情報処理学会論文誌 Vol.56 No.2 715–719 (Feb. 2015)

テクニカルノート

GitHub上の活動履歴分析による開発者分類

尾上 紗野[1,a)] 畑 秀明[1,b)] 松本 健一[1,c)]

受付日 2014年8月1日, 採録日 2014年10月8日

概要：オープンソースソフトウェア（以下，OSS）開発には多くの開発者が携わっており，異なる特徴を持つ開発者が存在すると考えられる．開発者の特徴を明らかにすることで，OSSプロジェクトの成功に必要な開発者を明らかにできるなどのソフトウェア工学における新しい観点の発見が期待される．本稿ではGitHubで活発なOSSプロジェクトであるhomebrewとnodeに参加する開発者を活動履歴からクラスタリングし，その結果から開発者を分類した．クラスタリングで得られた樹形図を分析した結果，活発なOSSプロジェクトには迅速・議論型，迅速・総合型，悠然・総合型などの異なったタイプの開発者がいることが分かった．

キーワード：OSS，GitHub，開発者分類

Developer Classification Based on Developers' Activities in GitHub

Saya Onoue[1,a)] Hideaki Hata[1,b)] Kenichi Matsumoto[1,c)]

Received: August 1, 2014, Accepted: October 8, 2014

Abstract: In open source software projects, there should be different types of developers. Clarifying the characteristics of developers may enable us to manage projects successfully. In this paper we studied developers in active projects in GitHub, homebrew and node. Based on the analysis of development activities in GitHub, we classified developers and found that there are different types of developers in active projects like fast-commenter, fast-generalist, and slow-generalist.

Keywords: OSS, GitHub, developer classification

1. はじめに

オープンソースソフトウェア（以下，OSS）プロジェクトには，多くの開発者が開発に携わっている．商業プロジェクトとは異なり誰でも参加可能なOSSプロジェクトには，異なる特徴を持つ開発者が多く存在すると考えられる．OSSプロジェクトに参加する開発者の特徴を明らかにすることで，ソフトウェア工学における新しい観点の発見が期待される．たとえば，活発なOSSプロジェクトに参加する開発者の特徴から，OSSプロジェクトの成功に必要な開発者を明らかにできると考えられる．OSSプロジェクトで開発者が行う活動は，コーディングを初め，コメントやバグ報告など多岐にわたる．OSSプロジェクトの成功には，これらの開発者がバランス良く開発に参加する必要があると考えられる．

従来からソフトウェア開発者の活動に着目した研究が行われている．Iharaらは，コミッタの過去の活動とその活動量を分析し，一般開発者の中からコミッタ候補者を見つけ出すためのコミッタ予測モデルを構築した[1]．Zhouらは，OSSプロジェクトのLong Term Contributors（LTC）の活動を分析し，開発者の活発さはプロジェクトの人気に左右されることを明らかにした[5]．これらの研究ではOSSプロジェクトの特徴や，不特定多数の開発者の傾向を明らかにすることは可能であるが，開発者個人の特徴を明らかにすることはできない．

また，開発者の性格分析に基づく研究が行われている．SfetsosらはMyers-Briggs Type Indicator（MBTI）を用い

[1] 奈良先端科学技術大学院大学
Nara Institute of Science and Technology, Ikoma, Nara 630–0192, Japan
[a)] onoue.saya.og0@is.naist.jp
[b)] hata@is.naist.jp
[c)] matumoto@is.naist.jp

てペアプログラミングでの相性の影響を調査し，良好なコミュニケーションを築くことができる組合せを示した [4]．Salleh らは人間の個性を特徴づける適応性，誠実性，外向性，同調性，神経質性の 5 つの観点から開発者を分析し，ペアプログラミングの影響を調査した [3]．しかし，性格テストはコストがかかり，偏りのないデータ取得が困難という欠点がある．

そこで，我々は開発者の活動履歴を用いて開発者の特徴を分析する．活動履歴を用いることで，開発者個人に対する分析を行うことができる．さらに，データ取得が容易で，偏りのないデータが得られる．

