を取り上げる．

数理系授業科目には「離散数学」「アルゴリズム入門」「データ構造とアルゴリズム・同演習」「形式言語とオートマトン」「組み合わせ最適化論」「ソフトウェア検証論」を分類した．「離散数学」は集合論，論理学，証明などを基礎とし，数え上げ，順列・組み合わせ，確率，グラフ理論などを扱う．「アルゴリズム入門」と「データ構造とアルゴリズム・同演習」「組み合わせ最適化論」はアルゴリズムにまつわる問題を扱う．「形式言語とオートマトン」は言語理論と計算理論の初歩を扱う．「ソフトウェア検証論」は形式手法を扱い，とくにモデル検査の実践を積む．

VLSI 系授業科目には「電気回路基礎・同演習」「電子回路」「過渡回路解析」「論理回路」「コンピュータアーキテクチャ」「集積回路設計」「ディジタルシステム設計」「VLSI 設計概論」ほかで構成される．このうち「ディジタルシステム設計」はプログラミング言語処理系も扱うため，システム開発演習科目の特徴も併せ持つ．

学部における卒業研究，大学院博士前期課程における特別研究は各研究室で個別指導を行う．筆頭著者の研究室では共同研究型インターシップを実施しており，名古屋大学の enPiT emb に参加している (4.6 節)．

4.2　ソフトウェア開発講義科目

ソフトウェア開発講義科目は「ソフトウェア工学概論」が該当する．この科目は学部において一通りソフトウェア開発演習科目を履修した後，大学院博士前期課程においてソフトウェア工学知識体系に基づいてソフトウェア開発の理論的側面を習得することを目的と

している．

「ソフトウェア工学概論」の教授方略として，ビジョン「自ら学ぶ力を持たせる」を踏まえ，リーディング・アサインメントとリサーチを中心とする SQRPR アプローチ [27] を考案した．SQRPR アプローチとは，授業全体を通して学生の意識を「教わる」から「学ぶ」へ転換する—これはまさにアクティブ・ラーニングの本質的目的である—ようにデザインすべく，次の要素を授業に盛り込んだ教授方略である．

要約 (Summarization) ：学生は教科書や配布資料を読み込んで，各トピックの要約をまとめる。漫然と講義を聴くのではない。これにより学生の意識を「教師から教わる」から「自分で学ぶ」ように変えることを狙う。

問い (Question) ：学生は要約をまとめる過程で生じた疑問点 (リサーチ・クエッション) も記述する。リサーチ・クエッションは必ずしも高度でなくとも良い。むしろ素朴な疑問の方が本質を捉えているものである。

研究 (Research) ：学生は自らが立てたリサーチ・クエッションについて，さまざまな文献資料をあたって調査する。これにより深く学ぶ。

発表 (Presentation) ：学生は研究成果をポスターセッションで発表する。また，研究成果をわかりやすくまとめた解説記事を記述する。研究成果を自分の言葉でまとめること，学生が互いに議論を深めあうことを意図する。

ふりかえり (Reflection) ：学生は最後にこの授業で得たものは何だったのか，自らの学びをふりかえる。ふりかえることで，学習したことの定着を図る。

SQRPR アプローチの前身は MSQGP アプローチである [17]．これはメタファー，要約，問い，ガイド，プレゼンテーションの頭文字をとって命名した．ここでいうメタファーとは，ソフトウェア工学で登場する概念の理解促進のために学生にとって身近な例を用いたり体験したりするようなアクティビティを指す．その一例としてグループ学習による体験授業を実施した [11] [12]．ただしメタファーは授業全体ではなくソフトウェア開発工程の特性に応じて断片的に採用することにしたので，SQRPR アプローチとして再編する際に要素として外した．またガイドは SQRPR アプローチでは発表に置き換えている．

4.3 ソフトウェア開発演習科目

ソフトウェア開発演習科目はプログラミング演習を中心とする授業科目である．科目としては「計算機演習 I/II」「ソフトウェア設計・同演習」「プログラミング・同演習」が該当する．このうち「ソフトウェア設計・同演習」には UML モデリング演習が含まれる．

