OSS開発におけるコミッター選出のための開発者の活動量に関する実証的分析

An Empirical Analysis of Developer's Activities for Selecting Committers in OSS Development

伊原 彰紀[*] **藤田 将司**[†] **大平 雅雄**[‡] **松本 健一**[§]

あらまし オープンソースソフトウェア（OSS）には，日々膨大な数の不具合が報告されており，修正にかかる時間が長期化している問題がある．不具合修正の長期化を解消する方法として，コミッター（プロダクトへの変更を直接加えることのできる開発者）の増員が挙げられる．コミッター1人あたりのプロダクトへの修正の手間を削減することによって，短期間でより多くの不具合への対応を行うことができると考えられる．しかし，選出を行う基準は明確に示されておらず，プロジェクトにとって信頼のおける非コミッターをコミッターとして選出することは容易ではない．そこで，本論文では，既存のコミッターが新たなコミッターを選出する際の指標に成り得る特徴を明らかにするため，コミッターに選出されるべき非コミッター（コミッター候補者）と分析期間内でコミッターに昇格していない非コミッター（一般開発者）の活動を分析する．32件のOSSプロジェクトを対象とし，不具合管理システム記録されている開発者の活動を比較した結果，コミッター選出に有効な指標はプロジェクトによって異なるものの，多くのプロジェクトでコメント数と参加期間が有効であることが分かった．

1 研究の背景と目的

近年，オープンソースソフトウェア（OSS）は，個人ユーザの利用のみならず，企業における業務での利用が進んでいる．OSSが広く普及するにつれて，OSSの社会的重要性は高まってきており，例えボランティアで開発が行われているといえども，発見された不具合に対して迅速な修正が強く求められる．一般的なOSSプロジェクトで，不具合の修正は次のように行われている．

(1) 利用者あるいは開発者は，発見した不具合をメーリングリスト (Mailing List: ML) や不具合管理システム (Bug Tracking System: BTS) を通じて報告する．
(2) 不具合報告を受けた開発者は，不具合解決のためにソースコードを変更し，変更を行った差分をパッチとしてMLやBTSに投稿する．投稿されたパッチは他の開発者による確認（レビュー）を受けた後，プロダクトへ反映される．

大規模なOSSプロジェクトには膨大な数の不具合が報告されており [3]，不具合の修正が長期化している．例えば上述の (1) に関連するものとして，不具合の報告内容に必要な情報が記載されていないため，開発者が不具合の内容を理解することが困難となり，修正に遅延が生じている．開発者が迅速に不具合修正に取りかかるために，開発者が必要とする不具合の情報について調査した研究がある [7] [8]．

また，(2) に関連するものとして，レビューが実施されていないパッチの増加 [9]，レビューが終了しているにも関わらずプロダクトに反映されていない等の問題 [1] が報告されている．不具合修正を効率化するためには，パッチレビュープロセスの改善が必要である．

[*]Akinori Ihara, 奈良先端科学技術大学院大学
[†]Shoji Fujita, 奈良先端科学技術大学院大学
[‡]Masao Ohira, 奈良先端科学技術大学院大学
[§]Ken-ichi Matsumoto, 奈良先端科学技術大学院大学

パッチレビュープロセスを効率的に実施する方法として，パッチを承認し，プロダクトへ反映する役割を担っているコミッターを増員することが挙げられる．多くの OSS プロジェクトで，新しいコミッターは，非コミッターの中から既存のコミッターによって選出される．しかし，非コミッターの中からコミッターを選出するための明確な指標はなく，コミッター選出は既存のコミッターの主観で決定せざる負えない．また，開発者は非コミッターからコミッターへ昇格する際に活動内容が変化する場合があり，既存のコミッターが新たなコミッターに期待する活動と，実際にコミッターの活動との差異を小さくすることが必要である．

本論文では，パッチレビュープロセスの効率化のための第一歩として，OSS プロジェクトにおけるコミッターの選出を容易にするために，有能な開発者の特徴を明らかにすることを目的とする．コミッターに昇格する開発者の特徴を明らかにするために，コミッター，コミッター候補者（コミッター昇格前の非コミッター），一般開発者（分析期間内にコミッターに昇格していない非コミッター）のパッチレビュープロセスにおける活動を，回数，参加期間，レスポンスの早さという観点から分析する．コミッター候補者と一般開発者の違いを提示することによって，非コミッターからコミッターへ昇格する開発者の特徴を明らかにする．また，コミッターを対象に，昇格前後の活動を比較することにより，開発者の活動内容の変化を明らかにする．コミッター昇格前後の活動の違いを分析することで，コミッターの選出を行う開発者が新たなコミッターに期待する活動と，実際のコミッターの活動との差異を小さくすることができると考えられる．

