Collaborative Writing on GitHub: A Case Study of a Book Project

Ei Pa Pa Pe-Than
Carnegie Mellon University
Pittsburgh, PA 15213, USA
eipapapt@cs.cmu.edu

James D. Herbsleb
Carnegie Mellon University
Pittsburgh, PA 15213, USA
jdh@cs.cmu.edu

Laura Dabbish
Carnegie Mellon University
Pittsburgh, PA 15213, USA
dabbish@cs.cmu.edu

CSCW '18 Companion, November 3–7, 2018, Jersey City, NJ, USA

ACM ISBN 978-1-4503-6018-0/18/11.
https://doi.org/10.1145/3272973.3274083

Abstract
Social coding platforms such as GitHub are increasingly becoming a digital workspace for the production of non-software digital artifacts. Since GitHub offers unique features that are different from traditional ways of collaborative writing, it is interesting to investigate how GitHub features are used for writing. In this paper, we present the preliminary findings of a mixed-methods, case study of collaboration practices in a GitHub book project. We found that the use of GitHub depended on task interdependence and audience participation. GitHub's direct push method was used to coordinate both loosely- and tightly-coupled work, with the latter requiring collaborators to follow socially-accepted conventions. The pull-based method was adopted once the project was released to the public. While face-to-face and online meetings were prominent in the early phases, GitHub's issues became instrumental for communication and project management in later phases. Our findings have implications for the design of collaborative writing tools.

Author Keywords
Co-creation; collaboration; coordination; collaborative writing; mixed-methods research; social computing.

Methods

We used a mixed-methods approach—a combination of semi-structured interviews and archival analysis of project artifacts.

Study Procedure: GitHub's non-software projects were identified through keyword searches and reviews of references in previous studies [6]. The projects were selected based on three criteria: (1) type of contribution (i.e., collaboration or collection), (2) amount of contribution, and (3) popularity based on GitHub's features such as stars, watchers, and forks.

For each selected project, we conducted interviews with both central users (either members of the core project team or five top contributors) and peripheral users (contributors who made at least one contribution). The project archives such as blog posts, wikis, and activities on GitHub were also collected.

ACM Classification Keywords

• Human-centered computing~Collaborative content creation

Introduction

More and more work nowadays—ranging from software development to text document creation—is being accomplished in networked digital environments. Beyond tools traditionally used for collaborative writing, people are increasingly using social coding platforms such as GitHub [2] to produce text artifacts such as books and policy statements [6, 8].

GitHub is a social coding platform which provides features specific for software development such as version control using Git, diff display, issues, pull requests, and forks and branches. These features are further complemented by social media style features such as star, followers/following, and badges [2]. GitHub has been widely adopted by the open source software development community, having more than 24 million developers working across 67 million repositories in 2017 [3].

Much research done to date has focused on the impacts of GitHub on collaboration in software development [2, 4]. These studies suggest that GitHub affords transparency of activities which, in turn, increases awareness of each other's activities, and reduces the need for additional communication [2]. However, very few studies focus on the use of GitHub for collaboration on text documents.

GitHub differs greatly from other tools used for writing such as wikis in which users can synchronously and directly edit an "Article" or content page and separately create comments on "Talk" page [5]. In contrast, GitHub allows collaborators to work in isolation by copying ("forking") the project repository, making changes in their local environments, and submitting their changes directly to the shared repository if they have commit access. Otherwise, they need someone to review their contributions, and hence submit as a "pull request". GitHub users can associate an issue with a pull request to discuss a specific part of the main article, which makes it easier for collaborators to navigate between discussion and actual changes [2].

Understanding some of the specific ways that GitHub's features facilitate collaboration on text documents is important because such understanding could contribute to the development of additional sets of features that can be applied to more familiar collaborative writing tools. Further, understanding how GitHub is used for non-code projects would allow us to understand the various ways that collaborative work on a wide variety of artifacts, both code and non-code, can occur within a networked digital environment.

We adopted a mixed-methods approach (see Box for detail)—a combination of interviews and the analysis of archival data—to study collaborative writing on GitHub. In this paper, we report the preliminary findings from interviews with contributors of a book project in GitHub—HoTT Book[1] (A textbook on homotopy type theory). Our interviews focused on why the team decided to put their project on GitHub, how GitHub features were utilized, what other tools they used along with GitHub, and what benefits and challenges they encountered.

[1] https://github.com/HoTT/book/

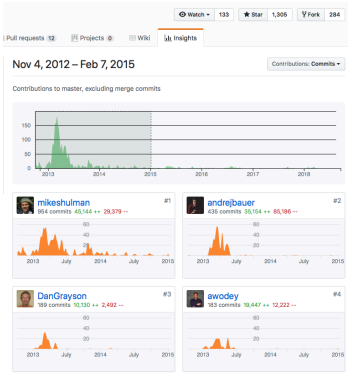

Figure 1: HoTT Book's contributor page in GitHub with four top contributors in the first two years.

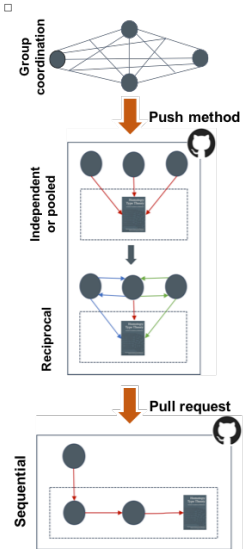

Figure 2: HoTT Book's workflow in GitHub.