学生のプログラミング能力は個人差が極めて大きい．そのため，早く学習を進める学生に授業進度を合わせれば脱落者が続出し，学習を遅く進める学生に授業進度を合わせれば退屈する学生が続出する．個別に学習が進められれば理想であるが，一斉授業スタイルでは授業進度を個別に調整することはできない．

このような場合に ID で採用されるアプローチが個別化教授システム (PSI: personalized system of instruction) である．PSI では学生たちは自習教材をそれぞれのペースで学習を進める．きちんと習得したかを確認するため，単元ごとに試験が用意されている．もし学生が試験に合格したならば次の単元に進み，試験に不合格ならば再学習して再試験に臨む．授業では講師を補佐するプロクターという人員が配置され，机間巡回をして学生のフォローアップをしながら試験の合否を判定する．PSI では ID に基づいて単元を順番に学習すると段階的に円滑に学習できるよう設計されている．また各単元には多数の演習課題が準備されており，十分に習得と定着を図れる．これにより PSI によって完全習得学習 (mastery learning) された状態が実現できる．PSI を採用した教材の例として向後らが開発したウェブ教材 [2] がある．筆頭著者が関わっているソフトウェア開発演習科目「計算機演習 I」「ソフトウェア設計・同演習」では PSI を採用している．

「計算機演習 I」では向後らのウェブ教材を採用した [2]．ほぼ全員の学生が最低限必要な学習内容をクリアする．全学習内容が早く終わってしまった学生に対しては，「ソフトウェア設計・同演習」で提供するような高度なプログラミング演習を先取りして個別に

フォローアップし学習させる．

「ソフトウェア設計・同演習」では UML モデリング演習とプログラミング演習から構成される．どちらもオリジナルの PSI 教材を開発して授業を実施している．最終的に PSI を実現するために，長年かけて少しずつ実績を積み上げてきた．

旧カリの「ソフトウェア設計論」では演習課題の充実を図って完全習得学習を実現することに注力した [15] [14] [21]．一方,「OOP 演習」では，当初，実践的なソフトウェア開発に必要な要素を段階的に学習するスパイラル・カリキュラムに取り組んだ [10] [9] [13]．その後，学部生の段階で必要な学習内容を厳選し，授業内容を大幅に見直した．

これらを「ソフトウェア設計・同演習」に再編・拡充するにあたって,「ソフトウェア設計論」の全内容と,「OOP 演習」の前半である電卓 GUI アプリ開発を必修内容として PSI 教材を構成した．全学習内容が早く終わってしまった学生に対しては，GUI アプリの自由課題を課す．UML モデリング演習については独自の学習管理システムを試験的に導入した．

2015 年秋より新規開講する「プログラミング・同演習」では,「ソフトウェア設計・同演習」の学習の進捗や意欲に応じ，いくつかの教育プログラムを並行して提供し，選択できるようにした．基礎的な学習内容として，デザインパターンに基づく画像処理 GUI アプリケーションの開発を準備している．これは「情報メディア工学実験 II」での学習成果である C 言語による画像処理プログラムを再利用し，適宜リファクタリングを進めてデザインパターンを適用していくという内容である．また応用的な学習内容として学内外のプログラミングコンテストを運営／活用する．

なお，このような教授方略を有効に機能させるためには，ソフトウェア開発能力に長けたスタッフの拡充が必要である．そこで，我々の場合は，筆頭著者の研究室の学生総勢 10〜15 名を徹底的に鍛え，ソフトウェア開発の指導補助，具体的には，開発上の助言，フォローアップ，コードレビュー，採点フィードバック補助，教材開発補助等が十分できる能力をもたせた．そのような体制を組めたことが成功には不可欠であったと考えている．具体的な研究室の体制作りについては，4.6 節で紹介する．