2 関連研究

OSS 開発におけるコミッター役割に関する研究が盛んに行われている．Fogel は，コミッターはプロダクトへ修正を加える際の手間を出来るだけ減らすために，多くの OSS 開発で採用されている役割であると述べており，信頼のおける開発者である場合，コミッターの人数が多いほど良いとしている [6]．

Jensen らは，OSS 開発における開発者の役割について調査を行っている [4]．コミッターは，プロジェクトに対する貢献度の高い非コミッターの中から既存のコミッターによって選ばれる．Jensen らは，OSS のドキュメントなどを調査し，コミッターへの昇格プロセスについて調査しているが，本論文では，不具合管理システムにおける開発者の活動を定量的に分析した点で異なる．

Shibuya らは，不具合修正プロセスに参加する開発者について分析を行っており，OpenOffice.org プロジェクトと GNOME プロジェクトにおいて，時間とともにコミッターの総数は増加しているが，毎月コミットを行っているコミッターの人数は一定であると述べている [2]．Shibuya らはコミッターの人数などのプロジェクトの現状について調査しているが，どのような活動を行う非コミッターがコミッターへ昇格しているかは調査していない．

また，Bird らは，PostgreSQL を対象に非コミッターがコミッターに昇格するまでの活動期間について分析しており，プロジェクトに参加してから約 1 年前後がコミット権限を付与する時期として適していると述べている [5]．活動期間が 1 年よりも長くなると開発者のモチベーションが低下する危険性があり，一方で 1 年よりも短い期間でコミッターの適正を判断するのは困難であると述べている．Bird らは，コミッター選出が困難であることを述べており，その問題を解決するために本論文ではコミッターの選出に参考となり得る指標を明らかにする．

3 OSS 開発におけるパッチレビュープロセス

不具合の修正は，図 1 のように開発者とレビュアーが既存のソースコードと修正後のソースコードとの差分（パッチ）の修正，レビューを繰り返して行われる．パッチレビュープロセスに参加する開発者の定義を表 1 に示す．以下はパッチがプロダ

表 1　開発者の定義

			定義
開発者	コミッター		構成管理ツールへのコミット権限を持つ開発者を指す.
	非コミッター	コミッター候補者	分析期間中にコミッターに昇格する開発者であり，コミッター昇格以前の開発者を指す.
		一般開発者	分析期間中にコミッターに昇格していない開発者を指す.

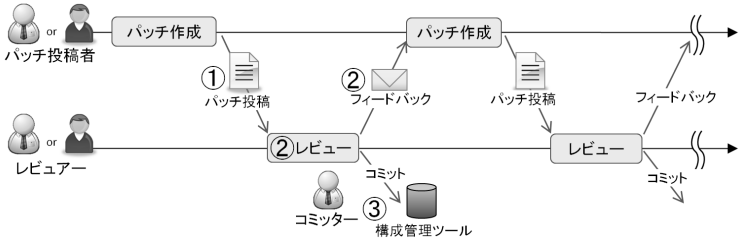

図 1　パッチレビュープロセス

クトへ反映されるまでの流れである.

1. 開発者は報告された不具合を修正するためのパッチを作成し，投稿する.
2. 投稿されたパッチは，他の開発者から修正内容が適切かどうかを確認するためにレビューを受け，フィードバックが返される.
3. パッチが適切であると判断された場合は，コミッターによってパッチを適用したソースファイルが，構成管理ツールへコミットされ，プロダクトに反映される.
4. パッチの投稿やレビューと並行して，コメントを通じて不具合に対する議論が行われる.

パッチの投稿やパッチに対するレビューは，コミッター・非コミッター共に行うことが可能であるが，修正されたソースコードを構成管理ツールへコミットを行うことができるのはコミッターのみである.

