RESEARCH ARTICLE

Free and Open Source Software organizations: A large-scale analysis of code, comments, and commits frequency

Tadeusz Chełkowski, Dariusz Jemielniak^{*}, Kacper Macikowski

MINDS (Management in Networked and Digital Societies), Department at Kozminski University, Warszawa, Poland

^{*} darekj@kozminski.edu.pl

Abstract

As Free and Open Source Software (FOSS) increases in importance and use by global corporations, understanding the dynamics of its communities becomes critical. This paper measures up to 21 years of activities in 1314 individual projects and 1.4 billion lines of code managed. After analyzing the FOSS activities on the projects and organizations level, such as commits frequency, source code lines, and code comments, we find that there is less activity now than there was a decade ago. Moreover, our results suggest a greater decrease in the activities in large and well-established FOSS organizations. Our findings indicate that as technologies and business strategies related to FOSS mature, the role of large formal FOSS organizations serving as intermediary between developers diminishes.

OPEN ACCESS

Citation: Chełkowski T, Jemielniak D, Macikowski K (2021) Free and Open Source Software organizations: A large-scale analysis of code, comments, and commits frequency. PLoS ONE 16(9): e0257192. <https://doi.org/10.1371/journal.pone.0257192>

Editor: Sergi Lozano, Universitat de Barcelona, SPAIN

Received: November 27, 2020

Accepted: August 25, 2021

Published: September 23, 2021

Peer Review History: PLOS recognizes the benefits of transparency in the peer review process; therefore, we enable the publication of all of the content of peer review and author responses alongside final, published articles. The editorial history of this article is available here: <https://doi.org/10.1371/journal.pone.0257192>

Copyright: © 2021 Chełkowski et al. This is an open access article distributed under the terms of the [Creative Commons Attribution License](https://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Data Availability Statement: All relevant data are within the manuscript and its [Supporting Information](#) files.

Introduction

Online communities in general and Free and Open Source (FOSS) communities in particular, have been a subject of stable academic interest since their inception [1–3]. Although individual FOSS projects have been the subject of many in-depth analyses, the organizations that manage and control FOSS projects have not yet garnered much academic interest.

Within the field of organization studies, researchers have studied topics such as the emergence of new teams from FOSS development networks [4], continued engagement [5], successful productization of peer production in software [6], group activity, dynamics, and social ties [7,8], diversity [9], leadership [10,11], interdependencies [12], the influence of leaders on project sustainability [13], network ties between projects [14], or and IP strategies [15].

As a new phenomenon, FOSS has often been described in terms of its innovative nature, market potential [16], surprising growth [17], the ability to “hack” capitalism [18], and its key differences from traditional software [19]. The gist of much of the literature is that the peak of the FOSS revolution is ahead of us and that we are still observing its growth and maturation [20,21] as organizational and economic regimes continue to change [22].

It is worth noting that “free software” and “open source software” are similar, but not identical, especially in activist circles where they are hotly debated. In order to avoid complex, ideological, and licensing-nuanced discussions we therefore attempt to stay neutral and use Free/Open Source Software (FOSS) as a caveat term [23–25].

Funding: This study was funded by the Polish National Agency for Academic Exchange (NAWA) to DK (PPN/BEK/2018/1/00009). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing interests: The authors have declared that no competing interests exist.

Using “FOSS” to refer to free and/or open source software is a way to capture two different philosophies: the one formulated by Richard Stallman in 1983 and Open Software as defined by Open Source Initiative [26]. We acknowledge that many researchers have traced the roots of open source to early as 1970 [27], but we understand that the term “open source” was coined in 1998 to separate the free programs from Open Source Initiative’s ideas of freedom. The term “Open Source” captures two distinct ideas, therefore it’s worth emphasizing that despite even though in many cases “Open Source” is used as a single term, it refers to two separate movements within the free software community. The first is the mission of promoting computers’ freedom to use software without any cost and copyright restrictions. The second refers to a more practical aspect of making software source codes accessible [28,29]. FOSS now incorporates philosophies and approaches as distant as leftist activism and corporate strategies [27]. For our purposes we are going to refer to FOSS mainly in its politically neutral field of collaborative and organizational practices. See also: <https://gnu.org/gnu/the-gnu-project.html> and <https://opensource.org/osd>.

