情報処理学会論文誌 Vol.62 No.4 1076–1085 (Apr. 2021)

READMEファイルの進化に関する実証的分析

亀井 靖高[1,a)] 清水 一輝[1,b)] 柏 祐太郎[1,c)] 佐藤 亮介[1,d)] 鵜林 尚靖[1,e)]

受付日 2020年7月16日, 採録日 2021年1月12日

概要：OSS（Open Source Software）開発プロジェクトにおいて，READMEファイルは一番初めに表示されるドキュメントであり，ソフトウェアを正しく利用するために必要不可欠である．それにもかかわらず，READMEファイルがどのように作成され，修正されているかなど，その進化過程は明らかにされていない．本研究では，READMEファイルの進化過程を明らかにするために，READMEの作成初期から最新バージョンまでの過程を4分割し，各期間でどのような内容が追加されたかや，READMEの変更回数を調査した．GitHubで公開されているプロジェクトを用いた実証実験の結果，READMEファイルの初期バージョンにはプロジェクトがどのようなものなのかを説明する内容や利用方法が記述されることが多く，その後，プロジェクトの状態やプロジェクトへの貢献者など開発者のための情報が追加されることが多いということが分かった．またREADMEファイルの作成には内容が多くなればなるほど，プロジェクト全体の変更に対するREADMEファイルへの変更の割合が高くなっており，平均で3.6%の工数を割いていることが分かった．

キーワード：README，オープンソースソフトウェア開発，実証的研究，GitHub

An Empirical Analysis of the Evolution of README Files

Yasutaka Kamei[1,a)] Kazuki Shimizu[1,b)] Yutaro Kashiwa[1,c)] Ryosuke Sato[1,d)]
Naoyasu Ubayashi[1,e)]

Received: July 16, 2020, Accepted: January 12, 2021

Abstract: In OSS (Open Source Software) development projects, README files are the first document that users and developers involve, and are inevitable to work the product properly. Nevertheless, the evolution process of README files, such as how they are created and modified, has not been still unrevealed. In this study, in order to understand the evolution of README files, we examine what contents were added in each period while dividing README development into four periods. In addition, we study the number and percentage of commits including README to clarify the efforts. The case study on GitHub projects demonstrates that the initial version of README files tends to contain a description of the project and its usage. Afterward, developers append the information of projects for developers on their README files (e.g., project status, project team, contributions). Additionally, on average 3.6% of commits include the modification of README files, in particular, the more content in the README file, the more frequently developers update them.

Keywords: README, open source software development, empirical study, GitHub

1. はじめに

オープンソースソフトウェア（OSS）の高機能・高品質化にともない，OSSはエンドユーザだけでなく製品開発に利用される事例も増えている [5]．たとえば，最新のクラウド基盤のためのソフトウェア（HadoopやOpenStackなど）は，利用しているベンダ企業がそのOSS開発プロジェ

[1] 九州大学
Kyushu University, Fukuoka 819–0395, Japan
[a)] kamei@ait.kyushu-u.ac.jp
[b)] shimizu@posl.ait.kyushu-u.ac.jp
[c)] kashiwa@ait.kyushu-u.ac.jp
[d)] sato@ait.kyushu-u.ac.jp
[e)] ubayashi@ait.kyushu-u.ac.jp

クトに参加し協力しながら開発を行っている [15]．OSS 開発プロジェクトでは，オンライン上に公開されているファイル群（ソースコードやドキュメントなど）に対して開発者が作業を行い，最終的に成果物（ソフトウェアシステム）をリリースする．

GitHub などのソースコードのホスティングサイトでは，最初に README ファイルを作成し，どのようなプロジェクトなのかや，誰がプロジェクトを作成し維持しているのかなどを記述することを推奨している[*1]．そのため，README ファイルは，プロジェクトの開発者や利用者がプロジェクトを閲覧する際，一番初めのページに表示されるようになっており，利用者だけでなく新規開発者にとっても重要な役割を担っている [7]．GitHub のストラテジ部門のバイス プレジデント Brian Doll 氏は「良い README ファイルを持つプロジェクトは最も使われる傾向にある」と主張している [1]．

しかしながら，README ファイルが OSS 開発プロジェクトにおいて重要な役割を担っている一方で，README ファイルは作成や修正に多くの労力が求められるため，必ずしもつねに最新の状態が維持されるわけではない．2017 年に行われた調査 [14] では，GitHub の利用者は README ファイルは重要であると考えているものの，内容が不完全であったり，古いままになったりしているものも多く存在するという課題が指摘されている．

本研究では，そのような課題を持つ REAMDE ファイルがどのように進化しているのか明らかにすることによって，最終的には OSS 開発プロジェクトにおけるドキュメント作成方法の推奨を目標とする．Steinmacher ら [13] は，開発者が OSS へ新規に参加する際の主な障壁として，ドキュメンテーション不足を指摘しているが，本研究で，プロジェクトの発足時期において，README に書くべき内容を明らかにし，OSS 開発プロジェクトにおけるドキュメントの充実が期待できる．

本論文では，目標を達成するための第 1 歩として，README ファイルの変更過程を調査する．具体的には，Prana ら [9] が分析した OSS 開発プロジェクトから，README ファイルを進化過程ごとに取得し，どのような記載内容が追加されているかを目視でカテゴリ分けした．そして，カテゴリ分けした README の記載内容を用いて，次の 3 つの調査課題（以下 RQ：Research Question）を解決する．

RQ1 README ファイルの初期バージョンには何が（どのカテゴリが）書かれているのか．

RQ2 最新バージョンまでの間に，どのカテゴリがどの期間に追加されるのか．

RQ3 README ファイルに対して，どの程度の労力が費やされているのか．

上記の RQ を達成することによって README ファイルの作成の際，作成初期の段階でどれほどの内容が書かれており，どれぐらいの作業工数を割いて最新バージョンの README ファイルを作成しているのかを理解することができると考える．本論文の貢献は，次のとおりである．

