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

ソフトウェア開発コース　矢吹研究室　1142032 小野寺航己

1. 研究の背景

　当研究では，バージョン管理システムGitHubを用いたソフトウェア開発プロジェクトにおける，開発フローを対象とする．

バージョン管理システムは，何度も変更を加えたファイルであっても，作成日時や，編集日時，変更点を随時保管しておくことで，複数の人間が過去のファイルや，変更点の確認，ファイルの状態を復元することなどの，管理を可能とするシステムである．

　代表的なバージョン管理システムとしてGitがある．Gitでは，このシステムを使っている各利用者が自由にできる領域（ローカルリポジトリ）に，Gitに保管されたデータや情報の全履歴を含んだ，完全な複製が作られる．このような仕組みがあるため，ネットワークにアクセスできなくても，履歴の閲覧や変更の記録といった，作業ができる．

　このGitを用いたウェブシステムとしてGitHubがある．GitHubはGitの機能を提供するウェブサービスであり，世界中の人々が自分の作品を保存、公開することもでき，ソフトウェア開発プロジェクトのための共有サービスでもある．

ソフトウェア開発では，一つのソフトウェアに対して複数のメンバが同時に編集を行うことで，複数のリリースバージョンがある中で，機能の追加やバージョン管理といった，作業を並行して行わなければならない状況がある．

Gitにはそのような状況を支援する機能としてブランチがある．ブランチは，履歴の流れを分岐して記録できる機能で，同じソフトウェアに対して複数の変更を，履歴を保持しつつ同時におこなうことができる．この機能があるため，ソフトウェア開発のツールとしてGitHubが用いられることが多い．

1. 研究目的

GitHubを用いたソフトウェア開発プロジェクト毎に，適切な開発フローを選択する基準を明確にする．

1. 研究方法

GitHubを用いる開発フローを網羅的に調査して，調査した開発フローのリスクを明らかにする．そのリスクをクラスター分析により分類し，その後，プロジェクトの性質に応じて適切な開発フローを選択できるようなガイドを作成する．

1. 結果

調査したGitHub Flow[1], Git Flow[1], はてなブログチームでの開発フロー[2]などの，13種類のGitHubを用いた開発フローを，発生しうるリスクを元にクラスター分析したところ，表1のような5つのクラスターに分類することができた．クラスター1は『フローが自動化されている』，クラスター2は『リポジトリを複数使用しているもの』，クラスター3は『使用したブランチを破棄するもの』，クラスター4は『アジャイル開発のような流れが可能なもの』，クラスター5は『使用するブランチが少ないもの』という共通点があった．

表1．開発フロー分析結果

|  |  |
| --- | --- |
| 1 | フィヨルドフロー,イストフロー |
| 2 | Amingフロー,サイボウズフロー, 矢吹研フロー①,矢吹研フロー② |
| 3 | ラクスルフロー, LINEフロー |
| 4 | Gitフロー, キャスレーフロー |
| 5 | GitHubフロー,はてなブログフロー,日本CAWフロー |

5．考察

　分析結果から，適切な開発フローを選択する基準は『フローを自動化することで，定期的にテストを行いたいかどうか』，『リポジトリを複数使いたいかどうか』，『ブランチを盛り込むべき機能やバージョンで管理したいのか』，『アジャイル開発でプロジェクトを行うのか』，『プロジェクトメンバにかかる負荷を少なくしたいのか』という点が開発フロー選択の基準になるのではないかと考える．

参考文献

[1] 大塚弘記. GitHub実践入門 Pull Requestによる開発の変革. p. 199-253. 技術評論社, 2014.

[2] 新野淳一. はてなブログチームの開発フローとGitHub（前編）. Publickey.

<http://www.publickey1.jp/blog/14/githubgithub_kaigi_2014.html（参照2014-09-19>）