# Classes-And-Metrics (CAM): a Collection of Snapshots of GitHub Java Repositories

Yegor Bugayenko
yegor256@huawei.com
Huawei, Russia, Moscow

## Motivation

First, research projects that analyze Java code usually mine it from places where open-source projects keep their files, such as GitHub. It is a common practice, in the papers explaining the results of the researches, fully disclose the coordinates of the open-source code being mined. However, source code is a very volatile substance: repositories change their locations and files are being modified, as was shown by Robles (2010). In order to guarantee *replicability* of their research results, paper authors must somehow guarantee that the source code they used at the time of research remains available and intact during the entire lifetime of the paper. One obvious solution would be making copies of the repositories being mined and then hosting them somewhere where they are "forever" available.

Second, research methods usually involve filtering out some types of files found in repositories, which are not source code, such as plain text documents or graphic images. Then, some source code files may need to be taken out because they are auto-generated or contain unparseable Java code: such files are not suitable for most methods of code analysis.

Third, most source code analysis researches involve collecting metrics from the files found in mined repositories, such as lines of code, complexity, cohesion, and so on. Most of these metrics are already known and their retrieval mechanisms are trivial, as was summarized by Nuñez-Varela et al. (2017).

Thus, there is an obvious duplication of work among different research projects: (a) they have to "host" mined data in order to guarantee desirable replicability, as noted by Cosentino et al. (2017), (b) they must implement filtering of source code fetched from GitHub, and (c) they have to collect popular metrics. Having a ready-to-use archive of downloaded, filtered, and measured source code files would help many research projects reduce the amount of work efforts.

## How It Works

In order to help research projects in all three tasks mentioned above, we created CAM[1] archive: an open-source collection of scripts regularly (at least once every six months) being executed in Docker containers in our proprietary computing environment with results published in form of an "immutable" ZIP archive as GitHub "asset" attached to the next release of CAM repository. Here, immutability is not technically guaranteed but promised: even though we, being the owners of CAM repository, are able to replace any previously created assets, we are not going to do so in order to not jeopardize the idea. Instead, new releases will be published retaining previously generated assets unmodified.

At the time of writing, CAM repository consists of scripts written in Makefile, Python, Ruby, and Bash, which do exactly the following:

- Fetches 1024 repositories from GitHub, which have `java` language tag, have more than 1K and less than 10K stars, and are at least as big as 200Kb;
- Removes files without `.java` extension, Java files with syntax errors, `package-info.java` files, files with lines longer than 1024 characters, and unit tests;
- Calculates KLoC, NCSS, Cyclomatic Complexity, Cognitive Complexity, number of attributes, number of constructors, number of methods, number of static methods, and some other metrics.

The size of the latest archive generated on the 14th of February, 2022 is 692Mb. It includes 97350 Java classes with 16 metrics for each class. It took us 73 hours on a server with four vCPU and 16Gb of RAM to generate the data.

## Conclusion

We expect CAM archive to be used by research teams analyzing Java source, which want (a) to guarantee replicability of their results and (b) to reduce data preprocessing efforts. We also expect open-source community to contribute to CAM scripts, making filtering more powerful and adding more code metrics to the collection.

## References

Cosentino, Valerio et al. (2017). "A Systematic Mapping Study of Software Development With GitHub". In: *IEEE Access* 5, pp. 7173–7192.

Nuñez-Varela, Alberto S et al. (2017). "Source Code Metrics: A Systematic Mapping Study". In: *Journal of Systems and Software* 128, pp. 164–197.

[1] https://github.com/yegor256/cam

Robles, Gregorio (2010). "Replicating MSR: A Study of the Potential Replicability of Papers Published in the Mining Software Repositories Proceedings". In: *IEEE Working Conference on Mining Software Repositories*, pp. 171–180.