様式 C-19

科学研究費補助金研究成果報告書

平成　22　年　6　月　1　日現在

研究種目：基盤研究（C）
研究期間：　2006 ～ 2009
課題番号：18500729
研究課題名（和文）　e-Learning を活用したソフトウェア工学教育の研究

研究課題名（英文）　A Software Engineering Education System with e-Learning Techniques

研究代表者
松浦　佐江子（MATSUURA SAEKO）
芝浦工業大学・システム理工学部・教授
研究者番号：10348906

研究成果の概要（和文）：
本研究では、ソフトウェア開発の全工程において、プロダクトおよびプロセスの妥当性検証方法を取り入れたオブジェクト指向開発技術の学習方法およびその評価方法を研究し、これらの方法に基づいた PBL （Project Based Learning）によるソフトウェア開発実習を最終ターゲットとしたカリキュラムならびに授業設計を Java および UML といったオブジェクト指向開発技術に基づき実施した。

研究成果の概要（英文）：
We have studied an instructional design for the related classes in such object oriented software development technologies as Java and UML. The goal of our software engineering education for undergraduate students is that they can develop suitable software products through all the software development process. To achieve this goal, the prerequisite classes need to be designed so that the students can learn how to define suitable products and verify them.

交付決定額

（金額単位：円）

	直接経費	間接経費	合　計
2006 年度	1,100,000	0	1,100,000
2007 年度	800.000	240,000	1,040,000
2008 年度	800,000	240,000	1,040,000
2009 年度	800,000	240,000	1,040,000
年度			
総　計	3,500,000	720,000	4,220,000

研究分野：ソフトウェア
科研費の分科・細目：科学教育・教育工学・教育工学
キーワード：ソフトウェア工学教育・オブジェクト指向開発技術・インストラクショナルデザイン・要求分析・モデル駆動開発・ソフトウェア開発技術者教育・Unified Modeling Language・Project Based Learning

１．研究開始当初の背景

ソフトウェア工学はソフトウェアの大規模・複雑化に伴い、様々なソフトウェア開発技術を生み出してきた。近年、ソフトウェアの需要は増え続け、ユビキタス・コンピュータや組み込みシステムにおいても、情報処理技術、とりわけソフトウェア工学が培ってきたソフトウェア開発技術の導入が必要となっている。それに伴い、ソフトウェア開発技術者の育成が緊急の課題となっており、大学

が課せられた課題でもある。

２．研究の目的

本研究では、ソフトウェア開発の全工程において、プロダクトおよびプロセスの妥当性検証方法を取り入れたオブジェクト指向開発技術の学習方法およびその評価方法を研究し、これらの方法に基づいた PBL (Project Based Learning) によるソフトウェア開発実習を最終ターゲットとしたカリキュラムならびに授業設計を行うことを目的とする。

３．研究の方法

(1)学部教育におけるソフトウェア開発技術の習得の目標を下記の４項目により定め、プログラミング教育の各科目における目標に設定し、カリキュラムの改善ならびに支援環境の開発を行う。

①プログラムを構文に従って正しく定義でき、コンパイル・実行することができる。
②機能的要求を満たすプログラムを適切なアルゴリズムで定義する。要求を満たすことをテストし、テスト結果により誤りを発見することができる。
③データ構造とアルゴリズムを工夫し、効率の良いプログラムを定義できる。
④保守性の高いプログラムを定義できる。すなわち、対象をモデル化し、モジュール構造を設計し、再利用性の高いモジュールや仕様変更時に少ない労力で変更が可能なモジュールの設計を行うことができ、モデルとプログラム間の整合性を検証できる。

特に、基本的なプログラミング能力は②の機能的要求を満たすプログラムを定義できることであり、要求を適切に認識する、認識した要求をプログラムとして定義できる、定義したプログラムが要求を満たしていることを確認するテストを定義できる、テストを実行し誤りを発見できることの４つの副目標を達成することを誤りがなくなるまで繰り返すことができることであると考える。

