リモートワークにおけるソフトウェア開発者間のコミュニケーション方法の調査

近藤 将成[1,a)]　崔 恩瀞[2,b)]　西浦 生成[1,c)]　水野 修[2,d)]

概要：近年のソフトウェア開発は DevOps 運動に代表されるように大きな変化を迎えている．その中で，従来のようにひとつのオフィスに人が集まる働き方だけでなく，リモートワークを推進する動きも活発となっている．しかし，リモートワークを行うためには，いくつもの問題を解決する必要がある．GitLab Inc. はリモートワークにおける問題と自社での解決策をまとめた資料を一般に公開している．一方で，実際にそれらの解決策が他の組織でも可能なのか，また，実際に行われているのかが明らかになっていない．そこで，本研究では，ソフトウェア開発者がリモートワークを実践する上で発生する問題の中でも，特に開発者同士のコミュニケーションに関連するものに注目し，GitLab Inc. が提示した解決策が実践されているのかどうかを調査する．ソフトウェア開発の中でも，特にオープンソースソフトウェア（OSS）の開発は，開発者が世界中に分散しており，リモートワークを行っていると言えるため，OSS を対象に調査を行った．調査結果より，否定的なコメントをより詳細に書くこと，及び，非同期的なコミュニケーションが行われていることがわかった．一方で，絵文字，及び，顔文字は使用されているとは言い難かった．

1. はじめに

近年のソフトウェア開発は DevOps 運動など大きな変化を迎えている [5]．DevOps には様々な概念が含まれており，そのひとつに，リモートワークがある．複数の国を跨いで活動する企業やソフトウェア開発プロジェクトでは，従来のようにひとつのオフィスに全ての開発者が集まって開発するだけではなく，リモートワークを実現することで，開発者同士が距離的に離れて開発する必要もある．GitLab Inc. は全ての従業員がリモートワークで働く企業のひとつであり，リモートワークの実施で得た知見を一般に公開している [6]．日本でも，社会的要請の影響もあり，全社員がリモートワークを実施する会社がでてくるなど [18]，リモートワークが大きく注目されている．

チームでソフトウェア開発を行う環境にリモートワークを取り入れる上で，「技術的な問題」と「文化的な問題」が存在する．技術的な問題については，Developers Summit 2020 にて講演が行われる [4] など，既存の技術を検討することで解決が可能である．一方で，GitLab Inc. の公開資料 [6] では，主に文化的な問題とその解決策について議論をしており，その重要性が述べられている．例えば，否定的な指摘を文章によって行う際には，その文章に具体的な問題点とその解決策を示す必要性について指摘している．しかし，具体的にこれらの文化的な問題に対する解決策が，どのようにソフトウェア開発において実現されているかは述べられていない．例えば，否定的な指摘とそうではない指摘にどの程度の差があるのかについて書かれていない．

オープンソースソフトウェア（以降，OSS）の開発では，多くの開発者が世界中に散らばり，ソフトウェアの開発を行っているため，リモートワークによるソフトウェア開発の代表的なモデルケースと考えられる．そのため，GitLab Inc. の公開資料で指摘されているようなリモートワークにおける文化的な問題の解決策を実践している可能性がある．しかし，我々の知る限り，リモートワークにおける文化的な問題に対する解決策が，OSS において実践されているかを調査した既存研究はない．

本研究では，OSS の課題管理に着目して，リモートワークの文化的な問題，特に，ソフトウェア開発者間のコミュニケーションにおける問題とその解決策を調査する．具体的には，GitLab Inc. の公開資料で議論されている 3 つのコミュニケーションにおける問題の解決策がどのように実践されているのかを調査する．問題と解決策とは，(1) 否定的なフィードバックを行った時に，受け手が内容に疑問を持っても，相手のところへ行き直接質問することが難しく

[1] 京都工芸繊維大学 大学院工芸科学研究科 設計工学専攻
[2] 京都工芸繊維大学 情報工学・人間科学系
[a)] m-kondo@se.is.kit.ac.jp
[b)] echoi@kit.ac.jp
[c)] k-nishiura@se.is.kit.ac.jp
[d)] o-mizuno@kit.ac.jp

解決に時間がかかるという問題を，あらかじめ否定の理由や解決策についてより詳細に述べることで解決すること，(2) 文章でのコミュニケーションのため，適切に自分の感情を表現し伝えることが難しいという問題を，絵文字を使用することで解決すること，及び (3) 世界中に開発者が散らばり，時差や働く時間の違いがあるという問題を，非同期的にコミュニケーションを取ることで解決することである．調査対象の OSS として，ソフトウェア開発者らが集まることができる固定された拠点が基本的に存在しない OSS（Apache Software Foundation の Avro，Tez，そして ZooKeeper，及び，Ruby の Rake）及び，ソフトウェア開発者らが集まることができる固定された拠点がひとつ以上存在する OSS（Docker for Mac，及び Netflix の Zuul，Falcor，及び，Hystrix）を選択し，完全なリモートワークである場合とそうでない場合での比較を行い，完全なリモートワークでは行えないコミュニケーションの影響を考慮できるようにした．コミュニケーションにおける問題とその解決策が実践されているのかあるいは実践されていないのかを調査するために，以下に示す 3 つの研究設問（Research Question，RQ）を設定する．

RQ1: 肯定的なコメントと否定的なコメントで差異はあるのか？

RQ2: コメントでのコミュニケーションに絵文字もしくは顔文字を使用しているのか？

RQ3: コメントの返答にかかる時間はどの程度か？

本研究は，リモートワークを成功させるために必要な，コミュニケーションにおける問題に対する解決策が，OSS の開発において実践されていることを明らかにした．リモートワークを導入する場合，否定的な文章を送る際にはその内容をより詳細に述べる必要があり，また，文章によるコミュニケーションでは非同期的なコミュニケーションが好まれていることを示唆している．一方で，絵文字及び顔文字を使うなど感情表現手法を使わずとも，リモートワークでのソフトウェア開発がうまくいくことを示している．これらの結果は，リモートワークに対して新たなマナー（例：否定的な文章を送るときは “必ず” 長い説明を付けなければならない）を持ち込むものではないが，リモートワークを成功させる上で，文化的な側面もまた重要であることを示唆している．

本研究の貢献として，調査から以下を発見した．

(1) 否定的なコメントを詳細に書くことは実践されていた．特に，否定的なコメントが行われた課題では，解決に時間がかかっており，またコメント数が多いなど，丁寧な議論が行われていた．