- 既存に公開されているデータセットを用いて，README ファイルの進化について定量的に分析を行った点（RQ1-3）
- 最新の予測手法を用いてデータセットを新規に追加し，公開，および，追加分析を行った点（考察）
- 本実験を再現するためのデータセットとスクリプトを公開している点[*2]

ドキュメント作成方法の推奨と本論文の位置付け．我々は OSS 開発プロジェクトにおけるドキュメント作成法の推奨の達成には，(1) README ファイルの進化過程を分析する，(2) 進化過程に基づき，README ファイルのカテゴリ作成・更新の指針案をまとめる，(3) その指針が実際の OSS 開発プロジェクトにおいて有用なものであるかをインタビューなどによって評価する，(4) 系統的文献レビュー [6] を実施し，OSS 開発プロジェクトにおけるドキュメントの種類を体系化する，(5) README ファイル以外のドキュメントについて工程 (1)–(3) を実施し，OSS 開発プロジェクトにおけるドキュメント作成方法を推奨する，という大きく 5 つの工程に取り組む必要があると考える．本論文で対象とする README ファイルの進化過程の調査は，上記工程の (1) に該当する．今後の見通しについては，7 章で述べる．

章構成．2 章では関連研究についての説明をし，3 章では研究で利用したデータセットの説明をする．4 章では調査の内容と結果について説明する．5 章では，追加実験と結果の意義について考察を行う．6 章で妥当性への脅威について述べ，7 章で結論を述べる．

2. 関連研究

本章では，従来研究として，OSS 開発プロジェクトへの導入支援，および．OSS 開発プロジェクトにおけるドキュメント分析に関する研究を述べる．

2.1 OSS 開発プロジェクトへの導入支援

Steinmacher らは，開発者が OSS 開発プロジェクトへの参加する際の主な障壁を明らかにする調査を行った [12]．調査では，新規にプロジェクトに参加する際の障壁が計 58 個ある報告しており，新規参入者の特徴や技術的ハードルなどに起因するものなどが存在する．

彼らは続く研究 [13] で新規参入者をサポートするポー

[*1] https://help.github.com/articles/about-readmes/

[*2] https://github.com/posl/readme_ipsj_replication

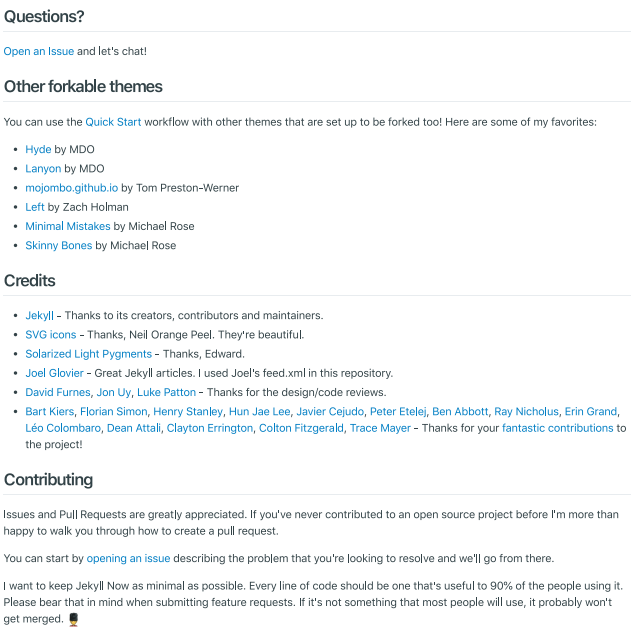
タルの作成および評価を行った．作成したポータルは新規参加者がプロジェクトにどのように貢献すればよいかなどを示し，いくつかの障壁の解決に役立った．しかし，技術的な障壁を解決するものとしては不十分という結果となった．また，あげられている障壁にはドキュメントの問題も含まれおり，本研究により，どのカテゴリから README を書き始めるのが望ましいかなど問題解決の手助けになると考える．

2.2 OSS 開発プロジェクトにおけるドキュメント分析

Moreno ら [8] は，ソフトウェアの情報を記述したドキュメントの 1 つであるリリースノートを自動作成する手法の設計，および，評価を行った．Moreno らは 990 件のリリースノートを目視で確認することでドキュメントに記述すべき項目を選定し，自動生成する手法を提案した．また，提案手法の有用性を実験によって実証的に評価している．Moreno らの研究ではリリースノートを対象としている一方で，本研究では README ファイルを対象としており，ドキュメントの分析という点で共通する部分があるが対象とするドキュメントが異なっている．

池田ら [16] はソフトウェアを説明するドキュメントの作成支援のために，GitHub に登録されている 143,239 件のプロジェクトが公開する README ファイルに記述されている項目の分析を行った．池田らの研究では README ファイルの見出しの単語を分析することによって，見出しとして使用している内容の上位 10 件について使用しているプロジェクト件数と全体のプロジェクトに対する記述率について調査を行った．分析の結果，対象プロジェクトの 30%以上が “install”，“usage”，“license” の項目を記述していることが分かった．見出しを抽出して分析している点では同じだが，本研究では README ファイルの進化について調査している点が異なる．

Prana ら [9] は README ファイルの品質向上と関連情報の発見の効率化を図るために，README ファイルの内容をセクションごとにどのカテゴリに属するかを自動分類し，ラベル付けする手法を提案している．README ファイルに適用された例が**図 1** であり，たとえば “Credit” というセクションには，“who” というラベル付けがされていた．Prana らは 393 件の OSS から取得したデータを用いてラベル分類を実施した．評価実験の結果，提案手法（マルチラベル分類器）は 0.746 の F1 値を達成し，有用性の評価に参加したソフトウェア開発者の多くは Prana らの手法が有用であると回答している．Prana らの研究では最新バージョンの README ファイルを予測・分類しているのに対して，本研究では README ファイルの進化について調査している点が異なる．



Questions?

