[解説]

On the Job Learning: 産学連携による新しいソフトウェア工学教育手法

小林 隆志，沢田 篤史，山本 晋一郎，野呂 昌満，阿草 清滋

要旨

本稿では，産学が密に連携し，企業における実際のソフトウェア開発を題材とし実践的な教育を行う新しい教育手法 On the Job Learning (OJL) について紹介する．

昨今，高度 IT 人材の不足が深刻な社会問題として指摘されているが，大学などの高等教育機関では，産業界が求める実践的な技術力を持つ人材を育成することが必ずしも容易ではない．この問題を解決するために，我々は産学の密な連携により，ソフトウェア開発の実践力をもった人材を育成する教育プログラムを展開してきた．OJL は，我々のプログラムの中核をなす実開発プロジェクト型の教育手法である．本稿では，実際に三年間実施した経験からその実施方法や教育効果について議論する．

1 はじめに

IT 分野の発展は目覚しく，産業界における高度 IT 人材の不足が深刻な社会問題として指摘されるようになってきた [2]．我が国の国際的な競争力の低下を防ぐためにも，効果的に高度 IT 人材を輩出するカリキュラムや教育方法が望まれている．我々は，文部科学省が進める先導的 IT スペシャリスト育成推進プログラム[2] の一拠点のメンバとして 2006 年度から活動してきた．この取り組みの中で，我々はソフトウェア工学に関する基礎的な技術力，それら技術を実践する力，変化へ適応する力を育成するための教育カリキュラムを設計し，実施してきた．

本解説では，大学などの教育機関や産業界における実践的 IT 教育に関する試みについて概観したあと，我々が先導的 IT スペシャリスト育成推進プログラムの一環として設計したソフトウェア工学教育カリキュラムについて，OJL（On the Job Learning）と呼ぶ新しい教育手法を中心に説明する．OJL は PBL（Project Based Learning）に代表される教育用の仮想プロジェクトを題材とする教育と，インターンシップや OJT（On the Job Training）のように実際の業務を体験することによる実務訓練を融合し，産学連携でより実践的な教育を行うことのできるプロジェクト参加型の教育手法である．産業界の協力のもと，実際のソフトウェア開発を題材にすることで，開発技術の実践的な内容を深く理解させ，それが訓練ではなく教育となるように教員が指導を行うという点が特徴である．

本稿では，OJL の実施方法を説明し他の教育手法との差異を議論するとともに，これまで複数の企業からの支援のもとで我々が実施してきた実施事例を紹介しその実施方法や教育効果について考察する．

以下，2 章では IT 教育における既存のプロジェクト参加型教育に関して概観し，それぞれの特徴を議論する．次いで，3 章では，我々の拠

[1]本解説は電子情報通信学会ソフトウェアサイエンス研究会にて発表した同題目の技術研究報告 [1] を研究会での議論を参考に，また最新情報を加筆し，改訂したものである．

On the Job Learing: A New Software Engineering Education Method on Collaboration between Academia and Industry

Takashi Kobayashi[*1], Atsushi Sawada[*2], Shin-ichiro Yamamoto[*3], Masami Noro[*2], Kiyoshi Agusa[*1]

*1 名古屋大学大学院情報科学研究科

Graduate School of Information Science, Nagoya University

*2 南山大学情報理工学部

Faculty of Information Sciences and Engineering, Nanzan University

*3 愛知県立大学情報科学部

Faculty of Information Science and Technology, Aichi Prefectural University

［解説］2010 年 3 月 20 日受付

[2]http://www.mext.go.jp/a_menu/koutou/it/index.htm

点が展開する教育カリキュラムの全体像について紹介する．4 章では OJL の概要を説明し，実施方法や教員や産業界からの人員の役割，テーマの選定方法に関して説明する．その後，5 章で我々がこれまでに実施してきた事例を紹介し，6 章で OJL の有効性や課題について考察する．7 章は本稿のまとめである．

2 実践的 IT 教育とプロジェクト参加型教育

知識の獲得状況を説明・評価する上で有名な分類に Bloom らの知識レベル [3] がある．それぞれを IT 教育に当てはめると，次のような関係になろう．

Knowledge 技術的な用語を教科書や講義などで見聞きしたことがあり，関連する資料を調べることができる

Comprehension 技術用語の意味を理解し，表面的な説明をすることができる

Application 用語に関する技術・技法を情報システムやソフトウェアの開発に利用することができる

Analysis 技術の背景にある制約や特性について理解し，特定の問題に対する向き・不向きの比較・分析ができる

Synthesis 技術の組み合わせや改良を含め，新たな開発問題への応用ができる

Evaluation その技術に関する知識や経験に基づいて，ソフトウェア開発問題に対する結果の評価，適否の評価ができる．

実践的な技術力について習熟するという観点からは，**Application** レベル以上の知識としてそれぞれの技術を身につけることが求められる．しかしながら座学を中心とした形態の授業を展開する従来型の大学や大学院教育において，あらゆる技術について **Application** レベルの知識を養うことは現実的ではない．このような従来型教育では，実践的な技術力を養成するために，学生実験やコンピュータ演習などと卒業研究や修了研究（修士論文執筆のための研究）を組み合わせたアプローチがとられてきた．そこで学生が自らに課せられた問題解決のために技術を利用しながら実践力を伸ばす機会は，実質的には卒業研究や修了研究における研究指導となるが，これらを技術教育と考えると徒弟制度的色彩が強く，学習内容や教育効果に偏りが大きくなりがちである．

このような問題意識から，欧米を中心に IT 技術者に求められるソフトウェア工学に関する知識体系 [4] が定められ，また大学でのソフトウェア工学教育カリキュラムの推奨モデル [5] が提案されてきた．日本においても J07 と呼ばれる標準カリキュラムにおいてソフトウェア工学教育の推奨モデルが定められてきた [6]．これらの標準的カリキュラムモデルにおいても，技術の実践力を養う開発実習の必要性が強調されており，日本の大学からも，組込みシステム開発などを題材とした実習の事例がいくつか報告されている [7,8].

