B-03

ソースコード理解に求められる知識が理解時間に与える影響の実験的評価
An empirical investigation on source code comprehension time and required knowledge to comprehension

田口 雅裕† 森崎 修司† 松本 健一†
Masahiro Taguchi Shuji Morisaki Ken-ichi Matsumoto

1. はじめに

ソフトウェアレビューはソフトウェア開発における成果物を静的に検査することでソフトウェアに含まれる欠陥や矛盾を発見する作業であり，開発プロセスの上流工程でも実施することができるのが特徴である．一般的にソフトウェア開発においては早い段階で欠陥や矛盾を発見できれば，修正にかかるコストが少なくなることが確認されており，レビューを実施することによって欠陥や矛盾を早期発見できれば，ソフトウェアの品質向上に大きな効果をもたらすことができる．

しかしながら，ソフトウェアレビューを正しく実行するための知見は明確ではなく，より効率的にレビューを実施するための知見やスキルを明らかにしていく必要がある．

本研究は，レビューの経験（読解の経験）と対象ソースコードがレビューアに対して求める知識とが，ソースコードを理解するために必要となる時間に与える影響を明らかにすることを目的とし，実務者を被験者とするソースコード読解実験を実施した．具体的には，理解のために GUI フレームワーク/ライブラリの知識を必要とするソースコード片，それら知識を必要としないソースコード片を用意し，被験者を”レビューの経験が豊富かつ，GUI フレームワーク/ライブラリの知識を多く持つ”グループ，”レビューの経験が少なく，GUI フレームワーク/ライブラリの知識も少ない”グループ，”GUI フレームワーク/ライブラリの知識を多く持ち，レビューの経験が少ない”グループの 3 つに分類し，それぞれの被験者の読解時間を比較する．

2. 実験

2.1 概要

Andrew らはソフトウェア開発者がソフトウェアの開発・保守においてソースコードをレビューする際に，馴染みのないコードに出会ったとき，開発者はコード内容を理解するために，そのコードに関連する情報をリファレンスまたはプログラムコード中から探し出し，収集すると述べている[1]．このため，知識がある場合には作業時間を減らすことができ，理解時間を減らすことができると考えられる．

本実験では，ソースコードの理解時間を始めとするデータを収集し，被験者の経験，知識をもとにソースコード理解時間を比較することで，知識とソースコード理解時間の関係を明らかにすることを目的とした

†奈良先端科学技術大学院大学 情報科学研究科，
Nara Institute of Science and Technology

ソースコードがレビューアに求める知識とソースコードの読解時間の関係について評価するために，ソフトウェア開発の実務者を対象にレビュー実験を行った．

レビュー対象となるソフトウェアは JAVA 言語による GUI を利用したペイントアプリケーションを用いた．

被験者にはあらかじめバージョン 1 のソースコードを理解しておいてもらい，その後，バージョン 1 のソースコードに対するアップデートバージョンをレビューしてもらった．ソースコードは紙面上，PC 上のどちらを利用しても良いものとし，レビュー中の関連情報の検索も自由に行えるものとした．

ソースコードに用いられる知識とレビューアが有する知識がレビューアの理解時間に与える影響を調査するために，被験者にアンケートに答えていただいた．アンケートの項目を以下に示す．

- 主に使用するプログラム言語
- クラスライブラリに関する知識・使用経験
- レビュー経験

また，実験結果として得られたデータを以下に示す．

- バージョン 1 の読解時間
- 各差分のレビュー，理解時間
- 被験者が感じたコードの難易度

以上の情報を収集した．

2.2 差分の概要

実験では GUI アプリケーションに対する修正・削除・既存の機能の変更を差分コードとして用意した．各差分は独立のものとして扱っており，差分同士がお互いに干渉することはない．つまり，ある差分 A が加えられた後に差分 B を加えるといったように差分同士に順列が存在しない．すべての差分が独立であるとはベースとなるソースコードに対し，各差分が別々の次元で加えられ，差分の数だけ更新プログラムが生成される．

被験者の GUI プログラミングの経験（以下 GUI 経験）の有無がソースコードを理解するための時間にどのように影響を与えるかを調査するために，用意した差分について GUI 知識を用いている差分とそうでないものとに選定した． GUI に関する知識が必要になる差分は GUI 上の変更を要求する差分で，例えば，ボタンの設置，イベントリスナなどの変更に対する修正・削除・追加に関する差分とした．

以下に比較で用いる差分の概要を示す．また，表 1 には各差分の追加行数などを示す．

■ 差分 1

イベントリスナに関する機能変更を行う差分．バージョン 1 ではマウスがキャンパス（絵の描画領域）以外の位置にある場合にもマウスポインタの座標を表示するようになっている．マウスがキャ

