Author
Bram de Vries
(A.C.devries-1@student.tudelft.nl)

Supervisors
Sebastian Proksch
Shujun Huang

GitHub Mining

The implementation of continuous integration pipelines on GitHub

June 28, 2023

1. Background

Continuous Integration (CI) is a software development practice that involves frequently merging code changes from multiple developers into a single shared repository. It has been proven that CI can improve the productivity of project teams[1] by automating the process of building, testing, and validating software changes.

2. Research Questions

While much previous research focussed on single CI serivces, this will combine results from the two most used services; Travis CI and GitHub Actions [2]. This will lead to a broader perspective on CI practices

How are Continuous Integration pipelines set up in GitHub software projects?

RQ 1
When are CI pipelines introduced into a project?

RQ 2
How are jobs triggered in CI pipelines?

RQ 3
How are jobs structured in CI pipelines?

RQ 4
What distinct types of jobs are set up in CI pipelines?

RQ 5
Which operating systems are used for CI pipelines?

A future goal is to combine the results with those of other members of the GitHub Mining Research Project group. This will be done to ultimately be able to inform and advise developers on maturing the CI implementation based on the context of their project.

3. Methodology

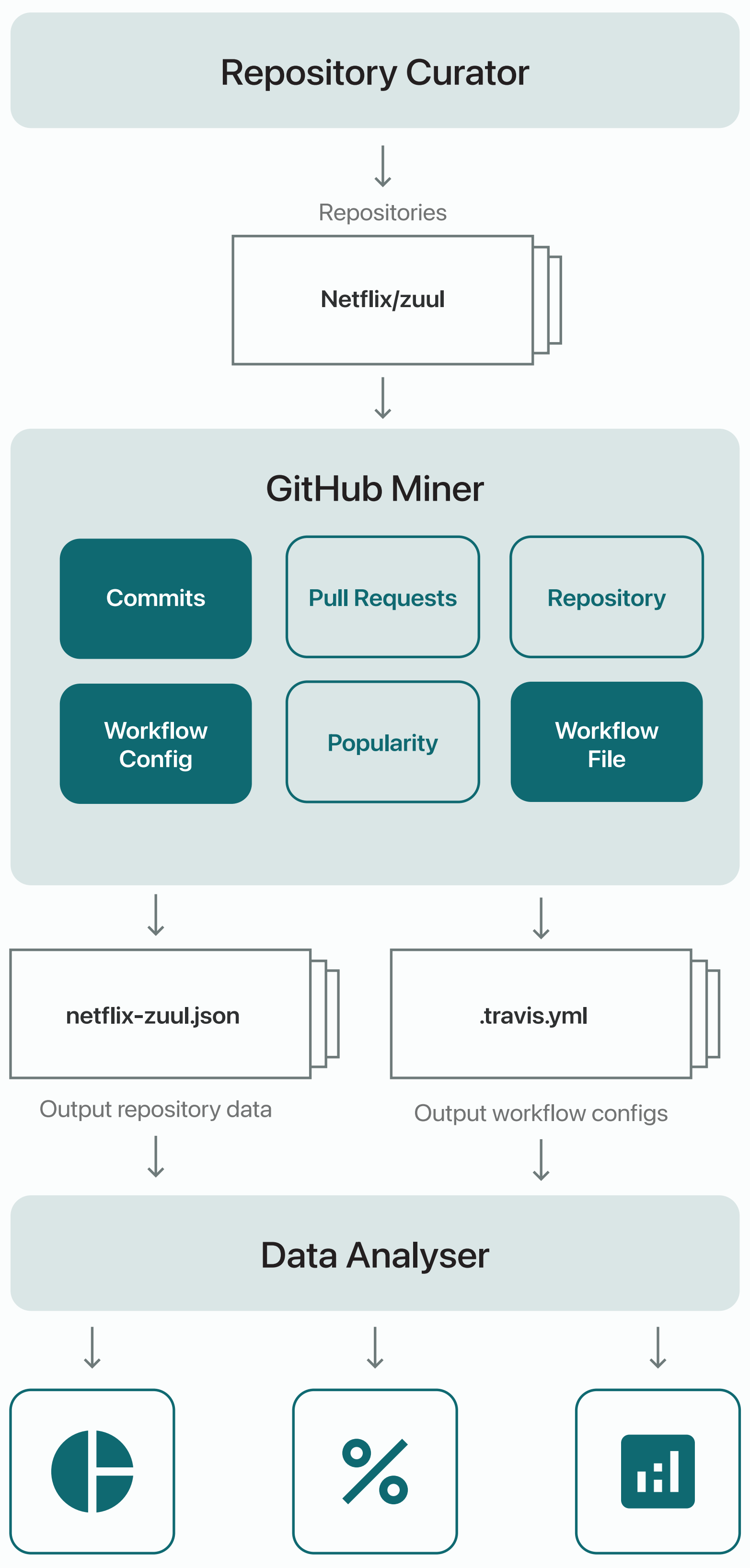

Figure 1: Illustration of method and components used to extract and analyse data.

1637
mined repositories

1574
GitHub Actions

63
Travis CI + GitHub Actions

4. Results

RQ 1: Time to Introduction

