A Dataset of Microservices-based Open-Source Projects

Dario Amoroso d'Aragona
Tampere University
Tampere, Finland
dario.amorosodaragaona@tuni.fi

Alexander Bakhtin
University of Oulu
Oulu, Finland
alexander.bakhtin@oulu.fi

Xiaozhou Li
University of Oulu
Oulu, Finland
xiaozhou.li@oulu.fi

Ruoyu Su
University of Oulu
Oulu, Finland
ruoyu.su@student.oulu.fi

Lauren Adams
Baylor University
USA
lauren_adams3@baylor.edu

Ernesto Aponte
Universidad del Sagrado Corazón
USA
eaponte81@sagrado.edu

Francis Boyle
Baylor University
USA
francis_boyle1@baylor.edu

Patrick Boyle
Baylor University
USA
patrick_boyle1@baylor.edu

Rachel Koerner
Baylor University
USA
rachel_koerner1@baylor.edu

Joseph Lee
University of Richmond
USA
Joseph.lee@richmond.edu

Fangchao Tian
University of Oulu
Oulu, Finland
fangchao.tian@oulu.fi

Yuqing Wang
University of Helsinki
Helsinki, Finland
yuqing.wang@helsinki.fi

Jesse Nyyssölä
University of Helsinki
Helsinki, Finland
jesse.nyyssola@helsinki.fi

Ernesto Quevedo
Baylor University
USA
ernesto_quevedo1@baylor.edu

Shahidur Md Rahaman
Baylor University
USA
shahidur_rahaman1@baylor.edu

Amr S. Abdelfattah
Baylor University
USA
amr_elsayed1@baylor.edu

Mika Mäntylä
University of Helsinki
Oulu, Finland
mika.mantyla@helsinki.fi

Tomas Cerny
SIE, University of Arizona
USA
tcerny@arizona.edu

Davide Taibi
University of Oulu
Oulu, Finland
davide.taibi@oulu.fi

ABSTRACT

Researchers in the microservices community often resort to demonstrating the impact of their proposed advancements on custom-made microservices projects. This is a possible source of bias that can reduce the trustworthiness of the results. Moreover, it is hard to compare advances in small projects, often developed due to lack of time. It is common across disciplines to recognize benchmarks that mitigate bias and unify the advancements' impact. To facilitate the identification of available open-source microservice projects (OSS-MS), we performed a comprehensive study to identify, curate, and catalog OSS-MS. We started with **389559** projects and filtered them down to **3804** projects that we manually labeled. After manual labeling, our dataset contains **378** projects with three or more microservices and with over 100 commits. We document the projects from many perspectives, including project size, platform, number of contributors, project purpose, and foundation support. This dataset can serve researchers as a roadmap to identify benchmarks, as our dataset can be used to answer questions such as whether the number of services impacts the issue count.

MSR '24, April 15–16, 2024, Lisbon, Portugal

ACM ISBN 979-8-4007-0587-8/24/04.
https://doi.org/10.1145/3643991.3644890

ACM Reference Format:
Dario Amoroso d'Aragona, Alexander Bakhtin, Xiaozhou Li, Ruoyu Su, Lauren Adams, Ernesto Aponte, Francis Boyle, Patrick Boyle, Rachel Koerner, Joseph Lee, Fangchao Tian, Yuqing Wang, Jesse Nyyssölä, Ernesto Quevedo, Shahidur Md Rahaman, Amr S. Abdelfattah, Mika Mäntylä, Tomas Cerny, and Davide Taibi. 2024. A Dataset of Microservices-based Open-Source Projects. In *21st International Conference on Mining Software Repositories*

(MSR '24), April 15–16, 2024, Lisbon, Portugal. ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3643991.3644890

1 INTRODUCTION

Microservice architectures, have been gaining momentum and becoming the de facto standard in modern cloud-based systems since being first proposed in 2014 by Lewis and Fowler [5]. Microservice-based systems are decomposed as suites of independently deployable, fine-grained, highly cohesive, and loosely coupled services. A number of studies have focused on investigating several topics of microservice architectures by analyzing different open-source (OSS) repositories. However, it is often hard to identify and analyze microservice practices due to the lack of public datasets of microservice-based systems [2].