In the early 21st century it seemed that FOSS would revolutionize society. Wikipedia conquered the market for online encyclopedias and marginalized Britannica [30], Linux became the No.1 server, breaking Microsoft’s monopoly [31], and Firefox was the most popular browser after Internet Explorer bundled with Windows [32]. All these successes led some researchers to hypothesize that FOSS in particular, and peer production in general had the potential to transform late capitalism [18,33]. Sharing and cooperation were expected to emerge as a new modality of economic production [34], leading to a groundbreaking transformation of markets and societies [35]. FOSS, through the creation of new forms of property, would “infect capitalism like a virus,” and challenge the dominant logic of private property and ownership [36–38]. The emergence of private collectives [39], creating new, a-hierarchical and loosely coordinated structures [29], and relying on creation of zero-reproduction costs goods, often of a non-competitive character [36] offered the promise of an entirely new organizational model that would gradually take over the existing ones. They also indicated a fundamentally different approach to organizational innovation [40].

On the surface, the narrative about constant growth and increase in importance seems very plausible. The development and global diffusion of FOSS are quite clear [16]. Even though some projects are naturally abandoned [41], there are certain patterns of growth and decline in FOSS projects [42], and we can reasonably expect FOSS organizations to grow and take over an increasing portion of market share from traditional organizations [43–45].

In the early 1980s, the open source community grew and open source sharing customs were embraced by a growing number of academic and non-academic organizations. In 1985, as a result of the conflict between AT&T and UNIX, Richard Stallman created a Free Software Foundation protecting the right to keep software freely available [46]. The institutionalization of the open source movement produced a variety of organizations structured around an idea, a project, a group of projects, or more recently, software vendors [35,47]. Once an open source group of collaborators reaches a certain size, the norms of sharing, licensing standards and maintenance duties of the community need to be maintained. FOSS organizations adopt or design FOSS licensing standards, distribution methods, software development standards, outside world communication representatives, quality and testing procedures and finally tools for community collaboration. FOSS organizations adopt traditional controlling structures to a degree that was needed to control the release process, but at the same time relaxed enough to preserve the free nature of the open source movement [46–48].

However, even though digital commons, peer production, and open collaboration are still perceived as showing great promise [49,50], and the dream of open organizing’s transformative powers has not been entirely lost [51], the situation has become much more fuzzy in the

past decade. While FOSS has always had some balance of for-profit and for-fun activities [26], large corporations have recently been able to incorporate elements of FOSS organization and approach into their traditional business development strategies [1], and to exploit FOSS software for closed and proprietary products [52]. In fact, even though the FOSS model initially proved a viable alternative to traditional software development methods, it has not been consistently successful in productization: the creation of products that the customers would find easy to understand and use [6,53]. Open organizing is a beautiful idea that showed enormous promise when it took the traditional modes of organizing by surprise, but it may be already past its peak. It is also much more hierarchical and bureaucratic than it originally assumed [54].

To understand the future and place of FOSS in management and society, it is more important than ever to measure engagement in FOSS projects over time across selected small, medium, and large projects. It should allow both the estimation of the general development of FOSS, and reveal the finer details, depending on the size of the organization. Our paper is an attempt to fill this gap.

Project rationale

The Apache Software Foundation is often cited as a paragon of FOSS organization [55–57]. According to Mark Driver, research vice president at Gartner, “The Apache Software Foundation is a cornerstone of the modern open source software ecosystem—supporting some of the most widely used and important software solutions powering today’s Internet economy.” (<https://blogs.apache.org/foundation/entry/apache-is-open>). Indeed, the Apache Software Foundation (ASF) is arguably the most prominent example of a large and successful FOSS organization. It is responsible for the fundamental components of the modern web architecture (Apache HTTP) [58], the backbone of data mining (Apache Hadoop, Apache Spark) and hundreds of tools essential for programming, integration and standardization of the internet as we know it [16,59,60].

Since 1999, Apache has been not only a place for project development, but also a model of open innovation and open collaboration, in many cases displacing traditional software development methods. However, although the Apache Software Foundation is proud of its continuous growth, it is worthwhile to look more closely at the fine-grained details of the community’s activity. For instance, data presented on the official Apache statistics pages (<https://projects.apache.org/statistics.html>), indicate an undeniable success in the growing code base; however, activity measured in community emails and issues presents very interesting fluctuations. According to Fig 1, ASF recorded the highest number of emails (78 846) in March 2016 but that number dropped to 42 814 in October 2017, a level last seen in May 2011 (42 400).

The observed decrease in the email communication can be explained by factors such as change in users’ behavior: many users moved away from email to integrated messaging systems in the code repository interface [61,62]. At the same time, there is empirical evidence of the correlation between the ratio of email messages in public mailing lists to versioning system commits [63] and consequently to project activity as a whole. Thus, it may be a possible signal of decreased participation in FOSS projects. However, even though emails and communication in FOSS have been studied as a proxy for project health and growth [56,64], and mature projects are known to rely on well-structured communication [65], research so far has focused on small samples, precluding a more definitive observation of larger trends over time. This observation has inspired us to conduct what we believe is one of the largest analyses of FOSS projects’ code, gathering data from 1314 individual projects and 1.4 billion lines of code managed.