Results

HoTT Book has 72 contributors who submitted 3651 commits, 554 issues, and 433 pull requests in 2018. We conducted interviews with four contributors of HoTT Book, three with central contributors and one with peripheral users. These interviews lasted between 44-85 minutes. We analyzed the collected data using grounded theory procedures [1]. Figure 1 and 2 respectively show the contribution activity and workflow of HoTT Book in GitHub.

Phase 1: Group Collaboration

The "group collaboration mode" was mainly adopted the very early phase of the project. The members performed tightly-coupled work, requiring them to work at the same time and provide immediate feedback [7]. These involve discussions about the content and organization of the book, division of labor, and coordination of work. The team used a wiki to record ideas, and a mailing list was also used to disseminate information among members and to organize events.

The adoption of GitHub was mainly driven by two factors: (1) having at least two members who were GitHub expert and the entire team consisted of a mix of people with varying levels of familiarity with GitHub, and (2) the team's prior experience with GitHub for code-based projects, i.e., proof assistant code. In addition to providing technical assistance to other members, one expert took on a "technical dictator" role by creating a collection of commands for mathematical formulas or symbols (i.e., macros), while the other expert took on a "technical editor" role by ensuring the consistency of terminology and macros throughout the book. One or two members were assigned as owners of each chapter or section.

Phase 2: Independent Coordination

During the actual writing process, "independent coordination mode" was adopted. The members worked in isolation for a loosely coupled task (or tasks with pooled dependencies [7]) such as content generation for different chapters or sections of the book. The "push method" was used and each chapter owner submitted their contributions directly to the project's GitHub repository. GitHub's issues were also used as a project management tool. For instance, the team created a task list (e.g., review a chapter) and assigned it to members with relevant skills or those who volunteered. This raised awareness about tasks being performed by members, thereby ensuring that only one person at a time edited a particular part of the book, which we called "social locking."

Phase 3: Sequential Coordination

Once the book was released to the public, the team adopted "pull-based method" [4]. That is, every change had to go through the mandatory review process and hence tasks in this phase had "sequential dependencies" [7]. One team member reviewed the requested changes and another member merged the approved changes. Most communication was done via issues and comments. There were occasional uses of email between a subset of team for private discussion.

Although the team aimed to crowdsource ideas or minor edits (e.g., typos and formatting issues) from the public, they realized that the members of the public had also contributed math-related content such as solutions for exercises, an improvement on a particular theorem, and bugs in mathematical proof. In addition, some contributors who were not part of a core project team had become regular contributors, suggesting that

GitHub facilitates the formation of a community around the project.

Conclusion

Our preliminary findings suggest that GitHub could be a useful tool for co-creation of text documents as its technical and social features supported well both loosely- and tightly-coupled work. However, socially-accepted conventions and practices (e.g., social locking, mandatory review) need to be in place for effective collaboration especially after putting the project on GitHub and making it public. We propose that task interdependency and participation of target audience should be considered in the development of such social conventions and practices. Taken together, GitHub could not only facilitate open collaboration among the members of core project team but also open up a possibility of participation from the public which may, in turn, lead to benefits for the project realizing things needed for improvement.

Acknowledgements

This work was supported by a grant from the Alfred P. Sloan Foundation, a gift from the Google Open Source Programs Office, and NSF grants 1546393 and 1633083.

References

1. Juliet Corbin and Anselm Strauss. 2014. *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*. SAGE Publications, Inc., Thousand Oaks, CA.
2. Laura Dabbish, Colleen Stuart, Jason Tsay, and Jim Herbsleb. 2012. Social coding in GitHub: transparency and collaboration in an open software repository. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work* (CSCW '12), 1277-1286. DOI: https://doi.org/10.1145/2145204.2145396
3. GitHub. The State of the Octoverse 2017. Retrieved July 1, 2018 from https://octoverse.github.com.
4. Georgios Gousios, Andy Zaidman, Margaret-Anne Storey, and Arie van Deursen. 2015. Work practices and challenges in pull-based development: the integrator's perspective. In *Proceedings of the 37th International Conference on Software Engineering* (ICSE '15), Vol. 1, 358-368.
5. Aniket Kittur and Robert E. Kraut. 2010. Beyond Wikipedia: coordination and conflict in online production groups. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work* (CSCW '10), 215-224. https://doi.org/10.1145/1718918.1718959
6. Justin Longo and Tanya M. Kelley. 2015. Use of GitHub as a platform for open collaboration on text documents. In *Proceedings of the 11th International Symposium on Open Collaboration* (OpenSym '15), Article 22, 2 pages. https://doi.org/10.1145/2788993.2789838
7. Andrew H Van de Ven, Andre L. Delbecq, and Richard Koenig Jr. 1976. Determinants of coordination modes within organizations. American sociological review, 322-338. http://www.jstor.org/stable/2094477
8. Alexey Zagalsky, Joseph Feliciano, Margaret-Anne Storey, Yiyun Zhao, and Weiliang Wang. 2015. The Emergence of GitHub as a Collaborative Platform for Education. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing* (CSCW '15), 1906-1917. DOI: https://doi.org/10.1145/2675133.2675284