Recently, a limited amount of work aimed at creating public datasets of microservice-based systems. Rahman *et al.* [9] curated a dataset composed of 20 GitHub projects. Brogi *et al.* [2] attempted to present a first set of microservice-based applications, but it was reported that the creation of this dataset was still ongoing. Differently from our work, they only selected and analyzed GitHub projects with less than 10 stars and the last commit before 1 October 2021. Other researchers manually selected different projects for their studies. For example, Márquez and Astudill, investigated the actual use of microservices architectural patterns was explored in 30 OSS-MS [8]. In another work Márquez and Astudill analyzed, the source code and documentation of 17 OSS-MS were examined to identify availability tactics to support the security design of microservice-based systems [7]. Furthermore, Waseem et al. investigated five OSS-MS on GitHub to explore the nature of issues faced by microservices system developers [14]. Schneider et al. proposed a dataset of the manually created dataflow diagrams of 17 OSS-MS on GitHub annotated with information on implemented security features [11]. In our previous work, we also proposed a small dataset of 20 manually selected microservices-based Java projects [10]. However, none of these studies contributed to the creation of public datasets with a large number of OSS-MS.

In this work, we present a manually labeled dataset of Dockerized OSS microservices. The dataset contains **378** projects, including industrial, academic, tutorials, and student works. We aim to provide the most comprehensive set of projects possible to allow researchers to decide how to select and filter the projects instead of filtering the projects based on a set of predefined criteria.

Thanks to our dataset, researchers will be able to conduct different types of studies, including mining software repository studies, process or product quality assessments, and many others. The availability of different types of projects will allow different types of work to be carried out. For example, researchers focusing on the learning aspects might consider specific aspects of student-developed projects, while researchers interested in some industrial characteristics might consider industrial projects only.

The rest of this work is structured as follows. Section 2 reports the process of project selection. Section 3 describes the created dataset. Section 4 discusses threats to the validity of the generated dataset. Section 5 concludes this paper and outlines plans to refine the dataset.

2 PROJECT SELECTION

In this section we present the steps performed to extract the data, to select the projects, and to manually label the projects. The general process for selecting microservice-based projects is shown in Fig. 1.

2.1 Metadata extraction

To search for OSS-MS projects we adopted the World of Code (WoC). The WoC is a computational and statistical infrastructure and Free/Libre Open Source Software (FLOSS) ecosystem to provide a research-ready, operational, updatable, and expandable dataset[6]. This giant dataset is curated by completely collecting and cross-referencing project objects (e.g., authors, projects, commits, blobs) mainly from three public version control systems (i.e., GitHub[1], Gitlab[2], Bitbucket[3]).

WoC enables getting information on project commits, blogs, and files. Therefore, starting from the 173M projects available in WoC, we query the dataset using the following inclusion criteria:

- I_1 **Systems with at least 1 commit in the past two years (2021-2022).**
- I_2 **At least 100 total commits**. This threshold was selected to exclude less active projects.
- I_3 **Community size of at least 3 contributors**. This threshold was selected to exclude personal projects

To enable the replicability of the search, the query used to extract WoC data is available in the replication package[4].

2.2 Docker information extraction

We first aim to retrieve all GitHub projects using Docker. Thus we cloned the projects and we excluded all the projects not having a Docker-Compose file. We extracted project metadata and service information from Dockerfile Docker-Compose files. Moreover, the file '*docker-compose*' documents the building process of a project to organize multiple images for containers and contains relevant information about services, containers, and networks. From the '*docker-compose*' file, we created the service dependency graph, to understand which services are connected and to exclude non-microservices (eg. databases, message buses adopted, API Gateways, monitoring tools). From this step, we obtained 389559 projects.