4　OSS 開発におけるコミッター選出

4.1　コミッター選出の現状

パッチレビュープロセスにおいてコミッターは，パッチを承認し，プロダクトへ反映する必要不可欠な役割を担う．Apache プロジェクトでは，非コミッターの不具合報告やパッチ投稿など，技術的な貢献を既存のコミッターが認め，コミッターに選出している [4]．しかし，Apache プロジェクトを含め，多くのプロジェクトは，開発者の中からコミッターを選出するための明確な指標や基準を提示していない．OSS プロジェクトの公式ドキュメントとして，コミッターになるための方法が記載されている場合があるが，詳細はコミッターへの昇格が決まった後の事務的な手続き（規約への同意など）のみである．コミッターの選出は，既存のコミッターが彼らの主観で非コミッターの貢献を評価し，コミッターに昇格させるべきか否かを判断している.

また，コミッターに昇格した開発者がプロジェクト管理者の期待する活動を実施するとは限らない．非コミッターの中には，コミット権限を取得することを目標とし，コミット権限取得後，プロジェクトへの貢献が減少する者も存在する [6]．非コミッターがコミッター昇格後に積極的に貢献する内容を既存のコミッターが予め理解できていれば，新しくコミッターを選出するときに適切な非コミッターを選出することが容易になると考えられる.

4.2　Research Questions

本節では，コミッターの選出に参考にすべき活動を明確にするために以下のよう

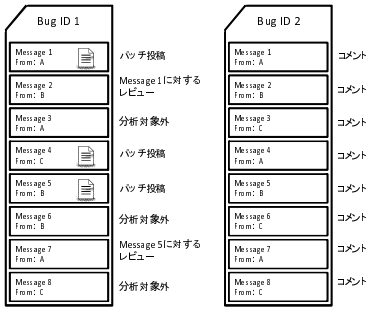

図 2　不具合管理システム内における，メッセージ毎の活動内容

な Research Questions を立てる．

RQ1 ：コミッター選出の指標として有効な活動は何か．
パッチレビュープロセスにおけるコミッター候補者，一般開発者の活動を比較し，活動量に違いがあった場合，その活動がコミッター選出のための指標に成り得ると考える．

RQ2 ：開発者はコミッターへの昇格前後でどのように活動が変化するか．
コミッター候補者がコミッターへ昇格後に積極的に貢献する活動を把握することによって，プロジェクト管理者がコミッター選出を行う際に，コミッターに期待する活動と，実際のコミッターの活動との差異を小さくすることができると考えられる．RQ2 では，既存のコミッターを対象に，コミッターへ昇格する前後の活動量を比較し，開発者の活動の変化を明らかにする．

4.3　開発者の活動内容の定義

本節では，Research Questions に答えるために，開発者毎に，パッチレビュープロセスにおける活動内容の計測を行う．パッチレビュープロセスにおける活動の分析項目として，図 1 より開発者の参加期間，活動回数，レスポンスの早さの 3 つのカテゴリが挙げられる．各カテゴリにおける活動内容のメトリクスとして，以下の 5 項目が挙げられる．

参加期間　各開発者毎の全分析期間において，1 件以上メッセージの投稿された月数．
平均パッチ投稿数　パッチ投稿数を各開発者のプロジェクト参加月数で割った数．
平均レビュー数　レビュー数を各開発者のプロジェクト参加月数で割った数．
平均コメント数　コメント数を各開発者のプロジェクト参加月数で割った数．
レビュー時間　開発者が過去に行ったレビューに要した時間（パッチの投稿された日時からレビューを行った日時までに要した時間）の中央値．

不具合管理システム内におけるメッセージの構成と，活動内容の定義について図 2 を用いて説明する．図 2 の左にはパッチを含む不具合報告の例，右にはパッチを含まない不具合報告の例を示す．それぞれの不具合報告の右には各メッセージがどの活動に該当しているかを示す．パッチを含む不具合票において，パッチが添付されたメッセージをパッチ投稿とし，パッチ投稿の次に投稿されたパッチを含まないメッセージ（ただし，パッチを投稿した開発者と同一人物だった場合はその次の開発者）をレビューと定義する．パッチを含まない不具合票におけるメッセージは全てコメントと定義する．また，パッチを含むメール内でパッチ投稿とレビューのどちらにも該当しなかったメッセージは，レビューとコメントどちらの可能性もあり，自動的に判別することが不可能なため，本研究では対象外とする．

