



Title	オブジェクト指向に基づくソフトウェア開発プロセス の分析：教育環境における評価実験
Author(s)	神谷，年洋；楠本，真二；井上，克郎 他
Citation	電子情報通信学会ソサイエティ大会講演論文集. 1996, 1996, p. 41-41
Version Type	VoR
URL	<a href="https://hdl.handle.net/11094/51097">https://hdl.handle.net/11094/51097</a>
rights	copyright©1996 IEICE
Note	

*The University of Osaka Institutional Knowledge Archive : OUKA*

<https://ir.library.osaka-u.ac.jp/>

The University of Osaka

## オブジェクト指向に基づくソフトウェア開発プロセスの分析

## —教育環境における評価実験—

Experimental analysis of object-oriented software development on educational environment\*

神谷 年洋<sup>†</sup>      楠本 真二<sup>†</sup>      井上 克郎<sup>†</sup>      毛利 幸雄<sup>‡</sup>  
 Toshihiro KAMIYA<sup>†</sup>    Shinji KUSUMOTO<sup>†</sup>    Katsuro INOUE<sup>†</sup>    Yukio MOHRI<sup>‡</sup>

<sup>†</sup>大阪大学 基礎工学部      <sup>‡</sup>日本ユニシス株式会社

<sup>†</sup>Faculty of Engineering Science, Osaka University      <sup>‡</sup>Nihon Unisys Corporation

## 1 はじめに

近年、ソフトウェアは大規模・複雑化する一方で、納期の短縮化、コスト削減、品質の向上が求められている。ソフトウェア部品の再利用は、それらの要求を実現するための有効な手法の一つである。ソフトウェアの部品化を効率良く行う手法として、オブジェクト指向に基づく開発方法論が提案され、多くの開発現場で実際に行われている。しかし、オブジェクト指向で開発されたプログラムに特有の複雑さやエラーについての分析は、従来の手続き指向のものに比べて必ずしも十分に行われていない。また、ソフトウェア部品の再利用を前提とした、コードの再利用性や保守性の評価手法も十分に検討されていない。本研究では、オブジェクト指向のプログラムの開発(再利用、保守を含む)プロセスを系統的に分析するための手法の開発の第一歩として、教育環境におけるオブジェクト指向開発プロセスを対象に、各種のプロセス・プロダクトデータの収集と分析を行う。

## 2 目的

教育環境におけるプログラム開発プロセスを観察し、開発プロセスに影響を与える要因について分析を行うことを目的とする。具体的には、次の(1)~(3)を行う。

- (1) オブジェクト指向プログラムに特有なフォールトを整理・分類し、開発コスト、品質に大きな影響を与えるフォールトを明確にする。
- (2) どのような特徴を持つソフトウェア部品やフォールトがプログラムの信頼性や再利用性に対して影響を与えるかについて分析する。
- (3) どのような特徴を持つプログラムが変更、あるいは、機能追加をしやすいかを調べる。また、効率のよいプログラムの変更方法についても検討する。

なお、(2)については、ソフトウェア部品の特性を次の項目を用いて評価する。

- (M1) 再利用性: 修正せずに部品として再利用できること。
- (M2) 信頼性: フォールトが少ないこと。
- (M3) 保守性: 変更が容易であること。

(M1)~(M3)は抽象的な概念であり、直接測定できる尺度ではない。しかし、文献[1]等で提案されているメトリクス手法を用いることで、間接的に評価することが可能である。

## 3 実験概要

日本ユニシス株式会社の新人研修におけるC++プログラム開発演習からデータを収集する。この演習では、8チーム(各チーム4、5名の新人で構成される)がそれぞれ、同じ課題(酒屋問題)をOMT法[5]を用いて設計し、引続きVisual C++開発環境[3]を用いてプログラムを作成する。

データとして、設計ドキュメント、プログラムリスト、作業報告書、フォールトレポート、プログラムテキストの更新履歴等を収集する。プログラムテキスト更新履歴の収集にはGINGERシステム[2]を利用する。他のデータはオフライン、あるいは、申告に基づいて収集する予定である。

## 4 分析方針

- (1) エラーの分類: フォールトレポートの内容に基づいて、フォールトの分類を行う。分類項目として、フォールトの影響範囲(例えば、オブジェクトの内部だけに影響を与えるのか、オブジェクトを用いる側にも影響を与えるか)、デバッグに要したコスト等を考えている。
- (2) 変更履歴の分析: プログラムテキストの更新履歴を用いて、フォールトを修正する場合や、機能を拡張した時に効率のよい変更を行っているパターンを分析し、その特徴を抽出する。

## 5 まとめ

本論文では、教育環境におけるオブジェクト指向ソフトウェア開発プロセスの現状把握を目的とした評価実験について検討した。

## 参考文献

- [1] E. M. Kim et al.: "Development of tool for calculating new metrics of C++ program complexity," Proc. of 16th Software Reliability Symposium, pp.34-39(1996).
- [2] S. Kusumoto et al.: "On a measurement environment for controlling software development activities," *IEICE Transactions on Communications Electronics Information and Systems*, E 74, 5, pp.1051-1054(1991).
- [3] 桜田, 田口: Visual C++プログラミング入門, ASCII(1994).
- [4] 山崎利治: "共通問題によるプログラム設計技法解説", 情報処理学会誌, 25, 9, p.934(1984).
- [5] J. Rumbaugh et al.: *Object-Oriented Modeling and Design*, Prentice-Hall(1991).