2.3 Project selection

In this step, we analyzed the data obtained from the metadata and from the docker information extraction steps, identifying four more Inclusion criteria. We used both metrics available in the WoC dataset and metrics calculated by us, selecting:

- I_4 **Systems with at least three docker-based microservices**. The number of microservices was again defined with a low threshold to provide a comparative number of projects in the dataset. We do not count as microservices database services, message buses, or API Gateways.
- I_5 **More than 12 active months**. Active months do not have to be consecutive. This characteristic was chosen to indicate at

[1] https://github.com
[2] https://gitlab.com
[3] https://bitbucket.org
[4] https://github.com/darioamorosodaragona-tuni/Microservices-Dataset.git

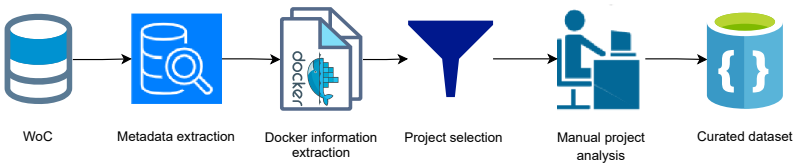

Figure 1: Overview of the dataset generation process

least one full year of active development. (Note that this is not in contrast with I_1, a project can have been committed only once in the last 2 years but can have 12 active months before the last 2 years.)

- I_6 **Projects at least one year old**. We included a minimum project age, since we aim at collecting only projects that are more established in architecture structure and system usage.
- I_7 **README exists and is in English**. There is a README with information about the project in English.

From this step we obtained 3804 projects.

2.4 Manual Project Analysis

To retrieve relevant information, and in particular to confirm if the project was developed with a microservice-based architecture, each of the 3804 project was analyzed and labeled by two authors. In case of disagreement, a third author resolved the disagreement. During the manual labeling, we used the following exclusion criteria:

- E_1 **No information in the README file**. No information in the README file. Projects in which the The README file was empty, or there was not enough information to understand the project's purpose and how to set it up.
- E_2 **Collections of tools/sw/books**. The GitHub repository was only used for collecting docker images or various tools or book chapters with code examples.

We classified projects based on the following criteria:

- *Application Type*
  - *Library to build other projects*: the project is a framework or library to build other projects.
  - *Monitoring other projects*: the project is developed to monitor or run other projects.
  - *Software System*: the project is a totally independent system not meant for running other projects (e.g., e-commerce systems).
  - *Part of a system*: the project is a portion of a full system, such as only the backend or the frontend in a distributed repository.
  - *Unknown*.
- *Application Purpose*
  - *Demo*: the project is developed as an example or demo for training, learning, or academic purposes.
  - *Student Project*: the project is developed for student assignment, examination, or hackathon (not for teaching how to use it, but to "demonstrate" if they learned).
  - *Production ready*: the project aims to be used in production. (e.g., Spinnaker[5]).
  - *Unknown*.
- *Developed by*
  - *Research Institute*: the project is developed by some academic units (e.g. universities and research institutes).
  - *Industry*: the project is mainly developed by companies.
  - *Community*: the project is developed by an OSS organization such as Otasoft [6] or foundation (e.g, Apache Software Foundation) or non-profit (e.g., WWF) or OSS community.
  - *Independent developers*: the project is developed by developers without academic, industrial, community affiliations.
  - *Government*: the project is a part of some government service and is therefore developed by government organizations.
  - *Others*.
- *Archived*
  - *Yes*: the project is officially archived and read-only or it is explicitly mentioned in README that the project is no longer maintained, or it is a student/demo project that is clearly abandoned after the course.
  - *No*: otherwise.
- *WIP/Incomplete*
  - *Yes*: the project is officially reported as "work in progress or "incomplete".
  - *No*: otherwise.
- *Is a Microservice*
  - *Yes*: it is clearly stated it the project documentation that is a microservices system .
  - *No*: there is no evidence in that the project is a microservices system.
  - *Unknown*: it is not stated that the project is a microservices system, but analyzing the docker-compose file could be a microservices system.

The manual classification allowed us to exclude:

- 474 projects that follow the exclusion criteria E_1,E_2.
- 280 projects that were *archived*
- 1536 projects for which it was not possible to ascertain whether they were truly microservices-based projects or not
- 1373 projects that were not microservices projects.

As a result, we selected **378** projects.

Note that one project can be excluded for multiple reasons (*e.g.*, archived and collections of tools/sw/books (E_2)).