Fig 1. Emails, topics and authors. Source: <https://projects.apache.org/statistics.html>.

<https://doi.org/10.1371/journal.pone.0257192.g001>

The strength of our study relies on a huge sample of commits, which allows us to make more certain observations about the changes, even though it also makes providing explanations more difficult. Additionally, one advantage of our study is its long-time focus: as only in the longer periods it is possible to observe incremental but clear shifts in the organizational landscape.

Research question

The aim of our study is to improve our understanding of the level of activity among a large sample of FOSS projects. Additionally, data stratification on the FOSS organization gives us a chance to analyze projects from the perspective of the FOSS organizational association. The importance of FOSS organizations such as Apache Software Foundation for the modern networked society could not be overestimated; any change in its community dynamic is an interesting factor from the vantage points of academia and business [60,61]. In order to understand it better, we explore the following research question:

What is the structure of commits, code and comments contribution among the selected Open Source Software Organizations over the last 20 years?

For this article, we've selected a stratified sample of small, medium and large FOSS organizations. We have collected 21 years of quantitative data describing commits frequency, code inserts and deletions as well as data about comments attached to code.

We classify this study as exploratory research on a large data sample, a first step in the direction of deeper case-oriented analysis. In order to answer our research question, we've quantified contributors' activities on the project level. To picture the activity across the analyzed projects, we use simple contingency tables collecting commits, comments, code projects as our main variables and its relationship as calculated variables. We argue that activity in FOSS projects measured in commits, source code, and comments has declined over the last 10 years.

Materials and methods

Data source

Our research data such as number of commits, comments and code lines was collected from Open Hub, a database and public directory of FOSS (Open Hub and the Open Hub logo are trademarks of Black Duck Software, Inc. in the United States and/or other jurisdictions). OpenHub's automated analytics software regularly visits the most popular source versioning systems such as Git, SubVersion, CVS and Bazaar. OpenHub curates data using its large online community where anyone is able to correct and edit OpenHub data entries. It is now arguably the largest and most trustworthy aggregated data source of FOSS. This article is based on the May 20, 2018 snapshot of the OpenHub.net repository. Since the frequency of updating the OpenHub repository may vary, APIs of the developers repositories change, and OpenHub needs time to adapt to these changes, to avoid data inconsistencies we have not used data after December 31, 2017, so as to make sure that the data completeness is as high as possible. Using the custom developed application and publicly available OpenHub API, we have collected a large data sample containing a comprehensive overview of the 20-year history of FOSS organizations.

Data sample and data collection

The result of programmers' work are code lines and comment lines distributed among the number of files, in most cases compiled into an executable software. Programmers are generally producing software using programming languages that fall into one of two categories: interpreted or compiled. Interpreted programming language code must be parsed, and executed each time the program is run. The compiled programs are translated by compilers into a very efficient lower level code that can be executed many times. Some programming languages are using a dual interpreted or compiled paradigm. The main artifact created in the process is a source code represented as a set of statements written in a programming language like Java, C, C++, JavaScript or XML, CSS or HTML tags. Software developed in collaborative environments is created in a series of commits; commits happen every time a developer wants to contribute a piece of work to a shared repository. This process is supported by concurrent developments in software such as Git [66].

The role of concurrent software development applications is to track the changes between the programmer's local environment and synchronize it with a remote repository, making sure that potential code changes and code conflicts are resolved and seamlessly merged (see: <https://git-scm.com/about>). A source-code modification such as adding, modifying, or removing lines of code, adding or removing files, changes in the documentation files, are typical examples of commits. Because of the open nature of software repositories and their accessibility, commits have been a subject of numerous software development studies [67,68], and the activity of developers measured in commits is known to be highly unequal in FOSS organizations.

Additionally, to make a source code clearer and easier for others to understand, programmers add comments to address the meaning of the code block or a code line. As noted by researchers, "source code comments are a valuable instrument to preserve design decisions and to communicate the intent of the code to programmers and maintainers" [69].

For the purpose of our research, each month we analyze the number of code lines and comment lines added by programmers for each project. To retrieve and collect the research information, we have developed an automated application retrieving and parsing data using the headless interface of OpenHub. Our application, which relies on REST API, listed the requested organizations and records, reflecting committers' activity ordered by projects in monthly snapshots.

Dataset selection and stratification

FOSS organizations differ in many ways—some like Eclipse represent large companies and their business goals while others like Apache others started as a single open project which, over time, attracted more programmers with new projects and ideas.