分析対象のデータとして，我々は GitHub [*1] で活動している開発者の活動履歴を用いる．GitHub は，バージョン管理システム Git を使用するソフトウェア開発プロジェクトのための共有ウェブサービスである．GitHub API から，開発者が行ったコーディングやコメントなどの活動履歴を取得できる．我々は先の研究で GitHub API から取得したデータを用いて開発者の活動履歴の分析を行い，開発者には異なる特徴があることを明らかにした [2]．

本稿では，開発者の活動履歴を用いて開発者のクラスタリングを行った．分析の結果，GitHub で活発な OSS プロジェクトである homebrew と node には迅速・議論型，迅速・総合型，悠然・総合型などの開発者が存在することを明らかにした．

2. 活動履歴の分析

2.1 分析対象の開発者

開発者を分類するため，分析対象とする開発者の選択を行った．活発な OSS プロジェクトで活動する開発者の分析を行うため，コミット数がともに 9,500 回を超えている homebrew [*2] と node [*3] の 2 つのプロジェクトを分析対象プロジェクトとし，そこから開発者を選択した．

homebrew には 3,395 人，node には 477 人の開発者が参加しているが，開発の中心となっている活発な開発者の特徴を明らかにするため，コミット数が 100 を超える開発者を分析対象とし，homebrew から 13 人（h1～h13），node から 10 人（n1～n10）の計 23 人の開発者を分析対象とした．

2.2 収集データ

GitHub API を用いて開発者が行った活動履歴を収集する．GitHub には開発者が行ったコーディングやコメントなどの活動イベントの履歴が記録されている．収集するデータは，開発者が行ったイベントの種類と，そのイベントを行った日時である．GitHub API で取得できる活動履歴は，各開発者ごとに取得できる上限が直近の 300 イベントであるため，取得可能である 300 イベントに対して分析を行った．収集したイベントは，分析対象プロジェクトに対して行っただけではなく，開発者が携わっているすべてのプロジェクトに対して行ったイベントである．開発者の活動履歴は 2013 年 12 月 27 日に収集した．

今回 GitHub API から収集するイベントは，コーディング関係の Create，Delete，Fork，Push，PullRequest と，コメント関係の CommitComment，PullRequestReviewComment，IssueComment，バグや問題を報告する issue と，リポジトリの動向を観察する watch の 10 種類とした．

また，開発者がイベントを行った日時のデータから，開発者が 300 イベントを行うのに要した期間求める．300 イベントを短期間で行う開発者は迅速に活動しており，長期間で行う開発者は緩やかに活動しているといえる．23 人の開発者が 300 イベントを行うのに要した最少日数は 8 日，最多日数は 659 日で，中央値は 51 日である．

3. 開発者の分類

3.1 手法

活動履歴のクラスタリング結果から，23 人の開発者を分類する．クラスタリングには，クラスタ間類似度の解釈が容易なウォード法による階層化クラスタリングを採用する．データは収集した 10 種類のイベントの実施回数と，直近の活動期間の，計 11 種類を使用する．

3.2 結果

階層化クラスタリングで得られた樹形図より，開発者を分類する．**図 1** に樹形図を示す．樹形図は開発者を大きく 2 つのクラスタに分けている．これらのクラスタに対し詳細な分析を行うため，図 1 の破線部でクラスタを区切り，計 5 つのクラスタに対し分析を行う．

概要：各クラスタに属する開発者の活動履歴を分析する．クラスタ C は 1 人の開発者しか属していない．この開発者は直近の活動期間が 659 日と，分析対象とした開発者の中央値である 51 日を大きく上回っており，他の開発者

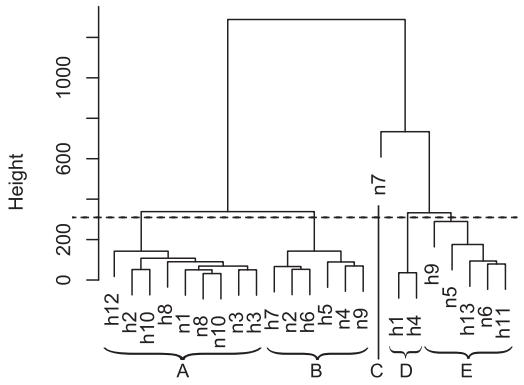

図 1 開発者の階層化クラスタリング
Fig. 1 Hierarchical clustering of developers.