5 分析方法

5.1 分析データの収集と整形

5.1.1 構成管理ツール

本論文では，コミッターと非コミッターを区別するために，Subversion（SVN）や Concurrent Versions System（CVS）などに代表される構成管理ツールの情報を用いる．構成管理ツールは，ファイルの各バージョンをリポジトリと呼ばれるデータベースに保持している．一般的な構成管理ツールは以下のように利用される．

1．ファイルをリポジトリに登録する
2．ファイルをリポジトリからローカル環境に取り出す（チェックアウト）
3．ローカル環境で，ファイルに対して変更を加える
4．変更したファイルをリポジトリに書き戻す（コミット）

多くの OSS では，このリポジトリ上のファイルが，リリースされる OSS プロダクトに直結しているため，コミットできる開発者を制限している．構成管理ツールには，ファイルがコミットされた履歴（コミットログ）を蓄積しており，コミッターに限らず誰でもコミットログを取得することが可能である．コミットログには 1 コミット毎に，リビジョン番号，コミットを行った開発者のアカウント名，コミット日時，変更行数，コミットを行った開発者のコメントが記録されている．本論文では，コミットログに記録された開発者アカウント名をリスト化し，不具合管理システムにおける，送信者メールアドレスとの一致がとれた開発者をコミッターと判断する．また開発者が初めて構成管理ツールにコミットを行った日時をコミッター昇格日時と定義する．

5.1.2 不具合管理システム

本論文では，多くの OSS 開発プロジェクトで使用されている，不具合管理システム”Bugzilla”における不具合票を対象とする．Bugzilla では，一つの不具合報告を一つの不具合票で管理しており，各不具合票には，不具合が発見されたプロダクトや不具合を修正するためのパッチ，修正に伴う議論が記録されている．本論文では，各開発者がコメントを投稿した日時，開発者名とメールアドレス，パッチの投稿回数，レビュー回数を収集する．

5.2 開発者の活動の把握

Research Questions に答えるために，パッチレビュープロセスにおけるコミッター，コミッター候補者，一般開発者の活動量を分析する．コミッター候補者の活動は各コミッターの昇格日時以前に行われた活動とし，コミッターの活動は各コミッターの昇格日時以降に行われた活動とする．各活動項目において，コミッター候補者・コミッター・一般開発者別の開発者の分布の比較にはウィルコクソンの順位和検定を用いる．本論文では，有意水準 5%で有意差のあった場合，コミッター選出の指標として使えると判断する．各 Research Question に対して，以下の分析を行う．

RQ1 ：コミッター候補者と一般開発者を比較，有意差の検定.
RQ2 ：コミッター候補者とコミッターを比較，有意差の検定.

RQ1 の分析は，4.3 節で述べた活動内容のメトリクス全てにおいて行う．RQ2 の分析は，4.3 節で述べた活動内容のメトリクスのうち，平均パッチ投稿数，平均レビュー数，平均コメント数，レビュー時間において行う．参加期間は，プロジェクトへの参加年数と共に，増加し続けるため，コミッター昇格前後の比較項目として不適切であると考え，RQ2 では対象としない．

5.3 分析の観点

OSS 開発の成功要因や，効率化を図った研究は数多く存在する．しかし，多くの研究では Research Questions や仮説を検証するために 2,3 件程度のケーススタディしか行われていない．検証したケーススタディにおいて，プロジェクトによって結