プロジェクト開発型の実習は，専門知識や技術を実践する能力を養う点で効果的であるとされ，先導的 IT スペシャリスト育成推進プログラムでも各拠点が PBL 型の教育プログラムを採り入れている [9–11]．ただし，いわば抽象化，一般化され，模範解答が用意された仮想的な開発を題材にする PBL では，技術や開発プロセスの実践，つまり **Application** レベルの達成が主目的となり，これを **Analysis** レベルにまで引き上げるのは難しい．**Analysis** レベルの知識を養うには，単に座学として簡単な例題に触れることや，特定の技術を適用するために設計された仮想プロジェクトを実践するだけではなく，現実問題に近い課題を設定し，問題を理解したうえで代替案とあわせて技術解を検討し，その技術を適用させる経験が重要となる．

現実問題に近い課題を設定したプロジェクト参加型教育は多く存在し，実システム開発を題材として用いた事例 [12] や，産学連携による取り組みが効果的であること [13] 等が報告されている．海外においても，産業界におけるソフトウェア開発課題を利用した教育を行う事例

に [14] 等がある．

また，産業界において実際の開発を題材とし教育を行う手法として，OJT [15] がある．OJT は，社員教育方法の一つで，実際の仕事を通じて，必要な技術，能力，知識，あるいは態度や価値観などを身に付けさせる教育訓練のことであり，多くの企業で導入されている．

学生に対する教育という点では，学生を一時的に企業の一員とし，就業体験をさせることで業務知識を獲得させるインターンシップも存在する．ただし，日本におけるインターンシップは，採用活動の一環として行われ，一ヶ月未満の短期のものがほとんどである．このように短期の実施では，実践的な技術力の大幅な向上は難しいのが現実である．日本型インターンシップのこのような問題を解消する試みもある．政策経営とヘルスケア分野へのIT活用をテーマとし，三ヶ月から半年以上にわたる長期インターンシップを展開する兵庫県立大学[3]の試みでは，学生の技術力や研究開発力が大幅に向上したことが報告されている．また，実践的技術力を養成する機会としてインターンシップを効果的に活用するために大学と企業の間を仲介する枠組みも産業界を中心として整備されつつある[4]．

プロジェクト参加型教育の中で現実問題を扱う場合は，教育効果が指導者の素質に強く依存することが問題になる．筆者らが [14,16] のメルボルン大学での実施担当者らにインタビューをした際も，指導者役の選定は大変重要かつ難しいポイントであるとの回答を得た．特定のドメインや企業における開発方法論の訓練ではなく教育を実施するという観点では，プロジェクトを運用し開発を進める実務能力と，それを題材に指導を行う教育能力の双方が必要となる．

3　OCEANのカリキュラム

我々は，先導的 IT スペシャリスト育成推進プログラムのもとで，本稿で紹介する OJL を教育の中核におく教育プロジェクトとして，名古屋大学を中心とし，南山大学，愛知県立大学，静岡大学，ほか連携企業六社が参画する「OJL による最先端技術適応能力を持つ IT 人材育成拠点の形成」（OCEAN: On the job Centered Education for Advanced engiNeers）[5] と銘打つ教育プロジェクトを実施してきた．

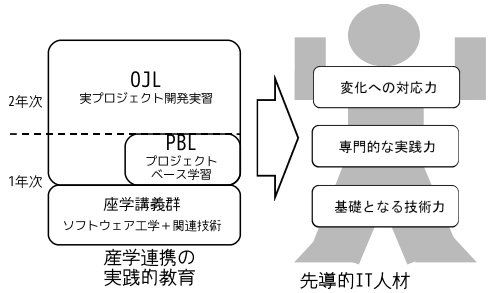

図 1: OCEAN の概要

OCEAN の概要を図 1 に示す．OCEAN では高度な専門性を持つ IT 人材に求められる素養を，ソフトウェア工学に関する基礎的な技術力とその実践力，変化への適応力と定義し，座学での講義群に，PBL および OJL という開発実習型の科目を組み入れたカリキュラムによって，大学院修士課程の学生へこれらの素養を教授することを狙っている．

座学での講義は修士課程の 1 年目に配置される．表 1 に示す通り，講義科目は，技術倫理，ネットワーク，ソフトウェア構築，ソフトウェア保守といった基礎科目に加え，ソフトウェア工学を中心とした技術要素に対して，深く掘り下げた内容を教授できるように，ソフトウェア要求工学，ソフトウェアアーキテクチャ，ソフトウェア設計技術，ソフトウェアモジュール化技術，正当性検証と妥当性確認，ソフトウェアプロジェクト管理の必修科目を配置している．また，各開発ドメインでの知識を教授するために，組込みシステム開発技術，分散システム開発技術，情報システム開発技術，実践的ソフトウェア開発技術といった講義科目を用意している．

各科目半期 1 講時ずつの授業時間 (1.5 時間×15 回の 22.5 時間) を割り当て，SE2004 [5] の知識体系のうちコンピュータサイエンス基礎と数学基礎を除く大項目を全科目でほぼカバーするよう設計されている．

[3] http://www.ai.u-hyogo.ac.jp/
[4] CeFIL：http://cefil.jp/
[5] http://www.ocean.is.nagoya-u.ac.jp/

