A Dataset of Enterprise-Driven Open Source Software

Diomidis Spinellis
Zoe Kotti
Konstantinos Kravvaritis
Georgios Theodorou
Panos Louridas
{dds,zoekotti,kravvaritisk,,louridas}@aueb.gr
Athens University of Economics and Business

ABSTRACT

We present a dataset of open source software developed mainly by enterprises rather than volunteers. This can be used to address known generalizability concerns, and, also, to perform research on open source business software development. Based on the premise that an enterprise's employees are likely to contribute to a project developed by their organization using the email account provided by it, we mine domain names associated with enterprises from open data sources as well as through white- and blacklisting, and use them through three heuristics to identify 17 264 enterprise GitHub projects. We provide these as a dataset detailing their provenance and properties. A manual evaluation of a dataset sample shows an identification accuracy of 89%. Through an exploratory data analysis we found that projects are staffed by a plurality of enterprise insiders, who appear to be pulling more than their weight, and that in a small percentage of relatively large projects development happens exclusively through enterprise insiders.

CCS CONCEPTS

• **Software and its engineering → Open source model**; • **Social and professional topics → Computing and business**; • **General and reference →** *Empirical studies.*

KEYWORDS

Software engineering economics, software ecosystems, open source software in business, Fortune Global 500, SEC 10-K, SEC 20-F, EDGAR, dataset

ACM Reference Format:
Diomidis Spinellis, Zoe Kotti, Konstantinos Kravvaritis, Georgios Theodorou, and Panos Louridas. 2020. A Dataset of Enterprise-Driven Open Source Software. In *17th International Conference on Mining Software Repositories (MSR '20), October 5–6, 2020, Seoul, Republic of Korea.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3379597.3387495

1 INTRODUCTION

Despite the size and wealth of software product and process data available on GitHub, their use in software engineering research can be problematic [9, 28], raising issues regarding the generalizability of the corresponding findings [47]. In particular, the open source nature of accessible GitHub repositories means that projects developed by volunteers through open source software development processes [12, 42] are overrepresented, biasing results, especially those related to software architecture or communication and organization structures, through the application of Conway's Law [8, 24]. In addition, many researchers are investigating differences between open source and proprietary software products and processes [3, 33, 38, 43].

Here we present a dataset of open source software developed mainly by enterprises rather than volunteers. This can be used to address the identified generalizability concerns and, also, to perform research on the differences between volunteer and business software development. One might think that open source software development by enterprises is a niche phenomenon. As others have identified [40] and also as is evident from our dataset, this is far from true. A series of queries on GitHub PushEvents published during 2017 found that companies such as Microsoft and Google had hundreds of employees contributing to open source projects [26].

The goal of the dataset's construction is to create a set of GitHub projects that are most probably developed by an enterprise. For the purposes of this work, we define as an enterprise project, one that is likely to be mainly developed by financially compensated employees, working full time under an organization's management. This definition excludes volunteer efforts such as Linux, KDE projects, VLC, and GIMP (even though some companies pay their employees to contribute to them), but includes for-profit company and funded public-sector organization projects that accept volunteer contributions, such as Google's Trillian, Apple's Swift, CERN's ALICE, and Microsoft's Typescript. Our aim is to minimize the number of false positives in the dataset, but we are not interested in the number of false negatives. We do not aspire to create a comprehensive dataset of enterprise projects, but one that contains a number sufficient to conduct generalizable empirical studies.

2 CONSTRUCTION AND EVALUATION

An overview of the dataset's construction process is depicted in an extended version of this paper [45]. The projects were selected from GitHub by analyzing the GHTorrent [17, 18] dataset (release 2019-06-01) by means of the *simple-rolap* relational online analytical processing and *rdbunit* relational unit testing frameworks [19].