(2) 絵文字の使用に関しては，今回調査した OSS においては一般的であるとは言い難かった．感情を伝えるための別の手法を検討する必要がある．

(3) 非同期での議論も実践されていた．リモートワークの開発では，同期的なコミュニケーションを期待するべきではなく，非同期的にソフトウェア開発を進めていく枠組みを考えることが重要である．

以降の本稿の構成を紹介する．第 2 章では，リモートワークの問題点及び OSS において文化的な側面を調査した関連研究について述べる．第 3 章では，調査対象の OSS とそのデータについて説明する．第 4 章では，それぞれの研究設問に対する調査結果をまとめる．第 5 章では，調査結果とソフトウェア開発との関係を議論する．第 6 章では，本研究の妥当性の検証を行う．第 7 章では，まとめと今後の課題を述べる．

2. 関連研究

近年，ソフトウェア開発サイクルの高速化などにより，DevOps という概念が注目を浴びている．DevOps は元々ソフトウェア開発チームと運用チームが分けられている事による弊害を無くすために立ち上がった運動であるが，近年では，技術的な側面と文化的側面の両方の様々な概念を含んだ言葉となっている [5]．DevOps が含む概念の中で，社会的な要請などの影響により近年注目を集めているのは，リモートワークである．一方で，ソフトウェア開発でリモートワークを成功させるためには，単純にオフィス以外で仕事をすれば良いわけでは無く，社員の孤独感，組織文化の継承，セキュリティなど様々な問題を解決する必要がある [6].

ソフトウェア開発におけるリモートワークに関する議論は既に行われている [20, 23]．例えば，Sepulveda [23] は，アジャイル開発におけるリモートワークを取り上げている．リモートワークの難しさのひとつにチームメンバーとの信頼関係を構築することがあると自身の経験より述べており，コミュニケーションを円滑に取る方法が必要であると述べている．Olson ら [20] は，リモートワークや多文化なチームでの開発における注意点を文化的な側面からまとめている．

GitLab Inc. はリモートワークのみで組織を構成した企業のひとつであり，リモートワークの実施で得られた知見を一般に公開している [6]．この中ではリモートワークにおける文化的な問題点の具体的な解決策についても議論されている．例えば，社員同士の繋がりを保ち，孤独感を減らすために，仕事とは関係がない雑談チャットを行うなどの取り組みを紹介している．しかし，これは GitLab Inc. における取り組みであり，紹介されている解決策が一般的に適用可能であり有効であるのか，また，実際に使われているのかについては議論されていない．

開発が頓挫せず長く続いている OSS は，リモートワークの成功事例である．従って，OSS の開発履歴を調査することで，文化的側面から見たリモートワーク成功のための取り組みが実際に行われているのかを調査することが可能で

ある．OSS の開発における文化を調査した研究は存在するが [3]，リモートワークに関する調査は行われていない．そこで，本研究では，GitLab Inc. が公開している資料の中で指摘されている，リモートワークにおける問題の解決策が，OSS の開発の中で実際に行われているのかを調査する．

3. 調査対象

3.1 用語の定義及び調査対象の OSS の選択

調査対象のデータを収集する上で，本研究における用語を定義する．

リモートワーク： ソフトウェア開発者らが場所及び時間を限定されず作業を行い，かつ，主たるコミュニケーションがオンラインで完結するソフトウェア開発形態．

完全リモートワーク： ソフトウェア開発者らが集まることができる固定された拠点が一切無いソフトウェア開発形態．

準リモートワーク： ソフトウェア開発者らが集まることができる固定された拠点がひとつ以上存在するソフトウェア開発形態．

ここで，リモートワークをリモートワーク全体の集合とすると，完全リモートワークはその部分集合であり，準リモートワークは完全リモートワークの補集合である．

調査を行う上で，OSS におけるリモートワークの開発プロセスだけではなく，主たるコミュニケーションが対面で行われる，内製の開発プロセスとの比較も必要である．OSS において内製での開発プロセスを得ることは難しいため，本研究では，完全リモートワーク及び準リモートワークの OSS を比較した．また，課題管理システム（JIRA）を使っている OSS と，GitHub の課題ページ（イシューページ）を使っている OSS を調査対象の OSS として選定した．表 1 にデータの概要を示す．この表で，開発種類は，完全リモートもしくは，準リモートのどちらであるかを示している．課題ページは課題が何を使って管理されているかを示している．ボット以外のコメント数とは実際に調査したコメント数である．なお，Apache の OSS 及び Docker for Mac では，ボットによるコメントが行われており，調査のノイズとなる可能性があったため，調査対象から除外した．除外されたボットによるコメント数はボットコメント数の列に書かれている．

準リモートワークとしたのは，Docker と Netflix の OSS である．この 2 つの組織はオフィスを持っており，社内にコアメンバーがいるため，完全リモートワークでの開発ではないと考えられる．一方で，完全リモートワークとしたのは，Apache と Ruby の OSS である．この 2 つの組織では，物理的なオフィスを基本的に持たない．従って，開発者がオフィスに集まって開発する状況を想定しづらいため，完全リモートワークとして調査した．

本研究では，課題管理システムに登録されている各課題のコメントを利用した．OSS における他のコミュニケーション手段として，メール，IRC，及び Slack がある．しかし，IRC 及び Slack はチャット形式であり，話題のまとまりを抜き出すことが難しい．また，メールは話題のまとまりを抜き出すことはできるが，議論していた内容が解決されたかを判断することが難しい．そのため，今回はリモートワークにおける問題に対する解決策が，OSS の開発で実施されているのかを調査するために，課題管理システムに登録されている各課題のコメントを分析対象とした．

表 1 調査対象の OSS と課題管理システムにおける課題数及びコメント数．

開発種類	組織	課題ページ	OSS	課題数	ボット以外のコメント数	ボットコメント数
完全リモート	Apache	JIRA	Avro	2,787	12,608	2,431
完全リモート	Apache	JIRA	Tez	4,124	25,164	94
完全リモート	Apache	JIRA	ZooKeeper	3,749	32,178	3,500
完全リモート	Ruby	GitHub	Rake	84	232	-
準リモート	Docker	GitHub	Docker for Mac	4,108	14,410	1,672
準リモート	Netflix	GitHub	Zuul	272	684	-
準リモート	Netflix	GitHub	Falcor	417	1,566	-
準リモート	Netflix	GitHub	Hystrix	671	3,143	-