(2)これらの目標を踏まえ、以下の項目を実施してきた。

① 演習に審査方式によるプログラムの評価方法を取り入れた[雑誌論文⑨]。
② 単体テストの方法を学習するツールを開発し、適用実験を行った[学会発表⑧]。
③ オブジェクト指向開発における統一モデリング言語 UML を用い、WebUI プロトタイプを自動生成ツールによる要求分析手法を提案し、適用実験を経て、大学院生を対象とした演習授業を設計、実施した[雑誌論文①②③④⑤⑥⑦⑧、学会発表①③⑤⑥]。
④ 学部３年次の PBL によるソフトウェア開発実習の授業設計をまとめ[雑誌論文⑩]、上記の要求分析モデリング手法の研究成果の基づき、授業改善を行った。
⑤ MDA(Model Driven Architecture)の考え方に基づき、自律走行車のハードウェアおよび実装技術 (LEGO の車体および Java) に非依存なモデルをトップダウンに設計し、段階的な走行目標に対する開発方法を提案した。組込みシステム技術協会が主催する ET ロボコンに参加して、地区大会の競技部門で１位になるという成果を得られた。さらに、迷路探索ロボットを事例として、実行可能 UML ツールを用いたモデリング実験を行っており、組込みシステム開発におけるトップダウンなモデリング方法を検討している。[学会発表②]
⑥ ソフトウェア工学的観点からのオブジェクト指向開発技術教育の充実のため、カリキュラムの改善を行った。具体的には、これまで半期で教育してきた内容を Java 言語による基礎プログラミング教育とオブジェクト指向開発技術の特長である再利用性・保守性についての教育を段階的に実施できるように科目を新設し、通年で教育を行った。

４．研究成果

計画当初のカリキュラムの見直しを行い、2 年次前期の科目を増やすことにより、より段階的な、ソフトウェア工学的観点の授業設計が実施できたと考える。開発した支援ツールを用いた適用実験を踏まえて、演習を実施しており、主観的ではあるが、学生のソフトウェア開発能力の向上も見られている。基礎的なプログラミング能力の育成には、多くのプログラミングを行わせ、解答例を提示し、テストプログラムによるテストを実施することが望ましいが、限られた教員で問題・解答例のプログラム・テストプログラムを開発し続けることはできない。演習の評価方式や採点支援等による支援を模索してきたが、学習者の意欲の向上や、教員の作業負荷の軽減に対する改善にはつながることはわかったものの、根本的な解決にはならない。そこで、つぎの２つの方針で、研究を進めており、これらの過程で得られた授業設計について述べる。第一に、学習者自身が、自己のプログラムをソフトウェアメトリクス等による客観的な指標や、構造および振舞いの観点から評価するツールを提供し、プログラムの改善点を学習することを支援する必要があることから、このための評価方法 (機能的要求を満たすテストケースの定義・品質評価のポイントと実践等) を授業に取り入れる。第二に、プログラム作成に至るまでの、対象システムのモデリング能力の育成支援を行う。これは、要求分析段階における、曖昧な仕様の定義と不十分な設計が、プログラムの複雑化の要因の一つであると考えられるからである。上流工程において、プロトタイプ自動生成ツール

による抽象度の高い実現による妥当性検証を支援することで、要求からソースコードまでのトレーサビリティの高い開発方法を確立することを目指す。

(1) オブジェクト指向開発技術教育の授業設計

本研究開始時には、オブジェクト指向開発技術の導入教育として、2 年後期の講義 1 コマ、演習 1 コマで実施していた Java 言語を用いたオブジェクト指向プログラミングの授業を、2 年前期に演習 5 回を含む講義を追加した 1 年通年での教育に変更した。この狙いは、C 言語を用いてプログラムの基礎を学習している学生に、「C と Java ではどこが同じで、どこが違うのか」の観点から、Java の仕組みを教育し、プログラミングそのものを十分に理解させることと、後期の演習においては、クラス設計とコードレビューに重点を置くことである。

