プログラミング初学者向けの Git/GitHub 操作支援手法・データ収集方法

玉田 春昭[1]

概要：今日の開発ではバージョン管理ソフトウェア（SCM）に加え，ソーシャルコーディングプラットフォーム（SCP）と呼ばれるサービスも併用していることが当たり前になっている．しかし，これらのソフトウェア，サービスを利用した開発に習熟することは容易なことではない．一般的に初学者は，プログラミング上の問題を解決することに多くの労力を投入し，開発の版管理や課題管理にまで手が回らない．加えて，版管理や課題管理の実施によって開発が進んだように見えないため，実施するモチベーションが高くない，というのも習熟困難である理由の遠因となろう．本稿では初学者がこれらの SCM や SCP を学習する際に，如何に支援できるかを議論する．

1. はじめに

Stackoverflow の調査によると，82.8%の開発者が開発におけるコミュニケーションツールとして GitHub を利用していると回答している[*1]．この値は 2 位の Slack（53.0%）を大きく引き離している．このように GitHub は開発用のツールとしてデファクトスタンダードとなっている．GitHub を利用するには，前提として git の利用も不可欠となる．これから開発を始めようという初学者にとって，これらのツールの利用はハードルが高い．ただでさえ複雑なフレームワークやライブラリを駆使してプログラムを書かなければならない．その上プログラムではない部分，すなわち，版管理や課題管理についても学習が必要となるためである．加えて，版管理や課題管理を丁寧に行ったとしても，開発の進捗が効率化するわけではない．そのため学習が後回しとなり，習熟が困難になっていると思われる．

これまで git や GitHub を利用した授業設計 [1], [2] や，チュートリアル[*2]は数多く提案されている．一方で，git や GitHub の操作自体を支援する方法についてはあまり議論されていない．そこで本稿では Git/GitHub の操作支援とそのために必要なデータ収集について議論する．

2. 初学者向け Git/GitHub 操作支援

初学者向けの Git/GitHub 操作支援は大きく，開発を行う前の支援（事前学習支援），プロジェクトの進行中での支援（インプロセス支援 [3]），プロジェクト終了後，もしくは中断中の支援（リフレクション支援）に分けられる．加えて，モチベーションを一定に保つためのモチベーション支援も考えられる [4]．それぞれについて，これまでの取り組みと今後の方向性について以降で述べる．

2.1 学習支援

git の書籍や学習サイト，チュートリアルなどが学習支援に該当する．大学などでの講義での学習も，この学習支援に該当するであろう．この学習支援では，シンプルな学習用のリポジトリや環境を用いることが多い．代表的な例として，Git-it[*3]や Getting Git Right[*4]，サル先生の Git 入門[*5]などが挙げられる．

2.2 インプロセス支援

先に述べたように，プロジェクト進行中に利用者の Git/GitHub 利用を支援する手法である．ペアプログラミングやモブプログラミングなどで初学者に対して熟練者が Git/GitHub 操作をサポートすることが該当するであろう．

我々の研究グループではより能動的に支援する方法として，git や GitHub 操作にインセンティブを与える手法 [5]，そして，CUI 上での git 操作の推薦手法 [6] を提案している．

2.3 リフレクション支援

リフレクション支援は，開発が終了した時，もしくは中

[1] 京都産業大学
Kyoto Sangyo University, Kyoto 603–8555, Japan

[*1] https://insights.stackoverflow.com/survey/2020
[*2] https://learngitbranching.js.org/
[*3] https://github.com/jlord/git-it-electron
[*4] https://www.atlassian.com/git
[*5] https://backlog.com/ja/git-tutorial/

表 1　環境・支援方法により収集するデータ

環境	学習支援	インプロセス支援	振り返り支援	モチベーション支援
エディタ		編集履歴	編集履歴	編集履歴，滞留時間
ターミナル	熟練者のコマンド履歴	コマンド履歴，実行結果	コマンド履歴	コマンド履歴，滞留時間
ブラウザ	ページ遷移履歴	操作記録，ページ遷移履歴	操作記録，ページ遷移履歴	
その他			git log	

断中にこれまでの開発状況の振り返りを実施するときの支援方法である．git などのログからのメトリクスによるバージョン活動の品質測定 [7] や GitHub flow の遵守度の測定 [8] が挙げられる．