To reduce sampling error and improve the precision of the results, we've divided the FOSS population into homogeneous subgroups before sampling (*stratification*). Subgroups (*strata*) were determined by the size of the FOSS organization, measured in numbers of projects [70–72]. Selection of a project as a stratification criterion has limitations that are discussed in the limitations, data and results section.

Stratum 1 - [LARGE] organizations with managed #Projects ≥ 100

Stratum 2 - [MEDIUM] organizations with $100 > \text{\#Projects} \geq 25$

Stratum 3 - [SMALL] organizations with #Projects < 25

Combined sample consists of ($n = 1314$) projects with MOE (Margin of Error) $\pm 2.30\%$ for the CL (Confidence Level) = 95% and MOE $\pm 3.03\%$ for the CL = 99%). It encompasses 15 FOSS organizations, 16 727 184 commits and over 1.4 billion lines of code. The collected attributes timespan ranges from 11 to 21 years. For each project in each year, we have collected a full 12-month history or a partial history (some project life spans are shorter than 21 years). In total, we have collected 3246 data months, in 9 cases the year data did not include the all months data.

Data record

Each record consists of raw attributes imported from data sources and variables derived from the collected data. Individual record represents a monthly activity for the analyzed project managed by the FOSS organizations. To understand the nature of projects' activity better, we have calculated additional attributes.

First, for better understanding of the project committers' level of activity and the nature of developed software, we measure a coefficient of code submitted per commit using the following equation:

$$\text{CODPC} = \Sigma \text{Lines of Code} / \Sigma \text{Commits}$$

CODPC might indicate the current project stage as frequent commits with a low number of submitted code may indicate that a project is in the maintenance phase [55]. Second, to identify the relationship between the lines of comments submitted in a single commit, we calculate comments per commit coefficient.

$$\text{COMPC} = \Sigma \text{Comments} / \Sigma \text{Commits}$$

This might be an interesting indicator of, for instance, the documentation phase of the project and code creation phase [55]. Lastly, we have calculated the ratio of comments per line of effective source code,

$$\text{COMPCOD} = \Sigma \text{Comments} / \Sigma \text{Code}$$

since a high number of comments per line of actual code may indicate more formal organization processes in a project. The list of variables, with types and source is demonstrated in Table 1.

Sample data record is presented in Table 2.

The number of collected observations varies among organizations, which is well represented in the frequency table (see Table 3).

A comparison of projects, commits, and code size is included in Table 4.

Results

Commits' analysis

Fig 2 shows that the drop in the commits' volume growth affecting large FOSS organizations (stratum 1) started around 2010, and continued to the end of the dataset history. A closer look at the

Table 1. The list of variables, with types and source classification.

Variable name	Description	Variable type	Source
Organization name	Open Source Software Organization name	Classification variable	Collected from the Data Source
Project name	Name of the Open source project	Classification variable	Collected from the Data Source
Organization size	Classification of the organization size—Large, Medium or Small.	Classification variable	Calculated, using managed projects number as a determination criterion.
Lines of code added	Number of code lines added in the observed time (one month).	Main quantitative variable	Collected from the Data Source
Lines of comments added	Number of comments lines added in the observed time (one month)	Main quantitative variable	Collected from the Data Source
Blank Lines added	Number of blank lines added in the observed time (one month)	Supporting quantitative variable	Collected from the Data Source
Commits	Number of commits lines added in the observed time (one month)	Main quantitative variable	Collected from the Data Source
Lines of code removed	Number of code lines removed from the code commit-to-commit comparison in the observed time (one month)	Supporting quantitative variable	Collected from the Data Source
Lines comments Removed	Number of comments lines removed from the code commit-to-commit comparison in the observed time (one month)	Supporting quantitative variable	Collected from the Data Source
Blank lines removed	Number of blank code lines removed from the code commit-to-commit comparison in the observed time (one month)	Supporting quantitative variable	Collected from the Data Source
Code per commits (CODPC)	Calculated value. Code/commits. Variable calculated for all collected cases.	Supporting variable.	Calculated variable.
Comments per commit (COMPC)	Calculated value. Comments/commits. Variable calculated for all collected cases.	Supporting variable.	Calculated variable.
Comments per code (COMPCOD)	Calculated value. Comments/code. Variable calculated for all collected cases.	Supporting variable.	Calculated variable.
Year	Time variable, observation year.	Main data variable	Collected from the Data Source
Month	Time variable, observation month.	Main data variable	Collected from the Data Source

<https://doi.org/10.1371/journal.pone.0257192.t001>

data with trimmed mean (top and bottom 5% of observations have been removed), reveals that the average annual growth for large FOSS organizations was 25.39%, 30.21% for medium FOSS organizations (stratum 2) and 35.68% for the small FOSS organizations (stratum 3). [Table 5](#) presents the combined growth rates for FOSS organizations of all three studied organizations sizes.