表 1: 座学講義一覧

科目名（各 22.5 時間）	SE2004 知識体系大項目 * PRF	MAA	DES	VAV	EVO	PRO	QUA	MGT	SAS
IT 技術倫理と社会	◎								
ソフトウェア要求工学		◎			○	○	○		
ソフトウェアアーキテクチャ		○	○		○	○	○	○	
ソフトウェア設計技術			◎		○	○			
正当性検証と妥当性確認				◎			○		
ソフトウェアモジュール化技術			○	○			○		
ソフトウェアプロジェクト管理					○	○	○	◎	
ソフトウェア構築			○		○		○	○	
ソフトウェア保守				○	○		○	○	
IT ネットワーク			○	○					◎ net**
組込みシステム開発技術 I			○	○					◎ emb**
組込みシステム開発技術 II			○	○					◎ emb**
情報システム開発技術			○	○					◎ inf**
分散システム開発技術			○	○					◎ net,inf**
実践的ソフトウェア開発技術	○	○	○	○	○	○	○	○	◎ emb,inf**

◎ — 講義の中心的内容として深く講述
○ — 講義に関連する話題として講述
* PRF: 専門職業倫理，MAA: モデルと分析，DES: 設計，VAV: V&V，EVO: 保守・発展，PRO: プロセス，QUA: 品質，MGT: プロジェクト管理，SAS: 専門領域知識
** net: ネットワーク中心システム，emb: 組込み・リアルタイムシステム，inf: 情報システム

カリキュラムの中核をなすのは OJL である．次章で詳しく説明するが，実製品に近いレベルのソフトウェア開発プロジェクトを通じた教育手法であり，連携企業との協調により実施される．一方，PBL の役割は座学の講義群と OJL との橋渡しであり，OJL において連携企業等の技術者と開発プロジェクトを行う前に，チームでの開発経験，各種 CASE ツールの活用経験，ソフトウェア品質に対する考え方を補うことを目的として行う．飛行船制御 [17,18] の組込みソフトウェア開発を題材にしてチーム開発を体験させる演習を設計した [19].

4 OJL：On the Job Learning

我々が提案する OJL とは，産学共同による新しいプロジェクト参加型の教育手法である．仮想的な課題で開発を学ぶ PBL では実製品レベルのソフトウェア開発における品質や管理の難しさを真の意味で実感することができない一方，特定の組織おける業務のあり方を身につける OJT ではそこで教授される技術や作法の普遍性，本質的な価値を学ぶ事ができない．OJL では，PBL と OJT を組み合わせ，互いにその欠点を補完しあうことでより高い教育効果を狙っている．本章では以下，OJL の概要及び教育目標，テーマの選定方法について説明する．

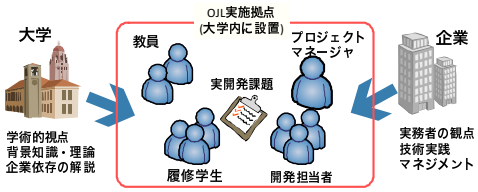

図 2: OJL の概要

4.1 概要

図 2 に OJL の概要を示す．上述の通り，OJL は PBL と OJT を組み合わせ補完することで，開発技術の表面的な理解だけでなく，その背景にある制約や特性についての深い理解を伴う実践的な技術力を養うことを目的とする．それを達成するための場として，OJL では，産学それぞれの立場から明確な役割を担う要員が参画する開発プロジェクト（OJL プロジェクト）を形成する．

典型的な OJL プロジェクトには，企業側から指導者と開発者が参画し，大学側から教員と学生が参画する．企業指導者は製品レベルの実システム開発課題をプロジェクトに持ち込むと

ともに，プロジェクト管理者の立場から，実システム開発のために必要な開発と管理のスキルを指導する．企業開発者は，学生とともにシステムやソフトウェアを開発する要員として必要に応じてプロジェクトに加わる．一方，大学教員は，プロジェクトにおいて学生の課題遂行を管理する役割を通じ，与えられた開発課題のドメインがもつ特徴や制約，さらに課題を解決するために用いられる開発や管理の技術を普遍的な視点からとらえ，問題と解法の本質を学生に教授する役割を担う．

表 2 は，OJL とほかのプロジェクト参加型の教育方法を比較したものである．OJL では実施拠点を大学内に設置することで，学生が長期間プロジェクトに参加することを可能としている．従事可能な期間が短いことから開発技術の十分な実践や教育ができないという問題は，インターンシップでしばしば問題となるが，OJL ではそのような事態に至ることはない．学生は，大学にいながら各 OJL プロジェクトが定める週報等の定期的な進捗報告や，工数と従事時間の管理，機密情報に関する取り扱い規則などを経験し，就業体験を超えたより実践的な学習を実施することができる．また教員が開発プロジェクトに参加をすることで，その技術や作法における普遍性，本質的な価値を教示し，特定の企業の開発スタイルに偏らない教育を実践することができる．

このように，OJL はあくまで大学における教育として行うものであるが，OJL プロジェクトで行うのは，連携企業から持ち込まれたテーマに関するソフトウェア開発作業である．さまざまな局面で機密情報を扱うことから，実施大学には図 3 に示すような，入退室が管理された OJL プロジェクト専用の部屋を用意し，機密の程度によっては開発作業やミーティングはこの部屋でのみ行うといった運用をする．

4.2 教育目標

OJL を通じた教育の主要な目標は次の三点である．

- 製品レベルの実システム開発の体験

図 3: OJL 専用室

- 開発や管理に関するスキルの修得
- 開発課題の特徴に応じた適用技術の取捨選択能力の獲得

一点目と二点目の目的は，製品に対する品質要求の高さ，高品質を達成するために現場で適用されている開発や管理に関わる諸技術について，実際にプロジクトに携わり，適用することで体験をすることであり，それを通じて教科書的に学んできた技術に対する制約や特徴を深く理解することにつながると期待している．三点目は，与えられた開発課題の特徴や制約を見抜く力を身につけることであり，それを通じてその課題に適用する技術を取捨選択し，また必要に応じてその課題に最適な専用の技術を創造することのできる能力が養われることを期待している．これらの目的は，産と学，どちらかのみが主体となる体制により実現できるものではなく，産学双方の強い協調関係の上にたってはじめて実現できるものである．