MSR '20, October 5–6, 2020, Seoul, Republic of Korea

ACM ISBN 978-1-4503-7517-7/20/05…$15.00
https://doi.org/10.1145/3379597.3387495

Following published recommendations the code and primary data associated with this endeavor are openly available online.[1]

The basic premise for constructing the dataset is that an enterprise's employees are likely to contribute to a project developed by their organization using the email account provided by it. Furthermore, it is unlikely that pure volunteer projects will have contributors using emails from a single enterprise-related domain address. Based on this premise, we identified projects where a large number of commits were contributed through accounts linked to the same enterprise email domain address. To increase the dataset's quality we then removed project clones [46], and only retained projects having more than the identified dataset's average stars (14) and commits (29). Finally, we created one table with diverse details regarding each selected project and one with details regarding each associated enterprise domain. The following paragraphs detail each step, starting from the creation of two tables: *valid enterprise domains* and *probable company domains*.[2]

Valid enterprise domains. This table was created by filtering all email domains found in the users' email table (Table *domains*). We did this by examining frequently occurring email domains, and creating rules to retain only those associated with enterprise development. Specifically, we removed from the set of domains a blacklist (Table *domain blacklist*) containing those associated with: email providers; top and second level organization domains, and thereby the many associated with volunteer open source organizations; open source hubs; top and second level educational domains and, explicitly, the domains of more than 20 hand-picked universities; individuals. We did not remove government organizations and research centers as these mainly operate as enterprises with professional developers. When in doubt, we looked up company emails in the RocketReach provider of company email format details.

Probable company domains. This table was created by identifying domains that are likely to belong to companies from publicly available data and domain heuristics. We obtained the domains associated with large companies in two ways. First, we screen-scrapped, downloaded, and filtered the data associated with the Fortune Global 500 companies: the largest corporations across the globe measured by revenue (Table *fortune global 500*). Second, we obtained the US Securities and Exchange Commission (SEC) yearly company filings that are made in machine readable form and extracted from them the company domains. Specifically, we obtained from EDGAR—the SEC's Electronic Data Gathering, Analysis, and Retrieval system—the XBRL files associated with two forms, namely a) Form 10-K, that gives a comprehensive summary of a company's financial performance (Table *sec 10 K domains*), and b) Form 20-F, that provides an annual report filing for foreign private issuers—non-U.S. and non-Canadian companies that have securities trading in the U.S. (Table *sec 20 F domains*).

We then extracted the internet domain (e.g. *intel.com*) associated with each company from the XBRL files. We obtained the company domains by looking at the XML name space used in the files, which in most cases contains the company's domain. We combined the three sources into the Table *distinct company domains* and complemented it with the Table *valid enterprise domains* filtered to include only records associated with top and second level commercial domains such as *.com*, *.co.uk*, *.com.au*.

From enterprise organizations to their projects. As a next step we combined the two tables with another listing domains registered for GitHub organizations (Table *org domains*), to get tables with user domains linked to GitHub organizations—Tables *valid enterprise users* and *probable company users*. The intuition here is that many companies developing software on GitHub will have configured a company organization under their domain name. Combining the two tables with the GHTorrent *Projects* table yielded the corresponding projects hosted under a GitHub organization: *valid enterprise projects* and *probable company projects*.

These two tables were then linked with a table of each user's email domain (Table *user domain*) and one identifying each commit's committer (Table *project commit committer domain*), giving the number of committers in each project associated with the corresponding organization: *valid enterprise domain committers* and *probable company domain committers*. This stage ended by selecting projects from organizations having a minimum number of committers appearing on GitHub with an email associated with the organization's domain giving the tables *multi committer valid enterprise projects* and *multi committer probable company projects*. The employed floor values (ten and five correspondingly) were selected to exclude projects associated with individuals operating under a personal but commercial-looking domain (e.g. *johnsmith.com*).

Enterprise-dominated projects. To cover enterprises that may not have GitHub organizations registered with emails under their domain, we also established in each project a rank of committers with valid enterprise email addresses according to their number of commits (Table *project committer domain rank*), and obtained those projects having committers from the same organizations as the topmost three (Table *same domain top committers*).