Table 2. Example of collected data record.

Organization	Apache
Size	Large
Project	Apache Commons Math
code added	257
code removed	891
comments added	24
comments removed	449
blanks added	17
blanks removed	136
Commits	10
contributors	1
Year	2016
Month	12
CODPC	25.7
COMPC	2.4
COMPCOD	0.093385214

<https://doi.org/10.1371/journal.pone.0257192.t002>

Table 3. Frequency distribution of collected observations.

	Frequency (Fc)	Percent	Cumulative Percent
Apache	22762	26.5	26.5
Debian	3555	4.1	30.6
Eclipse	14666	17.0	47.6
Gentoo	1314	1.5	49.2
GNOME	245	.3	49.5
JBoss	2821	3.3	52.7
Kde	18448	21.4	74.2
Mozilla	5866	6.8	81.0
nasa	1918	2.2	83.2
openstack	1091	1.3	84.5
OSGeo	3073	3.6	88.1
OW2	2788	3.2	91.3
OWASP	1320	1.5	92.8
tdf	248	.3	93.1
wikimedia	5909	6.9	100.0

<https://doi.org/10.1371/journal.pone.0257192.t003>

Code analysis

Fig 3 shows the source code growth dynamic, measured in the number of lines written. It is worth to notice differences between the source code contribution between the three analyzed groups and a dominance of the medium FOSS organizations. In order to have a clear view of the code base we use the code net value as a variable. Code net value represents a number of functional code lines, without banks and comment lines, additionally it deducts the deleted lines, since even a single commit can add and also remove code.

Table 4. Projects, commits and code size comparison.

	# of projects	Σ commits	Σ code added	Σ code removed
Large (#projects>100)				
apache	343	1 828 824	407 946 922	275 627 732
eclipse	172	1 532 770	449 599 501	295 623 314
kde	204	4 619 403	768 319 949	521 996 888
nasa	111	96 374	44 976 530	25 955 044
wikimedia	168	1 164 674	63 501 047	44 812 409
Medium (25<#projects<100)				
debian	29	789 435	710 261 271	460 060 995
JBoss	35	583 720	164 351 117	117 092 382
mozilla	94	2 559 615	777 124 625	516 245 203
OW2	40	501 288	519 017 687	260 478 326
OWASP	63	49 877	31 375 648	23 233 759
Small (#projects<= 25)				
gentoo	14	768 767	39 103 075	28 916 805
GNOME	1	523 280	37 933 312	28 153 328
openstack	13	926 429	76 950 341	51 594 436
OSGeo	25	383 002	70 590 585	48 623 746
tdf	2	399 726	60 838 350	41 705 135
Total	1 314	16 727 184	4 221 889 960	2 740 119 502

<https://doi.org/10.1371/journal.pone.0257192.t004>

Fig 2. Commits analysis in FOSS organizations 1997–2017.

<https://doi.org/10.1371/journal.pone.0257192.g002>

Comments' analysis

In order to have a clear picture of comments contribution among the analyzed FOSS organizations we have introduced a new metric: contributed lines of comments per lines of code (COMPCOD). COMPCOD, described in Table 6, shows differences among large, medium and small FOSS organizations in their code commenting behavior. In the medium FOSS organizations, the largest average value of 0.86 lines of comments per code line was recorded. It's important to emphasize that within the collected observations, only one organization, The Open Web Application Security Project (OWASP), is an outlier with over 2.46 comment lines per code line. Tables 6–8 show the mean COMPCOD, mean of comments line per code lines, and the number of source code lines per commit.

Discussion

Although all the analyzed organizations have grown over the past 20 years (Figs 2 and 3) we have observed lower and decreasing growth rates in the large FOSS organizations when compared to medium and small FOSS organizations (stratum 3). Furthermore, in recent years the commits volume of large FOSS organizations (stratum 1) started to drop by an average of 16.7% annually. The best example of this trend is the fact that in 2017, small FOSS organizations (stratum 3) surpassed the large FOSS organizations (stratum 1) commits volume by 10.8%. This is surprising, as 10 years earlier the commits volume of large FOSS organizations (stratum 1) was more than 6 times bigger than that of small FOSS organizations (stratum 3) (626 136 to 93 512) and over three and a half-time bigger than medium FOSS organizations (stratum 2) (626 136 to 172 843) (Fig 2).

Table 5. Combined growth rates for large, medium and small FOSS organizations 1997–2017.

	Large	Medium	Small
Trimmed mean (5%)	25.59%	30.21%	35.68%
Mean	28.27%	56.17%	44.42%
Max	142.27%	631.01%	273.87%
Min	-34.88%	-25.42%	-18.95%

<https://doi.org/10.1371/journal.pone.0257192.t005>

Fig 3. Source code committed in number of lines.