	2年前期:オブジェクト指向プログラミングI
1	プログラミングとは？ なぜ、オブジェクト指向なのか？
2	**演習** Javaのコンパイルとファイル管理
3	Javaプログラムの基礎・演習の復習
4	オブジェクトの定義 – クラスとインスタンス – フィールド・メソッド・コンストラスタ
5	**演習**
6	オブジェクトの関係　継承 事例によるオブジェクト定義の復習との拡張
7	**演習**
8	演習の復習 オブジェクトの関係 継承・ポリモルフィズム
9	オブジェクトの関係 関連・同等性
10	**演習**
11	オブジェクトの可視性 修飾子・パッケージ
12	例外
13	**演習**
14	ファイル操作と入出力

図 1　2 年次前期講義内容

図 1 は、2 年次前期の講義内容である。オブジェクト指向の特徴的な考え方である継承の概念は教えるが、初学者には理解しにくい抽象クラスやインタフェース等、再利用性に関わる項目については後期に回すこととした。演習を講義の間に挟むことで、実際にプログラムを動かしながら、それまでに経験してきたC言語と同じことと違うことを意識できるようにする。また、この時期は言語に関わらず、ファイル操作や、ファイル管理についての実践経験が少なく、身についていない学生が多い。これらはグループである程度の規模のプログラムを開発する場合には、必須の知識であることから、演習に取り入れ、実践させることとした。

図 2 は、2 年次後期の講義内容、図 3 は同時期に行う演習の内容である。

オブジェクト指向プログラミングの難しさは、適切なオブジェクトを抽出することにある。その結果、初学者のプログラムではオブジェクト指向の特徴であるオブジェクトの責務の分割、カプセル化、サブクラス化が行われていないことが多い。

リファクタリングはソフトウェアの外部的振舞いを保ったまま内部の構造を改善し、ソフトウェアの再利用性・保守性を高める技術であり、抽象化を実現する 1 つの方法になる。外部的振舞いは同じでも異なる定義が多様にできることと、その定義の利点と欠点を知ることは、プログラムの品質を知る上で重要である。そこで、一通り、Java のプログラミングに慣れた段階で、プログラムの表現の多様性を、リファクタリングの観点から教育する。これにより、抽象クラスやインタフェースの意義を説明することができる。また、こうした抽象化の観点や、新たな役割の付与の観点から、デザインパターンについても学習することにより、設計の利点、欠点を考慮できるようにする。GUI については、システムの多様な入力とし出力の 1 つの表現としての位置づけを明確にするように教育する。

	2年後期:オブジェクト指向プログラミングI
1	オブジェクト指向プログラミングの復習
2	オブジェクト指向におけるクラス設計
3	リファクタリング:マジックナンバーの置き換え
4	リファクタリング:制御フラグの削除
5	リファクタリング:メソッドの抽出・クラスの抽出
6	デザインパターン:イテレータ
7	デザインパターン:テンプレート・ファクトリ
8	デザインパターン:コンポジット
9	メタプログラミング
10	GUI:基本概念
11	GUI:出力の表現
12	GUI:入力の表現
13	マルチスレッド:基本概念
14	マルチスレッド:事例

図 2　2 年次後期講義内容

2 年次前期の講義の演習では、基本的なクラス定義とプログラムの作成手順を練習することと、エラーメッセージに着目することを念頭に問題を作成する。図 3 に示す 2 年次後期のプログラミング演習では、段階的な問題設計とチェックリストによるプログラムの自己評価ならびに 3 人程度のグループによるプログラムのレビューを実施する。どちら

の授業でも、プログラミング学習の早い段階から、UML モデルに慣れることを目的として、ソースコードをリバースしてクラス図により、プログラムの外観を見る方法や、アクティビティ図を用いて、問題を基本フローから段階的に分析する方法を開発のヒントとして提供する等の方法を授業に取り入れている。レビューのチェックリストには、課題の機能的要求の満足度に関する「課題項目」と、「品質項目」がある。「品質項目」は、クラス・フィールド・メソッドの命名の適切さ、変数のスコープの意識、メソッドの長さやクラスの大きさの適切さ、マジックナンバーや制御フラグ等による可読性の低下はないか等の評価を自分およびグループの他学生のプログラムに対して実施する。