なお，教育と指導の対象となるのはあくまで学生であるが，OJL は企業から参画する開発者にとってもメリットを享受できる良い機会であるといえる．学ぶ立場にある学生とチームを組み，技術を伝授する指導的な立場で開発を行いながら，指導者がプロジェクトを通じ学生をどのように使い，また管理するのかを見ることで，プロジェクト管理のノウハウを学ぶ機会になる．また，学からの普遍的な視点を入れることで，企業内に閉じた開発技術や仕事の作法の本質を理解することにもつながる．

表 2: OJL とプロジェクト参加型教育の比較

方式	対象者	指導者	題材	期間・規模	主な目的
PBL	学生・社員	教育担当者	教育用プロジェクト	6ヶ月	専門的スキルの伝授
OJT	社員	業務担当者	現実の業務	2ヶ月	社内業務への適応
インターンシップ	学生	業務担当者	現実の業務	1ヶ月	ジョブマッチング
OJL	学生・社員	業務担当者と 教育担当者が協力	現実規模のプロジェクト	18ヶ月	専門的スキルの伝授と 変化への対応力の育成

4.3 テーマ選定とそのタイプ

OJL プロジェクトを実施するにあたっては，そこで解決すべき開発課題を与え，教育目標を達成するためにとるべき手段，またそこで何に着目した指導を行うべきかについて決定しなければならない．本節では，開発課題を開発形態と管理形態の軸に沿った特徴に基づいて大まかに分類し，それぞれのタイプにおけるテーマ設定指針について議論する．

開発形態の軸では，表 3 に示すように，新規開発型と既存資産改良型に大別することができる．

新規開発型の OJL は，新製品などの先行的開発として設定することができる．市場，顧客，組織からの新たな要求や，新たなハードウェアデバイスの出現などに伴って必要となるソフトウェア開発を試験的に行うプロジェクトがこれにあたる．指導のポイントとして，顧客要求の獲得や分析に関わる技術やソフトウェアアーキテクチャ設計やインタフェース設計の技術が挙げられる．このタイプのプロジェクトでは，例えば要求からソフトウェアシステムが解決すべき問題の特徴や制約をとらえる技術や，製品に求められる仕様 (機能・非機能) を満たすために最適なアーキテクチャを設計したり代替案の中から取捨選択したりする能力を養うようなテーマ設定が可能である．

既存資産改良型の OJL は，既存の製品など，長年にわたる保守を経て品質が劣化したソフトウェアの品質改善や再設計のプロジェクトとして設定することができる．指導のポイントとして，再利用を意識したソフトウェア資産の構築や，保守性や可搬性を高めるための設計改善などリエンジニアリングの技術が挙げられる．このタイプのプロジェクトでは，例えば，再利用を意識したプロダクトラインの構築技術，品質向上のために最適なパターン技術を選択して適用できる能力を養うようなテーマ設定が可能である．

両方の型に共通して着目すべき開発技術として検証技術が挙げられる．いずれのタイプの OJL においても製品が満たすべき品質を確保するために適切な検証の技術を選択したり組み合わせたりする能力を養うようなテーマ設定が可能である．

一方，管理形態の軸では，表 4 に示すように，企業からの参画者の関与のしかたに応じて，請負型と共同開発型に大別することができる．

請負型の OJL では，企業からの指導者は製品の顧客に近い立場，あるいは開発元請けの立場からプロジェクトに参加する．したがって，プロジェクトは主に学生からなるチームにより運営され，学生は管理と開発の両方の役割を担うことになる．このタイプのプロジェクトの指導ポイントとしては，要求やリスクの分析に基づく開発計画や，開発目標の設定や計測，管理のための技術が挙げられ，計画技術や管理技術の実適用における特徴や制約に関する知識，課題の特徴や開発技術の制約に応じた管理技術の選択に関する能力を養うようなテーマ設定が可能である．

共同開発型の OJL では，企業からの指導者がプロジェクト管理者として参画し，学生は管理者のもとで開発を担う要員としてプロジェクトに携わる．このタイプのプロジェクトの指導ポイントとしては，構成管理や技術管理のための技術が挙げられ，製品の特徴に照らして適切な構成管理方式を選択する能力や製品の開発にとって適切な技術や技術の組合せを比較検討したり，開発のプロセスの改善を提案したりする能力を養うようなテーマ設定が可能である．

表 3: 開発形態軸に基づくテーマ設定

開発形態	開発目的	着目する開発技術
新規開発型	先行開発・試作	要求工学，設計，検証
既存資産改良型	品質改善	リエンジニアリング，検証

表 4: 管理形態軸に基づくテーマ設定

管理形態	指導者の役割	着目する管理技術
請負型	顧客・元請け	開発計画，開発管理
共同開発型	プロジェクト管理	構成管理，技術管理

5　実施事例と成果

1 年半の OJL 実施期間のうち，修士 1 年後期は座学での講義や PBL と並行して行うことから，開発問題の概要や背景知識の理解などにあて，本格的な課題への取り組みは修士 2 年次の 1 年間に行う．

表 5 はこれまで実施してきた OJL テーマである．⋆印は，本教育カリキュラム 1 期生（2007 年 10 月〜2009 年 3 月）と 2 期生（2008 年 10 月〜2010 年 3 月）の双方のテーマとして継続実施されたテーマである．

OJL は 1 年半に渡って実施されるが，半期に一度ずつ[6]，拠点内の全 OJL プロジェクト関係者が一堂に会し，3 回の技術交換会（中間報告／最終成果報告）を行う．OJL プロジェクトでの指導を受けている学生がプロジェクトの進捗や成果について発表し，他のプロジェクトとの情報交換を行う．OJL は新しい教育手法であり，教員や企業担当者にとってもこの技術交換会の意義は大きい．過去に行われた技術交換会では，成果の見せ方，開発の進め方，指導の方針などについて活発な意見交換が行われてきた．

