# GitHub 上のソフトウェア開発のためのフロー推薦手法

## 若月 純 † 矢吹 太朗 千葉工業大学 社会システム科学部 プロジェクトマネジメント学科 †

#### 1 序論

ソフトウェア開発では、複数のメンバが同時に開発を行うため、ファイルの最新バージョンが分からなくなる、同一ファイルに対する変更が競合する等の問題が発生する.このような問題を解決するため、バージョン管理システムを用いる.バージョン管理システムとは、変更履歴を管理するシステムのことである[1].

バージョン管理システムを提供するサービスに、GitHub がある. GitHub は、バージョン管理システムに加え、branch、Pull Request といった開発を補助する機能を提供するサービスである. branch とは、履歴を分岐して記録していくためのものである. branch を用いることにより、同一リポジトリ内で、別々の作業を並行して行うことが出来るようになる. Pull Request とは、自分のリポジトリから相手のリポジトリへ、変更を取り込んでもらうための要求を出す機能である. Pull Requestを用いることにより、変更が追加される前に確認することが出来る.

GitHub を使用する手順を開発フローと呼ぶ.現在わ かっている開発フローの数は 13 個ある [2]. 開発フロー の例を 2 つあげる . GitHub Flow は , 作業をする branch を作成し,完成したら統合する.といった開発フローで ある、この開発フローはとてもシンプルなため、開発フ ローを実施するまでの学習コストは抑えられるが,開発 規模が大きい場合, Pull Request がたまりやすく, コード レビューに時間がかかってしまうことがある. Git Flow は, develop branch から作業用 branch を作成する. 完 成したら Pull Request を行い, 作業用 branch を develop branch に統合する. リリースができるレベルになった ら,リリース用 branch を作成し,作業をする.リリー ス作業が終了すると master ブランチに統合され, バー ジョンタグを打ってリリースする.といった開発フロー である. branch 別にやることが決まっているため管理は 容易であるが, branch が複数あるため, Pull Request を 異なった branch に送ってしまう等の人的ミスが発生す

る場合がある[3].

このように開発フローは、メリットとデメリットがある.しかし、選択する基準は定められていないため、状況にあった開発フローを選択するのは難しい.そのため、適切でない開発フローを選択し、開発に悪影響を与える危険がある.このような事態を防ぐため、適切な開発フローを選択できるようにするための基準が求められる.

そこで、本研究は、適切な開発フローを選択できるようにするための基準を求めるため、GitHub上のプロジェクトを対象に、採用されている開発フローと、開発フローの採用に関わると思われる項目を調査し、分析した・

#### 2 目的

GitHub を用いたソフトウェア開発プロジェクトの性質において、適切な開発フローを選択できるようにするための基準を求める.この基準により、適切な開発フローを選択することができれば、プロジェクトを成功させるためのスケジュール、コスト、品質コントロールを補助することができる.

#### 3 手法

本研究は3段階に分かれる.

- 1. GitHub 上のプロジェクトから,採用されている開発フローと,開発フローの採用に関わると思われる項目を調査する.
- 2. 調査結果を分析する.
- 3. 分析結果の精度と再現率を求める.

初めに、GitHub 上のプロジェクトから、開発フローと開発フローの採用に関わると思われる項目を調査する、開発フローは、GitHub 上の branch と Pull Request の特性から求められる。branch に stable が用いられている場合は、Stable Flow である。master branch から記述的な名前の branch がある場合は、GitHub Flow である。develop branch と release branch がある場合は、Git Flowである・バージョンごとに branch が作られている場合は、LINE Flowである。Pull Request に WIP がある場合は、WIP Flowである。開発フローの採用に関わると思

Workflow recommendation method for software development on GitHub

<sup>&</sup>lt;sup>†</sup> Jun WAKATSUKI • Department of Project Management, Social System Sciences, Chiba Institute of Tchnology

われる項目は、GitHub 上のデータを用いる。GitHub 上のデータは、GitHub ブラウザに載っているデータと載っていないデータがある。載っているデータは、そのまま用いる。載っていないデータは、リポジトリをクローンして調査する。

次に,調査結果を分析する.調査したデータの分析は,決定木分析を行う.決定木分析により,プロジェクトがどのような性質を持つときに,どの開発フローが使われているかを明らかにする.决定木分析の目的変数は開発フロー,説明変数は,開発フローの採用に関わると思われる項目を用いる.