Final filtering and reporting. For the three types of possible enterprise projects we then formed their union (Table *candidate projects*), combined their metrics (Table *merged projects*), removed duplicate projects (Table *deduplicated projects*), combined records referring to the same project (Table *merged domain projects*), and joined them with the number of their commits (Table *project commit count*) and their stars (Table *project stars*), to select those with above average such metrics (Table *above average projects*). For each one of the shortlisted projects, we `git-cloned` from GitHub the project's repository and calculated its basic size metrics in terms of files and text lines (Table *size metrics*). (Due to churn from the date the GHTorrent dataset was published, not all repositories could be retrieved for measuring project size.)

Finally, to provide context for each project, we combined this table with each project's earliest and most recent commit (Table *commit range*), number of commits (Table *project commit committer domain count*) and committers (Table *project committer domains*) for each committer domain, number of commits (Table *project commit author domain count*) and committers (Table *project author domains*) for each author domain, total number of committers (Table *project*

[1] https://doi.org/10.5281/zenodo.3742973

[2] In the interest of readability, this text replaces the underscores in the table names with spaces.

committer count) and authors (Table *project author count*), size metrics (Table *project size metrics*), project license as provided by the GitHub API (Table *licenses*), as well as details about the derivation of the corresponding domain. This process created the table *enterprise project details* and the corresponding report *enterprise projects*.

We manually evaluated a random sample of an earlier version of this dataset,[3] following the systematic review guidelines by Brereton et al. [6]. The sample size was calculated at around 378 using Cochran's sample size and correction formulas [7] (95% confidence, 5% precision). To keep the raters alert we complemented the sample with 22 GitHub projects randomly selected from a set of projects with similar quality characteristics that were part of the dataset (Table *cohort projects*). The third and fourth authors were instructed to individually label the 400 projects as enterprise or not based on the definition in Section 1. To improve the labeling's reliability the two raters did not know the employed heuristics, and were also asked to complete the main reason the project was open source and write a few words to support their decision. Their ratings led to 78% inter-rater agreement and 29% reliability using Cohen's kappa statistic. The second author then resolved the conflicts by majority vote; after excluding the 22 irrelevant projects, 89% of the 378 projects were finally identified as enterprise. We used the bootstrap method [11] with 1000 iterations to establish a confidence interval (CI) for the percentage of enterprise projects in our sample; the 95% CI was calculated at [87–93]%. To generalize, 15 354 (CI: 15 009–16 044) projects of our dataset are expected to be truly enterprise-developed.

Regarding the dataset's external validity, note that although our evaluation addresses the dataset's precision, our method was not targeting a high recall and this was also not evaluated. Consequently, the dataset can be used to address empirical research generalizability concerns we identified in the introduction mainly by providing a set of enterprise-developed projects to be used in work employing stratified sampling, in cohort studies, or in case studies. Furthermore, the number of committers floor we employed in our selection means that the dataset excludes organizations that are small or have a tiny number of their employees committing on GitHub. Finally, the selection of above average projects in terms of stars and commits means that the dataset does not include stillborn or unpopular projects.

3 DATASET OVERVIEW

The dataset[4] is provided as a 17 264 record tab-separated file with the following 29 fields: `url`, the project's GitHub URL; `project_id`, the project's GHTorrent identifier; `sdtc`, true if selected using the same domain top committers heuristic (9 016 records); `mcpc`, true if selected using the multiple committers from a valid enterprise heuristic (8 314 records); `mcve`, true if selected using the multiple committers from a probable company heuristic (8 015 records); `star_number`, number of GitHub watchers; `commit_count`, number of commits; `files`, number of files in current main branch; `lines`, corresponding number of lines in text files; `pull_requests`, number of pull requests; `github_repo_creation`, time stamp of the GitHub repository creation; `earliest_commit`, time stamp of the earliest commit; `most_recent_commit`, time stamp of the most recent commit; `committer_count`, number of different committers; `author_count`, number of different authors; `dominant_domain`, the project's dominant email domain; `dominant_domain_committer_commits`, number of commits made by committers whose email matches the project's dominant domain; `dominant_domain_author_commits`, corresponding number for commit authors; `dominant_domain_committers`, number of committers whose email matches the project's dominant domain; `dominant_domain_authors`, corresponding number for commit authors; `cik`, SEC's EDGAR "central index key"; `fg500`, true if this is a Fortune Global 500 company (2 233 records); `sec10k`, true if the company files SEC 10-K forms (4 180 records); `sec20f`, true if the company files SEC 20-F forms (429 records); `project_name`, GitHub project name; `owner_login`, GitHub project's owner login; `company_name`, company name as derived from the SEC and Fortune 500 data; `owner_company`, GitHub project's owner company name; `license`, SPDX license identifier.