[*1] https://github.com/
[*2] http://brew.sh/
[*3] http://nodejs.org/

と大きく性質が異なったため 1 人だけ別のクラスタに属したのだと考えられる．また，クラスタ D は 2 人の開発者しか属していない．この開発者の活動履歴は Push 回数がそれぞれ 254 回，231 回で，それ以外のイベントはほとんど行っていない．活動内容が他の開発者と性質が異なったため，2 人だけ別のクラスタに属したのだと考えられる．

クラスタ A，B，E については，クラスタに属している開発者が 5 人以上いたため，箱ひげ図を用いて分析を行う．まず，**図 2** に直近の活動期間の箱ひげ図を示す．クラスタ E は他の 2 つのクラスタに比べて，直近の活動期間の中央値が 196 日と長く，緩やかに活動していることが分かる．一方，クラスタ A は 22 日，クラスタ B は 36 日と短く，クラスタ E に比べ迅速に活動していることが分かる．次に，**図 3** に 3 つのクラスタの特徴が著しく表れた，コーディング関係のイベントである Push とコメント関係のイベントである IssueComment の回数の箱ひげ図を示す．クラスタ B は Push 回数の中央値が 98 回，クラスタ E は 88 回と

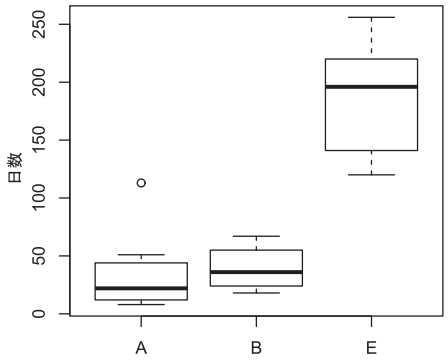

図 2 直近の活動期間．C は 659 日，D はそれぞれ 202 日，189 日である

Fig. 2 Boxplot of the activity periods of developers. The period of developer C is 659 days, and developers in D are 202 and 189 days, respectively.

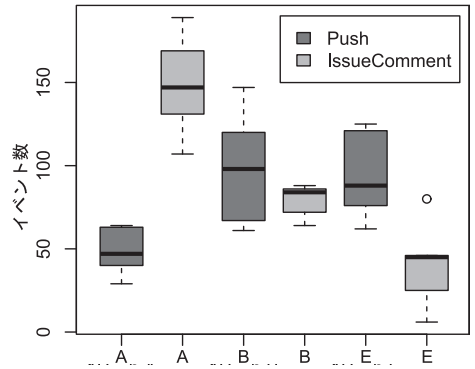

図 3 Push と IssueComment の箱ひげ図．C の Push は 89，IssueComment は 67，D の Push はそれぞれ 254，231，IssueComment は 3，13 である

Fig. 3 Boxplot of the number of Push events and IssueComment events. The number of Push events and IssueComment events of developer C are 89 and 67, and developers in D are 254, 3 and 231, 13, respectively

クラスタ A の 47 回に比べて多く，同じくコーディング関係のイベントである Create 回数においてもクラスタ B の中央値が 29 回，クラスタ E が 21 回，クラスタ A が 10 回と同様の傾向が見られた．IssueComment 回数はクラスタ A の中央値が 147 回と最も多く，次にクラスタ B の 84 回，クラスタ E の 45 回と続く．同じくコメント関係のイベントである PullRequestReviewComment 回数においてもクラスタ A の中央値が 16 回，クラスタ B が 6 回，クラスタ E が 2 回と同様の傾向が見られた．