データの取得は，JIRA 及び GitHub の Web API を利用した．JIRA のデータは 2020 年 4 月 11 日より収集を行った．GitHub のデータは 2020 年 4 月 19 日より収集を行った．

3.2 各課題のコメントの前処理

課題管理システムのコメントには，ソースコードやログメッセージが貼り付けられている場合がある．JIRA の場合は，多くのソースコードやログメッセージが特定の文字列に囲まれている．一方で，GitHub の課題ページから GitHub の Web API で取得したコメントからは，この種の文字列が消えてしまっている．そこで，我々はヒューリスティックスを定義して，ソースコード及びログメッセージを取り出すことを行った．ヒューリスティックスは，ソースコードやログメッセージ以外をできる限り取り除くことが発生しないように作成した．具体的には JIRA と GitHub の共通のヒューリスティックスとして，

- {} もしくは [] で囲まれた箇所を BRACKETS に置き換える．
- 改行後，//，$，<，>，#，[ERROR]，\d\d\d\d-\d\d-\d\d もしくは \d+\s+\d+\s+\d+:の正規表現とマッチするような文章のいずれかで始まる行を改行に置き換える．
- URL を URL という単語に置き換える．
- 改行以外の後に，; の次に改行がある行を改行に置き換える．

JIRA のみのヒューリスティックスとして，

- 改行後，[INFO]，[DEBUG]，[WARNING] で始まる行を改行に置き換える．
- {code から {code}，{noformat から {noformat}，{quote から {quote} で囲まれた箇所を CODE に置き換える．

GitHub のみのヒューリスティックスとして，

- 改行後，スペース，*もしくは [OK] で始まる行を改行に置き換える．

これらを作成し，適用した．

4. RQ とその回答

本章では，1 章で説明した 3 つの RQ の設定理由（動機），3 章で説明した調査対象を用いて各 RQ を答えるために行った調査方法（手法），及び，RQ への答えを述べる．

4.1 RQ1: 肯定的なコメントと否定的なコメントで差異はあるのか？

4.1.1 動機

肯定的な文章と異なり，否定的な文章をただ相手に送るだけでは，文章を受けた人は何が問題でどのような改善を行えば良いのかがわからない．そのため，否定的な文章を送る場合は，より詳しく具体的な意見を述べることで，ソフトウェア開発に対して有用なフィードバックを行うことができる [6]．この RQ では，各課題におけるコメントを全て収集し，それらを肯定的なコメントか否定的なコメントかに分類し，コメントの長さを比較する．

4.1.2 手法

各課題における各コメントが肯定的か否定的かに分類するために，感情分析手法を利用した．感情分析手法とは，ルールベースもしくは機械学習などモデルベースで，与えられた文章に対して感情値を推定する手法である．感情分析は，既にソフトウェア工学でも利用されている [8,16,17]．

感情分析手法には，ソーシャルメディアなどを利用して構築された手法（例：VADER [9]）や，ソフトウェア工学向けに調整された手法（例：Senti4SD [1]）が存在する．先行研究の結果 [10,11,19]，及び，公開された実装の有無と正解データを持つデータセット [21] を用いた感情分析手法の精度の比較実験の結果より，我々はソフトウェア工学向けに調整された Senti4SD [1] を採用した．[*1] Senti4SD は，スタックオーバーフローの質疑データを元にして，用語，キーワード，そして，文章の意味を考慮した特徴を生成し，それを利用して分類を行う手法である．

全ての調査対象のコメントに対して長さを計算する必要がある．我々は全てのコメントを Python の文字列型に変換した後に，len() 関数によって得た数値をコメントの長さとして利用した．最後に，肯定的なコメントと否定的なコメントでコメントの長さを比較した．

4.1.3 結果

表 2 に調査対象の OSS ごとの分析対象のコメント総数，肯定，否定，もしくは，感情無し（ニュートラル）に分類されたコメントの総数を記した．全ての調査対象の OSS において，肯定的なコメントは否定的なコメントよりも多く，1.5 倍から 3 倍程度の数であった．肯定的なコメントと否定的なコメントが極端にバランスが悪い OSS は存在しなかった．ニュートラルに分類されたコメント数は，肯定的なコメント及び否定的なコメントどちらよりも多かった．

表 2 Senti4SD によって分類された肯定的，否定的，そしてニュートラルなコメント数．

組織	OSS	総数	肯定	否定	ニュートラル
Apache	Avro	12,608	3,982	1,685	6,941
Apache	Tez	25,164	7,292	3,488	14,384
Apache	ZooKeeper	32,178	10,897	7,130	14,151
Ruby	Rake	232	69	28	135
Docker	Docker for Mac	14,410	4,102	2,479	7,829
Netflix	Zuul	684	187	63	434
Netflix	Falcor	1,566	399	128	1,039
Netflix	Hystrix	3,143	826	382	1,935

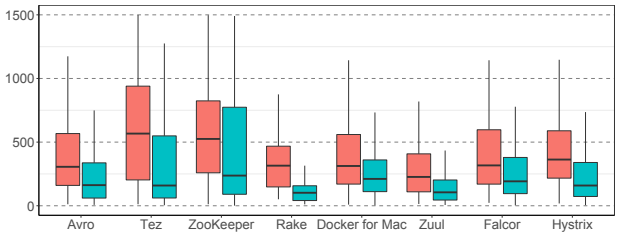


図 1 否定的なコメント（左手赤），肯定的なコメント（右手青）それぞれでの，コメントの長さの箱髭図．

全ての調査対象 OSS において，否定的なコメントは，肯定的なコメントより長い．図 1 及び表 3 に各感情を持ったコメントの長さの値を，箱髭図及びパーセンタイルで示した．肯定的なコメントの長さと否定的なコメントの長さのパーセンタイルを比較すると，全ての場合で否定的なコメントの値が大きい．また，P 値（表 3 の最後の列）からわかるように，マン・ホイットニーの U 検定の結果も非常に小さい（< 0.01）．この結果より，否定的なコメントが有意に長くなる傾向があるとわかる．