Open an Issue and let's chat!

Other forkable themes

You can use the Quick Start workflow with other themes that are set up to be forked too! Here are some of my favorites:

- Hyde by MDO
- Lanyon by MDO
- mojombo.github.io by Tom Preston-Werner
- Left by Zach Holman
- Minimal Mistakes by Michael Rose
- Skinny Bones by Michael Rose

Credits

- Jekyll – Thanks to its creators, contributors and maintainers.
- SVG icons – Thanks, Neil Orange Peel. They're beautiful.
- Solarized Light Pygments – Thanks, Edward.
- Joel Glovier – Great Jekyll articles. I used Joel's feed.xml in this repository.
- David Furnes, Jon Uy, Luke Patton – Thanks for the design/code reviews.
- Bart Kiers, Florian Simon, Henry Stanley, Hun Jae Lee, Javier Cejudo, Peter Etelej, Ben Abbott, Ray Nicholus, Erin Grand, Léo Colombaro, Dean Attali, Clayton Errington, Colton Fitzgerald, Trace Mayer – Thanks for your fantastic contributions to the project!

Contributing

Issues and Pull Requests are greatly appreciated. If you've never contributed to an open source project before I'm more than happy to walk you through how to create a pull request.

You can start by opening an issue describing the problem that you're looking to resolve and we'll go from there.

I want to keep Jekyll Now as minimal as possible. Every line of code should be one that's useful to 90% of the people using it. Please bear that in mind when submitting feature requests. If it's not something that most people will use, it probably won't get merged.

図 1 カテゴリ分類された README の例

Fig. 1 Example of README files categorized by the previous study [9].

3. データセット

本章では，4 章で述べる調査のために，先行研究から取得したデータについて記述する．

OSS 開発プロジェクト．本実験では Prana らの README ファイルの自動マルチラベル分類器の研究で使用された OSS 開発プロジェクトをデータセットとして利用する．Prana らの研究では GitHub 内の OSS 開発プロジェクトをランダムに 1,193 件クローンし，その中から README ファイルにタイトルしか書かれていないと思われるものといった実験に不適切なプロジェクトを除外した 393 件分を実験に利用している．

本研究では，README ファイルの進化を研究するため開発履歴が必要である．そのため，393 件の OSS 開発プロジェクトをクローンしたところ，すでに GitHub 上に存在していないプロジェクトが含まれており，実際にクローンできたプロジェクトは 365 件であった．それらのプロジェクト 365 件のプロジェクトをデータセットとした．

カテゴリー一覧．Prana ら [9] は，最新バージョンの README ファイルについて，各見出しごとを目視によってカテゴリ分けし，複数の著者によって分類分けの正しさを検証している．Prana らはそのカテゴリ分けしたものを，自動分類の性能評価に用いていた．本研究では，目視によって分類された各見出しを調査に用いる．各見出しは，**表 1** に示すカテゴリ一覧のうち，少なくとも 1 つが割り当てられている．なお，見出しは，マークダウン記法 [3] に基づき，“#” で記述された行が該当する．

表 1 カテゴリ一覧 [9] から引用
Table 1 List of categories cited from the previous study [9].

category	example section heading
what	introduction, project, background
why	advantages of the project, comparison with related work
how	getting started, how to run, installation, requirements, platforms, downloads, setup
when	project status, versions, project plans, roadmap
who	project team, community, mailing list, contact, acknowledgement, license,
references	API documentation, getting support, feedback, more information, translations, related projects
contribution	contributing guidelines
other	—

4. 調査と結果

README ファイルの進化の過程を調査するために 4 つの調査課題を設定し，調査を行った．この章では，その方法と結果について記述する．

4.1 RQ1：README ファイルの初期バージョンには何が（どのカテゴリが）書かれているのか

動機． README ファイルの進化についての知見を得る第一歩として，README ファイルの初期バージョンに何が書かれているのか調査する．最新バージョンで書かれているカテゴリのすべてが初期バージョンから記述されていることは少ないのではないかと考え，どのような内容が記述されているのかの知見を得ることで初期バージョンに記述すべき項目を議論する．

アプローチ． 初期バージョンの README に記載されている内容を明らかにするために，初期バージョンの README ファイルに出現する見出しを抽出し，抽出した見出しを目視およびカテゴリ分けを行う．**図 2** の左側には，アプローチの概要を示し，アプローチの詳細を以下に記載する．

まず最初に，プロジェクトのコードリポジトリをクローンし，最新バージョンのソースコードを用意した．次に，README ファイルに対して “git log” コマンドを実施し，初期バージョンのコミット ID を取得し，“git checkout” コマンドで初期バージョンの README ファイルを取得した．

次に，初期バージョンの README ファイルから機械的に見出しを抽出した．GitHub の README ファイルは

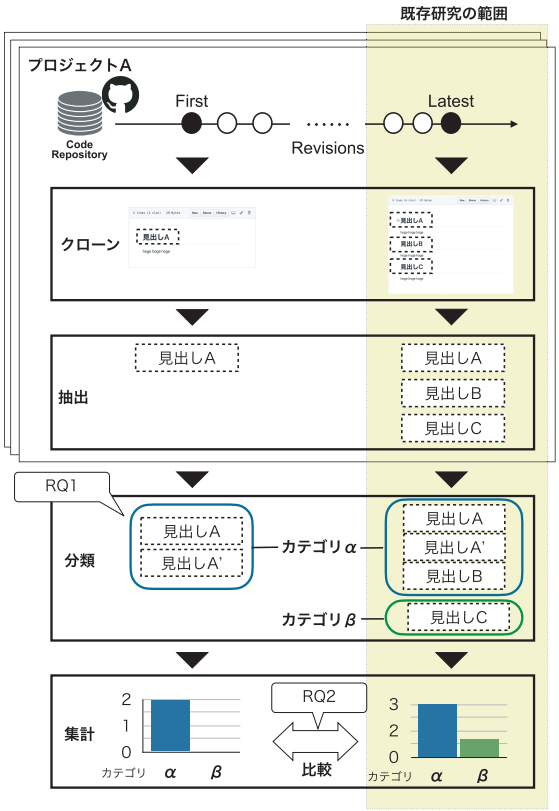


図 2 初期バージョンのカテゴリ分類と追加カテゴリの調査
Fig. 2 Overview of our investigation into the difference between the categories in the first version and in the latest version.