以下ではいくつかのテーマに対してその内容や実施体制，成果物に関して紹介する．

5.1　実施事例 1：トヨタ自動車株式会社との事例

トヨタ自動車株式会社の協力のもとで実施した OJL では，車載ソフトウェアのシステムバリエーションを構築・管理する際の支援手法およびツールを開発した．

[6]おおよその時期は，修士 1 年次の 3 月，修士 2 年次の 8 月と 1 月

図 4: OJL 技術交換会の様子

車載ソフトウェアは，対象となる車種のグレードやモデル，各国における仕様等の差異から，多様なシステムバリエーションを管理する必要がある．現時点でのこれらの構成管理は，版管理と，ifdef などのコンパイルオプションによる管理を併用しているが，プロジェクトや担当者によって方法が異なり，きちんとルール化されていない．近年では，複数の機能を一つの ECU に組み込むことや，製造コストの観点から動的に機能を選択することなど，新しい要求が出てきており，構成管理の効率化が望まれている．

この事例では，それらの問題を把握できるよう，対象ドメインの知識獲得を行ったあと，構成管理対象となるソースコードを開発した．構成管理対象には，Adaptive Cruise Control（ACC）を選択し，車両への搭載前にシミュレーションを行うための実装として，Simulink 上で S-Function として動作するプログラム（約 3,500 行）を実装した．その後，開発コードを対象に支援方式の検討を行い，

- コンパイルスイッチ間の関係抽出
- 関係定義に基づくコード検証

を行う Eclipse プラグインの開発を行った．

支援ツールはコンパイルスイッチ間の関係の解析結果をもとにその関係図と SMT 式を生成する．関係図として可視化することで，人間がコンパイルスイッチ間の関係について理解しやすくなる．また，コンパイルスイッチ間の親子関係や排他関係といった関係を利用し，制約上

表 5: これまでに実施した OJL テーマ

テーマ名 (協力企業)	実施大学
システム構成管理技術の研究（トヨタ自動車株式会社）⋆[5.1 節で紹介]	名古屋大学 愛知県立大学
次世代運転支援システムの情報表示系機能を検証するツールの開発（株式会社デンソー）⋆	名古屋大学
ラベルライターの外国語向けソフトウェア開発（ブラザー工業株式会社）	名古屋大学 愛知県立大学
インクジェット複合機での表示機能（ブラザー工業株式会社）	名古屋大学 愛知県立大学
次世代ＬＡＮプログラム開発（株式会社オートネットワーク技術研究所）⋆	名古屋大学
半導体製造装置のＦＡ通信ソフトウェア開発（東京エレクトロンソフトウェアテクノロジーズ株式会社）	南山大学
自動販売機制御ソフトウェアの再開発（富士電機リテイルシステムズ株式会社）⋆	南山大学
組み込みソフトウエアに向くコンポーネント仕様の研究（アイシン精機株式会社）⋆ [5.2 節で紹介]	名古屋大学
Java ソースコードの CDI(Code Inspection) ツールの開発（株式会社キャナリーリサーチ）[5.3 節で紹介]	南山大学
多軸制御装置のための NC 言語コンパイラの開発（株式会社エヌエスティー）	静岡大学
X 線画像処理システムの開発（富士フィルムイメージテック株式会社）	静岡大学
Optimized Link State Routing and Localization の開発（三菱電機株式会社）	静岡大学
大学事務システムの開発 (大学主導)	南山大学
Multicore 対応リアルタイム OS の開発（大学主導）⋆ [5.4 節で紹介]	名古屋大学

衝突しているコンパイルスイッチがないか，現在宣言しているコンパイルスイッチ群が十分であるかを検証することができる．

2 期 OJL では，上記の開発内容を発展させ，コンパイルスイッチなどを多用するバリエーション並行開発を支援する研究開発をテーマとした．バリエーション並行開発に特化した新しい構成管理手法を考案し，版管理ツールと統合開発環境 [20] や，制御ソフトウェアの固定小数点演算化手法とツール [21] の開発を行った．

この事例では，開発したツールおよび，その教育効果もさることながら，産学連携で教育や研究を実施するうえで利用できる車載ソフトウェアのソースコードは非常に稀である点で注目を集めている．当該 OJL で作成した上記 ACC のソースコードは現時点で複数の大学等から借用の依頼が来ている．

5.2 実施事例 2：アイシン精機株式会社との事例

アイシン精機株式会社の協力のもとで実施した OJL では，車載ソフトウェアに代表される組込みソフトウェアに対するコンポーネント化を促進するための支援方法およびツールの開発を行った．

車載ソフトウェアに代表される組込みソフトウェアでは，厳しい時間制約，リソース制約，デバッグの難しさなどから保守性や再利用性が低い記述を強いられているという問題がある．例えば大域変数の使用は，組込みソフトウェア以外のソフトウェアにおいてはモジュールの独立性を下げ，ソフトウェアの品質を下げる原因として避けられるのが一般的となっている．しかしながら時間性能やメモリ使用量の予測の容易さなどから組込みソフトウェアの開発の現場では使用されることも少なくない．

この事例では，まず，実際の製品ソースコード分析し，時間及び空間効率を犠牲にすることなく再利用性を確保する方法を検討した．上述のソフトウェアは，大域変数によって構成された黒板に対して，各処理モジュールが知識源としてアクセスし処理する黒板アーキテクチャであることに着目し，

- 各知識源に相当するモジュールの仕様外の入出力
- 黒板を介さない知識源同士の値の受け渡し

を禁止するための支援ツールを作成した．

支援ツールの開発では，まず黒板アーキテクチャに基づく黒板・知識源の仕様定義方法を定め，次にこの仕様に基づいて，

- コードテンプレートを生成する支援ツール BB-Generator
- ソースコードが仕様を満たすように黒板（大域変数）へアクセスしているかを確認するツール CX-Checker [22, 23]