- Newer projects introduce CI pipelines more quickly.

- On average, Travis CI pipelines are introduces more quickly than GitHub Actions.

- Projects created before the existance of GitHub Actions take between 2 and 6 years to still adopt it.

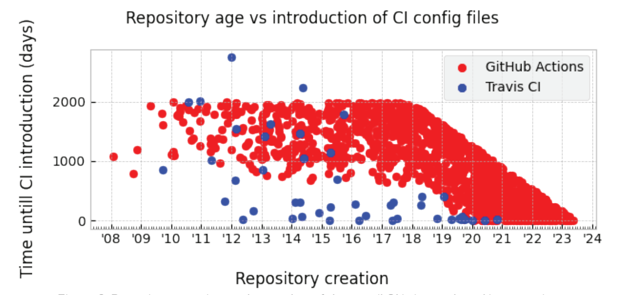

Figure 2: Repository creation vs the number of days until CI is in- troduced into a project.

RQ 2: Job triggers

- Across both platforms, a combination of *push* and *pull request* is most common.

- In Travis CI, some pipelines are disabled by turning both *push* and *pull request* triggers off.

- GitHub offers many more trigger events, with *schedule* and *workflow dispatch* being the most used.

		Push			
		on		*off*	
Pull request	*on*	28	(77.8%)	2	(5.6%)
	off	2	(5.6%)	4	(11.1%)

Table 1: Number of times combination of the push and pull request triggers are used in Travis CI pipelines.

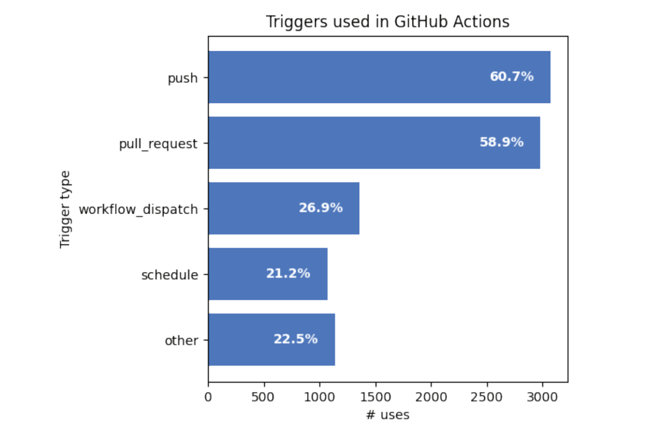

Figure 3: Number of times triggers are used in GitHub Actions pipelines. Percentages are relative to the total amount of pipelines.

RQ 3: Job configuration

- Travis CI allows for two methods of configuring jobs, through a single script or a job matrix.

- Most GitHub Actions projects configure three workflow files and six jobs within their repositories.

- GitHub Actions projects that set up considerably more workflow configuration files tend to have more jobs configured too.

			Config loc	
Method	#	%	avg	median
Single script	39	78%	33.54	24
Job matrix	6	12%	75.17	60
Combination	3	6%	169.34	168
Neither	2	4%	7.0	7

Table 2: Distribution of job configuration methods used in Travis CI.