表 2　基本統計量

対象プロジェクト名		分析期間	不具合報告数	対象開発者数	
				コミッター	非コミッター
Apache	Ant	2000/09-2008/12	4439	27	1564
	HTTP Server	2002/06-2010/03	6164	62	3053
	Tomcat	2000/09-2008/12	4271	44	2064
Eclipse	BIRT	2005/01-2010/12	20459	51	1470
	JDT	2001/10-2010/09	12960	54	2478
	Modeling	2002/09-2010/12	21636	75	1623
	PDE	2001/10-2008/12	5099	19	724
	Platform	2001/10-2009/12	82271	55	9497
	RT	2001/10-2010/12	16372	32	2258
	Technology	2004/01-2010/03	16383	85	1576
Fedora	Fedora	2001/06-2008/12	11735	34	3707
FreeDesktop	FreeDesktop	2003/01-2010/12	31169	77	8424
GCC	GCC	2000/03-2010/12	37028	268	2857
Globus	Globus	2002/02-2010/10	7108	21	590
Mess	Mess	2001/10-2010/10	1998	12	160
Mozilla	Bugzilla	1998/05-2010/12	15968	116	2968
	Calender	2001/11-2010/12	11312	106	3694
	Camino	2002/05-2010/12	9854	101	2287
	Fennec	2008/03-2010/10	4982	66	263
	Firefox	2004/01-2008/12	37478	216	13078
	SeaMonkey	2005/01-2010/10	30195	215	6521
	Thunderbird	2003/01-2010/10	30010	168	10874
OpenOffice	Database	2000/11-2010/05	5991	88	1471
	Drawing	2000/11-2010/05	3556	95	1324
	Framework	2000/11-2010/05	12168	81	3671
	Presentation	2000/11-2010/05	7223	99	2680
	Spreadsheet	2000/11-2010/05	11984	84	4712
	Wordprocessor	2000/11-2010/05	28866	81	9608
OsaFoundation	OsaFoundation	2003/01-2010/10	12761	53	383
Songbird	Songbird	2006/01-2010/05	20058	31	2363
Wireshark	Wireshark	2005/04-2010/10	6000	21	1406
Xfree86	Xfree86	2003/03-2010/10	1900	10	896

果や傾向が変わってくることが多く，得られた知見の OSS 開発に関する一般性が保証されていない場合が多い．本論文では，多くのプロジェクトを対象にケーススタディを行うことで，結果の一般性の向上が期待できる．

6　ケーススタディ

6.1　分析対象プロジェクト

本論文で対象とする OSS プロジェクトは，パッチの投稿・レビューの管理に不具合管理システム Bugzilla を使用しており，構成管理ツールに CVS もしくは SVN を使用している OSS プロジェクト 32 件を対象に分析を行う．以下に分析対象プロジェクトの概要を述べる．各プロジェクトに対する，分析期間，不具合報告数，コミッター候補者数，非コミッター数に関するそれぞれの統計量を表 2 に示す．

6.2　分析結果

6.2.1　コミッター選出の指標

図 3 に，コミッター候補者と一般開発者の活動量について，検定を行った結果を載せる．横軸に活動内容，縦軸にプロジェクト数の割合を示す．

参加期間　32 件中 27 件のプロジェクトプロジェクトでは，一般開発者の参加期間に比べてコミッター候補者の参加期間の方が長く，統計的有意差がみられた．よって，プロジェクトに長期間参加していることがコミッター選出の指標の 1 つと成り得ることがわかった．

月平均活動回数　平均コメント数に関して，32 件中 24 件のプロジェクトは，一般開発者間よりコミッター候補者の方が平均コメント回数が多く，統計的有意差がみられ，平均コメント数が多いことがコミッター選出の指標の 1 つと成り得ることがわかった．また，平均パッチ投稿数は 30 件中 11 件のプロジェクト，平均レビュー数は 28 件中 15 件のプロジェクトは，一般開発者よりコミッター候補者の方がそれぞれ多く，統計的有意差がみられたことから，平均パッチ投稿数が多いこと，平均レビュー数が多いこともプロジェクトによってはコミッター選出の指標に成り得ることがわかった．これらのことより，3 つの活動項目の