3 DESCRIPTION OF THE DATASET

The proposed dataset includes information on GitHub projects and historical analysis of public repositories. Information from GitHub is stored following GitHub Terms of Service[7], which explicitly allows extracting and redistributing public information for research purposes.

[5]https://spinnaker.io

[6]https://github.com/otasoft

[7]https://docs.github.com/en/site-policy/github-terms/github-terms-of-service

License The Dataset is licensed with a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International license[8].

Attributes The different types of metadata are:

- *Project Identification Information* to uniquely identify a project and the state of the project at the moment of this analysis.
- *Project Quantitative Information*: Information extracted from GitHub and the WoC dataset (e.g., number of commits, number of committers, number of stars).
- *Project Qualitative Information*: Information obtained manually analyzing the projects (e.g., type of application, goal).
- *Docker Descriptive Information*: Information extracted by parsing the docker and the docker-compose file (e.g., number of services, buses, monitors, databases).
- *Docker Qualitative Information*: Information obtained analyzing the information extracted from the docker files (e.g., service's dependencies graph, number of microservices).

Distribution The most frequent application type is *software system* (67.00%) while the most frequent purpose is *production ready* (56.8%). Most of the projects are developed by *independent developers* (39.75%). As for application type, 6% are *library* aimed at building other projects, while 7.25% are systems aimed at monitoring other systems. It is worth noting that 28.5% of the projects are developed by *industry* (Figure 2,Figure 3) .

Analyzing the project purposes by different developer types the overall high number of independent developer projects is evident, but it also shows how most of them are split between *production ready* and *demo*. *Community*, *Independent Developers*, *Government*, *Industry*, and *Research Institute* developed projects mostly consist of *production ready* projects. A description of the list of information retrieved and a figure showing these results are available in the replication package[4].

How to use the dataset The dataset is available online and is provided as 2 CSV files.
One CSV file contains [4]:

- Projects labeled as a microservices-based project
- Projects not archived

The other CSV file contains both archived and unarchived and both microservices and microservices-unknown projects [3].
The 2 CSV files contain Project Qualitative Information, Docker Descriptive Information, Docker Qualitative Information, and Project Quantitative Information.

4 THREATS TO VALIDITY

Threats to validity are discussed based on the guidelines by Brewer and Crano[1].

Internal Validity: One potential threat to internal validity is interpersonal bias during manual labeling projects. To alleviate this threat, a consensus meeting was held among the authors to understand the project labeling criteria and achieve agreement. Furthermore, any divergences and uncertainties of the labeled results were discussed among the authors.

External Validity: One threat is related to the generalization of the dataset. WoC infrastructure is not guaranteed to cover the entirety of microservice systems due to merely focusing on Git projects. However, our primary aim is not to include all microservice projects but to provide a comparable number of projects for research purposes. Therefore, it is believed that the microservice-based projects concluded 378 can be statistically representative of OSS projects. Furthermore, we are aware that some information is dynamic, and thus could change over time. For this reason, we published also the hash of the last commit at the moment of the data extraction.

Construct validity The main threat to construct validity is that we cannot validate the accuracy of our dataset creation, since there are no sizeable golden datasets that can be used to evaluate our microservice dataset. The approach of extracting service information from '*Dockerfile*' and '*docker-compse. yml*' files may incur some false positives of microservice projects. For example, some common components (e.g., databases) may be incorrectly identified as services. To mitigate this threat, microservices dependencies and nodes in a static dependency graph were used to filter non-microservice components (e.g., databases and monitors). Furthermore, we manually checked the file structure of projects to remove some false positives during labeling projects.

Reliability To mitigate the threats to reliability, we specified the process of project retrieval, filtering, information extraction, and project labeling. Furthermore, we also illustrated the script of sampling projects and the criteria for labeling projects. Therefore, a similar dataset can be achieved when other researchers duplicate this study.