次に課題数とコメント数が比較的近い完全リモートワークの OSS と準リモートワークの OSS を比較した結果を述べる．本研究では，Apache の Avro 及び Tez と Docker for Mac の比較，及び，Ruby の Rake と Netflix の Zuul の比較を行った．Apache の Avro 及び Tez と Docker for Mac の比較は，課題数及びコメント数がそれぞれ近い．Ruby の Rake と Netflix の Zuul の比較は，課題数及びコメント数が完全リモートワーク及び準リモートワークにおいて最小であり，かつ，その値が比較的に近い．

完全リモートワークと準リモートワークで，コメントの長さに一般化可能な差があるとは言い難い．表 4 に，対象のプロジェクト間における各感情を持ったコメントの長さの差のマン・ホイットニーの U 検定の P 値を示してい

[*1] GitHub リポジトリで事前実験の検討事項と結果を示している：https://github.com/MKmknd/preexp-ses2020-remotework-oss

表 3 全てのコメント，肯定的なコメント，否定的なコメントそれぞれでの，コメントの長さのパーセンタイル（25th，50th，及び 75th），及び，肯定的なコメントと否定的なコメントの長さを比較した時のマン・ホイットニーの U 検定の P 値（小数点以下 3 桁まで）．

OSS	全て			肯定			否定			P 値
	25th	50th	75th	25th	50th	75th	25th	50th	75th	
Avro	62.0	165.0	378.0	62.0	169.0	352.0	168.0	330.0	644.0	0.000
Tez	52.0	144.0	429.0	64.0	178.0	814.0	209.0	613.5	945.25	0.000
ZooKeeper	71.0	192.0	551.0	91.0	242.0	774.0	267.0	561.0	835.0	0.000
Rake	48.75	105.5	215.25	41.0	102.0	157.0	156.0	328.5	687.0	0.000
Docker for Mac	77.0	165.0	331.0	113.0	216.0	360.0	179.0	338.0	662.5	0.000
Zuul	47.75	93.0	205.5	45.5	115.0	217.0	112.5	248.0	444.5	0.000
Falcor	72.0	164.0	378.0	99.5	199.0	400.5	170.0	327.5	608.5	0.000
Hystrix	74.5	173.0	380.5	73.25	160.5	345.0	230.0	390.5	617.5	0.000

表 4 比較的規模が近い完全リモートワークの OSS と準リモートワークの OSS の，各感情分析結果におけるコメントの長さの差のマン・ホイットニーの U 検定の P 値（小数点以下 3 桁まで）．

OSS	全て	肯定	否定
Avro vs. Docker for Mac	0.495	0.000	0.161
Tez vs. Docker for Mac	0.000	0.009	0.000
Rake vs. Zuul	0.380	0.205	0.155

る．Tez と Docker for Mac の比較では，P 値が非常に小さい（< 0.01）．一方で，Avro と Docker for Mac の比較では，肯定的なコメントを除いて P 値が大きい．また，Rake と Zuul の比較でも P 値が大きい．そのため，完全リモートワークと準リモートワークで，コメントの長さに差がある場合と無い場合がある．

RQ1 の答え：否定的なコメントは，肯定的なコメントよりも長くなるという差異が存在した．一方で，コメントの長さは完全リモートワークと準リモートワークで一般化可能な差があるとは言い難い．

4.2 RQ2: コメントでのコミュニケーションに絵文字もしくは顔文字を使用しているのか？

4.2.1 動機

文章によるコミュニケーションにおいて，絵文字を使うことで，文章だけでは表現が難しい感情を伝えることができる [6]．実際，コミュニケーションにおける絵文字の有効性が既存研究で示されている [7, 14, 15]．

Lu ら [13] はソフトウェア工学分野において，絵文字がどのように利用されるのか調査を行った．一般的なソーシャルメディアのコミュニティとの絵文字の使い方の違いとして，絵文字を特別な意味として使用する（例：虫の絵文字を不具合のアイコンとして利用）などの特徴が報告されている．また，絵文字を肯定的に使用している傾向も観測されている．一方で，リモートワークという点での比較は行っていない．そこで，この RQ では，Lu らが比較した点を，完全リモートワークと準リモートワークで調査する．また，絵文字のみではなく，顔文字についても調査をする．

表 5 絵文字を持つコメント及び課題の割合．

OSS	絵文字の出現回数	絵文字を持つ課題割合	絵文字を持つコメント割合
Avro	2	0.000 (1/2,387)	0.000 (2/12,608)
Tez	16	0.000 (1/3,453)	0.000 (1/25,164)
ZooKeeper	6	0.001 (3/3,255)	0.000 (4/32,178)
Rake	8	0.078 (6/77)	0.034 (8/232)
Docker for Mac	473	0.078 (253/3,224)	0.023 (329/14,410)
Zuul	3	0.013 (3/227)	0.004 (3/684)
Falcor	30	0.058 (22/380)	0.016 (25/1,566)
Hystrix	27	0.037 (23/625)	0.009 (27/3,143)

表 6 顔文字を持つコメント及び課題の割合．

OSS	顔文字の出現回数	顔文字を持つ課題割合	顔文字を持つコメント割合
Avro	159	0.045 (108/2,387)	0.012 (148/12,608)
Tez	176	0.037 (129/3,453)	0.007 (166/25,164)
ZooKeeper	659	0.116 (378/3,255)	0.019 (603/32,178)
Rake	10	0.130 (10/77)	0.043 (10/232)
Docker for Mac	374	0.076 (246/3,224)	0.024 (352/14,410)
Zuul	13	0.040 (9/227)	0.019 (13/684)
Falcor	57	0.113 (43/380)	0.035 (55/1,566)
Hystrix	56	0.062 (39/625)	0.017 (52/3,143)

4.2.2 手法

本研究での絵文字とは，UTF-8 で定義されている絵文字のコードを指しており，顔文字は記号の組み合わせで作られるものとしている．絵文字の一覧は，Python のパッケージである `emoji` [12] で定義されている 2,811 個を対象とした．顔文字は，Wikipedia [24] よりラテン語のみによって構成された 139 個（2020 年 4 月 28 日時点）を利用した．

4.2.3 結果

顔文字は絵文字よりも頻繁に利用されている．その利用頻度は OSS ごとに異なり，完全リモートワークであるか否

かは影響しない．表 5 に絵文字を用いた調査対象の課題とコメントの割合をプロジェクトごとに示している．絵文字が使用されたコメントの割合は，最も割合の高い Rake でも 3.4%である．絵文字が使われたコメントを持つ課題の割合は，Rake 及び Docker for Mac で 7.8%であり，課題単位で見ると，使われる場合があると言える．一方で，課題単位でも 10%を超えることはない．また，Apache の 3 つのプロジェクトでは，課題単位でも 1%を超えることはない．