ンパスから離れていることを明示的に表すために，マウスがキャンパスから外れた時はマウス座標の表示を行わないようにする．

■ 差分 3
バージョン 1 に用意されているブラシツール機能を削除する差分．バージョン 1 では消しゴムツールやペンツールなどのボタンと共にブラシツールのボタンが配置されている．機能削除に伴って GUI ボタンの配置に空白が生じるので，GUI ボタンのレイアウトの整理も行う．

■ 差分 4
カラーパレットをモノクロ仕様に変更する差分．バージョン 1 ではカラーチューザ機能を用いてグレースケール以外の色（30×10 色）を描画色として選択できる．バージョン 2 ではカラーチューザのレイアウトのサイズを変更することでモノクロ化する．

■ 差分 5
スプレー機能を追加する差分．バージョン 1 では，消しゴムツールなどのボタンが配置されているツールパレット上にスプレー機能を実行するための GUI ボタンが存在している．しかし，ボタンを押しても機能を実装していないので利用することはできない．バージョン 2 ではスプレー機能を実行するための GUI ボタンを有効にする．GUI スプレー機能は選択された（アイコン選択）オプションに合わせてスプレーの散布の密度を変化する機能を持ち合わせている．

■ 差分 7
キャンパスクリアボタンを追加する．バージョン 1 にはキャンパスを白紙に戻すための GUI ボタンもその機能も存在しない．よって，GUI ボタンの配置とキャンパス内に描かれた内容を消去し，白紙に戻す機能を追加する．

■ 差分 12
カラーパレットの機能を追加する差分．バージョン 1 ではカラーパレットから色を選択する機能が付いている．バージョン 2 では Swing の JSlider, JSpinner クラスを用いてスライダ，スピナ機能を加える．加えたスライダ，スピナを用いて RGB 値の設定を行うことで，色の選択ができるように機能を拡張する．

文献[2]では，保守開発におけるソースコードの理解時間増加の主要原因として，仕様変更とソースコードの変更箇所の多さが挙げられている．今回の実験に用いたアプリケーションではソースコードの追加・削除・修正によって変更されるファイル数は高々2 ファイルとなり，更新箇所の分散によるソースコード理解への影響は小さいと考えられる．そのため，変更ファイル数が少ない差分については GUI 知識による影響がより計測しやすいコードと考えることができる．

表 1　各差分の規模と種別

差分	追加行数(削除)	変更ファイル数
1	6(0)	1
3	59(58)	2
4	2(1)	1
5	167(3)	2
7	33(0)	2
12	268(5)	1