マークダウン記法を利用して記述されているため，見出しを表す “#” から始まる行を走査することで見出しを抽出することができる．

最後に，調査対象の全プロジェクトから収集した初期バージョンの見出しをそれぞれ，Prana らのカテゴリのうちどれに該当するかを目視で分類した（分類手順は Prana らの研究に従う）．なお，抽出した見出しが，Prana らの分類に出現した見出しであれば，Prana らの分類と同じカテゴリに分類している．また，Prana らが作成したカテゴリのいずれにも該当しない見出しは，“no category” に分類した．

結果と考察． 調査の結果，**表 2** の左側（“初期” 列）に示した結果となった．初期バージョンで最も多かったのは “what” で 175 件であり，これは 1 つ以上のカテゴリを含む README ファイル 268 件のうち，65%を占める割合である．次に多いのは，“no category”（どのカテゴリも含まない）で 161 件であった．これは調査した 365 件の 44%を占めている．

本研究では最新バージョンの見出しに初期バージョンのカテゴリが依存する形になっているので，見出しが変更されると初期バージョンがカテゴリを含まないことになるので，161 件すべてが何も含まれていないと考えるのは適切ではないが，ほとんどが README の作成の際に最初は無記入，あるいはプロジェクト名あるいは適当な単語のみの

表 2 初期と最新バージョンでのカテゴリ件数および増分
Table 2 Number of categories in the initial version and the latest version as well as its increase.

カテゴリ	初期 (RQ1)	追加件数（RQ2）				最新	追加合計
		第 1 区間	第 2 区間	第 3 区間	第 4 区間		
what	175	+ 64 (+ 36.6%)	+ 42 (+ 17.6%)	+ 27 (+ 9.6%)	+ 43 (+ 14.0%)	351	+ 176
why	35	+ 12 (+ 34.3%)	+ 20 (+ 42.6%)	+ 9 (+ 13.4%)	+ 16 (+ 21.1%)	92	+ 57
how	144	+ 76 (+ 52.8%)	+ 35 (+ 15.9%)	+ 38 (+ 14.9%)	+ 26 (+ 8.9%)	319	+ 175
when	27	+ 13 (+ 48.1%)	+ 11 (+ 27.5%)	+ 11 (+ 21.6%)	+ 15 (+ 24.2%)	77	+ 50
who	73	+ 40 (+ 54.8%)	+ 31 (+ 27.4%)	+ 17 (+ 11.8%)	+ 32 (+ 19.9%)	193	+ 120
references	72	+ 55 (+ 76.4%)	+ 38 (+ 29.9%)	+ 24 (+ 14.5%)	+ 31 (+ 16.4%)	220	+ 148
contribution	32	+ 20 (+ 62.5%)	+ 13 (+ 25.0%)	+ 15 (+ 23.1%)	+ 19 (+ 23.8%)	99	+ 67
other	3	+ 4 (+133.3%)	+ 11 (+157.1%)	+ 1 (+ 5.6%)	+ 6 (+ 31.6%)	25	+ 22
no category	161	-	-	-	-	-	-

見出しで作成されると考えられる．

README ファイルの初期バージョンでは “what” が記述されていることが多い．

4.2 RQ2：最新バージョンまでの間に，どのカテゴリがどの期間に追加されるのか

動機．RQ1 で README ファイルの初期バージョンで出現するカテゴリを調べた．本 RQ では，初期バージョンには含まれないものの，開発が進むにつれて追加されるのはどのカテゴリなのかを調査する．RQ1 に加え，開発の途中で追加されるカテゴリを調査することで，開発の過程で求められる README ファイルの内容に関する知見が得られるのではないかと考える．また，カテゴリの追加が，どの期間で行われているのかを調べ，README ファイル作成におけるカテゴリ作成の順序の指針に関する知見を得る．

アプローチ．RQ1 で行ったように，初期バージョンと最新バージョンのカテゴリを比較し，初期バージョンには含まれないが，その途中で追加され，最新バージョンに含まれるカテゴリを調査した．

“git log” コマンドを用いて README ファイルに対するコミット数を抽出し，README ファイルの初期バージョンから最新バージョンまでをコミット数に基づいて等間隔で 4 つの区間に分け，各カテゴリがどの期間で追加されているのか調査した．初期バージョンから最新バージョンまでの各バージョンに対して “git checkout” コマンドを用いて，どういった見出しが追加されたかを取得し，その後，RQ1 と同様の方法で見出しに対するカテゴリ分類を行った．分類した README ファイルのカテゴリを比較することで，4 区間の追加カテゴリを調査した．

結果と考察．結果を表 2 の中央と右（“追加件数” 列と “追加合計” 列）に示す．追加件数を調べた場合，途中で追加されるカテゴリとして最も多かったのが “what” の 176 件であった．その次に “how” が，“what” に僅差の 175 件であった．特に，“what” と “how” は第 1 区間において多く追加されている（“what”：64 件，“how”：76 件）．つまり，“what” と “how” は，初期バージョンに限らず（RQ1），開発初期に記載される項目であることが分かった．

