オープンソース開発における開発者の貢献度とソースコード品質の関係について

山内一輝[†1] 阿萬裕久[†2] 川原稔[†2]

本稿は，オープンソース開発における“開発者”に注目し，各開発者の各ソースファイルに対する貢献度を定量化する基準について提案を行っている．

On the Relationship Between Developer's Degree of Contribution and Their Source Code Quality in Open Source Development

Kazuki Yamauchi,[†1] Hirohisa Aman[†2] and Minoru Kawahara[†2]

This paper focuses on “developers” in open source development, and proposes criteria for quantifying developer's degree of contributions to their source files.

1. はじめに

近年，ソフトウェア工学分野において，ソフトウェアリポジトリを分析し，そこから有益な知見を得ようとするソフトウェアリポジトリマイニングが盛んに研究されるようになってきている[1)]．最近では，ソースファイルの変更履歴のみならず，そこに携わる開発者についても研究が行われている[2)]．本稿はその一端として，これまでの研究ではあまり注目されなかった“開発者の貢献度”に着目し，貢献度を定量化してソースコードの品質評価へ活用することを考える．

2. 開発者の貢献度

2.1 定　義

開発者の貢献度を次の三つの視点から考える．

(1) 各ソースファイルの開発に携わった人数
(2) 各ソースファイルに対して各開発者が追加，削除及び変更を行ったコード行数
(3) 各ソースファイルの所定の版において，各開発者が最終作業者となっているコード行数

まず，(1) “携わった人数”について説明する．これは，当該ソースファイルの開発において，ソースコードの追加，削除または変更が何人の開発者によって行われたかを意味する．つまり，そのソースファイルの開発に貢献した人数である．ただし，ここでいう人数とは延べ人数ではなく，同じファイルに対して同一人物が複数回の修正を行っていたとしても，それは 1 人と数える．これは，携わった人数の多さとその後の品質との関係を分析することを目的としている．

次に，(2) “追加，削除及び変更を行ったコード行数”に関して説明する．視点 (1) では，過去に当該ソースファイルに関わったことがあるといっても，数行しか関わっていない場合と数百行関わっている場合とでは貢献度も異なると考えられる．そこで，そのソースファイルが現在に至るまでに，各開発者がどれだけの修正を施したのかにも着目し，その総量（累積行数）でもって貢献度を定量化することを考える．これによって一人一人の貢献の量がその後の品質に及ぼす影響について分析することを目指している．

最後に，(3) “各開発者が最終作業者となっているコード行数”について説明する．視点 (2) では，追加，削除または変更された行数を累積することを提案したが，その中には後の版で消えてしまったソースコードも含まれる．つまり，ある特定の版に着目して品質を議論しようとした場合，既に存在しない部分に対する貢献度も考慮するというのが (2) の視点である．これに対し，その時点で有効となっている行のみに着目して分析することも考えられる．視点 (3) は，対象としている版の各行について，その行の最終作業者，つまり，最後にその行を作成または修正した人物に着

†1 愛媛大学工学部情報工学科
Department of Computer Science, Faculty of Engineering, Ehime University
†2 愛媛大学総合情報メディアセンター
Center for Information Technology, Ehime University

表 1 開発履歴の例

コミット No.	イベント
1	A が f_1（100 行）を新規作成
2	B が f_2（80 行）を新規作成
3	A が f_1 に 20 行追加 f_2 を 10 行変更
4	C が f_3（80 行）を新規作成
5	B が f_2 に 5 行追加
6	C が f_3 から 5 行削除

目し，その行数でもって各開発者のそのソースファイルに対する貢献度を定量化しようというものである．これも視点（2）と同様の分析を目的としている．

2.2 例

前述した三つの視点について，簡単な例を示す．例として三人の開発者 A, B, C が三つのソースファイル f_1, f_2 及び f_3 を開発しているとする（**表 1**）．

視点（1）の場合，コミット 6 完了時点で各ソースファイルの開発に携わった人数は

- f_1: 1 名（A のみ）
- f_2: 2 名（A, B）
- f_3: 1 名（C のみ）

となる．

視点（2）の場合，各ファイルに対する各開発者の追加・削除・変更行数の累積値は，**図 1** より **表 2** のとおりとなる．例えば，ファイル f_2 の場合，開発者 B はコミット 2 と 5 でそれぞれ 80 行，5 行の追加を行っており，貢献は 85 行となる．これに対し，開発者 A はコミット 3 で 10 行の修正を行っており，これが A の貢献となる．開発者 C はこのファイルに関係していない．

視点（3）の場合，各ファイルにおける各行の最終作業者の分布は **図 2** のようになり，その結果，各開発者の貢献は **表 3** のようになる．ファイル f_1 に対しては開発者 A によるコード追加のみであるため，視点（2）と同じ結果となる．ファイル f_2 においては，コミット 3 で開発者 A がそれまでのソースコードのうち 10 行を変更しているため，その部分の最終作業者は B ではなく A になっている．その他の部分は B のままであり，コミット 5 で追加された分も含め，開発者 B の貢献は 75(= 80 − 10 + 5) 行となっている．ファイル f_3 では開発者 C しか登場しないが，コミット 6 において 5 行が削除されているため，最終的には 75(= 80 − 5) 行が開発者 C の貢献として残っている．

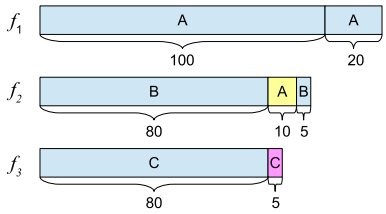

図 1 視点（2）での定量化

表 2 視点（2）での定量化

	A	B	C
f_1	120	0	0
f_2	10	85	0
f_3	0	0	85

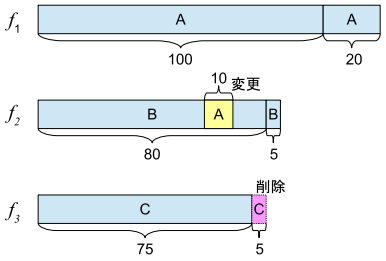

図 2 視点（3）での定量化

表 3 視点（3）での定量化

	A	B	C
f_1	120	0	0
f_2	10	75	0
f_3	0	0	75

この例のように，我々は三つの異なる視点から開発者の貢献度を定量化することを考えている．

3. おわりに

現在，GitHub において stars の多い上位 100 プロジェクト ⋆1 について，データの収集と整理，並びに分析を行っている．ワークショップでは，分析結果について議論を行いたい．

謝辞 本研究は JSPS 科研費 25330083（基盤研究（C））の助成を受けたものです．

参考文献

1) 門田暁人，伊原彰紀，松本健一：ソフトウェアリポジトリマイニング，コンピュータソフトウェア， Vol.30, No.2, pp.52–65 (2013).
2) Posnett, D., D'Souza, R., Devanbu, P. and Filkov, V.: Dual Ecological Measures of Focus in Software Development, *Proc. 2013 Int'l Conf. Softw. Eng.*, pp.452–461 (2013).

⋆1 2015 年 9 月末時点でのものである．ただし，5 位の node-v0.x-archive と 48 位の node は同じプロジェクトであり，前者は後者に引き継がれたプロジェクトであったため，実際には前者を除く 99 個のプロジェクトを対象とした．