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 研究の背景

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指標が求められる．

本研究では、GitHub 上の 32 件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 48 の調査結果を解析した。その結果、プロジェクトの性質に応じて，開発フローを推薦する手法を確立した。

2 研究の目的

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提供する．

3 プロジェクトマネジメントとの関連

ソフトウェア開発プロジェクトにおいて，プロジェクトマネージャーは, QCD を達成させるためにスケジュール，コスト，品質コントロールを行う．これらのコントロールを，GitHub を用いた開発フローを導入し，テストを自動化したり，誤操作を防いだりすることで補助することが出来る．

4 研究の方法

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. 調査結果を解析する．
3. 解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，Pull Request の特性から求められる．branch に stable が用いられてい

Workflow recommendation method for software development on GitHub

† Jun WAKATSUKI・Department of Project Management, Social System Sciences, Chiba Institute of Tchnology

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，調査したデータを分析する．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．

最後に，解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 22 件と 10 件の 2 種類に分ける。22 件のデータで決定木分析を行う．決定木分析結果と 10 件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．

5　成果物のイメージ

調査データを用いて決定木分析を行い，開発フローを選択する基準を求める．選択する基準は，次に書く例のように定量的になっている．人数が 5 人以上の時かつ，使われている言語が Ruby の時は，GitHub Flow を選択する．

また，選択する基準の精度と再現率を求める．精度と再現率は 50% 以上になることを目指す．

6　現在の進捗状況

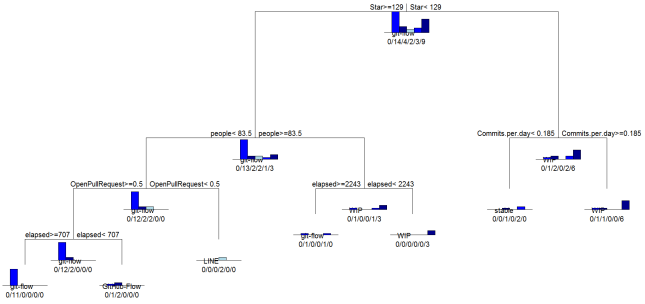

図 1　プロジェクトの性質により選択される開発フローの違い

GitHub 上の 32 個のプロジェクトから，プロジェクトの性質と開発フローを調査し，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。

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 Issues 数，Closed Issues 数，Issues 数，Open PullRequest 数，Closed PullRequest 数，PullRequest 数，Label 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 Commits 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。

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

決定木分析結果は，図 1 である．分析結果から，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．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，注目度を表す指標である．star 数が多い場合 git flow，star 数が少ない場合、wip flow が多く選択されている。ここから、常に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，常に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．

決定木の精度と再現率について記述する．精度の平均は 41%，95% 信頼区間は 26~56% だった．再現率の平均は 51%，95% 信頼区間は 29~73% だった．

7　今後の課題

この分析で，一日あたりの行数と Commit 数，1 人日あたりの行数と Commits 数は，平均を用いている．プロジェクトの傾向を考慮していないため，特定の曜日に Commit が増える場合や，Pull Request を確認する日等がある場合でも同様のフローになってしまう．そのため，より詳細なプロジェクトの傾向を調査し，分析を行えば，決定木の精度と再現率をあげられると考える．

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.