次に，前区間と比べて相対的に大きく増加するカテゴリを調べた．たとえば “what” の場合，初期に 175 件あり，第 1 区間で 64 件増えているので，36.6%（= 64/175）の増加である．各区間で最も増加率が大きかったカテゴリは，第 1 区間では “references” が最も増加しており，第 2 区間では why，第 3 区間では “contribution”，第 4 区間では “when” が最も増加していた．この結果から，OSS 開発プロジェクトでは，比較的初期の頃に，API document やフィードバックなどの参照情報（references）といったプロジェクトを利用するうえで必要な情報を記述し，中期の頃に，他のプロジェクトと比較した利点など（why）やプロジェクトの貢献者（contirbution）といったプロジェクトの意義や支援者の情報が記述される傾向があると考えられる．また，後期の頃には，プロジェクトの状態といった将来を見据えた情報（when の roadmap）が記述される傾向があることが分かった．

README ファイルに途中で追加されるカテゴリとしては “what” が最も多く，各区間での増加割合で調べると，第 1 から第 4 区間ではそれぞれ “references”，“why”，“contribution”，“when” が最も増加していた．

4.3 RQ3：README ファイルに対して，どの程度の労力が費やされているのか

動機．README ファイルの作成や編集に，プロジェクト全体としてどの程度の労力が費やされるかを調査する．README ファイルに関する研究において，進化過程に関する研究は存在しておらず，開発の過程において README ファイルにどの程度の工数を費やされているかは明らかになっていない．そのため，本 RQ を調査することで，プロジェクト全体として README ファイルに対してどの程度の工数を割り当てる必要があるのかの目安を提供できると

表 3 コミット数と追加カテゴリ数の関係
Table 3 Relationship between the number of the introduced categories and frequency of commits.

追加カテゴリ数	プロジェクト数	README のコミット数の平均	プロジェクト全体のコミット数の平均	README を含むコミットの割合
1	57	16	1,562	1.1
2	52	27	3,103	0.9
3	70	25	1,406	1.8
4	41	51	1,027	5.0
5	25	80	1,735	4.6
6	21	154	3,898	4.0
7	3	58	2,113	2.7
8	1	80	922	8.7
平均	33.8	61.4	1970.8	3.6

考える．

アプローチ．従来研究 [11] を参考に，Git リポジトリに対するコミットを労力の単位とし，README ファイルに対するコミットを README ファイルへの労力として計測する．具体的な計測方法は，各プロジェクトの README ファイルに対して，“git log --oneline -- README.md |wc -l” のコマンドを実行し，実行結果の行数をコミットの数として用いた．

また，初期バージョンに対して追加されたカテゴリが大きいほど，コミットの数も比例して大きい，つまり，多くの内容を記述している README ファイルはその分，多くの工数が割けられていると考えられる．たとえば，初期バージョンでは 1 つのカテゴリ（たとえば，“what”）があり，その後，5 つのカテゴリが追加されたプロジェクトは，1 つのカテゴリしか追加されなかったプロジェクトより多くのコミットがされている可能性が大きい．そのため，追加されたカテゴリ数ごとの README ファイルに対するコミット回数を取得し，追加カテゴリ数ごとに分けてコミット数も調査した．

結果と考察．README ファイルに対するコミット数に関する結果を**表 3** に示す．追加カテゴリ数ごとの該当するプロジェクト件数と，README ファイルに対するコミット数，および，プロジェクト全体のコミット数の平均，そして README ファイルに対するコミットの割合を示している．

表 3 に示すように，追加カテゴリ数が多くなるにつれて，README ファイルに対するコミット数，および，プロジェクト全体のコミット数に対する README ファイルへのコミット数の割合も増加している傾向が見られる．

また，追加カテゴリ数ごとの README ファイルへのコミット数の割合の平均をとると 3.6%であった．README ファイルに対してどの程度のカテゴリを追加するかにもよるが，大まかな指針として README ファイル作成における工数の割合は約 3.6%であると考えられる．

> 追加カテゴリ数が多くなるほど README ファイルのコミット数が増える．また，README ファイル作成にかかる工数の割合は 3.6%程度である．

5. 追加分析とまとめ

5.1 追加分析

動機．本実験では，Prana らが利用したプロジェクトと同じものをクローンし，初期バージョンや開発途上のデータを取得した．しかし，今回利用したプロジェクトは GitHub における一部のプロジェクトであり，これらが GitHub におけるプロジェクトの傾向を示しているかは疑問が残る．そこで，本章では異なるデータセットでも同様の結果が得られるかを確認する．異なるデータセットで同様の傾向が得られるかを確かめることは，結果の一般性向上に有用であると考える．

アプローチ．追加実験では，新しく GitHub から無作為に 50 リポジトリを選択およびクローンを行った．クローンしたプロジェクトから抽出できたセクションは計 665 件であった．

そして，3 章の方法と同様に，第 2 著者がセクションごとに目視で表 1 のカテゴリに従った分類を行った．その後，全セクションを Prana らの自動分類器に入力し分類を行って，分類結果が目視の結果と同一のものはそのままの分類結果を用いた．分類結果が異なるものについては，著者ら以外の者に独立して分類してもらい，多数決（2 名分の目視結果と自動分類器の結果）で決める．多数決で決められない（つまり，すべての結果が異なる）場合は，分類に携わった 2 名が議論によって解決する．

新規に作成したデータセット（3 章で抽出したデータを含まない，異なるデータセット）を用いて，4 章で実施した RQ1 から RQ3 と同じ方法で実験を行い，同様の傾向が得られるかを確認する．なお，本章で行う RQ は ARQ（Additional Research Question）と記載し，4 章で実施し

表 4 追加分析：初期と最新バージョンでのカテゴリ件数および増分
Table 4 Additional analysis: Number of categories in the initial version and the latest version as well as its increase.