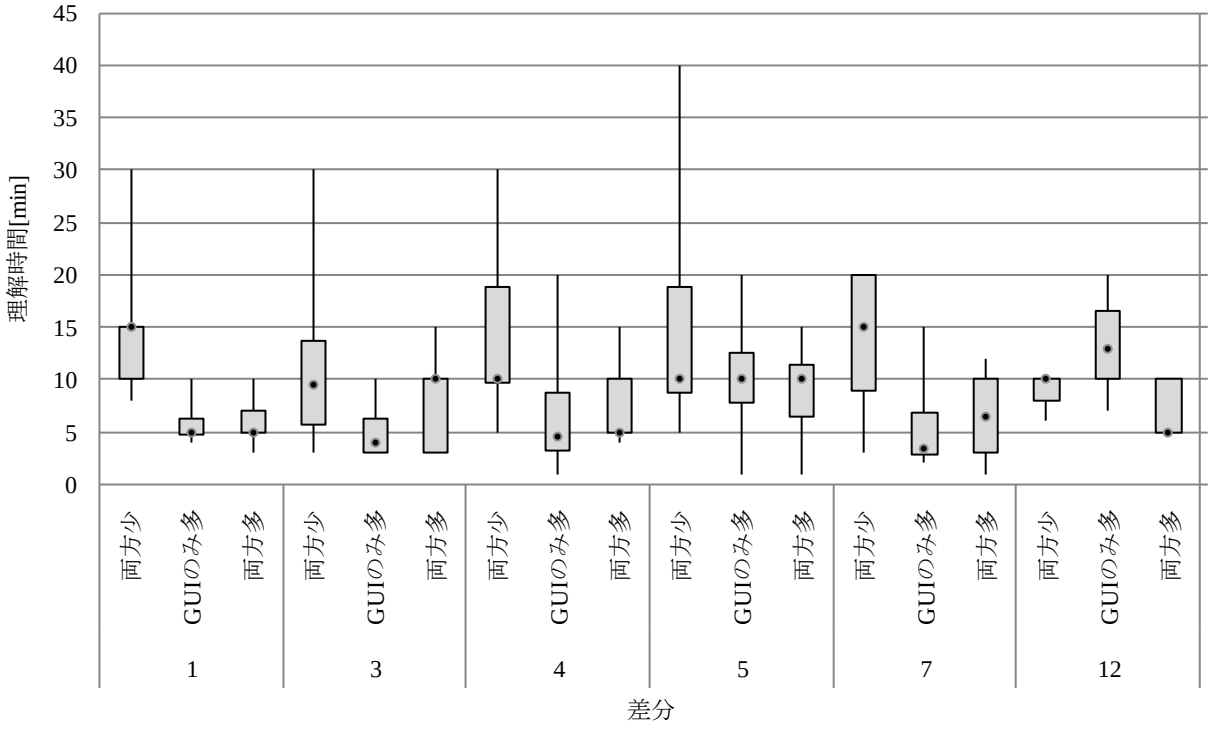

図 1　知識差のあるレビューアの理解時間比較

3. 結果

レビュー実験によって得られた結果より，レビューアの GUI に関する知識と各差分の理解時間の関係について示す．

なお，本稿で使用している略語の意味は以下に示すとおりである．

- 両方少
  レビュー経験が少ないかつ，GUI 経験も少ない被験者の集合．
- GUI のみ多
  レビュー経験は少ないが，GUI 経験は多い被験者の集合．
- 両方多
  レビュー経験も GUI 経験も多い被験者の集合．

3.1 知識差があるレビューアの理解時間比較

実験で得られた結果から各差分の理解時間を比較した．その際，被験者を GUI 経験毎に分けて比較した．

図 1 は両方少，GUI のみ多，両方多の被験者が各差分を理解すのに要した時間を箱ひげ図で表したものである．

以下，各差分ごとにレビューアの理解時間を比較する．

■ 差分 1
理解時間の中央値を比較すると，最も大きいのは両方少の被験者の 15 分となり，最小は GUI のみ多と両方多の 5 分となった．箱の位置は両方少の場合，10 分から 15 分の間に位置しており，最も高い．最低は GUI のみ多の場合で，4.75 分から 6.25 分に位置した．ひげの最大値は両方少の 30 分で，最小値は両方多の 3 分となった．

■ 差分 3
理解時間の中央値を比較すると，最も大きいのは両方多の被験者の 10 分となった．最小は GUI のみ多の 4 分となった．箱の位置は両方少の場合，5.75 分から 13.75 分の間に位置しており，最も高い．最も低いのは GUI のみ多の 3 分から 6.25 分となった．ひげの最大値は両方少の時の 30 分となり，最小値は両方少，GUI のみ多，両方多の全てで 3 分となった．

■ 差分 4
理解時間の中央値を比較すると，最も大きいのは両方少の 10 分で，最も小さいのは GUI のみ多の 4.5 分となった．箱の位置は両方少の場合に 9.75 分から 18.75 分に位置しており最も高い．最も低いのは GUI のみ多の場合で，3.25 分から 8.75 分に位置した．ひげの最大値は両方少の 30 分で最小値は GUI のみ多の 1 分となった．

■ 差分 5
理解時間の中央値を比較すると，レビューアの知識差に関わらず 10 分となった．箱の位置が最も高いのは両方少の 8.75 分から 18.75 分となり，最も低いのは両方多の 6.5 分から 11.5 分となった．ひげの最大値は両方少の 40 分が最大で，GUI のみ多，両方多の 1 分が最小値となった．

■ 差分 7
理解時間の中央値を比較すると，両方少の被験者が 15 分で最大となり，GUI のみ多の被験者が 3.5 分で最小となった．箱の位置が最も高いのは両方少の被験者で 9 分から 20 分に位置する．また最も箱の位置が低いのは GUI のみ多の被験者で，2.75 分から 6.75 分に位置する．ひげの最大値は GUI のみ多が最大の 15 分となった．最小値は両方多の 1 分となった．

■ 差分 12
理解時間の中央値を比較すると，GUI のみ多の被験者が 13 分で最大となっており，最小は両方多の被験者の 5 分となった．箱の位置が最も高い位置になったのは GUI のみ多の被験者で 10 分から 16.5 分に位置する．箱の位置が最も低いのは両方多の被験者で 5 分から 10 分に位置する．ひげの最大値は GUI のみ多の 20 分となり，最小値は両方少の 6 分となった．