最後に,分析結果の精度と再現率を求める.調査したデータをランダムに2種類に分ける.片方のデータで決定木分析を行う.决定木分析結果ともう片方のデータを照らし合わせて,決定木の精度と再現率を調べる.これを10回行い,信頼度95%区間の精度と再現率を用いる.

#### 4 結果

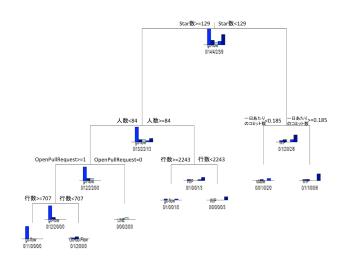


図1 プロジェクトの性質により選択される開発フローの違い

決定木分析結果は、図1である.分析から、プロジェクトの性質により選択される開発フローが明らかにされた.

GitHub 上の 32 個のプロジェクトから, プロジェクトの開発フローと性質を調査し,決定木分析を行った.また,決定木分析結果の精度と再現率を求めた.

開発フローは, Git Flow, GitHub Flow, LINE Flow, Stable Flow, WIP Flowの5種類だった.

プロジェクトの性質は、プロジェクト経過日数,行数,ファイル数,バイト数,Watch数,Star数,Fork数, Commit数,branch数,Release数,人数,Open Issue数, Closed Issue数,Issue数,Open PullRequest数,Closed PullRequest数,PullRequest数,Abel数,Open Milestone 数 , Closed Milestone 数 , Milestone 数 , Wiki 数 . 開発 人数 , 言語 , 一日あたりの行数と Commit 数 , 1 人日あ たりの行数と Commit 数を調査した . 言語は 26 種類の 項目を作成し , プロジェクトで使用している場合 1 , 使 用していない場合 0 で判別した . 26 種類に当てはまら ない場合は , その他にまとめた .

决定木の精度と再現率について記述する.32 件のデータをランダムに22 件と10 件に分け,精度と再現率を求めた結果,精度の平均は41%,95% 信頼区間は2656% だった.再現率の平均は51%,95% 信頼区間は2973% だった.

#### 5 考察

この決定木は、全データをまず Star 数で分類している.Star 数とは、注目度を表す基準である.Star 数が多い場合、プロジェクトを主に branch で管理する git flowを選択している.Star 数が少ない場合、プロジェクトを主に Pull Request で管理する wip flow が多く選択されている.ここから、チェックしているユーザが多いプロジェクトの場合、主に branch で管理し、チェックしているユーザが少ないプロジェクトの場合、主に Pull Request で管理していることが言える.

また,一日あたりのコミット数で分かれていることから,プロジェクトの生産性により適切なフローが代わることが言える.

### 6 結論

本研究により、プロジェクトの性質において、開発フローを推薦する手法を確立した.しかし、精度と再現率が低いことから、開発フローの採用に関わる項目を調査できていない可能性がある.プロジェクトの生産性といった時系列データにより、適切なフローが代わることから、コミットが増える時間帯や人数の増加傾向等、他の時系列データを調査することが今後の課題である.

#### 参考文献

- [1] 池田尚史, 藤倉和明, 井上史彰. チーム開発実践入門 ~共同作業を円滑に行うツール・メソッド. 技術評論 社, 2014.
- [2] 小野寺航己. バージョン管理システムを活用するソフトウェア開発の開発フロー. 卒業論文, 千葉工業大学, 2015.
- [3] 大塚弘記. GitHub 実践入門 Pull Request による開発 の変革. 技術評論社, 2014.