新人研修がソースコード品質に与える影響の調査

森田 大夢[1,a)]　平尾 俊貴[1]　石尾 隆[1]　新田 章太[2]　小西 俊司[2]　森 康真[2]　松本 健一[1]

概要：ソフトウェア開発者の育成を目的としたプログラミング研修では，プログラミングスキルの向上が期待される．本研究では，研修の効果を測定する試みとして，あるソフトウェア開発企業の新人研修で収集した 22 名のソースコードを用いて，研修前後でのプログラミングスキルの変化をソフトウェア品質の観点から調査した．その結果，研修前後でソースコード内の複雑度はあまり変化せず，宣言命令数が増加する傾向にあることを確認した．その要因として，研修後は変数を必要になった時点で宣言すると同時に初期化して使用するようにプログラムを記述する傾向が見られた．

キーワード：ソースコード分析，品質評価, データマイニング，プログラミング研修

1. はじめに

ソフトウェア開発では，効率的なプログラミング作業を実現するために，プログラミングスキルが重要である [1]．プログラミングスキルは，単純な開発経験年数の増加だけでは向上せず，大学での学習，例えば計算機科学に関する知識の修得も重要である [2]．しかし，実際の企業では，計算機科学系出身者のみからソフトウェア開発者を採用するわけではない．多様な分野から採用された様々な能力を持つ開発者候補に対し，最適な新人研修を実施することで，各企業の活動に適した知識とスキルを持つ開発者を育成することが求められる．

新人研修では，受講者の開発スキルを様々な観点で評価して，各受講者のスキルレベルに沿った学習内容が望まれる．また，研修前後でスキルレベルの成長度を定量評価できれば，より効果的な研修内容の選定に有効な情報になる．しかし，プログラミングスキルは様々な要素が関連するため，その測定は容易ではない．現状では，他の受講者と比較した相対的な自己評価が有力な指針になっている [3]．

本研究は，ある企業で実施された新人研修の効果を，実際に受講者が作成したプログラムの品質の観点から測定することを試みる．当該企業は，長期的なソフトウェア保守を重要視しており，プログラミングに関する知識に加えて，可読性および保守性を重視したプログラミングと，デザインパターン，リファクタリングといったソフトウェア工学の知識の講義・演習を研修に組み込んだ．従来の研修では，プログラミングに関するスキルの成長度の評価指標としてテストケースの通過率を用いていたが，受講者が可読性，保守性などのソフトウェア品質を考慮したプログラミングに取り組めているか否かを評価することが困難であった．そこで本研究では，新人研修前後で作成したソフトウェアの品質の変化を分析することで，研修の効果を測定する．

[1] 奈良先端科学技術大学院大学
Nara Institute of Science and Technology
[2] 株式会社ギブリー
Givery, Inc.
[a)] morita.hiromu.me8@is.naist.jp

2. 研修前後のソフトウェア品質の評価方法

本研究では，ある企業で実施された約 1 か月の新人研修に参加した 22 名への研修の効果を分析する．この研修前後ではプログラミングに関する理解度テストとして同一のプログラム作成課題を出題しており，その解答プログラムコードに対する分析を行う．

プログラム課題の内容は，入力された数値を初期値とし，条件が満たされるまでループを用いて数値を動かし，最終的に得られた値を出力するというものである．受講者は 1 時間以内に，数十個のテストケースをできるだけ多く通過するプログラムを提出する．課題で使用するプログラミング言語は制限しておらず，研修前に提出された解答には受講者ごとに知っている言語，具体的には C，C++，C#，Java，Python が使用されている．また，研修後に提出された解答はすべて Java が使用されている．

解答の評価指標は，あらかじめ準備されたテストケースの通過率から算出される得点である．100 点満点で問題の出力要件を満たすプログラムほど高得点となる．受講者の平均得点は研修前で 19.8，研修後で 59.6 である．すべての受講者の得点は研修後に増加するか，もしくは変化しな