2.4　モチベーション支援

git や GitHub の支援にあたり，学習者のモチベーションを保つための方策のための支援である．例えば，外発的動的付けとしての，トロフィー（実績）制度の導入 [4], [5] や，内発的動機付けとして，Git/GitHub のより具体的な必要性の教育などであろう．

3.　それぞれの支援方法のデータ収集

3.1　収集方法

初学者の Git/GitHub の利用支援には，インプロセス支援，リフレクション支援では特に，開発者の行動の把握が不可欠である．初学者の Git/GitHub に関係する行動は，主に IDE（エディタ），ターミナル，ブラウザで実施される．そのため，それぞれで行動の収集が必要である．

IDE（エディタ）からの情報収集は，細粒度の履歴管理 [9], [10] や，Language Server Protocol の利用が考えられる [11].

ターミナルでは，`history` コマンドや `kani`[6] により関連するコマンド履歴を記録できる．ただし，いずれの場合でも実行結果の記録は `script` コマンドを併用するなどしなければ収集できない．

一方，ブラウザでの開発者の行動履歴の記録は，ブラウザの訪問記録で一部の行動は収集できるものの，各ページで行った操作や滞留時間などの情報は収集できない．これらの情報を収集しようとすると何らかのプラグインの開発が必要となろう．ただし GitHub であれば，GitHub API を用いて操作履歴については収集可能である．

3.2　収集データ

では，どのようなデータを収集するとのちの分析に役立つのであろう．表 1 に，環境ごと，支援方法ごとに収集するデータを列挙した．ただし，具体的な支援方法は未定のものもある．

4.　まとめと議論の種

本稿では，Git/GitHub 操作学習のためのデータ収集フレームワーク並びに収集データについて議論した．支援の種類を学習，インプロセス，リフレクション，モチベーションに分け，また，開発に利用する環境ごとに収集すべきデータ，収集方法を整理した（表 1）．

今後，ターミナルでの git 操作のインプロセス支援，リフレクション支援を中心として，具体的な支援アプリケーションの構築を行っていく．そして，それら支援方法の有効性を評価していく予定である．

参考文献

[1] Hart, D.: A Survey of Source Code Management Tools for Programming Courses, *Journal of Computing Sciences in Colleges*, Vol. 24, No. 6, pp. 113—114 (2009).

[2] Kelleher, J.: Employing git in the classroom, *2014 World Congress on Computer Applications and Information Systems (WCCAIS)* (2014).

[3] 松村知子，森崎修司，勝又敏次，玉田春昭，吉田則裕，楠本真二，松本健一：問題の早期発見・改善を支援するインプロセスプロジェクト管理手法の実プロジェクトへの適用，電子情報通信学会論文誌，Vol. J92-D, No. 11, pp. 1974–1986 (2009).

[4] 玉田春昭：初学者向け GitHub flow 学習支援，ソフトウェアシンポジウム 2019 ワークショップ WS2 (2019).

[5] 西尾泰介，柳川龍太郎，玉田春昭：初学者を対象とした GitHub flow 支援ボット，ソフトウェアシンポジウム 2018 WS4 (2018).

[6] 増田亜里紗，玉田春昭：プログラミング初学者向け Git/GitHub 操作支援フレームワークの設計と実装，第 20 回情報科学技術フォーラム（FIT 2021）(2021).

[7] Elsen, R., Liem, I. and Akbar, S.: Software versioning quality parameters: Automated assessment tools based on the parameters, *Proc. 2016 International Conference on Data and Software Engineering (ICoDSE)* (2016).

[8] 井上拓海，小島遥一郎，藤原賢二，井垣　宏：版管理システム利用時のソフトウェア開発フロー遵守状況可視化手法の検討，信学技法, No.SS2017-55 (2018).

[9] 大森隆行，丸山勝久：プログラム開発履歴調査のための編集操作再生器，コンピュータソフトウェア，Vol. 28, No. 4, pp. 4_371–4_376 (2011).

[10] Proksch, S., Nadi, S., Amann, S. and Mezini, M.: Enriching in-IDE process information with fine-grained source code history, *Proc. IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER 2017)*, (online), DOI: 10.1109/saner.2017.7884626 (2017).

[11] 石田直人，神田哲也，嶋利一真，井上克郎：言語サーバを応用した細粒度編集履歴収集プラットフォームの構想，ソフトウェアエンジニアリングシンポジウム 2020 WS5 (2020).