カテゴリ	初期 (RQ1)	追加件数（RQ2）				最新	追加合計
		第 1 区間	第 2 区間	第 3 区間	第 4 区間		
what	29	+ 4 (+ 13.8%)	+ 2 (+ 6.1%)	+ 4 (+ 11.4%)	+ 8 (+ 20.5%)	47	18
why	8	+ 0 (+ 0.0%)	+ 5 (+ 62.5%)	+ 1 (+ 7.7%)	+ 2 (+ 14.3%)	16	8
how	33	+ 6 (+ 18.2%)	+ 0 (+ 0.0%)	+ 4 (+ 10.3%)	+ 6 (+ 14.0%)	49	16
when	6	+ 5 (+ 83.3%)	+ 1 (+ 9.1%)	+ 2 (+ 16.7%)	+ 4 (+ 28.6%)	18	12
who	22	+ 1 (+ 4.5%)	+ 1 (+ 4.3%)	+ 1 (+ 4.2%)	+ 5 (+ 20.0%)	30	8
references	23	+ 7 (+ 30.4%)	+ 2 (+ 6.7%)	+ 3 (+ 9.4%)	+ 3 (+ 8.6%)	38	15
contribution	13	+ 1 (+ 7.7%)	+ 3 (+ 21.4%)	+ 2 (+ 11.8%)	+ 4 (+ 21.1%)	23	10
other	1	+ 0 (+ 0.0%)	+ 0 (+ 0.0%)	+ 0 (+ 0.0%)	+ 1 (+ 100%)	2	1
no category	12	-	-	-	-	-	-

た Research Question（RQ）と区別をする（例：ARQ1）．

目視による分類結果．全 665 セクションに適用した結果，第 2 著者の分類と分類器の結果が一致したセクション数が 426（64.1%），異なるセクション数が 239（35.9%）となった．異なるセクションのうち著者ら以外の者を加えて多数決で解決したセクション数が 131（19.7%），議論で解決したセクション数が 108（16.2%）となった．

なお，議論で解決したセクションで最も多かった組合せが “reference” と “how” による不一致であり，議論で解決したセクションのうちの約 50%を占めた[*3]．本パターンでは，API の使用方法に関する説明文とサンプルコードが 1 つのカテゴリ内に書かれており，そのカテゴリに対して 1 名の担当者が “reference”，もう 1 名の担当者が “how”，分類器はそれ以外を付与している場合に発生した．最終的に，担当者らは議論を通して両方のラベルを付与することを決定した．

データセットの傾向．4 章の実験と本章の追加実験で用いた GitHub 上のプロジェクトの違いについて述べる．規模（論理行数）の中央値は，4 章の実験と本章の追加実験それぞれで，2,194 と 4,378 で，コミット数の中央値はそれぞれ 107 と 604 であった．また，選ばれたプロジェクトの主使用プログラム言語トップ 3 は，4 章の実験では JavaScript，Java，Ruby であったのに対して，本章の追加実験では Ruby，JavaScript，C であった．

結果（ARQ1：README ファイルの初期バージョンには何が（どのカテゴリが）書かれているのか）．表 4 の左側に初期バージョンと最新バージョンでのカテゴリ件数を示す．追加データセットで最も多かったものは “how” の 33 件であり，“what” の 29 件，“references” の 23 件，“who” の 22 件，が続いた．これらは RQ1 でも上位に位置していたが，“how” が ARQ1 で大幅に多くなり，RQ1 で最も多かった “what” を上回っている．また，RQ1 では，2 番めに多く 44%を占めていた “no category” が 12 件（24%）と大幅に少なかった．

結果（ARQ2：最新バージョンまでの間に，どのカテゴリがどの期間に追加されるのか）．表 4 の右側に追加合計を示す．RQ2 では，第 1 区間では，“what” が最も追加されるカテゴリであったが，ARQ2 でも同様に，“what” が最も多く 18 件であった．ただし，“what” が相対的に最も追加された期間は，RQ2 の第 1 区間と異なり，ARQ2 では第 4 区間（+20.5%）であった．

相対的に最も増加したカテゴリを各期間ごとに確認すると，第 1 区間は “when” の 83.3%（RQ2 では “when” は 5 位で，“references” が 1 位），第 2 区間は “why”（RQ2 でも “why” が 1 位），第 3 区間は “when”（RQ2 では “‘when” は 2 位で，“contribution” が 1 位），第 4 区間は “when”（RQ2 でも “when” が 1 位）であった[*4]．第 1 区間を除き，相対的に最も増加したカテゴリは類似していた．

結果（ARQ3：最新バージョンまでの間に，どのカテゴリがどの期間に追加されるのか）．

表 5 に，README ファイルの変更を含むコミット数の平均を示す（追加カテゴリ数別）．まず，RQ3（表 3）では，プロジェクトにおける全コミットのうち，README ファイルを含むコミットの平均割合は 3.6%であった．一方，ARQ3 では，追加カテゴリ数 “7” で出現する 1 プロジェクトが 17.4 と大きく上振れていたため，その平均は 6.3%と大きく異なった．しかし，その 1 プロジェクトを除くと，平均 4.5%とおおむね類似した増加傾向が見られた．

RQ と ARQ を比較した結果，すべての結果が同様の結果を示したとはいえないものの，初期に出現しやすいカテゴリや全コミットのうち README の変更割合など多くの項目において同様の傾向が観察された．

[*3] 今回の目視分類でも Prana ら [9] と同様，少なくとも 1 つのカテゴリを割り当てたため，担当者 1 が “reference” を付与，担当者 2 が “reference” と “how” を付与などの組合せも不一致としても含む．

[*4] RQ2 と同様に other は比較の対象から外している．

表 5 追加実験：コミット数と追加カテゴリ数の関係
Table 5 Additional analysis: Relationship between the number of the introduced categories and frequency of commits.