	2年後期:プログラミング演習II
1	授業の進め方の説明とプログラミング理解度調査
2	テーマ1　整列アルゴリズムの応用　クラス設計
3	レビュー
4	発表
5	テーマ2　探索アルゴリズムの応用　クラス設計
6	レビュー
7	発表
8	テーマ3　四則演算計算する仮想計算機　クラス設計
9	レビュー
10	発表
11	テーマ4　アルゴリズムの可視化　GUI　クラス設計
12	レビュー
13	プログラムの改善
14	発表

図３　２年次後期演習内容

	3年前期:情報実験I
1	● 実験の目的の説明 ● プログラムの仕様の説明 ● プログラムのテストに関する講義 ● テストケースを考える。 ● 基本フローのプログラムを実装する
2	●2人1組になり、テストケースを交換し、自分のプログラムをテストする。 •テスト結果を報告・テストケースについて再考 ●機能拡張のためのリファクタリングについて •自分のプログラムの評価 •入出力インタフェースの改善 ●拡張要求： •色付長方形　＋　残りの操作と例外フロー •テストケースの追加
3	●2人1組になり、自分のテストケースで相手のプログラムをテストする。 •テスト結果を報告する・テストケースについて再考 ●ＧＵＩの講義（1） ●仕様変更要求：出力の仕様
4	●2人1組になり、自分のテストケースで、相手のプログラムをテストする。 •テスト結果を報告する・テストケースについて再考 ●ＧＵＩの講義（2） ●仕様変更要求：入力の仕様

図４　３年次前期実験内容

３年時前期の実習では、ソフトウェアの開発工程（要求分析からテストまで）の中で、ソフトウェア開発には何が求められているかを教育する。高品質なソフトウェアを効率よく開発することが、ソフトウェア開発の目標であることから、仕様変更に対して、少ない工数で、品質を低下させない開発とは何かを、CUI から GUI への仕様変更とテストケースの定義により、4 段階の開発工程を経ることで学習する。ここでは、2 人で 1 組となり、互いのプログラムをテストすることにより、実際にテストがプログラムの品質に影響を与えることを体感させる。

(2) モデリング教育

UML を用いたモデリング方法については、3 年次前期のソフトウェア設計論の講義で、複数の具体的な事例を通して、UML 図を用いた分析方法を教育している。

開発者がシステムの振舞いと入出力データを UML のアクティビティ図・クラス図・オブジェクト図を用いて要求分析モデルとして定義し(図 5、6 参照)、そのモデルから Web アプリケーション・プロトタイプを自動生成するツールを開発した(図 7 参照)。プロトタイプを用いて、顧客が要求の妥当性を確認すると同時に、開発者自身も最終製品を模したプロトタイプで確認しながら、モデルを洗練することができる。本ツールでは複数のモデルに対して段階的なプロトタイプ生成が可能であり、ユースケース単位のテストケース開発を支援することができる。

本要求分析手法では、基本的にはユースケース分析と同様に、要求の最も核となる機能要求を中心に扱う。ユーザが直接操作するシステムの UI(User Interface)に機能要求が顕在する部分（これを、ユーザとシステムのインタラクションと呼ぶ。図 5 の中央のパーティションがこれに相当する）を分析対象とする。

業務プロセスをシステムのサービスとして定義する場合、つぎの４つの観点から分析を行う。

① 業務規則に基づくサービス成立時の、システムの基本的な処理の手順は何か。
② サービスが成立するために必要な入力データとその条件および条件が不成立な場合のシステムの応答は何か（不成立な場合に、どの処理に移行するするのか、どのようなメッセージを出力すればよいか)。
③ サービスが提供する出力データの項目および値の形式と制約は何か。図 6 に示すように、アクティビティ図のオブジェクトノードに定義されたデータをクラス図を用いて定義する。その具体値はオブジェクト図を用いて定義する。

