オープンソース開発におけるコードクローン含有率の収束傾向に関する調査

本田　暁[†]　阿萬　裕久[††a)]（正員）
佐々木隆志[††]　川原　稔[††]（正員）

Investigation of Convergence Tendency of Code Clone Ratio in Open Source Development

Akira HONDA[†], *Nonmember*, Hirohisa AMAN[††a)], *Member*, Takashi SASAKI[††], *Nonmember*, and Minoru KAWAHARA[††], *Member*

[†] 愛媛大学工学部情報工学科，松山市
Department of Computer Science, Faculty of Engineering, Ehime University, 3 Bunkyo-cho, Matsuyama-shi, 790–8577 Japan
[††] 愛媛大学総合情報メディアセンター，松山市
Center for Information Technology, Ehime University, 3 Bunkyo-cho, Matsuyama-shi, 790–8577 Japan
a) E-mail: aman@ehime-u.ac.jp

あらまし　コードクローンは保守を困難にする要因の一つと言われているが，その一方で全てのクローンが有害とは限らないとの指摘もあり，全てを除去するのが適切ともいえない．そこで本論文ではその適切な分量について検討するため開発・保守の進行に伴うクローン含有率の動向を調査し，おおむね20%程度に収束する傾向にあることを報告している．

キーワード　コードクローン，クローン含有率，メトリックス，定量的指標

1. まえがき

産業界において，大規模かつ複雑なソフトウェアに対する効率的な保守は重要な課題の一つである．その際，コードクローンの存在は保守作業を困難にする要因になるといわれている [1]．コードクローンとはソースコードの中に存在する互いに一致または類似したコード断片のことであり，あるコード断片に修正が必要な場合，それとクローン関係にある他の全てのコード断片にも同様の修正が必要となる可能性があり，それらを全て確認しなければならなくなる．それゆえコードクローン検出ツール（例えば CCFinderX [2]）を活用して該当する箇所を見つけ出し，そこにリファクタリングを施して保守性を高めるといった対応がなされている [3]．

しかしその一方で，全てのコードクローンがソフトウェア保守に悪影響を及ぼしているわけではないという報告も少なくなく [4]〜[7]，全てを除去するのが適切であるとも断言できない．直感的に言えば，“コードクローンが多すぎるというのは保守作業の困難化を招きやすい”といえるが，同時に“ある程度はコードクローンが存在していても問題ではない（普通である）”ともいえる．ソフトウェア開発や保守が長期間にわたって進行していけば，その中で問題となるようなクローンは解消され，やがては無害なクローンが残っていくのではないかと考えられる．

そこで本論文では，実際に広く使われているソフトウェアに対してデータ収集を行い，ソースコードに含まれるコードクローンの割合（含有率）が開発や保守の進行とともにどのように変動していくのかという観点からコードクローンの調査と考察を行う．

2. 調査対象と方法

2.1 対　　象

調査対象ソフトウェアの一覧を表 1 に示す．調査の一般性を高めるため，主要開発言語が C/C++であるという以外には特別な基準を設けず，開発規模，開発期間並びにドメインは敢えて不揃いなものを選択した．なお，言語が異なった場合，コードクローンの生み出され方に違いが生じている恐れがあり，その種の懸念を排除するために今回は C/C++に限定した．他言語でも同様の調査を行うことは今後の課題としたい．

2.2 方　　法

対象ソフトウェアのソースファイルを 3ヶ月ごとに入手し，それぞれコードクローン検出ツールを用いてクローン含有率を算出する．検出ツールにはコードクローン研究で広く使われている CCFinderX を使用する．クローン含有率としては CCFinderX で算出される CVRL を使用する．これは“クローンとなっているコード断片の登場するソースコード行が全体に占める割合”を表したメトリックスである．

表 1　調査対象ソフトウェア
Table 1　Surveyed software products.

ソフトウェア	データ収集期間
Firefox(注1)	2004/11/09〜2013/08/06
Linux kernel(注2)	1994/03/13〜2013/09/02
gimp-for-painters(注3)	1997/12/01〜2013/06/01
handbrake-jp(注4)	2006/03/01〜2011/03/01
Password Safe(注5)	2002/03/01〜2013/09/01
Cool Reader(注6)	2007/03/15〜2013/09/01

(注1)：http://www.mozilla.jp/firefox/
(注2)：https://www.kernel.org/
(注3)：http://sourceforge.jp/projects/gimp-painter/
(注4)：http://sourceforge.jp/projects/handbrake-jp/
(注5)：http://passwordsafe.sourceforge.net/
(注6)：http://coolreader.org/

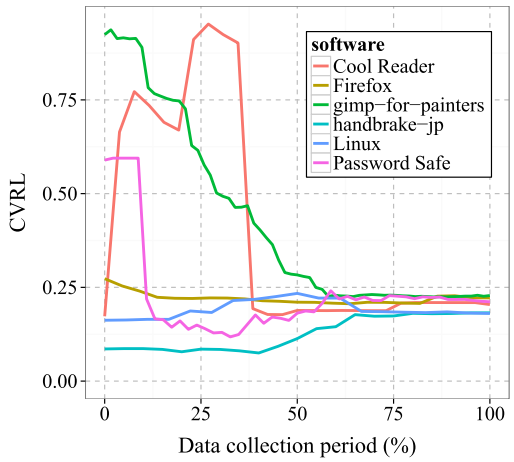

図 1　クローン含有率の動向
Fig. 1　Changes in CVRL.

表 2　データ収集最終日におけるクローン含有率
Table 2　CVRL on the final day of data collection.