<https://doi.org/10.1371/journal.pone.0257192.g003>

Moreover, as Fig 2 shows, the drops in the commits' volume growth is affecting large FOSS organizations disproportionately, and this phenomenon was not observable in the first 10 + years. As Table 5 shows, the trimmed mean (top and bottom 5% removed) the change is dramatic: The drop in the large FOSS organizations activity measured in commits can be demonstrated by one comparison—commits in 2017 represented only approximately 30% of commits compared to the record high in 2010 (2010–951 294, 2017–287 907).

In the first decade of the period under analysis (1997–2017), we observed a steady growth in the medium and large FOSS organizations, while small FOSS code growth dynamics tended to fluctuate. However, from 2010, the organizations managing over 100 projects started to receive less new source code than in the previous decade. Compared to 2010, when users

Table 6. Mean COMPCOD (mean).

	Mean of comments lines per code line
Large(#projects>100)	
apache	0.61270
eclipse	0.51178
kde	0.32221
nasa	0.39455
wikimedia	0.32492
Medium(25<#projects<100)	
debian	0.30002
JBoss	0.43842
mozilla	0.44154
OW2	0.68189
OWASP	2.48573
Small(#projects<= 25)	
gentoo	0.22984
GNOME	0.14378
openstack	0.20929
OSGeo	0.44055
Tdf	0.67851

<https://doi.org/10.1371/journal.pone.0257192.t006>

Table 7. Mean of comments line per code lines (COMPCOD).

Mean values of comments lines per code line (COMPCOD)	Large(#projects>100)	Medium (25<#projects <100)	Small (#projects< = 25)
1997	0.15529	0.25613	0.16807
1998	0.14792	0.24806	0.21714
1999	0.21548	1.15850	0.31579
2000	0.26096	0.26674	0.28325
2001	0.31828	0.60631	0.23779
2002	0.28140	0.29048	0.33766
2003	0.47377	0.93073	0.65974
2004	1.36562	1.60681	0.37094
2005	0.42731	0.43290	0.36057
2006	1.07082	0.31924	0.31417
2007	0.37597	1.00306	0.27286
2008	0.63819	0.29977	0.26000
2009	0.36495	0.32552	0.28688
2010	0.64023	2.14643	0.95249
2011	0.35907	0.26990	0.23549
2012	0.44546	0.34084	0.49329
2013	0.33473	0.45488	0.24373
2014	0.31285	0.50129	0.22618
2015	0.29581	0.89144	0.33060
2016	0.41132	0.23855	0.32675
2017	0.59586	0.21995	0.22574
Trimmed mean (5% bottom and 5% top)	0.4199	0.5495	0.3157
Mean	0.4520	0.6099	0.3390
SD	0.27530	0.50011	0.16956

<https://doi.org/10.1371/journal.pone.0257192.t007>

committed over 100 million of the net new source code lines, in the year 2012 large FOSS organizations (stratum 1) received only 36.1 million lines. In that same period, medium FOSS organizations (stratum 2) surpassed large FOSS organizations and by 2012 they had received almost 3.5 times more net new source code than large FOSS organizations. One of the most surprising findings is a noticeable “time shift” of the growth dynamic between the medium and large FOSS organizations. The observed decrease in the code base growth dynamic starts two years later in medium FOSS organizations (stratum 2) and two years later than that in small FOSS organizations.

In a deeper analytical look at our proposed metric COMPCOD (Table 6), studying the distribution reveals that one of the OWASP projects, “The OWASP Zed Attack Proxy,” described as “. . . one of the world’s most popular free security tools and is actively maintained by hundreds of international volunteers,” (<https://github.com/zaproxy/zaproxy/wiki>) includes the code of conduct, instructions and even the elements of documentation in the comments sections. Regardless of the outliers, projects associated with large FOSS organizations registered less activity than projects in the medium FOSS organizations (stratum 2) with comments per line 0.45 ratio.

Finally, small FOSS organizations (stratum 3) are the least active in code comments, providing approximately 1 line of comment for every 3 lines of code (COMPCOD = 0.34). Additionally, an analysis of 20 years of comments per code history shows that standard deviation in large and small FOSS organizations (stratum 3) is smaller.

Table 8. Number of source code lines per commit (CODPC).