4.4 システム開発講義科目

システム開発講義科目に該当するのは「コンピュータシステム」である．この科目は旧カリの「プログラミング言語処理系」「オペレーティングシステム (OS)」という 2 つの科目を統合した科目として開講した．統合する理由としては，2 科目に渡って詳細を学習させるよりも，より普遍性の高い基礎の習得に絞り 1 科目に統合して，空いた時間で他の科目を学ばせるようにしたほうが良いと考えたからである．また「コンピュータシステム」には,「形式言語とオートマトン」での学習内容を継承し，より高次な後続科目群であるシステム開発授業科目，ソフトウェア開発授業科目，VLSI 系授業科目への知的好奇心を喚起するという役割も期待されている [16]．

「コンピュータシステム」のコンセプトづくりはブログ記事 [16] に詳述した．次の順番でコンセプトを確定していった．

1. カリキュラムの中での授業の役割や位置付け
2. どのような学生が対象なのか，学生にどうなってほしいのか
3. 学習手段として何を使うのか
4. 教え方の方針やスタイル
5. 授業内容の範囲

1 については前述した通りである．2 については，プログラミングの苦手意識の克服と自ら学ぶ力を習得させたいという 2 点である．3 については「プログラミング言語処理系」で開発した自習教材，学習管理システム，PC 演習室や広い作業卓のある教室，学生の ICT 環境の有効活用を考慮した．4 については，アクティブ・ラーニングや反転授業を採用することにした．5 については，授業内容にコンピュータの動作原理を加えた上で基礎に絞ることを決めた．

後続科目群への学習意欲を喚起すること，アクティブ・ラーニングを採用することから，概念や原理を深く理解することを意図した．概念や原理を深く理解することはどういうことかを考察した結果，ガニェの多重に統合された目標 (multiple integrated objectives) [1] として次の 3 つの学習目標に整理した [19]．

用語の暗記 ：概念や原理に関連する用語とその意味を対応させて説明できる

直観的な理解 ：概念や原理，用語同士の関連について，例示や図示をしながら説明できる

応用技能 ：与えられた問題に対し，原理を適用して解を導き出せる

それぞれに合わせた教授方略を採用した．用語の理解については，ひたすら用語を暗記する．この目的で，用語を詳細に解説したテキスト，用語リスト群を提示しての調べ学習，e ラーニングによる小テストなどを開発した．

もちろん，それだけでは生きた知識にならないので，直観的な理解として，グループ学習によって原理を体験する，身近なメタファーにたとえて用語を説明させることを行った．前者の例としては，コンピュータの動作原理として CPU がバスを通じてメモリや I/O にアクセスしながら計算を進める様子を体感するロールプレイング・ワークショップ [31] [7] [6] を実施した．後者の例としては，マルチタスクの単元で，単元の初回にマルチタスクに関連する用語を覚えるのと並行して学生にマルチタスクを身近な例にたとえさせ (解答例としては料理，事務作業など)，単元の 2 回目で，初回の身近な例に沿って覚えた用語を一通り説明させる (たとえば事務作業におけるプリエンプションは上司の指示によって現在行っている事務作業を中断することに相当する) という指導方略を実施した [19]．応用技能としては，学んだ内容を活用してシステムプログラミングをするといったことである．

ビジョン「自ら学ぶ力を習得させたい」を体現化するため，授業全 15 回の前半では動画を用いるなどして懇切丁寧に指導して足場づくりをし，後半では調べ学習を中心にするなどして徐々に足場を外して自立学習を促すようにした [25]．

具体的な授業づくりについては文献 [28] [25] [24] を参照されたい．成果物として，動画 [23]，テキスト [22] [20] の各教材を公開している．歴史的経緯については文献 [26] を参照されたい．

4.5 システム開発演習科目，実験授業科目

システム開発演習科目としては「組込みソフトウェア」が該当する．「組込みソフトウェア」では実践的な組込みソフトウェア開発を学習させることが求められていた．舘と共同で授業を開発した [5] [4]．