追加カテゴリ数	プロジェクト数	README のコミット数の平均	プロジェクト全体のコミット数の平均	README を含むコミットの割合
1	9	47	640	7.3
2	9	46	1,222	3.8
3	7	39	2,371	1.6
4	3	84	1,285	6.5
5	3	33	1,479	2.2
6	1	31	553	5.6
7	1	232	1,330	17.4
8	0	0	0	0
平均	4.7	73.1	1268.6	6.3

*カテゴリ数 8 は平均の計算対象から除いている

5.2 結果のまとめと意義

調査の結果によって，プロジェクトでは平均 3.6%と少なくないエフォートを README ファイルに配分しており，特にカテゴリを多く追加するプロジェクトほど，README の更新回数は増え，全体を占める README のエフォートの割合は大きかった．また，追加されるカテゴリは，主に序盤（第 1 区間）に追加されており，他区間が 142〜201 件追加されたのに対して，序盤では 284 件のカテゴリが追加されていた．

序盤に追加されたカテゴリで特に多いカテゴリは “what” や “how” で，初期と序盤をあわせて半数以上のプロジェクトが README に記載していた（全 393 プロジェクト中，“what” が 239 プロジェクト，“how” が 220 プロジェクト）．また “references” も第 1 区間に追加されることから，新規プロジェクトが初期に意識するべき項目は，「当該ソフトウェアはどういったもので，どのように使えるのか」などのユーザを定着させる要素が必要であると考えられる．

序盤以降は，“when” や “who”，“why”，“contribution” など，プロジェクトに関する情報が逐次更新されており，新規開発者がプロジェクトに参加しやすくしたり，既存開発者との情報共有や敬意を伝えたりするなど，プロジェクトを持続的に発展させるための README 活用方法だと考えられる．

このように，README の活用方法は使い方だけを書くべきものではなく，プロジェクトの繁栄にとって重要である可能性がある．README に書くべき内容により，どのようにユーザや開発者にどのような影響（スター数や参加開発者数など）を与えるかは今後の課題とする．

6. 妥当性への脅威

本研究の調査結果には以下の妥当性に対する脅威が存在していることに注意されたい．

内的妥当性：本研究では Prana らの方法と同様に，2 名の担当者がそれぞれ独立して，GitHub の README ファイルのセクションを目視で確認しながら分類した．2 名の担当者が付与したラベルが同じでない場合は，議論を通してラベルを決定することで，誤ったラベルが付与されないようにしている．2 名の担当者は，ソフトウェアの開発経験がそれぞれ 5 年と 6 年であり，一定のソフトウェア開発の知識はあるものの，OSS ソフトウェア開発の README ファイルの更新経験は深くはない．そのため，すべてのプロジェクトにおける固有のコンテクストを把握することは難しく，また，セクション内の文章のみで分類しているため，誤ったラベルを付与している可能性が存在する．

外的妥当性：本研究では GitHub で公開されているプロジェクトのうち，365 件のプロジェクトを無作為に抽出して調査を行った．さらに考察では一般性の向上を目的として，再度 50 件を GitHub から無作為に抽出および同様の調査を実施し，類似した結果が得られることを確認した．5 章の追加分析で収集したデータセットの特徴が必ずしも Parna らの 365 件の特徴と一致するものではないものの，5.1 節で述べるよう，同様の傾向が得られたことから，結果の一般性は向上したと考える．

ただし，追加実験の調査対象は GitHub プロジェクトであり，多くはボランティアベースのソフトウェア開発である [10]．OSS プロジェクトには，企業の開発者が主に開発するオープンソースソフトウェア（OpenStack など）[2] や，ボランティアの開発者と雇用された開発者が混在しているようなプロジェクト（Firefox など）が存在する [4]．雇用された開発者が存在するプロジェクトの README とボランティアベース以外のプロジェクトでは README の進化過程やそのエフォートは異なる可能性がある．

構成概念妥当性：本論文におけるカテゴリ分類では，Prana らの研究に従って，“what” や “why” などに分類し，それらが各期間で存在するか否かを判定している．しか

し，分類の粒度も大きく，内容が変更されていても同じカテゴリであれば，“当該カテゴリが存在する” とのみ判定している．したがって，各カテゴリがどのように進化したかまでは追跡していない．各カテゴリの更新回数など，詳細な進化過程の分析は今後の課題とする．

また，カテゴリ分類時に見出しごとにセクションとして分けて調査を行ったが，一般的な README の記載方法であるマークダウン記法（“#” から始まる）[3] を利用して抽出を行った．しかし，開発者が必ずしもこの記法に則って README を作成しているとは限らず，分類の対象から外れてしまった可能性も存在する．

7. まとめと今後の課題

本論文では README ファイルの初期バージョンからの進化に着目して 365 件のプロジェクトに対して調査を行った．GitHub で公開されているプロジェクトを用いた実証実験の結果，README ファイルの初期バージョンにはプロジェクトがどのようなものなのかを説明する内容や利用方法が記述されることが多く，その後，プロジェクトの状態やプロジェクトへの貢献者など開発者のための情報が追加されることが多いということが分かった．また，README ファイルの作成には含まれるカテゴリが多くなればなるほど，プロジェクト全体の変更に対する README ファイル変更を含む割合が高くなっており，平均で 3.6%の工数を割いていることが分かった．

1 章の「ドキュメント作成方法の推奨と本論文の位置付け」で述べたように，本論文ではドキュメント作成方法の推奨に向けて考えられる 5 つの工程のうちの，工程 (1) に取り組んだ．また，5.2 節でのまとめは，README ファイルのカテゴリ作成・更新の指針案のまとめにあたる工程 (2) の一部になると考える．今後は RQ3 で行った調査の複数カテゴリを持つ README ファイルはどのような順番で追加されるのについてさらに詳しく調査したり，ソースコードやマニュアル内で頻出する単語と README ファイルの関連性などに着目して調査し，記述方法の推奨・評価である工程 (3)，さらにドキュメントの一般化に向けて工程 (4)，(5) に取り組むことで，OSS 開発プロジェクトにおけるドキュメントの記述方法の提案を目指す．

謝辞 本研究の立ち上げに際して，調査や分析プログラムの実装に協力してくれた佐渡島悠樹氏に感謝の意を表する．本研究の一部は，中島記念国際交流財団による助成，JSPS 科研費 JP18H03222 および JSPS・国際共同研究事業による助成を受けた．

参考文献

[1] Begel, A., Bosch, J. and Storey, M.D.: Social Networking Meets Software Development: Perspectives from GitHub, MSDN, Stack Exchange, and TopCoder, *IEEE Software*, Vol.30, No.1, pp.52–66 (2013).