5 CONCLUSION

Microservice architecture has been adopted by enterprise IT as a predominant architectural style to develop larger software systems (e.g., Amazon and Netflix). However, fewer studies focus on collecting and sharing datasets of microservice-based systems, applications, or practices. To fill this gap, this paper presents a shared dataset of microservice-based systems comprising 1218 projects with more than two services and the issue contents of these projects. Moreover, the data extraction pipeline was illustrated with the support of publicly available executed scripts.

We expect that the shared dataset can serve as a starting point and spur research based on mining microservice systems. The creation of our dataset will help researchers and practitioners analyze many research questions. For example, this dataset can be leveraged to investigate : (1) what and how issues are caused and solved in microservice-based systems; (2) whether the number of services of dockers impacts the issues status (e.g., closed issues and open issues); (3) which microservice architectural tactics and patterns are frequently used in microservice-based systems in academic and industry, respectively; (4) how microservice anti-patterns [13] or smells [12] are refactored in OSS-MS. OSS projects continually evolve in many aspects, such as issue and commit number. In future work, this dataset will be updated and extended regularly by collecting more projects for other periods.

ACKNOWLEDGEMENTS

This work is based on work supported by a grant from the Research Council of Finland (grants n. 349487 and 349488 - MuFAno) and by the National Science Foundation under Grant No. 2245287.

[8] https://creativecommons.org/licenses/by-nc-sa/4.0/

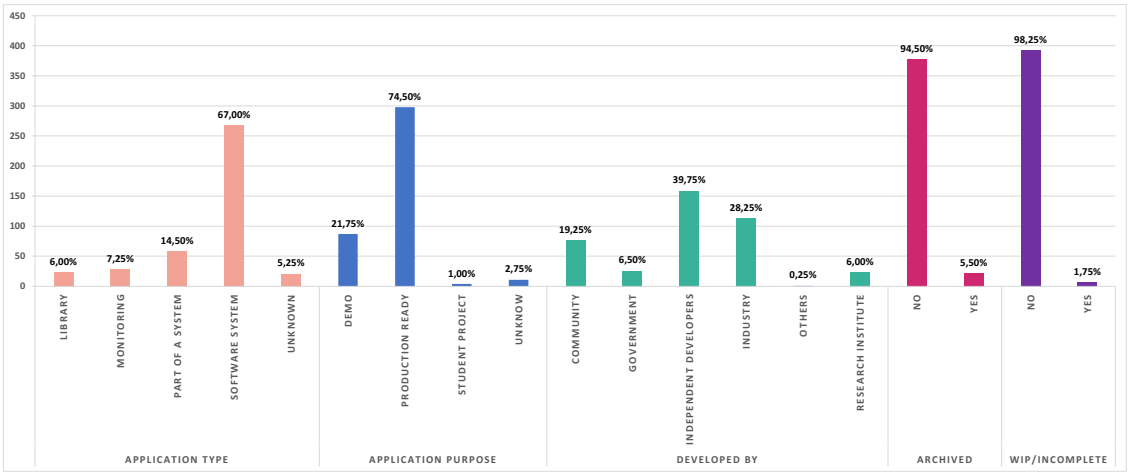

Figure 2: Types of labeled projects

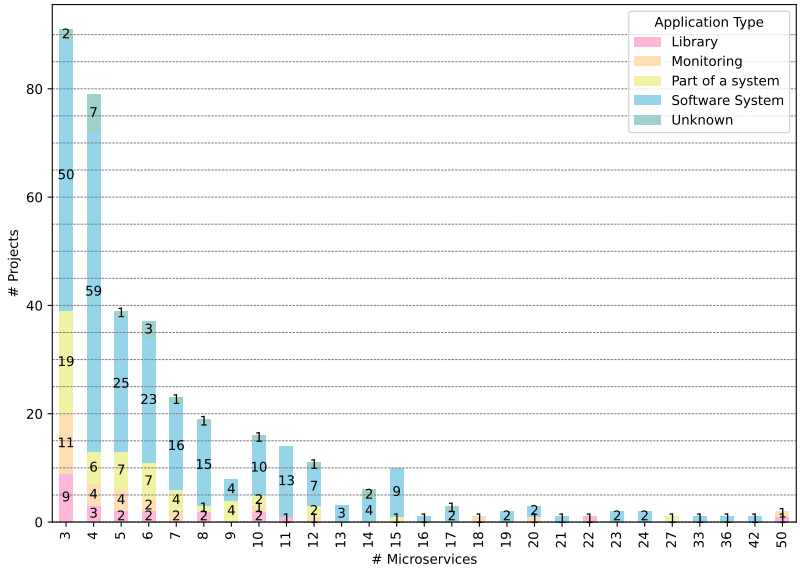

Figure 3: Number of projects per number of microservices with Application Type information