Overall, we see that projects are staffed by a plurality of enterprise insiders, who appear to be pulling more than their weight. Regarding the distribution of contributors, across all identified projects in the dataset we found that 33% of the authors and 24% of the committers are associated with the project's dominant domain. Similarly, regarding the distribution of work, 45% of the commits are made by the enterprise's authors, and 41% of the commits are made by the corresponding committers.

The ten most popular out of the 110 top level domains associated with projects are: *com* (13 494 projects), *io* (763), *de* (383), *gov* (339), *net* (256), *ru* (142), *fr* (134), *cn* (120), *br* (118), and *uk* (111). Similarly, out of 5 097 owners, those associated with the highest number of GitHub projects are: *Microsoft* (855 projects), *Azure* (328), *google* (123), *twitter* (93), *18F* (90), *udacity* (82), *SAP* (79), *Netflix* (79), *hashicorp* (77), and *GoogleCloudPlatform* (77).

In very few projects does development appear to be exclusively controlled by the enterprise: we found 90 projects (0.5%) where all commits came from an enterprise committer and 220 projects (1.3%) where all commits came from an enterprise author. We were expecting these projects to be small, but in fact they sport an average line count of 453k for projects with exclusively enterprise authors and 976k for projects with exclusively enterprise committers. Considerable development seems to happen through pull requests, with 95% of the projects having pull requests associated with them, with an average of 161 pull requests per project.

In total, according to their SPDX identifiers, the projects are licensed using 29 different open source licenses. The two most common licenses used are the MIT (4 340 projects) and Apache 2.0 (3 761 projects), with the GPL version 2 or 3 license used only by 780 projects. This finding indicates that few enterprise open source projects seem to follow a business model based on relicensing GPL code for proprietary development. Surprisingly, for 3 535 projects no license was found, and for 3 374 projects the license did not match one with an SPDX identifier.

We compared the earlier version of this dataset mentioned in Section 2 against the Reaper dataset of engineered software projects [36] in terms of stars, commits, pull requests (PRs), authors, and committers (see Table 1). Reaper initially contained 1 853 205 projects in the

[3] https://doi.org/10.5281/zenodo.3653878 and https://doi.org/10.5281/zenodo.3653888. This was updated following the peer review suggestions, and differs by 64 projects (0.37%—26 removed, 38 added) from the currently supplied one.
[4] https://doi.org/10.5281/zenodo.3742962

Table 1: Enterprise (E) and Reaper (R) Dataset Metrics

	Min		Max (k)		Avg		Stddev	
Metric	E	R	E	R	E	R	E	R
Stars	15	0	80	51	355	11	1661	221
Commits	30	0	304	383	1159	70	5323	1196
PRs	0	0	25	42	161	3	672	94
Authors	1	0	26	5	27	2	213	10
Committers	1	0	26	5	22	2	208	7

form *login-name/project-name*, from which 1 849 500 were successfully associated with a project ID of GHTorrent. Null values were substituted with zero in both datasets, thus metrics were calculated on the basis of the entire dataset sizes (17 252 for this, 1 849 500 for the Reaper). It appears that in all dimensions this dataset is considerably richer than the Reaper one. The difference most likely stems from this dataset's considerable selectivity, as it contains two orders of magnitude fewer projects than Reaper.