	Large(#projects>100)	Medium (25< #projects<100)	Small (#projects <= 25)
1997	159.6457	213.6047	188.9899
1998	121.7883	1367.6196	63.7804
1999	122.4032	99.2152	585.9229
2000	100.4900	95.5775	169.4701
2001	137.2856	129.0426	116.8105
2002	112.5782	169.6384	147.2365
2003	75.3101	217.3441	133.9808
2004	202.8971	304.0662	77.2385
2005	254.2520	251.1138	107.8710
2006	208.5257	484.9788	188.4939
2007	209.4148	373.9491	165.4688
2008	218.2877	374.0096	110.8150
2009	212.1694	300.7204	149.9105
2010	222.8003	296.1230	206.8572
2011	252.7280	376.7635	352.3639
2012	218.6569	715.4053	131.8826
2013	209.0371	500.2177	131.1219
2014	226.6461	612.1060	130.7347
2015	250.4403	1202.5672	102.1072
2016	215.5444	555.9114	105.3622
2017	227.4132	184.8912	143.2757
Trimmed mean (5% bottom and 5% top)	190.9870	387.4562	150.5258
Average	188.4912	420.2317	167.1283
SD	51.88962	251.80445	110.31318

<https://doi.org/10.1371/journal.pone.0257192.t008>

Conclusions

Our results indicate a shift in contribution activity across FOSS projects of different sizes and growth stages over time, and that the largest organizations are slowing their growth at a faster pace than medium and small organizations.

There are many possible reasons for the observed phenomenon. We cannot exclude the possibility that the modalities of cooperation have changed over time and that the measures we are using do not hold a stable accuracy over the whole period.

However, if the results reflect the actual changes in FOSS organizations and projects, they are quite troubling for the open source movement.

One possible interpretation of this phenomenon is that open source, as an approach to developing projects, has lost some of its appeal. It is worth remembering that at first, FOSS principles were interpreted as bringing together an ideological paradigm shift (openness), governance and technological innovations [52,73]. These three areas were conflated into one, and raised the hopes of early enthusiasts that openness as a social norm is inseparable from and consequent to the other two, and supports a redefinition of labor leading to the reshaping of capitalism. In other words, the dominant assumption was that as new forms of governance and technology promote open organizing, we can expect traditional organizing to be gradually replaced, and the far-reaching consequences, according to some authors, may even change the capitalism as we know it [35].

Our study does not allow us to make claims about causality, and as our interpretation here is speculative, it should be treated with caution. However, what we believe may be happening

is the result of FOSS technology and organizational model maturing and becoming mainstream. While initially the governance and technological innovations indeed led to a wider adaptation of openness as a dominant logic, the traditional organizations soon learned how to use (and sometimes abuse) these two innovations to create closed ecosystems and gatekeep their position.

The successes of Google in leveraging Android to win commercially on a mobile market, or of WordPress to build a regular business based on FOSS principles, as well as a series of take-overs, such as acquiring GitHub by Microsoft for 7.5 billion dollars, and acquiring RedHat by IBM for the staggering sum of 34 billion dollars, all show that rather than transforming society, FOSS may be trimmed and harnessed for traditional corporate goals. While open source may be on the rise as an effective organizing principle [74], it has been disentangled from at least some of its original premises. The principles of sharing economy, rooted in collaborative, pro-social, and anti-commercial ideals [75] have also been used rhetorically and adjusted for the mainstream economy, leading to further exploitation and inequality [76]. In a way, FOSS movement has both “won and lost the war” [77], as it has been widely accepted as a form of software development, but the profits deriving from it have largely been appropriated by corporations. In its 2.0 version, FOSS development becomes yet another business model [78], bordering freemium more than a revolutionary society-changing movement.

The ideologies of openness, sharing, and collaborating are being repurposed for business as usual [79,80]. The openness of software have become routine factors for influencing productivity and efficiency [81,82]. Moreover, open collaboration software development turned out to be much less collaborative in an actual daily practice had been assumed [83,84].

Moreover, far from being stable, FOSS organizations underwent major adaptations to the environment. One of the major roles of FOSS organizations to nurture interactions among community members, calling actions, setting guiding principles or developing tools to facilitate collaborative software development and streamline coordination [85,86]. Benefits provided for the FOSS developers and users by the FOSS organizations, such as Apache Software Foundation, Mozilla Foundation or Linux Foundation, include project governance and vital institutional support infrastructure [87,88]. Users or contributors can rely on an organizational framework for intellectual property rights management as well as for legal support and well-defined development and maintenance processes. In many cases FOSS organizations exist as communities of practice, where people engage in collective development, learning and solving similar problems [89].