REFERENCES

[1] Marilynn B. Brewer and William D. Crano. 2014. *Research Design and Issues of Validity* (2 ed.). Cambridge University Press, 11–26.

[2] Antonio Brogi, Andrea Canciani, Davide Neri, Luca Rinaldi, and Jacopo Soldani. 2018. Towards a reference dataset of microservice-based applications. In *Software Engineering and Formal Methods: SEFM 2017 Collocated Workshops: DataMod, FAACS, MSE, CoSim-CPS, and FOCLASA, Trento, Italy, September 4-5, 2017, Revised Selected Papers 15*. Springer, 219–229.

[3] Dario Amoroso d'Aragona. 2023. Microservices Dataset - Complete Version. (12 2023). https://figshare.com/articles/dataset/Microservices_Dataset_-_Complete_Version/24722163

[4] Dario Amoroso d'Aragona. 2023. Microservices Dataset - Filtered Version. (12 2023). https://figshare.com/articles/dataset/Microservices_Dataset_-_Filtered_version/24722232

[5] James Lewis and Martin Fowler. 2014. Microservices. (25 Mar 2014). https://martinfowler.com/articles/microservices.html Accessed: 8 December 2023.

[6] Yuxing Ma, Tapajit Dey, Chris Bogart, Sadika Amreen, Marat Valiev, Adam Tutko, David Kennard, Russell Zaretzki, and Audris Mockus. 2021. World of code: enabling a research workflow for mining and analyzing the universe of open source VCS data. *Empirical Software Engineering* 26 (03 2021).

[7] Gastón Márquez and Hernán Astudillo. 2019. Identifying Availability Tactics to Support Security Architectural Design of Microservice-Based Systems. In *Proceedings of the 13th European Conference on Software Architecture - Volume 2* (Paris, France) *(ECSA '19)*. 123–129.

[8] Gastón Márquez and Hernán Astudillo. 2018. Actual Use of Architectural Patterns in Microservices-Based Open Source Projects. In *2018 25th Asia-Pacific Software Engineering Conference (APSEC)*. 31–40.

[9] Mohammad Imranur Rahman, Sebastiano Panichella, and Davide Taibi. 2019. A curated dataset of microservices-based systems. In *SSSME-2019 (CEUR Workshop Proceedings)*. CEUR-WS. jufoid=53269; Joint of the Summer School on Software Maintenance and Evolution ; Conference date: 02-09-2019 Through 04-09-2019.

[10] Mohammad Imranur Rahman, Sebastiano Panichella, and Davide Taibi. 2019. A curated dataset of microservices-based systems. *SSSME-2019* (2019).

[11] Simon Schneider, Tufan Özen, Michael Chen, and Riccardo Scandariato. 2023. microSecEnD: A Dataset of Security-Enriched Dataflow Diagrams for Microservice Applications. In *2023 IEEE/ACM 20th International Conference on Mining Software Repositories (MSR)*. IEEE, 125–129.

[12] Davide Taibi and Valentina Lenarduzzi. 2018. On the definition of microservice bad smells. *IEEE software* 35, 3 (2018), 56–62.

[13] Davide Taibi, Valentina Lenarduzzi, and Claus Pahl. 2020. Microservices anti-patterns: A taxonomy. *Microservices: Science and Engineering* (2020), 111–128.

[14] Muhammad Waseem, Peng Liang, Mojtaba Shahin, Aakash Ahmad, and Ali Rezaei Nassab. 2021. On the Nature of Issues in Five Open Source Microservices Systems: An Empirical Study. In *25th International Conference on Evaluation and Assessment in Software Engineering* (Trondheim, Norway) *(EASE '21)*. 201–210.