4 RELATED WORK

While the relationship between academic or semi-academic institutions and open source software has been favorable [29], with large open source projects such as the Berkeley Software Distribution (BSD) [39] originating from them, this has not always been the case for business. The relationship between business and open source software was often tense in the past, with GPL-licensed software described as "an intellectual property destroyer", un-American, and "a cancer" [34]. Meanwhile, others asserted that open source was compatible with business [22], and researchers quickly identified several business models that are based on open source software [1, 4], as well as significant industrial adoption of open source software products [44]. In short, research associated with the involvement of enterprises in open source software can be divided into four areas [27]: a) company participation in open source development communities [5, 23]; b) business models with open source in commercial organizations [4, 20]; c) open source as part of component based software engineering [2, 30]; and d) usage of open source processes within a company [13, 31].

We consider our study part of the first area. According to Bonaccorsi et al. [5], companies participated in one third of the most active projects on SourceForge as project coordinators, collaborators in code development, or code providers. Hauge et al. [21] also identified the role of component integrator. By providing their proprietary software to the open source community, companies can benefit from reduced development costs, advanced performance, repositioning in the market, and additional profit from new services [27]. Still, the provided software should be accompanied by adequate documentation and information to help the community members engage in it [21].

Although companies marginally participated in open source projects in the past, the participation has recently increased, especially in the larger and more active projects, with a crucial part of the open source code being provided by commercial organizations, particularly small and medium-sized enterprises (SMEs) [32]. For instance, 6%–7% of the code in the Debian GNU/Linux distribution over the period 1998–2004 was contributed by corporations [41].

Similarly, German and Mockus [14] linked identical contributors of CVS repositories with multiple names or emails of different spelling. Using their infrastructure they identified the top contributors of the Ximian Evolution project, and found that the top ten contributors were Ximian employees and consultants, and also that private companies such as RedHat, Ximian and Eazel, severely affected the development of the GNOME project [15], similarly to the way the Mozilla project was mainly developed by Netscape employees [35].

5 RESEARCH IDEAS

The provided dataset can be employed in various ways. First, it can be used to study the involvement of enterprises in OSS development by examining whether they are mostly *takers* or *givers*, their roles within projects, and how they shape a project's evolution and success [5]. Second, it can be employed in studies regarding OSS business models, to investigate how their choice is affected by different enterprise characteristics such as the employees' education level, the enterprise's age, size, service variety, and whether it is family-owned or not [20]. Third, it can be used for research on the composition and structure of OSS supply chains and value chains, particularly to identify the added, deleted, and unchanged dependencies and their effect between releases for different types of packages such as build and test tools [10]. Furthermore, it can be employed in studies concerning enterprise-driven global software development, to measure benefits and tackle issues induced from the physical separation among project members such as strategic, cultural, communication, and knowledge management issues [25]. Another use involves identifying product or process differences between enterprise and volunteer-driven software development in terms of cost, service and support, innovation, security, usability, standards, availability, transparency, and reliability [37]. Finally, it can be used to study enterprise regulatory, compliance, and supply chain risks, to investigate the risk domains that enterprises face when engaging in OSS development, the available sources of risk mitigation, and the heuristics by which managers apply this understanding to manage such projects. From these insights, formalized risk mitigation instruments and project management processes can be developed [16].

ACKNOWLEDGMENTS

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 825328.

REFERENCES

[1] Stephanos Androutsellis-Theotokis, Diomidis Spinellis, Maria Kechagia, and Georgios Gousios. 2011. Open Source Software: A Survey from 10,000 Feet. *Foundations and Trends in Technology, Information and Operations Management* 4, 3–4 (2011), 187–347. https://doi.org/10.1561/0200000026

[2] Claudia Ayala, Øyvind Hauge, Reidar Conradi, Xavier Franch, Jingyue Li, and Ketil Velle. 2009. Challenges of the Open Source Component Marketplace in the Industry, Vol. 299. 213–224. https://doi.org/10.1007/978-3-642-02032-2_19

[3] Adrian Bachmann and Abraham Bernstein. 2009. Software Process Data Quality and Characteristics: A Historical View on Open and Closed Source Projects. In *Proceedings of the Joint International and Annual ERCIM Workshops on Principles of Software Evolution (IWPSE) and Software Evolution (Evol) Workshops (IWPSE-Evol '09)*. Association for Computing Machinery, New York, NY, USA, 119–128. https://doi.org/10.1145/1595808.1595830

[4] Andrea Bonaccorsi, Silvia Giannangeli, and Cristina Rossi. 2006. Entry Strategies Under Competing Standards: Hybrid Business Models in the Open Source Software Industry. *Management science* 52, 7 (2006), 1085–1098.