ソフトウェア	CVRL
Firefox	0.222
Linux kernel	0.180
gimp-for-painters	0.228
handbrake-jp	0.182
Password Safe	0.212
Cool Reader	0.204

適切なデータ収集間隔の設定については議論の余地があるが，これが短すぎるとその間にソースコードの変更がほとんど起こらず，クローン含有率の動向をうまくつかめない恐れがある．今回の分析対象とは異なるが，Eclipse の開発で “同一のソースファイルに対して複数回のコード変更が起こる間隔の期待値が約 100 日であった” という報告 [8] を参考にし，ここでは 3ヶ月を一つの目安として調査を行うこととした．

3. 調 査 結 果

結果を図 1 及び表 2 に示す．図 1 は各ソフトウェアにおけるクローン含有率（CVRL）の動向を表したグラフである．グラフの横軸は，データ収集開始日を 0，最終日を 100%として正規化した期間になっている．縦軸はクローン含有率である．表 1 はデータ収集最終日におけるクローン含有率をまとめたものである．

いずれのソフトウェアのクローン含有率も 20%程度に収束していく傾向にあることが見てとれる．なお，紙面の都合上詳細は割愛するが，いずれのソフトウェアもクローン含有率の変動が小さくなり安定した時期であっても開発が終了していたわけではなく，それぞれ開発は継続的に行われていたことを付記しておく．

表 3　クローン含有率のソフトウェア間での平均と標準偏差の動向
Table 3　Changes in the mean value of CVRL and the standard deviation value of CVRL within surveryed software products.

データ収集期間（%）	平均	標準偏差
10	0.389	0.336
20	0.340	0.281
30	0.339	0.320
40	0.213	0.107
50	0.202	0.057
60	0.202	0.031
70	0.200	0.022
80	0.204	0.018
90	0.207	0.020
100	0.205	0.020

同様にオープンソースソフトウェアにおけるクローン含有率を調査した先行研究として Uchida らの論文 [9] がある．文献 [9] では多数のオープンソースソフトウェアを対象にクローン検出を行い，その割合の平均値やクローンの内容について考察している．今回の調査では，ソフトウェアの成熟を経てクローン含有率がそれぞれ一定値に収束しているという傾向が見られたが，それとともにソフトウェア間での “個体差” も小さくなっているということも見てとれた（表 3）．つまり，先行研究で含有率の平均が一つの目安として報告されていた(注7)が，ソフトウェアが成熟するにつれソフトウェア間での “分散” も小さくなり，いずれも同様の値（約 20%）へ収束していく傾向を確認できたことが本論文の貢献である．

今回は含有率の動向について調査を行い，その値の収束傾向を確認できた．しかしながら，現実にどのようなコード断片がコードクローンとして残り続けているのか，逆にどのようなコード断片は途中でクローンでなくなっていくのかという観点では十分に追跡できていない．その点についても詳細に追跡していき，保守性への影響について考察していくことは重要な今後の課題としたい．

4. む　す　び

本論文ではコードクローンの含有率に着目し，その値がソフトウェアの成熟とともにどのように変化して

(注7)：文献 [9] ではクローン含有率の平均値として 11.3%という数値が報告されている．ただし，これはトークン数の割合に基づいた数値であるため，本論文の結果（行単位での含有率）と直接比較するのは難しい．

いくかを調査した．規模やドメインの異なる 6 種類のソフトウェアについて追跡していくといずれも 20%程度に収束していく傾向が観測された．

通常，どのようなソフトウェアであっても多少のコードクローンは存在していると思われるが，今回の調査結果はその適度な量を議論する一つの資料になると考える．今後の課題として，コードクローンにはさまざまな生成要因が混在しているため，それらを分類しつつ開発過程を解析していき，どういったクローンは残り続け，どういったクローンは除去されていくのかという観点から保守性への影響について解析することが挙げられる．また，他言語でも同様の調査を行い，言語の違いについても考察する必要がある．

文　献

[1] 井上克郎，神谷年洋，楠本真二，“コードクローン検出法,” コンピュータソフトウェア，vol.18, no.5, pp.47–54, Sept. 2001.

[2] T. Kamiya, “the archive of ccfinder official site,” Aug. 2010. http://www.ccfinder.net/

[3] 肥後芳樹，吉田則裕，“コードクローンを対象としたリファクタリング,” コンピュータソフトウェア，vol.28, no.4, pp.43–56, Nov. 2011.

[4] J. Krinke, “Is cloned code more stable than non-cloned code?,” Proc. 13th Int. Working Conf. Source Code Analysis and Manipulation, pp.57–66, 2008.

[5] T. Kamiya, “Classifying code clones with configuration,” Proc. 4th ICSE Int. Workshop on Software Clones, pp.75–76, 2010.

[6] J. Harder and N. Gode, “Clone stability,” Proc. 15th European Conf. Software Maintenance and Reengineering, pp.65–74, 2011.

[7] 堀田圭佑，肥後芳樹，楠本真二，“生成抑止，分析効率化，不具合検出を中心としたコードクローン管理支援技術に関する研究動向,” コンピュータソフトウェア，vol.31, no.1, pp.14–29, Feb. 2014.

[8] 阿萬裕久，“オープンソース開発におけるソースコードの安定性予測について,” 情処学ソフトウェア工学研報，vol.2006, no.125, pp.57–64, Nov. 2006.

[9] S. Uchida, A. Monden, N. Ohsugi, T. Kamiya, K. Matsumoto, and H. Kubo, “Software analysis by code clone in open source software,” Journal of Computer Information Systems, vol.45, no.3, pp.1–11, April 2005.

（平成 26 年 2 月 5 日受付，2 月 28 日再受付）