5-2021

Research artifact: The potential of meta-maintenance on GitHub

Hideaki HATA

Raula KULA

Takashi ISHIO

Christoph TREUDE
Singapore Management University, ctreude@smu.edu.sg

Research Artifact: The Potential of Meta-Maintenance on GitHub

Hideaki Hata,* Raula Gaikovina Kula,* Takashi Ishio,* Christoph Treude†
*Nara Institute of Science and Technology
{hata, raula-k, ishio}@is.naist.jp
†University of Adelaide
christoph.treude@adelaide.edu.au

***Abstract*—This is a research artifact for the paper "Same File, Different Changes: The Potential of Meta-Maintenance on GitHub". This artifact is a data repository including a list of studied 32,007 repositories on GitHub, a list of targeted 401,610,677 files, the results of the qualitative analysis for RQ2, RQ3, and RQ4, the results of the quantitative analysis for RQ5, and survey material for RQ6. The purpose of this artifact is enabling researchers to replicate our mixed-methods results of the paper, and to reuse the results of our exploratory study for further software engineering research. This research artifact is available at https://github.com/NAIST-SE/MetaMaintenancePotential and https://doi.org/10.5281/zenodo.4456668.**

I. Study Overview

Online collaboration platforms such as GitHub have provided software developers with the ability to easily reuse and share code between repositories. With clone-and-own and forking becoming prevalent, maintaining these shared files is important, especially for keeping the most up-to-date version of reused code. Different to related work, we propose the concept of meta-maintenance—i.e., tracking how the same files evolve in different repositories with the aim to provide useful maintenance opportunities to those files.

To explore the potential of meta-maintenance, we conducted an exploratory study with (i) a quantitative analysis of 27,994,587 seed files from 32,007 Git repositories to establish the prevalence of seed files, the extent to which seeds evolve, and the uniqueness of seeds; (ii) a qualitative analysis of a stratified sample of 1,011 seed files to determine the kinds of seeds, the relationships among seed families, and main drivers for changes in the variants; and (iii) a survey for developer feedback. We constructed the following six research questions to guide our study.

(RQ1): *How prevalent are seed files in software repositories?*
(RQ2): *What kinds of seed files are there?*
(RQ3): *Are there relationships among repositories in seed families?*
(RQ4): *What was the main driver of the changes for variants?*
(RQ5): *How uniquely do variants evolve in seed families?*
(RQ6): *How do developers consider changes for variants?*

Our results indicate that a majority of active repositories on GitHub contains at least one file which is also present in another repository, and that a significant minority of these files are maintained differently in the different repositories which contain them. We manually analyzed a representative sample of shared files and their variants to understand which changes might be useful for meta-maintenance. Our findings support the potential of meta-maintenance and open up avenues for future work to capitalize on this potential.

TABLE I: Collected repositories

language	# candidates	# obtained (%)
C	3,262	2,962 (91%)
C++	4,219	3,824 (91%)
Java	5,911	5,427 (92%)
JavaScript	9,017	7,960 (88%)
Python	6,606	6,087 (92%)
PHP	3,877	3,388 (87%)
Ruby	2,639	2,359 (89%)
sum	**35,531**	**32,007 (90%)**

II. Contents

This artifact includes a list of studied 32,007 repositories on GitHub shown in Table I, a list of targeted 401,610,677 files used for **RQ1**, the results of the qualitative analysis for **RQ2**, **RQ3**, and **RQ4**, the results of the quantitative analysis for **RQ5**, and survey material for **RQ6**. We used the following coding guides for our qualitative analysis.

A. Coding guide for *RQ2*

library: a program that contains a collection of code used by applications
utility functionality: a system software for controlling the operation of a computer, devices, etc.
configuration: Configuration files
other: We used this code mostly for header files or files containing version information

B. Coding guide for *RQ3*

related: There is an explicit relationship among repositories, e.g., one is a fork of another, their names are similar or identical, or because one mentions the other prominently in its documentation.
non-related: On the other hand, many seed families do not contain any evidence to suggest that there is a connection among the repositories, apart from using at least one common file.

978-1-6654-1219-3/21/$31.00 ©2021 IEEE
DOI 10.1109/ICSE-Companion52605.2021.00084

C. Coding guide for ***RQ4***

reference to a known origin: For many repositories in our sample, the origin is obvious—this applies in particular to the various Linux variants. In those cases, changes that have been applied to seed files are often the same commits that have been applied to the origin.

library updates: Library updates.

project-specific changes: The most interesting case for meta-maintenance are project-specific changes which are not library updates or made in reference to a known origin.

tangled updates: In case the commit history contained changes that could not easily be localized to the file under investigation, we applied this code.

other: For change histories which did not fit any of the previous categories, we applied this code.

D. Quantitative analysis for ***RQ5***

We provide a CSV file of seed family samples. Each line represents a variant v of a seed family. Duplicate variants are excluded.

strata: the strata including the seed family.

blob1: the blob ID of the seed file. The variants having the same blob1 belong to the same seed family.

loc: the number of lines of the seed file.

url: GitHub link to see the variant file v.

commit: the commit including the seed file in the repository.

head: the revision analyzed in the paper.

modificationcount: the number of commits that modified the variant file.

blob2: the blob ID of v in the head revision.

sha1: the SHA-1 hash of the file content of v.

retention: the value of $Retention\_f(v, seed)$ in the paper.

uniqueness: the value of $u(v$, the seed family members other than $v)$ in the paper.

E. Survey material for ***RQ6***

As part of the questionnaire, we asked (a) how important the seed file was to the project, (b) what kind of maintenance activities were the developers interested in regarding the file, and (c) whether they would be interested in a meta-maintenance approach. See material for all questions.

III. Conclusion

Based on this work which has established the prevalence of seeds in GitHub projects, their multiple categories of seed variants, uniqueness and practical useful potential of meta-maintenance, there are many open avenues and challenges for future work: understanding how to manage all seed variants in seed families, further studies of what are useful changes, and tool support to extract specific needs of a seed family to query other repositories, to name a few.