[5] Andrea Bonaccorsi, Dario Lorenzi, Monica Merito, and Cristina Rossi. 2007. Business Firms' Engagement in Community Projects. Empirical Evidence and Further Developments of the Research. In *Proceedings of the First International Workshop on Emerging Trends in FLOSS Research and Development (FLOSS '07)*. IEEE Computer Society, USA, 13. https://doi.org/10.1109/FLOSS.2007.3

[6] Pearl Brereton, Barbara A. Kitchenham, David Budgen, Mark Turner, and Mohamed Khalil. 2007. Lessons from Applying the Systematic Literature Review Process Within the Software Engineering Domain. *J. Syst. Softw.* 80, 4 (April 2007), 571–583. https://doi.org/10.1016/j.jss.2006.07.009

[7] William G. Cochran. 1977. *Sampling Techniques* (3rd ed.). John Wiley & Sons, Inc., USA.

[8] Melvin E Conway. 1968. How do Committees Invent? *Datamation* 14, 4 (1968), 28–31.

[9] V. Cosentino, J. L. C. Izquierdo, and J. Cabot. 2016. Findings from GitHub: Methods, Datasets and Limitations. In *MSR 2016: IEEE/ACM 13th Working Conference on Mining Software Repositories*. 137–141.

[10] Tapajit Dey and Audris Mockus. 2018. Are Software Dependency Supply Chain Metrics Useful in Predicting Change of Popularity of npm Packages?. In *Proceedings of the 14th International Conference on Predictive Models and Data Analytics in Software Engineering*. 66–69.

[11] B. Efron. 1979. Bootstrap Methods: Another Look at the Jackknife. *Ann. Statist.* 7, 1 (Jan. 1979), 1–26. https://doi.org/10.1214/aos/1176344552

[12] Joseph Feller, Brian Fitzgerald, et al. 2002. *Understanding Open Source Software Development*. Addison-Wesley, London, UK.

[13] Gary Gaughan, Brian Fitzgerald, and Maha Shaikh. 2009. An Examination of the Use of Open Source Software Processes as a Global Software Development Solution for Commercial Software Engineering. In *Proceedings of the 2009 35th Euromicro Conference on Software Engineering and Advanced Applications (SEAA '09)*. IEEE Computer Society, USA, 20–27. https://doi.org/10.1109/SEAA.2009.86

[14] Daniel German and Audris Mockus. 2003. Automating the Measurement of Open Source Projects. *Proceedings of the 3rd Workshop on Open Source Software Engineering* (Jan. 2003).

[15] Daniel M. German. 2002. The Evolution of the GNOME Project. In *Proceedings of the 2nd Workshop on Open Source Software Engineering*. 4. https://flosshub.org/sites/flosshub.org/files/German.pdf

[16] Matt Germonprez, Brett Young, Lars Mathiassen, Julie E Kendall, Ken E Kendall, and Warner Brian. 2012. Risk Mitigation in Corporate Participation with Open Source Communities: Protection and Compliance in an Open Source Supply Chain. In *Proceedings of the 7th International Research Workshop on IT Project Management (IRWITPM '12)*.

[17] Georgios Gousios. 2013. The GHTorrent dataset and tool suite. In *Proceedings of the 10th Working Conference on Mining Software Repositories (MSR '13)*. IEEE Press, Piscataway, NJ, USA, 233–236. https://doi.org/10.5555/2487085.2487132

[18] Georgios Gousios and Diomidis Spinellis. 2012. GHTorrent: Github's Data from a Firehose. In *9th IEEE Working Conference on Mining Software Repositories (MSR)*, Michele Lanza, Massimiliano Di Penta, and Tao Xie (Eds.). IEEE, 12–21. https://doi.org/10.1109/MSR.2012.6224294

[19] Georgios Gousios and Diomidis Spinellis. 2017. Mining Software Engineering Data from GitHub. In *Proceedings of the 39th International Conference on Software Engineering Companion (ICSE-C '17)*. IEEE Press, Piscataway, NJ, USA, 501–502. https://doi.org/10.1109/ICSE-C.2017.164 Technical Briefing.