[2] Bitergia: available from ⟨https://blog.bitergia.com/2013/04/04/companies-contributing-to-openstack-grizzly-analysis⟩ (accessed 2020-06).

[3] Github guides: available from ⟨https://guides.github.com/features/mastering-markdown/⟩ (accessed 2020-06).

[4] Hertel, G., Niedner, S. and Herrmann, S.: Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel, *Research Policy*, Vol.32, No.7, pp.1159–1177 (2003).

[5] IDC Japan:：2016 年度国内オープンソースソフトウェア利用実態調査結果 (2016).

[6] Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J. and Linkman, S.: Systematic Literature Reviews in Software Engineering – A Systematic Literature Review, *Information and Software Technology*, Vol.51, No.1, pp.7–15 (2009).

[7] Mens, T. and Goeminne, M.: Analysing the evolution of social aspects of open source software ecosystems, *Proc. 3rd International Workshop on Software Ecosystems*, pp.1–14 (2011).

[8] Moreno, L., Bavota, G., Penta, M.D., Oliveto, R., Marcus, A. and Canfora, G.: ARENA: An Approach for the Automated Generation of Release Notes, *IEEE Trans. Softw. Eng.*, Vol.43, No.2, pp.106–127 (2017).

[9] Prana, G.A.A., Treude, C., Thung, F., Atapattu, T. and Lo, D.: Categorizing the Content of GitHub README Files, *Empirical Software Engineering*, Vol.24, No.3, pp.1296–1327 (2019).

[10] Riehle, D.: *The Five Stages of Open Source Volunteering*, pp.25–38, Springer Berlin Heidelberg (2015).

[11] Robles, G., González-Barahona, J.M., Cervigón, C., Capiluppi, A. and Izquierdo-Cortázar, D.: Estimating development effort in free/open source software projects by mining software repositories: A case study of openstack, *Proc. 11th Working Conference on Mining Software Repositories*, pp.222–231 (2014).

[12] Steinmacher, I., Chaves, A.P., Conte, T.U. and Gerosa, M.A.: Preliminary empirical identification of barriers faced by newcomers to Open Source Software projects, *2014 Brazilian Symposium on Software Engineering* (*SBES*), pp.51–60 (2014).

[13] Steinmacher, I., Conte, T.U., Treude, C. and Gerosa, M.A.: Overcoming open source project entry barriers with a portal for newcomers, *Proc. 38th International Conference on Software Engineering*, pp.273–284 (2016).

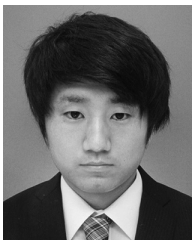
[14] Survey, O.S.: available from ⟨http://opensourcesurvey.org/2017/⟩ (accessed 2020-06).

[15] 伊原彰紀，大平雅雄：オープンソースソフトウェア工学：シリーズオープンソースソフトウェア工学，コンピュータ ソフトウェア，Vol.33, No.1, pp.1_28–1_40 (2016).

[16] 池田祥平，伊原彰紀，ラウラ ガイコビナクラ，松本健一：GitHub における README 記述項目の分析，第 24 回ソフトウェア工学の基礎ワークショップ，pp.135–140 (2017).

亀井 靖高 （正会員）

2005 年関西大学総合情報学部卒業．2009 年奈良先端科学技術大学院大学情報科学研究科博士後期課程修了．同年日本学術振興会特別研究員（PD）．2010 年カナダ Queen's 大学博士研究員．2011 年九州大学大学院システム情報科学研究院助教．2015 年同大学同研究院准教授．博士（工学）．マイニングソフトウェアリポジトリ，ソフトウェアメトリクスの研究に従事．ESEM 2007 Best Paper Award，MSR 2014 Distinguished Paper Award，2015 年度情報処理学会論文賞等各賞受賞．ACM，ソフトウェア科学会，電子情報通信学会各会員．IEEE Senior Member．

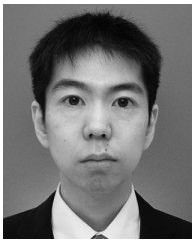


清水 一輝

2020 年九州大学工学部電気情報工学科卒業，現在，同大学大学院システム情報科学府博士前期課程在学中．ソフトウェア工学，特にマイニングソフトウェアリポジトリの研究に従事．

柏 祐太郎 （正会員）

2013 年和歌山大学システム工学部卒業．2015 年同大学大学院システム工学研究科博士前期課程修了．2015 年（株）日立製作所に入社，システムエンジニアとして勤務．2017 年日本学術振興会特別研究員（DC1）．2020 年和歌山大学大学院システム工学研究科博士後期課程修了．同年九州大学大学院システム情報科学研究院特任助教．ソフトウェア工学，リリースエンジニアリングの研究に従事．



佐藤 亮介 （正会員）

2013 年東北大学大学院情報科学研究科博士課程後期 3 年の課程修了．同年東京大学大学院情報理工学研究科特任研究員．2017 年九州大学大学院システム情報科学研究院助教．2020 年東京大学大学院情報理工学系研究科助教．ソフトウェアの形式的検証の研究に従事．

鵜林 尚靖 （正会員）

1982 年広島大学理学部数学科卒業．1999 年東京大学大学院総合文化研究科広域科学専攻広域システム科学系博士課程修了．博士（学術）．1982〜2003 年（株）東芝に勤務．2003 年九州工業大学情報工学部助教授，2010 年九州大学大学院システム情報科学研究院教授，現在に至る．2003 年度情報処理学会山下記念研究賞受賞．ソフトウェア工学，プログラミング言語モデルの研究に従事．日本ソフトウェア科学会，電子情報通信学会，ACM，IEEE-CS 各会員．本会フェロー．