④　サービスが成立しない場合の条件とその時のシステムの応答は何か（不成立な場合に、どの処理に移行するのか、どのようなメッセージを出力すればよいか）。

図５の赤い丸の部分が②と④のサービスが成立しない場合の条件の分析に相当する。

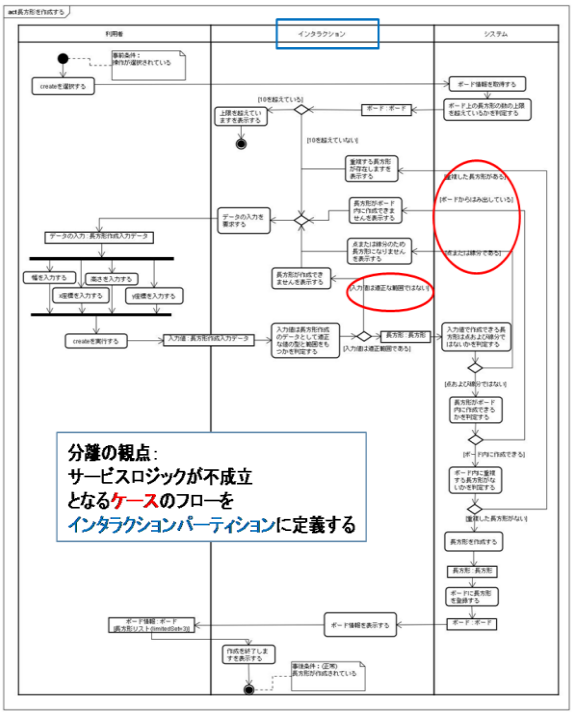

図５　要求分析モデル

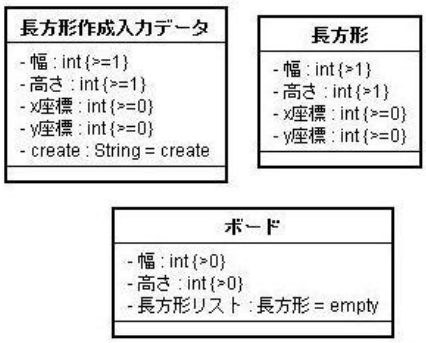

図６　データの定義

上記の分析の観点に基づき、Web アプリケーション・プロトタイプ自動生成ツールを用いて、以下の実験を行った。

- 顧客と開発者を想定した学部４年生による開発実験では、従来のユースケース分析で行われるユースケース記述と手動のプロトタイプ定義による開発との比較を行い、本手法が顧客の要求を定義可能であり、顧客が確認した項目を要求仕様として正確に定義できることが確認できた。工数に関してもユースケース記述と手動で定義することに比べ、優位である。
- 企業からの受託研究により、小規模ではあるが、その製品開発工程と本手法による開発の比較を行い、現場の開発における問題点の改善に貢献することが確認できた。現状ではドキュメントとソースコードに乖離があり、再利用の基盤として利用することは難しいが、本手法により、そのトレーサビリティが向上できることがわかった。
- 大学院授業において、顧客と開発者役を設けて、要求分析モデリング演習を行った。短期の講習で、最終的なシステムのイメージを通して要求分析モデルの妥当性を学生自身が確認しながら開発することを実施できた。しかし、要求分析自体の意義を十分に理解していない学生もおり、顧客としての要求を明確に出せていないケースが見られた。

図７　自動生成されたプロトタイプ

(3) 今後の課題

学部３年次後期の PBL において、試験的に(2)で述べたプロトタイプ自動生成ツールを用いたところ、自分たちが分析している内容を最終製品の形態で確認でき、非常に分かりやすいとの感想を得ることができた。しかし、分析の観点を明確に理解していない初学者にとって、要求分析モデルを十分に定義するには、もう少し練習が必要である。ここ数年は、前提となる授においてもアクティビティ図を用いていることから、要求分析段階でアクティビティ図を用いることが徐々にできるようになっている。今後は、プロトタイプ自動生成ツールの公開により、利用者の拡大を図りたいと考えている。

５．主な発表論文等

（研究代表者、研究分担者及び連携研究者には下線）

〔雑誌論文〕（計 10 件）

① Shinpei Ogata, Saeko Matsuura, A Method of Automatic Integration Test Case Generation from UML-based Scenario, WSEAS TRANSACTIONS on INFORMATION SCIENCE and APPLICATIONS, Issue 4, Volume 7, pp.598-607, April 2010. 査読有