APPENDIX

Table 1: Top 50 projects sorted by number of microservices

GitHub Identifier	FirstCommit	#CoreDevelopers	#Commits	Application Type	#Microservices
taskcluster/taskcluster	31.12.2013	6	40023	Library	50
jatinbajaj1/xpresso	10.7.2020	8	134	Monitoring	50
FudanSELab/train-ticket	3.1.2018	15	984	Software System	42
dojot/docker-compose	20.2.2017	13	961	Software System	36
nocproject/noc	20.7.2008	5	77366	Software System	33
lblod/app-demo-editor	10.3.2018	4	1191	Part of a system	27
alchemy-fr/Phraseanet	16.2.2011	7	22182	Software System	24
mjoniec/Utils	15.1.2019	1	263	Software System	24
tokend/developer-edition	17.10.2018	7	648	Software System	23
lightstep/opentelemetry-examples	15.1.2020	3	390	Software System	23
akka/alpakka	10.5.2016	72	10185	Library	22
grin-pool/grin-pool	26.5.2018	5	853	Software System	21
marein/php-gaming-website	15.11.2017	1	482	Software System	20
appwrite/appwrite	8.4.2019	3	6468	Monitoring	20
dotnet-architecture/eShopOnDapr	25.8.2020	2	237	Software System	20
georchestra/docker	27.12.2014	3	457	Software System	19
strangesast/core	12.10.2018	1	678	Software System	19
Combitech/codefarm	31.1.2017	3	1315	Monitoring	18
coopcycle/coopcycle-web	28.10.2016	1	11438	Software System	17
smrealms/smr	2.2.2007	3	11141	Unknown	17
lblod/app-toezicht-abb	22.12.2018	4	485	Software System	17
rodrigorodrigues/microservices-design-patterns	8.12.2018	1	603	Software System	16
danionescu0/docker-flask-mongodb-example	3.12.2016	3	192	Software System	15
OADA/oada-srvc-docker	21.4.2017	4	1222	Software System	15
icebob/catalyst	23.9.2019	1	317	Software System	15
StackStorm/st2-docker	2.3.2017	7	875	Software System	15
FAForever/faf-stack	31.1.2016	4	1158	Part of a system	15
Cingulara/openrmf-docs	27.1.2019	3	942	Software System	15
pelias/docker	11.6.2018	16	1177	Software System	15
ovh/cds	11.10.2016	4	20501	Software System	15
CaliOpen/Caliopen	20.6.2014	3	7232	Software System	15
andryyy/mailcow-dockerized	9.12.2016	27	8300	Software System	15
vietnam-devs/coolstore-microservices	1.6.2018	2	776	Software System	14
4teamwork/opengever.core	3.9.2009	6	29361	Unknown	14
hashintel/engine	15.7.2019	8	660	Software System	14
aspnetrun/run-aspnetcore-microservices	14.4.2019	2	205	Software System	14
OpenLMIS/openlmis-ref-distro	15.6.2016	18	1240	Software System	14
ministryofjustice/opg-use-an-lpa	22.2.2019	8	6605	Unknown	14
blackducksoftware/hub	4.4.2017	14	309	Software System	13
creativesoftwarefdn/weaviate	30.3.2016	4	6886	Software System	13
asc-lab/micronaut-microservices-poc	25.7.2018	3	625	Software System	13
bugsnag/bugsnag-js	5.2.2013	7	5575	Part of a system	12
astarte-platform/astarte	9.6.2017	2	5546	Software System	12
hmcts/tribunals-case-api	7.9.2017	13	8564	Software System	12
claranet/spryker-demoshop	22.8.2013	57	125908	Software System	12
fedspendingtransparency/usaspending-api	10.8.2016	14	17686	Part of a system	12
abixen/abixen-platform	19.10.2016	2	2087	Monitoring	12
microservices-patterns/ftgo-application	10.9.2017	4	501	Software System	12
geoserver/geoserver-cloud	9.7.2020	1	546	Software System	12
barnumd/idp-in-a-box	5.6.2017	3	561	Unknown	12