表 6 に顔文字を用いた調査対象の課題とコメントの割合を示している．Tez を除いた全てのプロジェクトで 1%以上のコメントに顔文字が使用されている．課題単位でみると ZooKeeper，Rake，及び，Falcor で 10%を超える割合となった．そのため，顔文字は絵文字よりも頻繁に利用されている．

これらの結果より，Apache の OSS で絵文字はほぼ使用されておらず，他の OSS でもその使用割合が異なることがわかる．顔文字は絵文字と比較するとより多く使用されており，また Apache の OSS でも利用されている．この違いは，完全リモートワークであるか，準リモートワークであるかは関係がない．

RQ2 の答え：絵文字の使用は一般的とは言い難い．一方で，顔文字は Tez を除いて 1%以上のコメントで使用されており，絵文字よりも高い頻度で利用されている．しかし，その頻度も高いとは言い難い．リモートワークであるなしに関わらず，絵文字及び顔文字の使用は一般的とは言い難い．

4.3 RQ3: コメントの返答にかかる時間はどの程度か？

4.3.1 動機

文章によるコミュニケーションは同期的にも非同期的にも行えるコミュニケーションである．非同期的コミュニケーションでは，時差がある場合，子供がいるなどで働く時間が異なる場合でも，コミュニケーションを取ることができる．

課題管理システムにおいて，開発者らは該当課題に対して掲示板形式で議論を重ねているため，同期的にも非同期的にも議論を行うことができる．GitLab Inc. の公開資料 [6] では非同期的なコミュニケーションの重要性も述べている．この RQ では，開発者らが各課題において短時間で返答して議論を行うことを好むのか（同期的議論），それとも，返答までに時間差があることを許容して議論を行うことを好むのか（非同期的議論）を調査する．

4.3.2 手法

各課題において，コメント間の時間差を計算した．ここで，JIRA の課題の説明文，及び，GitHub の課題の最初のコメントは調査から除外している．これらは課題の報告と説明を行うものであり，課題が報告されてから実際にその課題に開発者らが取り組み始めるまでにタイムラグがあると考えられるためである．得られた時間差に対して，返答時間のパーセンタイルを計算した．また，時間差の箱髭図を作成した．

表 7 コメントの返答時間のパーセンタイル（25th，50th，及び 75th）．

OSS	25th	50th	75th
Avro	0d-0h-31m-33s	0d-6h-8m-59s	2d-4h-36m-23s
Tez	0d-0h-30m-46s	0d-2h-22m-4s	0d-18h-13m-28s
ZooKeeper	0d-0h-22m-14s	0d-3h-0m-0s	1d-13h-36m-18s
Rake	0d-0h-7m-25s	0d-3h-11m-43s	3d-0h-36m-55s
Docker for Mac	0d-0h-52m-23s	0d-16h-21m-24s	6d-21h-14m-26s
Zuul	0d-1h-22m-13s	0d-14h-51m-50s	7d-17h-48m-31s
Falcor	0d-0h-31m-11s	0d-4h-43m-31s	3d-4h-46m-5s
Hystrix	0d-0h-39m-31s	0d-9h-54m-53s	4d-9h-2m-26s

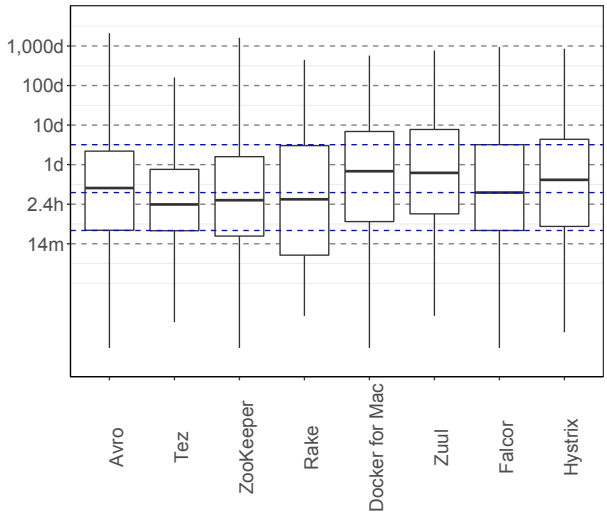


図 2 コメントの返答時間の箱髭図．縦軸は対数スケール．m は分間，h は時間，d は日数を表す．

4.3.3 結果

半分程度のコメントは返答に 2 時間以上の時間がかかっているが，多くのプロジェクトでは 4 分の 1 のコメントは 30 分程度もしくはそれより短い時間で返答されていた．表 7 はコメントの返答時間のパーセンタイルを示す．この表より，Apache の 3 つの OSS 及び，Netflix の 2 つの OSS（Falcor 及び Hystrix）では，25th パーセンタイルでおおよそ 30 分前後の返答時間である．Ruby の Rake の 25th パーセンタイルは非常に早く 7 分で行われていた．従って，これらの OSS では，4 分の 1 のコメントの返答は短時間（同期的）に行われていると言える．一方，50th パーセンタイルは少なくとも 2 時間 22 分（Tez）の返答時間がかかっており，半分のコメントの返答は非同期的に行われている．

準リモートワークは完全リモートワークよりもコメントの返答時間が伸びる場合がある．表 7 より，Docker for Mac 及び Zuul では，25th パーセンタイルが 1 時間前後で

あり，準リモートワークの一部の OSS では，ほとんどのコメントが非同期的に行われている．図 2 では，コメントの返答時間を箱髭図で示している．準リモートワークの中で全てのパーセンタイルで最も返答時間が早いのは，Netflix の Falcor であり，青の破線として示している．中央値が Falcor よりも大きい（返答時間が長い）完全リモートワークのプロジェクトは Avro のみであり，75 パーセンタイルでは，全ての完全リモートワークのプロジェクトがより短い時間で返答していた．よって，準リモートワークは完全リモートワークよりも返答時間が長い傾向にある．

RQ3 の答え：少なくとも半分のコメントは返答に 2 時間以上の時間がかかっており，非同期的な議論が好まれている．完全リモートワークでない場合，返答時間がより長くなる場合がある．