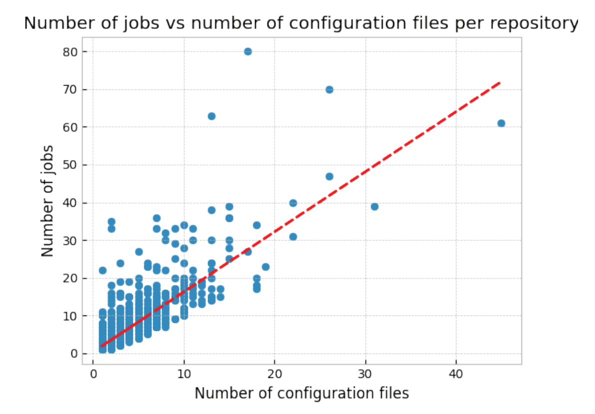

Figure 7: Total number of jobs in a repository vs number of config- uration files in a repository.

RQ 4: Job types

- Using previous large-scale research on jobs in Travis CI [3], 9 different job categories were devised.

- There seems to be a stronger focus on jobs outside of building and testing in GitHub Actions.

Category	# Jobs	GHA %	TCI %
Testing	2040	20.56%	58.64%
Building	1883	18.98%	8.30%
Analyzing	1605	16.17%	0.18%
Releasing	817	8.23%	1.43%
Organisation	716	7.22%	-
Documentation	401	4.04%	3.26%
Formatting	69	0.70%	1.82%
Communication	63	0.63%	0.07%
Unknown	2329	23.47%	26.26%

Table 3: Distribution of categorized jobs in GitHub Actions and Travis CI.

RQ 5: Operating systems

- GitHub offers more optionsin customising which OS to run pipelines on.

- Across both platforms, the most common OSs are: *Ubuntu*, *MacOS* and *Windows*.

- The majority of Travis CI pipelines have no OS configured, defaulting to Ubuntu.

	GHA Jobs		TCI Pipelines	
OS	#	%	#	%
Ubuntu	6386	73.7%	16	29.1%
MacOS	216	2.5%	4	7.3%
Windows	242	2.8%	1	1.8%
Not specified	717	8.3%	34	61.8%
Other	1102	12.7%	-	-
Ubuntu total	7003	82.0%	50	90.9%

Table 4: Distribution of operating systems used in GitHub Actions jobs and Travis CI pipelines. The last row shows the total usage of Ubuntu, as that is the default when no system is specified.

5. Conclusions

-The YAML files and services give some structure to how to configure CI, but a lot is left up to decide by the developers.

- The relatively new CI platform GitHub Actions is on the rise, which is more tightly integrated into GitHub.

- This is shifting the focus from mainly on building and testing to including code analysis and automating organisational tasks.

6. Limitations

- Little repositories that use Travis CI and no repositories that exclusively use it were found. While this relatively low number of Travis CI repositories could be a limiting factor for the results of this research, the repository curation process was chosen to be as general and unbiased as possible.

- The categorisation of the GitHub Actions jobs. The, the exact keywords used in classifying Travis CI[2] were not disclosed, so they had to be recreated, which was done with the help of GitHub Copilot.

7. Future work

- These results can be combined with those of my team of peers to find a relation between a repository's contextual factors and the configuration of its CI.

- To be able to advise developers on how CI could be matured for their projects, more research needs to be done on what effect different configurations have on performance and efficiency.

- There is also room for improvement regarding code duplication inside CI configuration files. Some repositories use many separate files for similar tasks, like testing on multiple operating systems. More research could determine whether this is a flaw of the CI services or developers and how this could be resolved.

8. References

[1] Bogdan Vasilescu, Yue Yu, Huaimin Wang, Premkumar Devanbu, and Vladimir Filkov. Quality and productiv- ity outcomes relating to continuous integration in github. In Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering, ESEC/FSE 2015, page 805–816, New York, NY, USA, 2015. Association for Computing Machinery.
[2] Mehdi Golzadeh, Alexandre Decan, and Tom Mens. On the rise and fall of ci services in github. In 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), pages 662–672, 2022.
[3] Thomas Durieux, Rui Abreu, Martin Monperrus, Tegawende F.Bissyande, and Luis Cruz. An analysis of 35+ million jobs of travis ci. In 2019 IEEE International Conference on Software Maintenance and Evolution (ICSME), pages 291–295, 2019.