An Analysis of Code Clones in GitHub Actions Workflows

Guillaume Cardoen[1], Alexandre Decan[1,2] and Tom Mens[1]

[1]Software Engineering Lab, University of Mons, Belgium
[2]F.R.S.-FNRS Research Associate

Abstract

GitHub Actions is the built-in CI/CD service of GitHub. While it promotes the use of reusable components, copy-pasting from existing workflows remains a frequent practice, which may lead to code clones. This paper explores the occurrences of code clones inside workflows. We conduct a quantitative analysis of 352K+ code clones instances in a dataset of 117K+ active workflows across GitHub. We observe that most workflow files contain non trivial code clones, mainly at the level of workflow steps. This study characterises code clones in GitHub Actions workflows, hence constituting the basis for understanding the impact of code clones in workflows.

Collaborative software development is an essential practice for globally distributed project teams. It relies on social coding platforms such as GitHub, providing a multitude of collaboration tools such as version control, issue and pull request management, quality analysis, code reviewing and continuous integration/deployement (CI/CD). GitHub is the most popular social coding platform, with over 5.2 billion contributions by over 100 millions users worldwide to more than 518 million repositories according to GitHub's 2024 Octoverse report [1].

In November 2019 GitHub released GitHub Actions (abbrevatied to *GHA*) as its built-in CI/CD tool, becoming the dominant CI/CD service in GitHub repositories in less than 18 months [2]. The GHA service requires repository maintainers to define and store *workflows* as YAML files in their repositories. These workflows declare how to automate repetitive tasks (e.g., testing, building, issue triaging, quality analysis, deploying) in reaction to one or more events (e.g., a push to a branch, a new pull request, a fixed schedule). When such an event occurs, a runner executes the corresponding workflow.

Workflows may use *Actions*, one of GHA reusable components. Actions can be of multiple types. A JavaScript Action is a single JavaScript file that is executed during runtime. Composite Actions use a similar syntax to that of GHA workflow, and can be used to declare a sequence of steps in a single reusable component callable from other workflows. Finally, *reusable workflows* share the syntax of GHA workflow and allow maintainers to reuse complete jobs.

Despite these reuse mechanisms, creating new workflows based on existing ones (either from the same author or from someone else) is a common practice [3]. Copy-pasting from one's own workflow is a frequent reuse mechanism used by GHA's users. This copy-paste reuse tends to lead to duplicated fragments within and across workflow files.

Such duplicated code is commonly referred to as *code clones*. Code cloning has been, and remains to be, a very active topic of research [4]. A common definition of code clones are code fragments that are similar according to some similarity measure [4].

Depending on the reasons of code cloning, it may have positive effects or to the contrary, be detrimental to the overall quality and maintainability of a codebase [5]. From the problematic side, clones tend to be considered as bad smells needing refactoring [6]. Clones may result in bug propagation, inconsistent bug fixes, reduced readability, increased code size, and obscuring the origin of the code [5]. From the beneficial side, code clones can enhance code readability and robustness while reducing development time [5]. Moreover, in languages lacking robust reuse mechanisms, they may be the only viable option for extending or adding a functionality without reinventing the wheel.

As GHA is widely used [2] as part of the supply chain of many software projects, there is a need to

BENEVOL 2025: The 24th Belgium-Netherlands Software Evolution Workshop Enschede, 17-18 November 2025
guillaume.CARDOEN@umons.ac.be (G. Cardoen); alexandre.DECAN@umons.ac.be (A. Decan); tom.MENS@umons.ac.be (T. Mens)
0009-0005-2008-3565 (G. Cardoen); 0000-0002-5824-5823 (A. Decan); 0009-0005-2008-3565 (T. Mens)

understand how cloning impacts GHA workflow maintainability, especially since the GHA documentation advocates to avoid duplication by relying on reusable components such as Actions and reusable workflows. A first step in this direction is to quantify the prevalence of code clones in GHA workflows, and to which extent they could be avoided.

We therefore shed more light on the characteristics of code clones within and across workflows in GitHub repositories. More specifically, we present a quantitative analysis of 352K+ instances of code clones identified in a large dataset [7] of 117K+ recently active workflows in 31K+ GitHub repositories to answer four research questions:

RQ1: *How prevalent are code clones in workflows?* This question aims to establish to which extent code clones occur in GHA workflows. We observed that a majority of workflow files contain non trivial code clones.

RQ2: *Which parts of workflows are subject to code clones?* While code clones are present in a majority of workflows, it remains unclear where they are located. Finding their location is essential to understand if refactoring is needed and possible. We found that the duplication of one or multiple shell commands or calls to reusable components represents four out of five code clones.

RQ3: *How are code clones distributed across different scopes?* GHA allow to declare reusable components callable from multiple workflows in the same GitHub repository or across the same GitHub organisation. This question analyses the characteristics of code clones across these different scopes.

RQ4: *To which extent could code clones be avoided by using appropriate reuse mechanisms?* This research question explores the possibility of refactoring workflows to remove the detected code clones by using GHA's existing reuse mechanisms.

Overall, we found evidence of widespread occurence of code clones inside GHA workflows and studied the characteristics of such clones. This study paves the way for further studies of code clones in workflows, as well as the impact of such clones.

Acknowledgments

This research is supported by F.R.S.-FNRS research projects T.0149.22 , F.4515.23 and J.0147.24.

References

[1] GitHub, Octoverse report 2024: The state of open source software, https://octoverse.github.com/, 2024. [Accessed 23-09-2025].
[2] M. Golzadeh, A. Decan, T. Mens, On the rise and fall of CI services in GitHub, in: International Conference on Software Analysis, Evolution and Reengineering (SANER), IEEE, 2022.
[3] H. Onsori Delicheh, G. Cardoen, A. Decan, T. Mens, Automation and reuse practices in github actions workflows: A practitioner's perspective, 2025. doi:10.5281/zenodo.15422635.
[4] M. Zakeri-Nasrabadi, S. Parsa, M. Ramezani, C. Roy, M. Ekhtiarzadeh, A systematic literature review on source code similarity measurement and clone detection: Techniques, applications, and challenges, Journal of Systems and Software (2023) 111796.
[5] C. J. Kapser, M. W. Godfrey, "cloning considered harmful" considered harmful: patterns of cloning in software, Empirical Software Engineering 13 (2008) 645–692.
[6] M. Fowler, Refactoring: improving the design of existing code, Addison-Wesley Professional, 2018.
[7] G. Cardoen, T. Mens, A. Decan, A dataset of GitHub Actions workflow histories, in: International Conference on Mining Software Repositories, ACM, 2024.