② Shinpei Ogata, Saeko Matsuura, Evaluation of a Use-Case-Driven Requirements Analysis Tool Employing Web UI Prototype Generation, WSEAS TRANSACTIONS on INFORMATION SCIENCE and APPLICATIONS, Issue 2, Volume 7, pp.273-282,February 2010. 査読有

③ 小形, 松浦：UML 要求分析モデルからの段階的な Web UI プロトタイプ自動生成手法, 日本ソフトウェア科学会, コンピュータソフトウェア, Vol.27, No.2, pp.14-32, 2010. 査読有

④ 小形, 松浦：Web UI プロトタイプ自動生成ツールを用いたユースケース駆動要求分析の評価実験, 情報処理学会 FIT2009, RB-001,pp73-80.2009. 査読有

⑤ 小形, 松浦：妥当性確認可能なモデルベース要求仕様からの統合テスト仕様自動生成, 情報処理学会ソフトウェアエンジニアリング最前線 2009, pp.127-132,2009. 査読有

⑥ 小形, 松浦：UML 要求分析モデルからの段階的な Web UI プロトタイプ自動生成, 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論文集,pp.79-86,2008. 査読有

⑦ S.Ogata and S.Matsuura, A UML-based Requirements Analysis with Automatic Prototype System Generation, Communications of SIWN, Vol.3, pp. 166-172, June 2008. 査読有

⑧ 松浦：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, Vol.48, No.8, pp.2578-2595, 2007.査読有

⑨ 松浦：プログラミングレポート採点支援ツールと課題設計による評価方法の改善, 論文誌ＩＴ活用教育方法研究,（社）私立大学情報教育協会, 第 9 巻, 第１号, pp.36-40, 2006. 査読有

⑩ 松浦:ソフトウェア開発実験における PBL の評価方法, 情報処理学会 FIT2006 情報科学技術レターズ, Vol.5, pp.401-404, 2006.査読有

〔学会発表〕（計８件）

① S.Ogata and S.Matsuura, Towards the Reliable Integration Testing: UML-based Scenario Analysis using an Automatic Prototype Generation Tool, Proc of SOFTWARE ENGINEERING, PARALLEL and DISTRIBUTED SYSTEMS (SEPADS '10),pp.151-159, 2010. 査読有

② K.Ito and S.Matsuura, Model Driven Development for Embedded Systems, Proc of SOFTWARE ENGINEERING, PARALLEL and DISTRIBUTED SYSTEMS (SEPADS '10),pp.102-109, 2010. 査読有

③ S.Ogata and S.Matsuura,An Evaluation of a Use Case Driven Requirements Analysis Using Web UI Prototype Generation Tool ,Proc. of The 9th WSEAS International Conference on APPLIED COMPUTER SCIENCE(ACS'09),pp.235-240, 2009. 査読有

④ M.Amakawa, S.Ogata and S.Matsuura, A System Development Method based on a Service Independent Interaction Model, Proc. of The 9th IASTED International Conference on Software Engineering and Applications (SEA2008), pp. 154-159, 2008. 査読有

⑤ S.Ogata andS.Matsuura, Scenario-based Automatic Prototype Generation, Proc. of 32nd Annual IEEE International Computer Software and Applications Conference (COMPSAC2008), pp.492-493, 2008.査読有

⑥ S.Ogata and S.Matsuura, Automatic Generation of UML-based Web Application Prototypes, Proc. of Tenth International Conference on Enterprise Information Systems (ICEIS2008), pp.244-251, 2008. 査読有

⑦ S.Matsuura: IMPROVEMENT IN MARKING OF PROGRAMMING EXERCISES USING A MARKING SUPPORT TOOL AND SUBJECT DESIGN, CATE 2007, pp.76-80, 2007.査読有

⑧ N. Kamigochi and S. Matsuura: A Learning Support Tool for Testing Java Programs, The IASTED International Conference on Software Engineering SE2007, pp.273-278, 2007.査読有

〔その他〕
プロトタイプ自動生成ツールの公開
http://www.sayo.se.shibaura-it.ac.jp/

６．研究組織
(1)研究代表者
松浦　佐江子（MATSUURA SAEKO）
芝浦工業大学・システム理工学部・教授
研究者番号：10348906