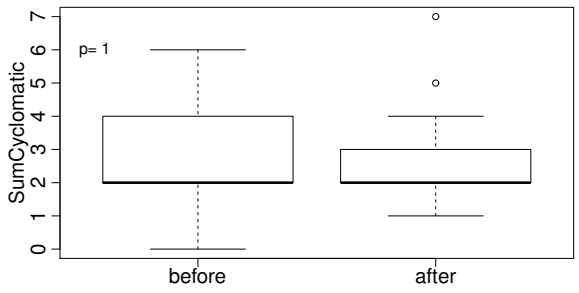

図 1 研修前後に作成したプログラムの複雑度の分布

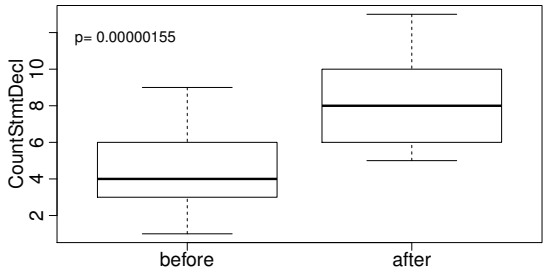

図 2 研修前後に作成したプログラムの宣言命令数の分布

いことを確認した．

本分析では，研修によって可読性の高いプログラムが書けるようになるため，プログラムの複雑度が低下し，かつ必要十分な個数の変数を適切に使用できるようになるという仮説を立てた．この仮説を検証するため，解答プログラム全体のサイクロマチック複雑度と，変数や関数などの宣言を行った数（以降，宣言命令数）という 2 つの指標を研修前後でそれぞれ計測して比較した．これらの指標として，商用ソフト SciTools Understand 5.1 から SumCyclomatic, CountStmtDecl メトリクスの値を取得した．

3. 結果および考察

図 1 に研修前後の複雑度の分布を示す．研修後の受講者の解答プログラムは研修前よりも少ない制御命令を用いて，すなわち低い複雑度で簡潔に解答できるという仮説を立てた．しかし，研修前後で統計的な有意差は確認できなかった．これは，研修で出題した問題が比較的難易度の低い問題であり，複雑な制御構造を必要としなかったことが要因であると考えられる．

図 2 に研修前後の宣言命令数の分布を示す．研修前の平均 4.4 に対して，研修後は 8.3 と大幅に増加した．t 検定の結果より，統計的有意差（$p < 0.0001$）を確認した．研修後のプログラムの内容を目視で確認したところ，必要に応じて変数を宣言すると同時にその値を初期化するようにプログラムを記述する傾向を確認した．なお，研修前後で用いた言語が違う受講者がいることも，結果には影響している．たとえば研修前にクラス宣言を必要としない C 言語や，関数宣言なしに直接命令を記述できる Python を使用していた受講者にとっては，Java に言語を切り替えた際，クラス等の宣言数が増加している．研修の効果のみを効果的に計測できるような研修の設計，実施は今後の課題である．

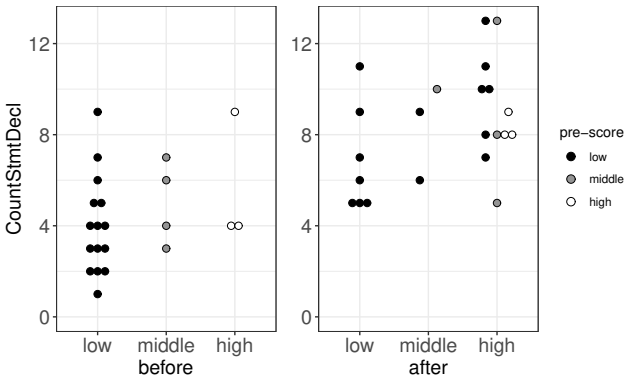

図 3 得点別による宣言命令数の推移

図 3 は，研修前の得点を low，middle，high の 3 段階でレベル分けを行い，研修後の宣言命令数の変化の推移を示したものである．研修前から高得点（high）だった受講者の 3 人中 2 名は，研修後に宣言命令数が 2 倍に増加していた．同様に，研修前後とも低得点（low）であった受講者の宣言命令数も増加する傾向にあった．これらの受講者については得点の変化はなかったが，プログラムの品質に関する指標を通じて研修の影響を確認することができた．

4. おわりに

本研究では，研修前後のプログラミングスキルの変化を，ソースコード品質の観点から調査した．研修後の受講者が作成したプログラムでは，宣言命令数が増加する傾向を確認した．変数宣言の増加が常に可読性の向上につながるとは限らないが，変数を用途ごとに適宜宣言して値を代入するようなプログラムの書き方の傾向として，可読性を重視する研修の効果が表れたと考えられる．

今後の調査では，複雑な制御構造を必要とする課題を使用して比較実験を実施し，プログラミング言語に対する習熟度とプログラムの複雑度，可読性の関係を調査する．さらに，本調査で扱わなかったソースコード品質評価の指標も導入していく予定である．

参考文献

[1] Bergersen, G. R., Sjberg, D. I. K. and Dyb, T.: Construction and Validation of an Instrument for Measuring Programming Skill, *IEEE Transactions on Software Engineering*, Vol. 40, No. 12, pp. 1163–1184 (2014).

[2] Dieste, O., Aranda, A. M., Uyaguari, F., Turhan, B., Tosun, A., Fucci, D., Oivo, M. and Juristo, N.: Empirical evaluation of the effects of experience on code quality and programmer productivity: an exploratory study, *Empirical Software Engineering*, Vol. 22, No. 5, pp. 2457–2542 (2017).

[3] Siegmund, J., Kstner, C., Liebig, J., Apel, S. and Hanenberg, S.: Measuring and modeling programming experience, *Empirical Software Engineering*, Vol. 19, No. 5 (2014).