の二つを並行開発した．

CX-Checkerは，黒板アーキテクチャ仕様以外にも，コーディング規約をチェックできるEclipseプラグイン開発された．MISRA-CやGNUコーディングスタンダードといった代表的なものだけでなく，社内独自のコーディング規約に対して柔軟に対応できるよう，チェックするルールに対する高い柔軟性を実現している．ソースコードの抽象構文木をXMLによってマークアップした中間形式を採用し，XML文書に対するXPath記述によるルール記述や，構文要素の複雑な関係を利用するためのDOM及び独自のAPIを利用したJava言語によるルール記述をサポートする．

開発は，企業側4名，学生2〜5名（1期生が2名，後半より2期生3名も参加），教員3名によって，3フェーズのインクリメンタル開発によって実施された．毎週の全体ミーティングおよび週報による報告に加え，メールベースでの議論・指導がおこわなれた．また，開発されたツールを実際にアイシン・コムクルーズ株式会社において使用して頂く導入試験や，黒板アーキテクチャに基づく制約を満たすようにリファクタリングを行ったソースコードを，実車に組み込み走行試験を行うなど，製品や業務に直結した開発を行った．

5.3 実施事例3:株式会社キャナリーリサーチとの事例

株式会社キャナリーリサーチの協力のもとで実施したOJLでは，Javaソースコードの静的解析（CDI: Code Inspection）ツールの開発を行った．近年，多くの現場でCOBOLなどに代表されるプログラミング言語で記述されていたレガシーソフトウェアがJavaへ移植されつつある．これらのJavaソースコードには，オブジェクト指向の概念に基づく再設計に基づいて記述されているとは限らないものも含まれており，セキュリティや保守性などの品質低下が懸念されている．一方，レガシーソフトウェアの量は膨大であり，これらを効果的に検査するツールが求められている．

このような課題を受け，企業からのプロジェクトマネージャ1名と教員3名のもと，学生7名 (1期生3名，2期生2名，3期生2名) がこれまで2年半にわたり開発を行ってきた．

開発の初期には，既存のオープンソースソフトウェアや商用ツールを調査し，それらが提供する検査機能の整理を行った．その上で，上記のようなJavaソースコードに対する現場からの検査要求を満たすようなツールをスクラッチから開発し，機能拡張を行ってきた．

本OJLでは，プロダクトライン型ソフトウェア開発 [24] を実施したことが大きな特徴である．静的解析ツールでは，抽象構文木や制御フローグラフ，データフローグラフなどのデータ構造を基礎に，個々の検査項目はこれらの構造をトラバースしながらソースコード上の問題点や欠陥を指摘するアルゴリズムとして実現される．本プロダクトラインのコア資産としてこれらのデータ構造とそれらをトラバースするための処理のアーキテクチャを定義した．構造と処理のアーキテクチャ設計では，デザインパターン [25] を参考に，コア資産としての再利用性だけでなく，空間効率や保守性，拡張性を高めることを目的とした．また，アーキテクチャに検査項目を追加する開発手順や，検査項目に対するテストケースの生成からテスト実施に至る検証手順についても典型的なプロセスの雛型を作成し，コア資産の一部として定義した．

さらにこれらのコア資産に基づき，配列の定義外参照やリソースリークといった，Javaソースコード上の問題点を検査するアルゴリズムを合計で31項目 (それぞれの項目には複数の検査オプションが存在する) 実装した．本ツールはEclipseプラグインとして実装されており，Eclipseプロジェクトにより提供されている開発ツールとの連携も可能である．

OJLの成果として基本設計と実装を行ったこのCDIツールは，連携企業側でさらなる検証とブラッシュアップを経て，2009年12月に製品化[7]された．

現在，本OJLプロジェクトでは検査項目の拡張や性能向上に関する開発を進めているが，開

[7] JCI (Java Code Inspector): http://www.canaly.co.jp/jci/

発効率が向上していることや，新しいプロジェクトメンバへの技術伝承が容易に行えていることからも，これまでに構築したプロダクトラインが有効に機能していることが実感されている．

5.4 実施事例 4：学内組織主導での事例

この事例では，NPO 法人 TOPPERS プロジェクトの協力を得て名古屋大学大学院情報科学研究科附属組込みシステム研究センターが主導して，汎用性と拡張性を備えたトレースログ可視化ツール TLV を開発した [26].

マルチプロセッサ環境で有効なデバッグ手法として，プログラム実行履歴であるトレースログを解析する手法が有効であり，これまでに多くのトレースログ可視化ツールが開発されている．しかしながら，これら既存のツールが扱うトレースログは，形式が標準化されておらず，環境（OS やデバッグハードウェア）毎に異なるので，可視化対象が限定されおり，汎用性に乏しい．さらに，可視化表示項目がツールごとに固定であり，追加や変更が容易ではないなど，拡張性に乏しいといった問題がある．

TLV（TraceLogVisualizer）は，トレースログを一般化した標準形式と各種フォーマットへの変換ルールを定め，その汎用性を確保している．また，トレースログの可視化表現を指示する仕組みを抽象化し，可視化ルールとして形式化することで，高い拡張性を実現している．当該 OJL で作成したこのツールは，その完成度と有用性から Toppers プロジェクトにおいてリリースされる [27] など実用性の面でも高い評価を受けている．

TLV の開発では，企業出身者 2 名と教員 1 名がプロジェクトマネージャを務め，学生 3 名と企業出身者 2 名が開発実務を担当した．進捗の報告は，週に 1 度のミーティングと週報の提出により行われた．開発は初期段階でプロトタイプを作成し，そのプロセスと成果物から，GUI の評価や要求の再抽出，アプリケーションドメイン分析を通じた設計と実現方法を検討し，その後，ユースケース駆動によるアジャイル開発によって開発が行われた．ツールは C#で実装され，総行数は約 18500 行であった．