さまざまな試行錯誤の末，たどり着いたコンセプトはビジョン「自ら学ぶ力を習得させたい」に沿ったコンセプト「技術の学び方を学ぶ」である．それを体現化するため，仕様書や回路図，取扱説明書のような技術文書を読んで問題を解決するという授業である [5] [4]．ソフトウェア開発演習科目と同様，学生個別に学習を進める．機材の都合上，ペアを組んで作業を行うが，授業前に補習を兼ねてプログラミング・モデリング能力を診断し，能力水準の近い学生同士でペアを組む．その理由としてはソフトウェア開発の能力水準に差があるペアでは，能力の高い学生がほとんどの作業を行ってしまって，能力の低い学生の学習を阻害してしまう問題を防ぐためである．演習課題が早く終わった学生への発展課題も用意した．

このように大学院において成果が得られたので，2013 年度以降入学生を対象に学部実験授業科目である「情報メディア工学実験 Ⅲ」に，「組込みソフトウェア」を学部生に合わせて簡易にした授業を実施した．

さらに，多様化するシステム開発の実情を考慮し，また「情報メディア工学実験 Ⅲ」を履修済みの学生が大学院に進学した時に「組込みソフトウェア」がより高度な授業内容が求められることを勘案して，2015 年度の「組込みソフトウェア」では，制御するデバイスの種類を大幅に増やし，より高度な問題解決を行う授業にする予定である．

4.6 研究指導

筆頭著者の研究室では，研究者ではなく実践的なエンジニアを育成する方針を採っている．ビジョンに掲げた 3 つの活動に次のような順番で取り組む．

1. 個性に合わせて長所を伸ばす
2. 自ら学ぶ力を持たせる
3. 現実社会の問題解決の経験を積ませる

「個性に合わせて長所を伸ばす」についてはブログ記事 [30] に詳述した．学生個々に面談を重ね，心理診

断テストを活用しながら，学生それぞれの個性を引き出し，学生の将来のビジョンを一緒に考えることを行う．その結果として学生それぞれの専門を設定する．

「自ら学ぶ力を持たせる」については，前述のように設定した専門について，近隣の勉強会への参加と，その内容について研究室内での勉強会の実施を奨励する方策を採っている．それにより，自主的に研究や勉学に勤しみ，学生同士で相互に学び合う学習環境が自然と形成されている．

そして「現実社会の問題解決の経験を積ませる」ことを具現化するために，研究室で最も力を入れている活動が，共同研究型インターンシップである．

経験的に，プログラミング能力を高めるためには，ソフトウェア開発のアルバイトに従事し，エンジニアなどの専門家の指導を受けながらプログラミングの経験を多く積むのが一番である．また，企業の採用活動，学生の就職活動でも，インターンシップによって実務で協働することで学生の適性や業務の詳細を共有することが有益である．しかしこれらを多く密度の濃い経験を積もうと思うと負担が大きくなり，学業と両立させることが課題となる．そこで，インターンシップ，アルバイトの活動を統合し，研究室と企業との共同研究として位置づけることで学業との両立を図るのが共同研究型インターンシップである．

共同研究型インターンシップは 2014 年度までに 7 テーマ 4 社 1 個人事業主 1 大学の案件で実施した．主業務は受託ソフトウェア開発であるが，他にもウェブのマーケティング，デザイン，サイト改善等も行っている．これらの業務では，パートナーとなるエンジニア，プロジェクトマネージャ，デザイナ，アナリスト等の専門家による監修・指導を受けている．組込みシステム開発が関連する案件では enPiT emb 名古屋大学 OJL とも連携している．

このように鍛えた上級生を演習・実験科目の教育補助に従事させ，下級生に個別指導する体制を整えている．それにより下級生のソフトウェア開発の能力と意欲が高まる．このような下級生が進級した後でさらなる下級生を指導するという好循環が形成されている．