5. 議論

本章では，4 章で得た結果をまとめた．また，追加の調査を行い，それらの結果がソフトウェア開発とどのように関係しているのかをまとめた．

5.1 リモートワークの問題の解決策は OSS の開発で実践されているのか？

否定的なコメントを詳細に書くことは実践されていた．非同期での議論も実践されていたが，準リモートワークでも非同期での議論を行っている OSS があり，必ずしもリモートワークであるため採用しているとは言い難い．絵文字の使用に関しては，今回調査した OSS においては一般的であるとは言い難かった．

5.2 ソフトウェア開発とコメント内容の関係は何か？

RQ1 の結果より，否定的なコメントは長くなる傾向にあった．このコメントがソフトウェア開発に対して与える影響及び使用状況を調査するため，調査対象のコメントを肯定的で長い（PL），肯定的で短い（PS），否定的で長い（NL），否定的で短い（NS）の 4 つのコメントタイプに分類し，NL のコメントが行われた課題とその他を比較した．ここで，短いコメントと長いコメントは各感情に分類されたコメントの長さの中央値以上か未満であるかで分類した．

NL のコメントが行われた課題はコメント数が多くなる傾向がある．表 8 に，Avro における各コメントタイプのコメントを持つ課題におけるコメント数のパーセンタイルと，NL とその他のコメントタイプを持つ課題とを比較したときのマン・ホイットニーの U 検定の P 値を示す．また，視覚的にわかりやすいように，図 3 に箱髭図形式で図示した．NL のコメントが行われた場合，他よりコメント数が多く，また，P 値も小さい．P 値が小さくならない（> 0.05）場合

表 8 Avro における各コメントタイプを最低ひとつは持つ課題におけるコメント数のパーセンタイル，及び，NL なコメントを含んだ課題と他のコメントタイプを含んだ課題とのコメント数の差のマン・ホイットニーの U 検定の P 値（小数点以下 3 桁まで）．

コメントタイプ	課題数	25th	50th	75th	P 値
PL	986	3.0	6.0	10.0	0.000
PS	1,196	3.0	5.0	9.0	0.000
NL	491	5.0	8.0	15.0	-
NS	612	4.0	6.0	11.0	0.000

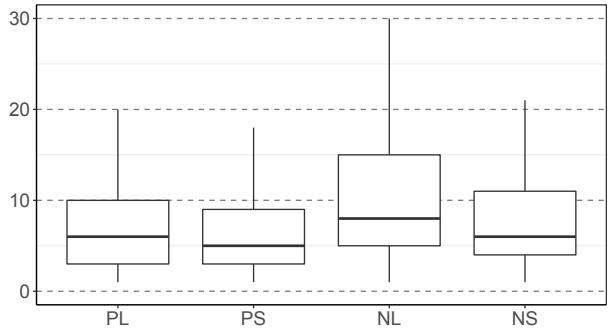


図 3 Avro における各コメントタイプを最低ひとつは持つ課題におけるコメント数の箱髭図．

表 9 Avro における各コメントタイプを最低ひとつは持つ課題における課題解決時間（秒）のパーセンタイル．

コメントタイプ	25th	50th	75th
PL	91,221.0	975,557.0	5,994,369.5
PS	88,682.5	620,212.0	3,757,364.0
NL	274,967.25	1,404,913.5	8,588,003.75
NS	256,838.25	1,283,548.0	7,352,132.75

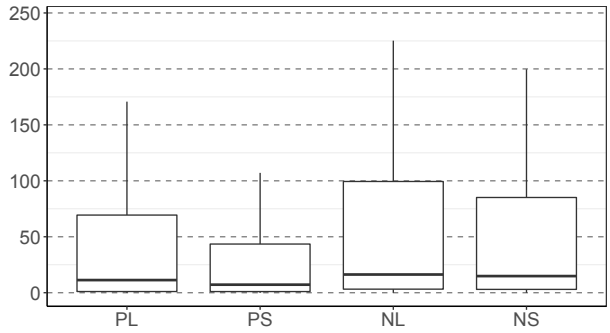


図 4 Avro における各コメントタイプを最低ひとつは持つ課題における課題解決時間（日）の箱髭図．

が Ruby の Rake，Docker for Mac，及び Netflix の Falcor で存在したが，コメント数が多くなる傾向はどの OSS でも確認できた．

完全リモートワークのプロジェクトにおいて，NL のコメントが使用される課題では，解決までに時間がかかる傾向にある．表 9 に，Avro における各コメントタイプのコメントを持つ課題における課題解決時間のパーセンタイル

表 10 絵文字もしくは顔文字が使用されている課題における NL なコメントを含んだ課題の割合及び，NL なコメントを含んだ課題における絵文字もしくは顔文字が使用された課題の割合．

OSS	絵文字もしくは顔文字における割合	NL なコメントにおける割合
Avro	0.532 (58/109)	0.118 (58/491)
Tez	0.504 (65/129)	0.079 (65/821)
ZooKeeper	0.611 (232/380)	0.189 (232/1,227)
Rake	0.333 (5/15)	0.556 (5/9)
Docker for Mac	0.486 (211/434)	0.292 (211/723)
Zuul	0.364 (4/11)	0.154 (4/26)
Falcor	0.262 (16/61)	0.302 (16/53)
Hystrix	0.316 (18/57)	0.149 (18/121)

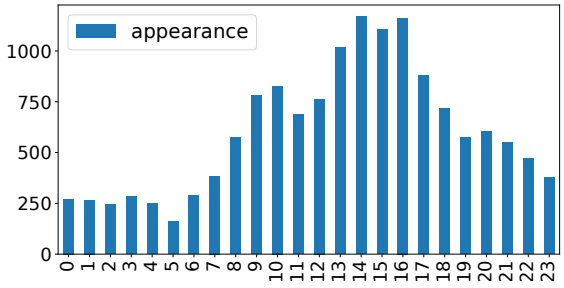
と，NL とその他のコメントタイプを持つ課題とを比較したときのマン・ホイットニーの U 検定の P 値を示す．先ほどと同様に，視覚的にわかりやすいように，図 4 に箱髭図形式で図示した．NL のコメントが行われた課題の解決時間が他より長い．他の完全リモートワークのプロジェクトでも同様の結果であった．一方で，準リモートワークでは，Zuul を除く Docker for Mac，Falcor，及び，Hystrix でこの傾向がみられなかった．よって，完全リモートワークのプロジェクトにおいて，NL のコメントが行われる課題は，解決までに時間がかかる傾向にある可能性がある．