各クラスタ：樹形図の分析結果から，各クラスタの特徴を**表 1** にまとめる．イベント項目には各クラスタで特に回数が多かったイベントを示す．クラスタ A は直近の活動期間の中央値が 22 日と短く，IssueComment 回数が 147 回とイベントの半数を占めている．よって，迅速に活動しており，主にコメントを行っているため迅速・議論型といえる．クラスタ B は直近の活動期間の中央値が 36 日と比較的短く，特に回数が多かったイベントは Push の 98 回，IssueComment の 84 回，Issues の 23 回となっている．よって，迅速に活動しており，各イベント回数に偏りがないため迅速・総合型といえる．クラスタ C は直近の活動期間が 659 日と非常に長く，特に回数が多かったイベントは Push の 89 回，IssueComment の 67 回，Issues の 47 回となっている．よって，非常に緩やかに活動しており，各イベント回数に偏りがないため長期・総合型といえる．クラスタ D は直近の活動期間の中央値が 196 日と長く，Push 回数が 243 回とイベントの大部分を占めている．よって，緩やかに活動しており，主にコーディングを行っているため悠然・不言実行型といえる．クラスタ E は直近の活動期間の中央値が 196 日と長く，Push 回数の中央値が 88 回，IssueComment 回数が 45 回，Create 回数が 29 回となっている．よって，緩やかに活動しており，各イベント回数に偏りがないため悠然・総合型といえる．

各プロジェクト：各プロジェクトの開発者が，分類したクラスタに占める割合を**図 4** に示す．homebrew では 8 人，node では 7 人と，2 つのプロジェクトの大部分の開発者は迅速型である．また，迅速・議論型の開発者は homebrew に 5 人，node に 4 人，迅速・総合型の開発者は homebrew に 3 人，node に 3 人，悠然・総合型の開発者は homebrew に 3 人，node に 2 人属している．2 つのプロジェクトには，活発にコメントを行う開発者や，活発に総合的なイベントを行う開発者，緩やかに総合的なイベントを行う開発者が属していることが分かる．

4. 議論

4.1 考察

3.2 節では，homebrew と node に参加する 23 人の開発者を 5 つのタイプに分類した．いずれのプロジェクトにも迅速・議論型，迅速・総合型，悠然・総合型の 3 つのタイ

表 1 結果のまとめ
Table 1 The median of the number of events, activity periods, and types of each groups.

クラスタ	期間（中央値）	イベント（中央値）	タイプ
A	22 日	IssueComment（147 回），Push（47 回）	迅速・議論型
B	36 日	Push（98 回），IssueComment（84 回），Issues（23 回）	迅速・総合型
C	659 日	Push（89 回），IssueComment（67 回），Issues（47 回）	長期・総合型
D	196 日	Push（243 回）	悠然・不言実行型
E	196 日	Push（88 回），IssueComment（45 回），Create（29 回）	悠然・総合型

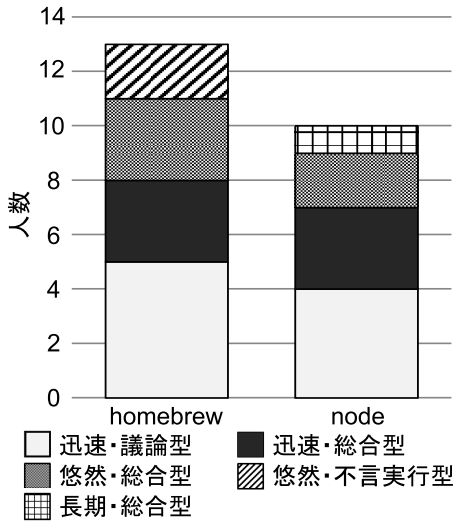

図 4 プロジェクトごとの開発者の内訳
Fig. 4 The composition of the existing types of developers.