4. 考察

図 1 で得られた結果から，ソースコード理解に GUI 知識が必要となる差分をレビューする際には，GUI 経験が多い方がコード理解に必要な時間が少なくなることが全体の傾向として表れている．この傾向は特に，両方少と GUI のみ多の被験者を比較したときに顕著に表れている．GUI のみ多の被験者はレビュー経験と GUI 経験のどちらもが豊富な被験者(両方多)と比較しても，同等かそれ以上の速さでソースコードを理解できるようになっている．特に，差分 1 に関しては両方少と両方多の被験者をウィルコクソンの有意検定で検定した場合，有意水準 5%で有意差を確認できる．しかし，GUI のみ多と両方多の被験者の有意差は有意水準 5%で現れない．差分 1 はキャンパスパネルからマウスが離れたことを知らせるイベントリスナを追加することで，キャンパス外にマウスポインタがあるときのマウス座標の表示を行わないようにしている．追加するコード自体の規模は小さく，他のコードに影響を与えるわけでも可読性が低いわけでもない．Java の GUI に関する知識があれば，理解に苦しむようなコードではないと考えられる．このことから差分 1 はレビュー経験に関わらず GUI 知識を身に付けることで理解が早まることがわかる．

差分 3，4，7 については両方多の被験者よりも GUI のみ多の被験者の理解時間が早くなっている．これは，レビュー経験が多いことによってソースコード理解時に注意すべき点が増加し、GUI のみ多の被験者よりも理解時間が多くなっていることが考えられる．しかし，ソースコードのどのような要素がレビュー経験に対して理解時間の増加を要求しているのかは現段階では確認できていない．考えられる要因として，ソースコードの追加・修正・削除においてはレビューアの理解にはベースとなるソースコードの理解が大きく影響すると Dunsmore らは述べているが[3]，今回はバージョン 1 に関する実験結果にまで言及出来ていない．よって，差分 3，4，7 については各被験者のバージョン 1 のソースコードの理解度を調査し，

知識差があるレビューアの比較を行う必要があると考えられる．

差分 12 は差分中で追加・削除の行数が最も大きい差分である．差分 1，3，4，5，7 の追加・削除行数は順に 6，59，2，167，33 行であるが差分 12 に関しては 268 行である．差分 12 は追加・削除行数が多いため，GUI 経験が多くなってもレビュー経験が少ないままでは理解時間が減少しなかったものと考えられる．

また，差分 5 は 2 番目に追加・削除行数が多いが，この差分についても両方少の被験者と GUI のみ多の被験者を比較すると理解時間の中央値に差が表れていないことが分かる．さらに，この差分についてはスプレー描画を表現するために制御構文がネストされていて，GUI 知識以外の複雑さが大きくなっている．文献[4]では複雑なコード変更はソフトウェア開発の遅延の原因になると述べられている．この影響を考慮すると，差分 12 には GUI 知識以外にも要求するスキルが含まれており，それが差分 12 との違いと考えられる．

5. まとめ

レビューの経験（読解の経験）と対象ソースコードがレビューアに対して求める知識とが，ソースコードを理解するために必要となる時間に与える影響を明らかにすることを目的とし，実務者を対象としてソースコード理解時間を計測する実験を実施した．

GUI 知識が必要になるソースコードを理解するための時間は，レビューアの GUI 経験が多ければ短くなり，レビューの経験が少ない場合でも，その傾向が認められた．

今後の展望としては，考察でも述べたがバージョン 1 の理解時間とバージョン 2 の理解時間の依存関係について調査していくことで，より詳細に知識と理解時間の関係を明らかにできるのではないかと考えている．また，被験者が主に使用している開発言語を考慮した知識差比較なども考えられる．さらに，各差分に対応した同規模の差分を用意してレビュー実験を行なうことでより深く知識とコードの理解時間の関係について調査していく．

謝辞

本研究の一部は，文部科学省「次世代 IT 基盤構築のための研究開発」の委託に基づいて行われた．また，本研究の一部は，文部科学省科学研究補助費（若手 B:課題番号 21700033）による助成を受けた．本論文の分析対象データの一部は IBM Academic Initiative Program（http://www.ibm.com/developerworks/university/academicinitiative/）の支援によって収集された．

参考文献

[1] A J. Ko, B A. Myers, M J. Coblenz, and H H. Aung, “An Exploratory Study of How Developers Seek, Relate, and Collect Relevant Information during Software Maintenance Tasks,” IEEE Transactions on Software Engineering, Vol. 32, No. 12, pp.971-987, 2006.

[2] H C. Benestad, B Anda, E Arisholm, “Understanding cost drivers of software evolution: a quantitative and qualitative investigation of change effort in two evolving software systems,” Empirical Software Engineering, Vol. 15, No. 2, pp.166-203, 2010.

[3] A Dunsmore, M Roper, M Wood, “The role of comprehension in software inspection,” The Journal of Systems and Software, Vol 52, 2000, pp.121-129.

[4] A E. Hassan, “Predicting Faults Using the Complexity of Code Changes,” Proceedings of the IEEE 31st International Conference on Software Engineering, pp.78-88, 2009.