次に，絵文字もしくは顔文字が使用されたコメントを持つ課題と，NL のコメントを持つ課題の関係性について分析する．表 10 に，絵文字もしくは顔文字が使用されたコメントを持つ，もしくは，NL のコメントが使用されている課題における双方が使用されている割合を示している．母集団が小さい方を元に計算すると，Netflix の OSS では約 3 割，それ以外の OSS では約 5 割存在していた．そのため，絵文字及び顔文字は，NL のコメントが使用されている課題において使用される割合が高い．ただし，絵文字及び顔文字の使用率は高くないため，ソフトウェア開発におけるコミュニケーションへの影響は小さく，また文化としても根付いていないと考えられる．

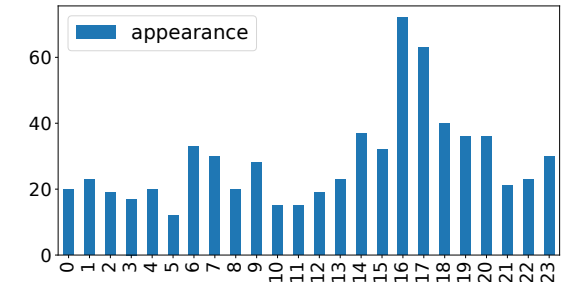
NL のコメントが使用されている課題では，コメント数が多くなり，また，完全リモートワークではその解決時間もより長くかかっている．このことから，NL のコメントが行われる課題はより丁寧な議論が行われている可能性がある．一方で，感情表現手法としての絵文字や顔文字の利用はその効果が指摘されているが，ソフトウェア開発においては一般的ではなく，GitHub の課題のコメントに付与できるスタンプのような，その他の手法の利用を考えていくと良いと考えられる．

表 11 コントリビューターひとりあたりの課題数．

OSS	割合
Avro	20.050 (2,787/139)
Tez	152.741 (4,124/27)
ZooKeeper	35.705 (3,749/105)
Rake	0.532 (84/158)
Docker for Mac	256.750 (4,108/16)
Zuul	6.800 (272/40)
Falcor	8.176 (417/51)
Hystrix	6.330 (671/106)



(a) Docker for Mac



(b) Zuul

図 5 Docker for Mac と Zuul における UTC 時間における各時間ごとのコメントが行われた回数．

5.3 ほとんどのコメントが非同期的に行われる OSS とそうでない OSS の差は何か？

RQ3 より，準リモートワークである Docker for Mac 及び Zuul では，25th パーセンタイルでも，コメントの返答時間が 1 時間前後であり，時間がかかっている．この理由を明らかにするために，まず，GitHub リポジトリにおけるコントリビューター 1 人あたりの課題数を算出した．コントリビューター 1 人あたりの課題数を調べることで，開発者に対して報告される課題数が多すぎないかを調査する．次に，開発者らがコメントを返している時間帯を算出した．開発者は主に通常のオフィスアワーで働いており [3]，準リモートワークでは，コア開発者らが同一オフィスの同一タイムゾーンにいることが返信が遅れる理由と考えたためである．

表 11 に，各プロジェクトにおけるコントリビューター

ひとりあたりの課題数を示す．Docker for Mac では，他の OSS と比較して，コントリビューターひとりあたりの課題数が極端に多くなっていることが確認できる．一方で，2 番目にコントリビューターひとりあたりの課題数が多いのは Tez であるが，完全リモートワークのプロジェクトの中で，50th 及び 75th パーセンタイルの返答時間が最も短い（表 7）．また，Zuul に関しても課題数の割合が高いとは言い難い．そのため，コントリビューターひとりあたりの課題数は，Docker for Mac で返答時間が長くなっている一要因である可能性はあるが，一般化可能な因果とは言い難い．

図 5 に，Docker for Mac 及び Zuul における各時間帯に行われたコメント数を示している．Zuul では 16 時及び 17 時にコメントが偏っていることが確認できる．一方で，Docker for Mac ではこのような偏りがみられない．そのため，返答している時間帯の偏りも，Zuul で返答時間が長くなっている一要因である可能性はあるが，一般化可能な因果とは言い難い．

コメントの返答時間にはプロジェクトごとに差があったが，コントリビューターひとりあたりの課題数及び返答時間帯は，一般化可能な原因であるとは言い難い．そのため，追加の調査が今後必要である．

一方で，RQ3 の結果より，少なくとも半分のコメントは返答に 2 時間以上の時間がかかることがわかっており，リモートワークでは，オフィスほどのコミュニケーション速度を求めるべきではなく，非同期的にコミュニケーションを行いソフトウェア開発を円滑に進めていく枠組みが重要であると考えられる．

6. 妥当性への脅威

構成概念妥当性： リモートワークにおけるコミュニケーションを円滑に進めるために推奨されている方法が実際に行われているかを調べ，ソフトウェア開発との関係を議論したが，因果関係については調査していない．そのため，今回得られた結果がリモートワークにおける問題の解決策のみに起因するものであるのかの調査はできていない．また，リモートワークを行っていないソフトウェア開発プロジェクトを調査していない．そのため，今回得られた結果が普遍的なリモートワークの特徴であるのか，及び，OSS における特徴との差異の調査はできていない．OSS には企業の活動として貢献している人とそうでない開発者がいる [22] ため，開発者の属性を調査し，その割合を考慮に入れることで，分析対象のプロジェクトをより細かく分類でき，結果の妥当性を向上させることができる．しかし，今回は課題管理システムを中心に分析しており，既存手法 [22] の分類手法を使用することができなかった．GitHub の課題にはその課題の優先度や難易度を示すラベルがつけられていないことが多く，課題の難易度別の分析はできていない．

内的妥当性： RQ2 で顔文字の調査を行ったが，顔文字は記号の集合であり，誤検出が発生する．ただし，今回調査した OSS での手作業での分析の結果，この割合は高くはなく，結果への影響は限定的である．コメントに含まれているソースコードやログメッセージを取り除くために，ヒューリスティックスを用いた．手作業による分析より，ソースコードやログメッセージ以外を取り除くことはほとんど発生しなかった．一方で，ソースコードやログメッセージを見逃してしまうことは発生しており，調査結果に影響を及ぼした可能性がある．