5 将来課題

このような活動をさらに拡充していきたいと考えている．直近では，ソフトウェア開発演習科目の「計算機演習 II」とシステム開発演習科目と VLSI 系授業科目の両方の性格を持つ「ディジタルシステム設計」への適用に筆頭著者が関与することとなった．また実験科目へのさらなる関与・支援も要請されている．

さらにソフトウェア開発や教育に関わる組織・団体に積極的に普及活動を行っている．たとえば，関連する学会で発表・講演し，コミュニティに関与し，勉強会を主催するなどを行っている．またブログ [8] や SNS 等での発信も積極的に行っている．

6 まとめ

本発表では北九州市立大学でのソフトウェア開発の教育について，ビジョン，成果，行っている活動を紹介した．また参考文献として今までの成果物を紹介した．今後の更新や新たな活動については，随時ブログ [8] や SNS 等を通じて発信していく．

謝辞

今まで学内外の多くの方たちに支えられて研究と実践を重ねることができました．全ての人に深く感謝します．

参 考 文 献

[1] Gagné, R., Wager, W., Golas, K., and Keller, J.: *Principles of Instructional Design*, Wadsworth Pub, Belmont, CA, 5th edition, 2004. 邦訳: 鈴木 克明, 岩崎 信 (監訳): インストラクショナルデザインの原理, 北大路書房, 京都, 2007.

[2] 向後千春: ネコのぶきっちょと学ぶ C 言語, 2002. available at http://bit.ly/1JaSbM3.

[3] 鈴木克明: 教材設計マニュアル-独学を支援するために, 北大路書房, 京都, 2007.

[4] Tachi, N. and Yamazaki, S.: A Training Course for Embedded Software Engineers: Learning How to Learn New Technologies, *2014 International Conference on e-Commerce, e-Administration, e-Society, e-Education, and e-Technology (e-CASE), Nagoya, Aichi, Japan*, 2014.

[5] 舘伸幸, 山崎進: 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, Vol. 62, No. 4(2014), pp. 4_33–4_38.

[6] Yamazaki, S., Satoh, T., Jiromaru, T., Tachi, N., and Iwano, M.: Instructional Design of a Work-

shop“How a Computer Works”Aimed at Improving Intuitive Comprehension and Motivation, *Proc. 3rd International Conference on Learning Technologies and Learning Environments (LTLE 2014), Kitakyushu, Japan*, IIAI, 2014. presentation is available at http://bit.ly/1fuA2QH.

[7] Yamazaki, S., Satoh, T., Jiromaru, T., Tachi, N., and Iwano, M.: Instructional Design of a Workshop“How a Computer Works”Facilitating Intuitive Comprehension with Driving Motivation, *Information Engineering Express*, Vol. 1, No. 2(2015), pp. 21–30.

[8] 山崎進: ZACKY's Laboratory. available at https://zacky1972.github.io.

[9] 山崎進: 【中間報告】スパイラルカリキュラムによるソフトウェア工学教育, 情報処理学会論文誌, Vol. 51, No. 12(2010), pp. 1–5.

[10] 山崎進: スパイラルカリキュラムによるソフトウェア工学教育, ランチョンセミナー第 *58* 回, 熊本大学 *e* ラーニング推進機構, 2010. available at http://bit.ly/1KzoRid.

[11] 山崎進: グループ学習によるソフトウェア工学の授業設計, 日本教育工学会第 *27* 回全国大会, 2011.

[12] 山崎進: グループ学習によるソフトウェア工学の授業設計, 第 *25* 回 *SEA* 教育ワークショップ, ソフトウェア技術者協会教育分科会 SEA SIGEDU, 2011. avaiable at http://bit.ly/1hyOQ2O.

[13] 山崎進: スパイラルカリキュラムによるソフトウェア工学 e-learning 教材の開発, ランチョンセミナー第 *84* 回, 熊本大学 *e* ラーニング推進機構, 2011. available at http://bit.ly/1NzwD0I.

[14] 山崎進: UML モデリング教育, 第 *26* 回 *SEA* 教育ワークショップ, ソフトウェア技術者協会教育分科会 SEA SIGEDU, 2012. avaiable at http://bit.ly/1hPEA5T.