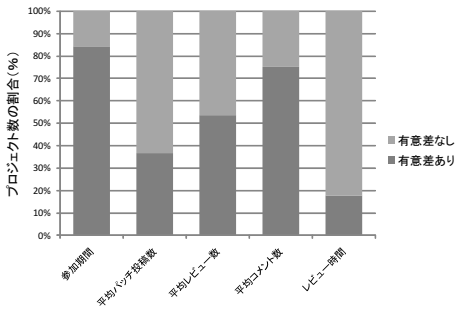

図 3　コミッター候補者と一般開発者での有意差検定結果

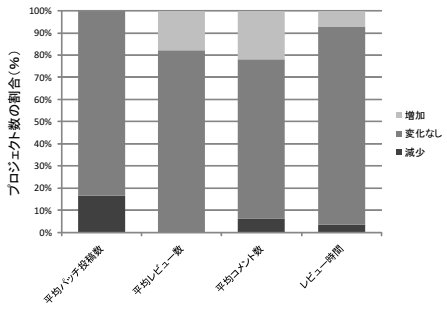

図 4　コミッター昇格前後での有意差検定結果

表 3　大規模プロジェクト内での平均活動回数が指標として有効なプロジェクトの割合

	平均パッチ投稿数	平均レビュー数	平均コメント数
Apache	2 件/3 件中 (67 %)	1 件/2 件中 (50 %)	3 件/3 件中 (100 %)
Eclipse	4 件/6 件中 (67 %)	4 件/6 件中 (67 %)	6 件/7 件中 (86 %)
Mozilla	3 件/7 件中 (43 %)	5 件/7 件中 (71 %)	5 件/7 件中 (71 %)
OpenOffice	1 件/6 件中 (17 %)	0 件/4 件中 (0 %)	3 件/6 件中 (50 %)

うち平均コメント数が最もコミッター選出に有効であることがわかった．

レスポンスの早さ　28 件中 23 件のプロジェクトでは，コミッター候補者と一般開発者のレビュー時間に統計的有意差はみられず，コミッター選出の指標として使えないことがわかった．一方で，RT，Calender，Firefox，Spreadsheet，Wordprocessor の 5 プロジェクトにおいて，コミッター候補者が一般開発者に比べてレビュー時間が短く，統計的有意差があったことから，レビュー時間が短いことがコミッター選出の指標の 1 つと成り得ることがわかった．

6.2.2　開発者の活動の変化

図 4 に，コミッター昇格前後の活動内容について，検定を行った結果を載せる．横軸に活動内容，縦軸にプロジェクト数の割合を示す．

月平均活動回数　コミッター候補者とコミッターに有意水準 5%で有意差があるプロジェクトは，どの活動内容においても半数以下である．（パッチ投稿数：30 件中 5 件，レビュー：28 件中 5 件，コメント数：32 件中 9 件）パッチ投稿数に変化のあったプロジェクトは 5 件中 5 件がコミッター昇格後パッチ投稿数が減っていることがわかる．レビュー数に変化のあったプロジェクトは 5 件中 5 件がコミッター昇格後レビュー数が増えていることがわかる．コメント数は一番多くのプロジェクトにおいて統計的有意差があり，コミッター昇格によって活動量が変化していることがわかる．

レスポンスの早さ　Spreadsheet，Wordprocessor，Osafoundation の 3 プロジェクトは，コミッター昇格によってレビュー時間に変化があることがわかった．また，変化のあった 3 プロジェクト中，Spreadsheet と Wordprocessor はコミッター昇格後にレビュー時間が増加し，Osafoundation では減少していることがわかる．レスポンスの早さはコミッター昇格によってどのように変化するか，予測することは困難であるといえる．

7　議論

7.1　コミッター選出の指標

RQ1 の結果から，多くのプロジェクトで参加期間や月平均活動回数がコミッター選出に有効であることが示された．しかし，Mozilla や Eclipse のように複数のプロ

表 4　Eclipse platform の開発者の活動量

	参加期間（月）			平均パッチ投稿数			平均レビュー数		
一般開発者	1.0	1.0	2.0	1.0	1.0	2.0	1.0	1.0	1.0
コミッター候補者	4.0	7.0	12.0	3.6	6.7	12.4	1.0	1.5	2.0
	平均コメント数			レビュー時間（分）					
一般開発者	1.0	1.5	2.3	313.8	1463.0	17657.0			
コミッター候補者	2.8	7.6	15.0	121.3	872.0	6091.8			

ジェクトが存在する場合，全てのプロジェクトについて RQ1 の結果が有効であるとは限らなかった．表 3 は，大規模が抱えるプロジェクトの中で平均活動回数がコミッター選出の指標として使えるプロジェクトの割合を示している．

表から，OpenOffice プロジェクトは，他の大規模プロジェクトと比べてパッチ投稿やレビューの指標が使えないプロジェクトが多い．これは，一般的なプロジェクトでは有能な開発者をコミッターによって選出される一方で，OpenOffice プロジェクトは開発者自身がコミッターへの昇格を希望し，承認される場合があるからと考えられる．