6 考察

教育手法の効果を定量的に議論することは非常に困難であり，現時点では OJL という教育方式の客観的かつ定量的な効果を示すことはできない．しかしながら，参加している教員や実際に指導を受けた学生の意見から，企業の実際のプロジェクトマネージャの管理のもと，実務者と一緒に開発作業を行うこと，特に品質や納期，工数に関する意識の差に関しては高い学習効果があることが見受けられた．また，企業側担当者からも,「インターンシップでは，企業側ができる範囲で作業をしてもらうことになる．それに対し OJL では，教員がともに参加し教員からの知見を反映しながら実施するので，企業としても大変やりがいがある」「学生が対象ということで，OJL テーマのレベルを心配していたが，基礎知識があり，学習意欲が高いので順調にプロジェクトを実施することができている」という意見があるなど，OJL の一定の効果が確認できる．

また，学生の学習結果の評価方法に関しても，OJL に特化した評価方法が必要である．現段階では半期毎の技術交換会において，プロジェクトの開発成果を報告するだけではなく，そこから何を学んだかを報告させ評価するようにしている．

重要な観点は学生が実問題を対象とした開発を通して何を学んだかであり，普段の指導においては「なにを」ではなく「どう」開発したかに重点をおく．したがって，開発対象の新規性や有用性よりも，期間内に開発問題をどう解いたか，座学で学んだ技術をどのように適用したのかの説明を求める．また，**Application** レベルの知識を獲得できているかを確認するために，「なぜ」その方式や技術を選択し実施したのかの説明も求め，対象問題を抽象化し得失を検討できているかを確認している．

定例ミーティングや半期毎の技術交換会において，どのように技術解を求めたかを振り返らせることは，学生自身が体験した内容を **Application** レベルの知識として洗練するのに非常に効果的であると考えている．

7 おわりに

本稿では，PBL に代表される教育用の仮想プロジェクトを題材とする教育と，インターンシップや OJT のように実際の業務を体験することによる実務訓練を融合した，新しい教育手法 OJL を紹介した．既存のプロジェクト参加型の教育手法との差異を明確にし，我々が 3 年間実施してきた事例をもとにその有用性を議論した．

OJL のテーマ設定や知財権の取扱い，OJL プロジェクトにおける学生の指導方法などについては，先導的 IT スペシャリスト育成推進プログラムの拠点として活動した成果として印刷製本 [28] し，同様の試みを行う組織等の参考に供する予定である．

今後は，履修生に対する追跡調査などを行うことで，OJL による教育効果を明確にするとともに，有効な評価基準や方法を検討していく必要がある．また，ソフトウェア分野に限らず他の分野へ適用範囲を拡大すること，特定の IT 職種の教育に特化した教育手法として洗練することも検討課題である．

謝辞

OJL による教育の実践は，文部科学省研究拠点整備等補助金（先導的 IT スペシャリスト育成推進プログラム）による助成のもとで行われた．OJL は連携企業関係各位の教育に対する熱意により支えられている．ここに記して謝意を表す．

参考文献

[1] 小林 隆志，沢田 篤史，山本 晋一郎，野呂 昌満，阿草 清滋：“On the Job Learning: 産学連携による新しいソフトウェア工学教育手法”，電子情報通信学会技術研究報告，ソフトウェアサイエンス研究会，Vol. 109, No. 170，pp. 95–100，2009.

[2] 山下 徹：高度 IT 人材育成への提言～国際競争力の復権にむけて，NHK 出版，2007.

[3] B. Bloom, M. Englehart, E. Furst, W. Hill and D. Krathwohl: *Taxonomy of Educational Objectives: The Classification of Educational Goals. Handbook I: Cognitive Domain*, Longmans, 1956.

[4] A. Abran, J. W. Moore, P. Bourque and R. Depuis (eds.), *Guide to the Software Engineering Education Body of Knowledge (SWEBOK)*, 2004 Version, IEEE Computer Society, 2004.

[5] The Joint Task Force on Computing Curricula: *Software Engineering 2004 — Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering*, A Volume of the Computing Curricula Series, IEEE Computer Society and Association for Computing Machinery, 2004.

[6] 阿草 清滋，西 康晴，沢田 篤史，鷲崎 弘宜：“特集　情報専門科目カリキュラム標準 J07：ソフトウェアエンジニアリング領域（J07-SE），” 情報処理，Vol. 49，No. 7, pp. 743–749, 2008.

[7] 酒居 敬一，荻原 剛志：“情報系学部教育におけるアセンブリ言語を用いた組込みシステム開発実験の試み，” 組込みシステムシンポジウム 2007 論文集，情報処理学会シンポジウムシリーズ，Vol. 2007，No. 8, pp. 6–14，2007.

[8] 小倉 信彦，渡辺 晴美：“ロボットコンテストを利用した組込み教育の実践，” 情報処理学会論文誌，Vol. 49，No. 10，pp. 3531–3540，2008.

[9] 文部科学省；先導的 IT スペシャリスト育成推進プログラム 8 拠点のプロジェクト概要 — 世界最高水準の高度 IT 人材育成を目指して —，2008.

[10] 坂本 憲昭，深瀬 光聡，峯 恒憲，日下部 茂，中西 恒夫，大森 洋一，北須賀 輝明，ウッディン モハマッド メスバ，荒木 啓二郎，福田 晃，安浦 寛人：“大規模な産学

連携による高度 ICT 人材育成に向けての取り組み”，情報処理学会論文誌，Vol. 49, No. 8，pp. 2830–2842，2008.

[11] 坂本 憲昭，峯 恒憲，日下部 茂，深瀬 光聡，荒木 啓二郎，福田 晃：“大規模な産学連携による高度 ICT 人材教育におけるインターンシップの役割とその効果”，情報処理学会論文誌，Vol. 49，No. 10，pp. 3388–3398，2008.