[20] Elad Harison and Heli Koski. 2010. Applying open innovation in business strategies: Evidence from Finnish software firms. *Research Policy* 39, 3 (April 2010), 351–359. https://doi.org/10.1016/j.respol.2010.01.008

[21] Øyvind Hauge, Carl-Fredrik Sørensen, and Andreas Røsdal. 2007. Surveying Industrial Roles in Open Source Software Development. In *Open Source Development, Adoption and Innovation*, Joseph Feller, Brian Fitzgerald, Walt Scacchi, and Alberto Sillitti (Eds.). Springer US, Boston, MA, 259–264.

[22] Frank Hecker. 1999. Setting up Shop: The Business of Open-Source Software. *IEEE software* 16, 1 (1999), 45–51.

[23] Joachim Henkel. 2008. Champions of revealing—the role of open source developers in commercial firms. *Industrial and Corporate Change* 18, 3 (Dec. 2008), 435–471. https://doi.org/10.1093/icc/dtn046 arXiv:https://academic.oup.com/icc/article-pdf/18/3/435/2415321/dtn046.pdf

[24] James D. Herbsleb and Rebecca E. Grinter. 1999. Splitting the organization and integrating the code: Conway's law revisited. In *ICSE '99: Proceedings of the 21st international conference on Software engineering*. IEEE Computer Society Press, Los Alamitos, CA, USA, 85–95.

[25] James D Herbsleb and Deependra Moitra. 2001. Global Software Development. *IEEE software* 18, 2 (2001), 16–20.

[26] Felipe Hoffa. 2017. Who contributed the most to open source in 2017 and 2018? Let's analyze GitHub's data and find out. Available online https://medium.com/@hoffa/the-top-contributors-to-github-2017-be98ab854e87. Accessed January 25th, 2020. Optional.

[27] Martin Höst and Alma Oručeviundefined-Alagiundefined. 2011. A Systematic Review of Research on Open Source Software in Commercial Software Product Development. *Inf. Softw. Technol.* 53, 6 (June 2011), 616–624. https://doi.org/10.1016/j.infsof.2010.12.009

[28] Eirini Kalliamvakou, Georgios Gousios, Kelly Blincoe, Leif Singer, Daniel M. German, and Daniela Damian. 2016. An In-Depth Study of the Promises and Perils of mining GitHub. *Empirical Software Engineering* 21, 5 (01 Oct. 2016), 2035–2071. https://doi.org/10.1007/s10664-015-9393-5

[29] Josh Lerner and Jean Tirole. 2001. The open source movement: Key research questions. *European Economic Review* 45, 4-6 (May 2001), 819–826. https://ideas.repec.org/a/eee/eecrev/v45y2001i4-6p819-826.html

[30] Jingyue Li, Reidar Conradi, Christian Bunse, Marco Torchiano, Odd Petter N. Slyngstad, and Maurizio Morisio. 2009. Development with Off-the-Shelf Components: 10 Facts. *IEEE Softw.* 26, 2 (March 2009), 80–87. https://doi.org/10.1109/MS.2009.33

[31] Juho Lindman, Matti Rossi, and Pentti Marttiin. 2008. Applying Open Source Development Practices Inside a Company. In *Open Source Development, Communities and Quality*, Barbara Russo, Ernesto Damiani, Scott Hissam, Björn Lundell, and Giancarlo Succi (Eds.). Springer US, Boston, MA, 381–387.

[32] Bjørn Lundell, Brian Lings, and Edvin Lindqvist. 2006. Perceptions and Uptake of Open Source in Swedish Organisations. In *Open Source Systems*, Ernesto Damiani, Brian Fitzgerald, Walt Scacchi, Marco Scotto, and Giancarlo Succi (Eds.). Springer US, Boston, MA, 155–163.