また，Mozilla プロジェクトは他の大規模プロジェクトより，レビュー数を指標として有効であるプロジェクトが多い．Mozilla では，コードの品質を向上させるためにパッチの検証を専門的に行う Superreviewer という役割が存在し，Superreviewer を目指す非コミッターが積極的にレビューを行っていることが理由として考えられる．

本論文で挙げた指標の有効性は，プロジェクトの運用方針やプロジェクト規模によって異なる場合がある．プロジェクト管理者はプロジェクトに参加する開発者の目的や，どのようなコミッターを必要としているかを考慮して指標を選択する必要があると考えられる．

コミッター選出の目安として，全ての指標で有意差が見られた Eclipse platform に参加する非コミッター，コミッター候補者の活動期間と月平均活動回数の第 1 四分位数，中央値，第 3 四分位数を表 4 に示す．Eclipse platform では，参加期間，平均パッチ投稿数，平均コメント数は非コミッターとコミッター候補者に明確な違いがあるため，目安となると考えられる．

7.2　開発者の活動の変化

コミッターへの昇格前後で活動回数に変化があり，統計的有意差の見られたプロジェクトは半数以下であった（パッチ投稿数：30 件中 5 件，レビュー：28 件中 5 件，コメント数：32 件中 9 件）．この中で，Eclipse Platform は，パッチ投稿数，レビュー数，コメント数の 3 項目全てにおいてコミッター昇格前後の活動量に有意差があったプロジェクトである．図 5 に，Eclipse Platform における，コミッター昇格前後の活動の分布を示す．Eclipse Platform のコミッターは非コミッターからコミッターへ昇格することによって，パッチ投稿数は減少，レビュー数，コメント数は増加している．Eclipse Platform 以外のプロジェクトでも，パッチ投稿数に変化のあったプロジェクトはコミッター昇格後にパッチ投稿数が減少，レビュー数に変化のあったプロジェクトはコミッター昇格後にレビュー数が増加，コメント数に変化のあった 9 件中 7 件のプロジェクトでコミッター昇格後に増加している．コミッター昇格後，コミッターは一般開発者へのアドバイスや議論，パッチの検証にコストをかけるためパッチの投稿数が現象していると考えられる．

7.3　本論文の制約

本論文では，不具合管理システムとして Bugzilla，構成管理ツールとして CVS もしくは Subversion を利用しているプロジェクトのみを分析対象とした．近年では分散型リポジトリ GIT や Mercurial を利用しているプロジェクトが増加している．分散型リポジトリの特徴の一つは，全ての開発者がそれぞれローカルの計算機でリポジトリを管理しており，誰でもコミットを行うことができる点である．ただし，最終的にマージを行うことができるのは権限を持った一部の開発者のみである．集中

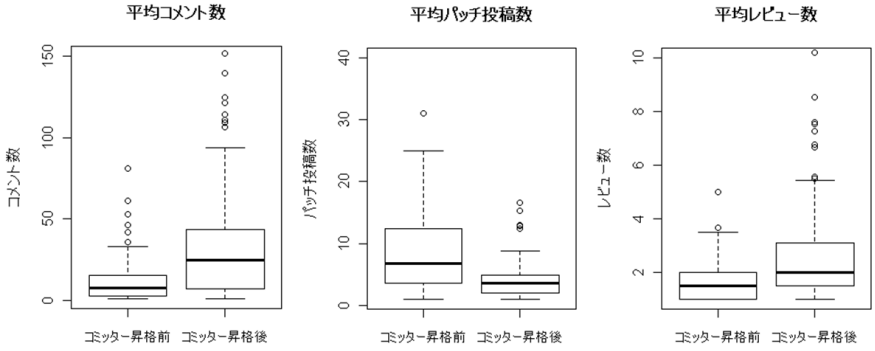

図 5　Eclipse Platform のコミッター昇格前後の活動回数

型リポジトリと分散型リポジトリでの開発者の役割を単純比較することは不可能であるため，本研究では分析対象外とし今後の課題とする．

また，一部のプロジェクトでは，ボランティアの開発者だけでなく企業の開発者も参加している場合があり，企業の開発者は優先的にコミッターとして選出されている可能性が考えられる．また，各プロジェクトにおける分析期間の初めから既にコミッターとして活動を行っている開発者も多く存在しており，分析結果に影響している可能性がある．