外的妥当性： 今回は，実験対象の OSS として，Apache の OSS，Ruby の OSS，Doker の OSS，及び，Netflix の OSS を使用した．また，オフィスを持つ OSS と持たない OSS も含んでいる．ただし，これらの OSS のみで十分な一般化ができるのかは疑問の余地がある．今後，より多くの OSS での分析を検討していく必要がある．顔文字は常に新しいものが生まれており，また記号の集合であることから全てを調査対象とすることは難しい．そのため，今回の調査結果は，一般的に有名な顔文字に限定したものとなる．

7. まとめと今後の課題

本研究では，OSS の課題管理に着目して，リモートワークにおけるコミュニケーションの取り方について，GitLab Inc. が公開している資料 [6] における解決策が実践されているのか及び，解決策とソフトウェア開発との関係を調査した．否定的なコメントは肯定的なコメントより詳細に書かれていた．非同期的なコミュニケーションも実施されていた．一方で，絵文字及び顔文字の使用は一般的ではなかった．

今後の課題として，リモートワークを行っていないソフトウェア開発における課題管理でのコミュニケーションと，リモートワークにおけるコミュニケーションとの差異の調査を行うこと，及び，今回調査した解決策がソフトウェア開発にどのような影響を与えるのか，因果関係を明らかにすることが必要である．この際に，より多くの OSS での分析を追加する予定である．また，絵文字の解釈について，国による差異が少ないとする研究 [14] 及び差があるとする研究 [2] が存在する．そのため，開発者の情報と組み合わせることで，より深い分析をする必要がある．

謝辞 本研究は，JSPS 科研費 JP19J23477，JP19K20240，及び JP18H04094 の助成を受けた．

参考文献

[1] Calefato, F., Lanubile, F., Maiorano, F. and Novielli, N.: Sentiment polarity detection for software development, *Empir. Soft. Eng.*, Vol. 23, No. 3, pp. 1352–1382 (2018).

[2] Heeryon, C.，稲葉利江子，石田　亨，高崎俊之，森由美子：絵文字コミュニケーションにおけるセマンティクス，技術報告 110(2006-ICS-145) (2006).

[3] Claes, M., Mäntylä, M. V., Kuutila, M. and Adams, B.: Do programmers work at night or during the weekend?, *Proc. of ICSE*, pp. 705–715 (2018).

[4] CodeZine 編集部：エンジニアの働き方改革を実現する、快適リモートワーク環境を構築する方法【デブサミ 2020】，Shoeisha Co., Ltd.（オンライン），入手先　〈https://codezine.jp/article/detail/12027?ocid=AID740624_TWITTER_oo_spl100001227025951〉（参照 2020-4-19）.

[5] Davis, J. and Daniels, K.: Effective DevOps: ―4 本柱による持続可能な組織文化の育て方，オライリー・ジャパン (発行所), オーム社 (発売元)，1st edition (2018).

[6] GitLab Inc.: The Remote Playbook from the largest All-Remote company in the world, GitLab Inc. (online), available from 〈https://about.gitlab.com/company/culture/all-remote/〉 (accessed 2020-4-9).

[7] Guibon, G., Ochs, M. and Bellot, P.: From emojis to sentiment analysis, *Proc. of WACAI* (2016).

[8] Guzman, E., Azócar, D. and Li, Y.: Sentiment analysis of commit comments in GitHub: an empirical study, *Proc. of MSR*, pp. 352–355 (2014).

[9] Hutto, C. J. and Gilbert, E.: Vader: A parsimonious rule-based model for sentiment analysis of social media text, *Proc. of ICWSM* (2014).

[10] Islam, M. R. and Zibran, M. F.: Leveraging automated sentiment analysis in software engineering, *Proc. of MSR*, pp. 203–214 (2017).

[11] Jongeling, R., Datta, S. and Serebrenik, A.: Choosing your weapons: On sentiment analysis tools for software engineering research, *Proc. of ICSME*, pp. 531–535 (2015).

[12] Kim, T. and Wurster, K.: carpedm20/emoji, GitHub (online), available from 〈https://github.com/carpedm20/emoji〉 (accessed 2020-4-28).

[13] Lu, X., Cao, Y., Chen, Z. and Liu, X.: A First Look at Emoji Usage on GitHub: An Empirical Study, *arXiv preprint arXiv:1812.04863* (2018).

[14] 宗森　純，大野純佳，吉野　孝：絵文字チャットによるコミュニケーションの提案と評価，情報処理学会論文誌，Vol. 47, No. 7, pp. 2071–2080 (2006).

[15] 宗森　純，重信智宏，丸野普治，尾崎裕史，大野純佳，吉野　孝：異文化コラボレーションへのマルチメディア電子会議システムの適用とその効果，情報処理学会論文誌，Vol. 46, No. 1, pp. 26–37 (2005).

[16] Murgia, A., Tourani, P., Adams, B. and Ortu, M.: Do developers feel emotions? an exploratory analysis of emotions in software artifacts, *Proc. of MSR*, pp. 262–271 (2014).

[17] 中川　要：ソフトウェア開発プロジェクトにおける感情極性推移の分析，修士論文，京都工芸繊維大学 (2018).

[18] 中川雅博：「全社一斉リモートワーク」1 カ月で見えた極意クラウド会計「freee」佐々木大輔 CEO が語る，Toyo Keizai Inc.（オンライン），入手先 〈https://toyokeizai.net/articles/amp/344405〉（参照 2020-4-16）.

[19] Novielli, N., Girardi, D. and Lanubile, F.: A benchmark study on sentiment analysis for software engineering research, *Proc. of MSR*, pp. 364–375 (2018).

[20] Olson, J. S. and Olson, G. M.: Culture surprises in remote software development teams, *Queue*, Vol. 1, No. 9, pp. 52–59 (2003).

[21] Ortu, M., Murgia, A., Destefanis, G., Tourani, P., Tonelli, R., Marchesi, M. and Adams, B.: The emotional side of software developers in JIRA, *Proc. of MSR*, pp. 480–483 (2016).

[22] 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. of MSR*, pp. 222–231 (2014).

[23] Sepulveda, C.: Agile development and remote teams: learning to love the phone, *Proc. of ADC*, pp. 140–145 (2003).

[24] Wikipedia: List of emoticons, Wikipedia (online), available from 〈https://en.wikipedia.org/wiki/List_of_emoticons〉 (accessed 2020-4-28).