[33] Alan MacCormack, John Rusnak, and Carliss Y. Baldwin. 2006. Exploring the Structure of Complex Software Designs: An Empirical Study of Open Source and Proprietary Code. *Management Science* 52, 7 (2006), 1015–1030. https://doi.org/10.1287/mnsc.1060.0552

[34] Joseph Scott Miller. 2002. Allchin's Folly: Exploring Some Myths About Open Source Software. *Cardozo Arts & Entertainment Law Journal* 20 (2002), 491.

[35] Audris Mockus, Roy T. Fielding, and James D. Herbsleb. 2002. Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 11, 3 (2002), 309–346. https://doi.org/doi:10.1145/567793.567795

[36] Nuthan Munaiah, Steven Kroh, Craig Cabrey, and Meiyappan Nagappan. 2017. Curating GitHub for Engineered Software Projects. *Empirical Software Engineering* 22, 6 (01 Dec 2017), 3219–3253. https://doi.org/10.1007/s10664-017-9512-6

[37] N Pankaja and PK Mukund Raj. 2013. Proprietary Software versus Open Source Software for Education. *American Journal of Engineering Research* 2, 7 (2013), 124–130.

[38] James W. Paulson, Giancarlo Succi, and Armin Eberlein. 2004. An Empirical Study of Open-Source and Closed-Source Software Products. *IEEE Transactions on Software Engineering* 30, 4 (April 2004), 246–256.

[39] Eric S. Raymond. 1999. *The Cathedral and the Bazaar* (1st ed.). O'Reilly & Associates, Inc., USA.

[40] Carol A Robbins, Gizem Korkmaz, José Bayoán Santiago Calderón, Daniel Chen, Claire Kelling, Stephanie Shipp, and Sallie Keller. 2018. Open Source Software as Intangible Capital: Measuring the Cost and Impact of Free Digital Tools. In *Paper from 6th IMF Statistical Forum on Measuring Economic Welfare in the Digital Age: What and How*. 19–20.

[41] Gregorio Robles, Santiago Dueñas, and Jesus Gonzalez-Barahona. 2007. Corporate Involvement of Libre Software: Study of Presence in Debian Code over Time, Vol. 234. 121–132. https://doi.org/10.1007/978-0-387-72486-7_10

[42] Walt Scacchi, Joseph Feller, Brian Fitzgerald, Scott Hissam, and Karim Lakhani. 2006. Understanding Free/Open Source Software Development Processes. *Software Process: Improvement and Practice* 11, 2 (2006), 95–105. https://doi.org/10.1002/spip.255

[43] Diomidis Spinellis. 2008. A Tale of Four Kernels. In *ICSE '08: Proceedings of the 30th International Conference on Software Engineering*, Wilhelm Schäfer, Matthew B. Dwyer, and Volker Gruhn (Eds.). Association for Computing Machinery, New York, 381–390. https://doi.org/10.1145/1368088.1368140

[44] Diomidis Spinellis and Vaggelis Giannikas. 2012. Organizational Adoption of Open Source Software. *Journal of Systems and Software* 85, 3 (March 2012), 666–682. https://doi.org/10.1016/j.jss.2011.09.037

[45] Diomidis Spinellis, Zoe Kotti, Konstantinos Kravvaritis, Georgios Theodorou, and Panos Louridas. 2020. A Dataset of Enterprise-Driven Open Source Software: Extended Description. https://doi.org/10.5281/zenodo.3742854

[46] Diomidis Spinellis, Zoe Kotti, and Audris Mockus. 2020. A Dataset for GitHub Repository Deduplication. In *17th International Conference on Mining Software Repositories (MSR '20)*. Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3379597.3387496

[47] Hyrum K. Wright, Miryung Kim, and Dewayne E. Perry. 2010. Validity Concerns in Software Engineering Research. In *Proceedings of the FSE/SDP Workshop on Future of Software Engineering Research (FoSER '10)*. Association for Computing Machinery, New York, NY, USA, 411–414. https://doi.org/10.1145/1882362.1882446