プの開発者が属している．このことから，OSS プロジェクトには同じ特徴を持つ開発者が存在する可能性が提示できる．今回は 2 つの OSS プロジェクトに対してのみ開発者の分類を行ったが，分析対象プロジェクトを増やすことで，活発な OSS プロジェクトに参加する開発者の特徴が明らかにできると考えられる．また，活発でない OSS プロジェクトも分析対象とすることで，それぞれのプロジェクトに参加する開発者の特徴が比較でき，そこから OSS プロジェクトの成功に必要な開発者が明らかにできると考えられる．

4.2 妥当性への脅威

GitHub API では，開発者の活動履歴は直近の 300 イベントしか収集できないため，直近の 300 イベントのみに対して分析を行っている．しかし，開発者によっては 300 イベントを数週間，または数日で終えてしまうものもいるため，直近の 300 イベントのみでは十分な分析が行えるとはいえない．さらに，活動頻度に波がある開発者も存在すると考えられる．迅速型として分類された開発者でも，今回取得したデータの中でのみ迅速であった可能性もある．また，2 つの OSS プロジェクトにそれぞれ参加する 23 人の開発者に対してのみ分析を行っているため，得られた分析結果は一般化されたものではない．そのため，今後の課題として複数の OSS プロジェクトから一定期間ごとにデータを収集し，より多くの開発者の長期間のデータを分析する必要がある．

5. おわりに

本稿では，OSS プロジェクトに参加する開発者を GitHub API から取得した活動履歴を用いてクラスタリングし，分類した．クラスタリングは活発な OSS プロジェクトである homebrew と node に参加する開発者の活動履歴を用いて行った．分析の結果，活動内容や活動の活発さから開発者を迅速・議論型，迅速・総合型，長期・総合型，悠然・不言実行型，悠然・総合型の 5 つのクラスタに分類できた．2 つのプロジェクトには，活発にコメントを行う開発者や，活発に総合的なイベントを行う開発者，緩やかに総合的なイベントを行う開発者が属していた．

OSS プロジェクトに参加する開発者の構成を調べることで，成功している OSS プロジェクトはどのような開発者で構成されているかを明らかにできる可能性がある．

謝辞 本研究は JSPS 科研費 26540029 の助成を受けた．

参考文献

[1] Ihara, A., Ohira, M. and Matsumoto, K.: An Analysis Method for Improving a Bug Modification Process in Open Source Software Development, *Proc. IWPSE-Evol '09*, pp.135–144, ACM (2009).
[2] Onoue, S., Hata, H. and Matsumoto, K.: A Study of the Characteristics of Developers Activities in Github, *Proc. IWESEP '13*, pp.7–12 (2013).
[3] Salleh, N., Mendes, E., Grundy, J. and Burch, G.S.J.: An empirical study of the effects of conscientiousness in pair programming using the five-factor personality model, *Proc. ICSE '10*, pp.577–586, ACM (2010).
[4] Sfetsos, P., Stamelos, I., Angelis, L. and Deligiannis, I.: An experimental investigation of personality types impact on pair effectiveness in pair programming, *Empirical Softw. Eng.*, Vol.14, No.2, pp.187–226 (2009).
[5] Zhou, M. and Mockus, A.: Does the Initial Environment Impact the Future of Developers?, *Proc. ICSE '11*, pp.271–280, ACM (2011).

尾上 紗野

2013 年奈良女子大学理学部情報科学科卒業．現在，奈良先端科学技術大学院大学修士 2 年．ソフトウェア開発者の活動分析に関する研究に興味を持つ．

畑 秀明 （正会員）

2007 年大阪大学工学部電子情報エネルギー工学科卒業．2009 年同大学大学院博士前期課程修了．2012 年同大学院博士後期課程修了．2013 年奈良先端科学技術大学院大学助教．博士（情報科学）．ソフトウェア不具合予測手法，ソフトウェアリポジトリのマイニングに関する研究に従事．電子情報通信学会，日本信頼性学会，ACM，IEEE 各会員．

松本 健一 （正会員）

1985 年大阪大学基礎工学部情報工学科卒業．1989 年同大学大学院博士課程中退．同年同大学基礎工学部情報工学科助手．1993 年奈良先端科学技術大学院大学助教授．2001 年同大学教授．工学博士．エンピリカルソフトウェア工学，特に，プロジェクトデータ収集/利用支援の研究に従事．電子情報通信学会，日本ソフトウェア科学会，ACM 各会員，IEEE Senior Member．