[12] 井上 明，金田 重郎：“実システム開発を通じた社会連携型 PBL の提案と評価，” 情報処理学会論文誌，Vol. 49，No. 2，pp. 930–943，2008.

[13] 松澤 芳昭，大岩 元：“産学協同の Project-based Learning によるソフトウェア技術者教育の試みと成果，” 情報処理学会論文誌，Vol. 48，No. 8，pp. 2767–2780，2007.

[14] T. Mahmood, K. Lister, S. Karunesekera and E. Kazmierczak: “Industry based Learning in Software Engineering Successes and Challenges”, *Proc. Workshop on Software Engineering Education,* pp. 16–21, 2007.

[15] 寺沢 弘忠: “管理者のための OJT の手引” 日本経済新聞社, 1999.

[16] Faculty of Engineering, University of Melbourne: “433-340 Software Engineering Project” Subject Information `http://unimelb.edu.au/HB/subjects/433-340.html`

[17] MDD ロボットチャレンジ編集委員会編，MDD ロボットチャレンジ 2004：産学連携による組込みソフトウェア開発の実践，情報処理学会，2005.

[18] MDD ロボットチャレンジ編集委員会編，MDD ロボットチャレンジ 2005：産学連携によるモデルベース組込み開発の実践，情報処理学会，2006.

[19] 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：“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，Vol. 50, No. 11，pp. 2677–2689，2009.

[20] 横山祐司, 日高隆博, 山本晋一郎, 小林隆志, 手嶋茂晴, 阿草清滋: “バリエーション並行開発のための版管理ツールと統合開発環境”, 情報処理学会研究報告 2010-SE-167-6, 2010.

[21] 関文貴, 日高隆博, 山本晋一郎, 小林隆志, 手嶋茂晴, 阿草清滋: “制御ソフトウェアの固定小数点演算化ツールの設計と実装”, 情報処理学会研究報告 2010-SE-167-27, 2010.

[22] 大須賀 俊憲，小林 隆志，間瀬 順一，渥美 紀寿，山本 晋一郎，鈴村 延保，阿草清滋：“CX-Checker: C 言語プログラムのためのカスタマイズ可能なコーディングチェッカ”，ソフトウェア工学最前線 2009（情報処理学会ソフトウェアエンジニアリングシンポジウム 2009 論文集），近代科学社，pp.119–126，2009.

[23] CX-Checker プロジェクト: `http://www.sapid.org/cxc/` 2010.

[24] L. M. Northlop: “SEI’s Software Product Line Tenets”, *IEEE Software,* Vol. 19, No. 4, pp. 32–40, 2002.

[25] E. Gamma, R. Helm, R. Johnson and J. Vlissides: *Design Patterns: Elements of Reusable Object-Oriented Software,* Addison-Wesley, 2005.

[26] 後藤 隼弐, 本田 晋也, 長尾 卓哉, 高田 広章: “トレースログ可視化ツールの開発”, 電子情報通信学会技術研究報告，コンピュータシステム研究会，Vol. 108，No. 463, pp. 73–78，2009.

[27] NPO 法人 TOPPERS プロジェクト：“トレースログ可視化ツール TraceLogVisualizer（TLV）の一般公開について”，`http://www.toppers.jp/press/release-0911-1.pdf`，2009.

[28] 沢田 篤史，小林 隆志，山本 晋一郎，野呂 昌満，阿草 清滋（編）: OJL ハンドブック，文部科学省先導的 IT スペシャリスト育成推進プログラム名古屋大学拠点「Project OCEAN：OJL による最先端技術適応能力を持つ IT 人材育成拠点の形成」事務局，2010．(発行予定)

著者略歴

(1) 小林 隆志 2004 年東京工業大学理工学研究科計算工学専攻博士課程修了．同大学学術国際情報センター助手，名古屋大学大学院情報科学研究科附属組込みシステム研究センター特任准教授を経て，2009 年より同研究科情報システム学専攻准教授．工学博士．ソフトウェア設計方法論，ソフトウェア再利用技術，複合メディアコンテンツの管理・検索，Web サービス連携などの研究に従事．

(2) 沢田 篤史 1995 年京都大学大学院工学研究科情報工学専攻博士後期課程研究指導認定退学，奈良先端科学技術大学院大学情報科学研究科助手，京都大学大学院工学研究科助手，同大学大型計算機センター助教授，同大学学術情報メディアセンター助教授，南山大学数理情報学部教授を経て，2009 年より同大学情報理工学部教授（先導的 IT スペシャリスト育成推進プログラム担当）．博士（工学）．ソフトウェア工学，組込みシステム工学，ネットワーク情報家電などの研究に従事．

(3) 山本 晋一郎 1987 年名古屋大学工学部卒業後．同大学大学院に進学，同大学助手，講師，愛知県立大学情報科学部助教授を経て，2007 年より同大学准教授．プログラミング言語処理系，ソフトウェアの形式的開発手法，ソフトウェア開発環境に関する研究に従事．近年は，細粒度のソフトウェア・リポジトリに基づいた CASE ツール・プラットフォームに関する研究を進めている．

(4) 野呂 昌満 1986 年慶應義塾大学大学院工学研究科管理工学専攻博士後期課程単位取得退学．南山大学情報理工学部ソフトウェア工学科教授．工学博士．ソフトウェアアーキテクチャ，アスペクト指向計算，プログラミング言語の意味論および処理系などの研究に従事．

(5) 阿草 清滋 1970 年京都大学工学部電気第二学科卒業後，同大学大学院工学研究科電気工学第二専攻修士課程，同博士課程，1974 年京都大学情報工学科助手，講師，助教授を経て 1989 年より名古屋大学教授．工学博士．ソフトウェア開発方法論，知的開発環境，ソフトウェアデータベース，仕様化技法，再利用技法，マンマシンインタフェースなどの研究に従事．