[15] 山崎進: ソフトウェアモデリング教育の開発〜初心者の C(自信) を重視した授業づくり〜, ランチョンセミナー第 *119* 回, 熊本大学 *e* ラーニング推進機構, 2012. available at http://bit.ly/1PD9JEY.

[16] 山崎進: 授業づくりはまずコンセプトづくりから〜事例に学ぶコンセプトづくり, 2013. available at http://bit.ly/1Kexljc.

[17] 山崎進: 授業構想:MSQGP アプローチ〜反転授業の応用, ランチョンセミナー第 *154* 回, 熊本大学 *e* ラーニング推進機構, 2013. available at http://bit.ly/1TXsb0Y.

[18] 山崎進: 教育理念:まとめ, 2013. available at http://bit.ly/1hB2Too.

[19] 山崎進: 理系知識習得科目をどう教えるか〜概念と原理のディープラーニング, 2013. available at http://bit.ly/1EdDZFk.

[20] 山崎進: C 言語のアセンブリ言語コード化〜直観編, 2014. available at http://bit.ly/1hPE5Zu.

[21] Yamazaki, S. and Jiromaru, T.: Instructional Design of Exercise-Centric Teaching Materials on UML Modeling, *3rd International Conference on Learning Technologies and Learning Environments (LTLE2014), Kitakyushu, Fukuoka, Japan.*, 2014. presentation is avaiable at http://bit.ly/1UWSod6.

[22] 山崎進: コンピュータの基本構成と動作原理〜知識編, 2014. available at http://bit.ly/1NpVjtE.

[23] 山崎進: ワークショップの手引き「コンピュータシステム」コンピュータの基本構成と動作原理〜直観編, 2014. movie is available at http://bit.ly/1J9BK2n.

[24] 山崎進: 変化の激しい時代に生き残れる工学部学生を育てる二つのアプローチ, 第 *15* 回反転授業オンライン勉強会, 反転授業の研究 Facebook グループ, 2014. avaiable at http://bit.ly/1hPDSpt.

[25] 山崎進: 情報工学教育における知識定着と直観的理解を意図した反転授業／アクティブラーニングの授業設計, 第 *3* 回 反転授業公開研究会, 大阪府, 関西大学教育開発支援センター, 2014. presentation is available at http://bit.ly/1PpzgRr.

[26] 山崎進: 教材設計マニュアルからアクティブ・ラーニングへの道のり, 第 *28* 回 *SEA* 教育ワークショップレポート, ソフトウェア技術者協会教育分科会 SEA SIGEDU, 2014. avaiable at http://bit.ly/1hPDAPo.

[27] 山崎進: アクティブ・ラーニングの理想像を目指す授業づくり, 教師が変わる, 学生も変わる—ファカルティ・ディベロップメントへの取り組み *(*シリーズ 北九大の挑戦 *3)*, 北九州市立大学, 中溝幸夫, 松尾太加志 (編), 九州大学出版会, 2015, chapter 5-1, pp. 153–164.

[28] 山崎進: コンピュータシステムの原理を理解させるアクティブ・ラーニング, 電子情報通信学会総合大会公開シンポジウム 「教育改革と人材育成」, IEICE, 2015. presentation and movie are avaiable at http://bit.ly/1LmTP01.

[29] 山崎進: プロフィールにビジョンとミッションをこめて, 2015. available at http://bit.ly/1JcOqFM.

[30] 山崎進: 山崎進研究室のひみつ〜個性に合わせた長所を伸ばす研究指導, 2015. available at http://bit.ly/1LrBM94.

[31] 山崎進, 佐藤敬, 次郎丸沢, 舘伸幸, 岩野正史: 直観的な理解を促進するワークショップ「コンピュータの動作原理」の授業設計, パーソナルコンピュータ利用技術学会論文誌, Vol. 9, No. 1-2 合併号 (2015), pp. 40–48.