本論文では，開発者同士による IRC などの記録に残っていない活動についての影響は調査していない．今後，開発者へのインタビューなどを通じて今回対象としなかった開発者の活動の影響についても調べる必要がある．

本論文では非コミッターが昇格後，プロジェクトへの貢献について分析していない．そこで，Eclipse platform の開発者が昇格後も継続しているか否かを調査した．コミッター昇格前にパッチ投稿，レビュー，コメントのいずれかの活動を行っている 55 人の開発者のうち 44 人（約 80%）は昇格後も 1 年以上継続している．Eclipse platform プロジェクトのみ確認を行った結果，本論文で挙げた指標は，継続して活動するコミッター選出に有用であると考えられる．今後はコミッター昇格後も継続してプロジェクトに貢献するコミッターを推薦する方法を検討する．

8　おわりに

本稿では，OSS 開発におけるコミッター選出のために，パッチレビュープロセスにおける活動を定義し，開発者の活動の特徴を分析した．32 件の OSS プロジェクトを対象にケーススタディを行った結果，得られた知見は以下の通りである．

- 参加期間，パッチ投稿数，レビュー数，コメント数は，約半数のプロジェクトにおいて有効な指標と成り得る．その一方で，レビュー時間は，一部のプロジェクトにおいてのみ有効な指標と成り得る．
- 一部のプロジェクトの開発者は，コミッターへの昇格によって活動量が変化することがある．また，活動量が変化する際にパッチ投稿数は減少，レビュー数は増加する傾向が強い．
- コミッター選出の指標や開発者の活動の変化はプロジェクト毎に異なるため，プロジェクトの特性に合わせたコミッター選出が必要とされる．

従来まで，プロジェクトのコミッターの主観で非コミッターの中からコミッター選出を行ってきた．本研究の分析を通して，一般開発者とコミッター候補者の活動に違いがあることに実証的な裏付けを与えることができた．今後，本論文で得られ

た知見を用いることで，プロジェクト管理者がコミッターの選出を容易に行うことが出来るようになると考えられる．各プロジェクトで有用なコミッター選出を実現し，不具合修正の長期化を解消することを期待する．

謝辞 本研究の一部は、文部科学省「次世代 IT 基盤構築のための研究開発」の委託に基づいておこなわれた．また、本研究の一部は、文部科学省科学研究補助費（基盤 B：課題番号 23300009，若手 B：課題番号 22700033）による助成を受けた．

参考文献

[1] Akinori Ihara, Masao Ohira and Ken-ichi Matsumoto: An analysis method for improving a bug modification process in open source software development, Proceedings of the joint international and annual ERCIM workshops on Principles of software evolution and software evolution workshops (IWPSE-Evol'09), pp.135–144, 2009.

[2] Bianca Shibuya and Tetsuo Tamai: Understanding the process of participating in open source communities, Proceedings of the 2009 ICSE Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development (FLOSS'09), pp.1–6, 2009.

[3] Brent Hailpen and Padmanabhan Santhanam: Software debugging, testing, and verification, IBM Systems Journal, pp4–12, 2002.

[4] Chris Jensen and Walt Scacchi: Role Migration and Advancement Processes in OSSD Projects: A Comparative Case Study, Proceedings of the 29th International Conference on Software Engineering (ICSE'07), pp.364–374, 2007.

[5] Christian Bird, Alex Gourley, Prem Devanbu, Anand Swaminathan and Greta Hsu: Open Borders? Immigration in Open Source Projects, Proceedings of the Fourth International Workshop on Mining Software Repositories (MSR'07), 2007.

[6] Karl Fogel: Producing Open Source Software: How to Run a Successful Free Software Project, O'Reilly Media, Inc, 2005

[7] Nicolas Bettenburg, Sascha Just, Adrian Schröter, Cathrin Weiss, Rahul Premraj and Thomas Zimmermann: What makes a good bug report?, Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of software engineering (FSE'08), pp.308–318, 2008.

[8] Pieter Hooimeijer and Westley Weimer: Modeling bug report quality, Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering (ASE'07), pp34–43, 2007.

[9] Yi Wang, Defeng Guo and Huihui Shi: Measuring the evolution of open source software systems with their communities, SIGSOFT Softw. Eng. Notes, Vol.32, No.6, 2007.