Yet, as technologies develop and organizational practices mature, some functions that had previously been crucial in FOSS development and provided by FOSS organizations may be replaced by software and online services. While this paper does not analyze the new emerging FOSS organizations especially created after 2017, the analyzed data provides evidence that activities measured as commits are declining, and may have dire side effects for the entire FOSS movement. The existence of large FOSS organizations has made big policy and activism possible. Promoting big ideological changes in the areas of open licensing, fairness in digital files sharing [90], sharing rather than selling as a principle of contemporary society, or openness in general as a strong social norm [91] would not have been possible without their support. Large FOSS organizations brought grand projects, such as new operating systems (Linux) or productivity suites (such as OpenOffice) into existence. These large projects were essential for the belief that the emerging peer-to-peer economy and the new commons may make a larger impact on the society that went beyond isolated cases of software [43,92].

It is possible that ideological manifestos, postulating openness as a new principle of social organizing, having a potential for transformative influence on capitalism, may not have had as much appeal as it seemed. Yes, FOSS organizations paved the way to distributed structures and to making openness an organizing principle, and according to some measures their

influence on capitalism may have been profound. They also developed tools and processes that made virtual collaboration more effective. Yet, our results may indicate that as soon as the traditional organizations caught up on both of these fronts, FOSS organizations, and especially the large ones, started to lose momentum.

It may be that the demand for a revolution simply was not there, and the general public couldn't care less about openness. Even though projects with a non-market sponsor, as well as with open licenses used to be able to attract greater user interest over time in the past [93], the successes of services such as TripAdvisor, Quora, Google Guides or Yelp have made it abundantly clear that many users do not have a problem with creating collective content for a for-profit company, which uses this content on a restrictive license, and relies on corporate-decided community governance without any open collaboration in regards to organizational structures and roles. They just enjoy a friendly UI, and a peer production mode of contributing. The final nail in the coffin has been the rise of centralized cloud services such as GitHub or BitBucket, which have met many of the organizational and cooperative needs of developers that were previously addressed by open designs.

Limitations of the research model, data and results

Our study relies on data from the period of 1997–2017, and does not cover the most recent changes in the open source environment. While this approach is reasonable because of the data availability and comparability, it should be noted that in recent years FOSS organizations have explored new ways of supporting open source projects, and new ways of managing coordination, including ways more difficult to measure and compare to previous years.

This paper is a quantitative conceptualization of the activity levels in a stratified sample of projects associated with FOSS organizations. Even though OpenHub is a reliable source of data, the results should be considered within the trust boundary of the data source. There is no guarantee that all projects, commits, comments, or organizations are fully represented in the OpenHub database. It is also worth mentioning that the results are applicable only to FOSS projects associated with formal FOSS organizations, thus the results do not represent the full population of FOSS projects. The proposed perspective of looking at the FOSS organizations through the lens of projects, commits, submitted code, and code commits may not fully represent all behind-the-scenes activities, including important cooperative behaviors not related to coding, but providing the much-needed social glue of interactions. It is widely accepted that communication among FOSS collaborators happens in many different channels [94–97], and we have studied only the structured, technical ones. There are many activities that foster cooperation, and that are not code-centric [98,99]. Researchers used different methods to understand the nature of FOSS collaborations such as Social Network Analysis or dedicated metrics for understanding the nature of the FOSS models [58,63,84,100].

Despite these trends, our findings need to be reconfirmed through other methods. Our research raises many questions about the potential change in the way that FOSS processes are organized. Since the selected data source and perspective criteria introduce natural bias into our results, these results should not be unreflexively used to generalize to other FOSS communities or organizations. Moreover, as our results' main strength is the sample size, it is also its major weakness, as it makes an explanatory approach—seeking correlations, reasons, and causes—much more difficult.

Final remarks

Our study is an attempt to determine the basic quantitative indicators of growth of FOSS organizations. We have discovered interesting trends in commits, comments and code growth

dynamics, indicating that there has been a change in the activity levels across all types of FOSS organizations. FOSS organizations are still gaining new code, but the collaborative efforts measured in commits, committed code, and comments are lower than they were in 2020. Medium and small FOSS organizations seem to be less affected by the overall slowdown, still attracting new users but not as quickly as in the past. These results might be explained by the increasing adoption of FOSS collaborative online services such as GitHub and BitBucket. With more tools and simpler collaborative processes there may be a diminishing need for organizational proxies, because people can create ad hoc short-lived structures without dedicated processes and formal committees. If the original success of FOSS was even partly a result of this form of organization substituting for what can be more easily achievable through online services and software tools, it is quite understandable that FOSS organizations develop less dynamically. However, if this is what is happening, the practical implications are considerable: instead of revolutionizing the society or even just software development, FOSS will turn out to be a modest innovation, one that temporarily helped resolve some structural and communication issues, but only until the mainstream organizations have absorbed some of its model, and until regular project management tools have sufficiently evolved.

Another possible explanation may be that we observe the maturation and aging of the FOSS development model: it not only no longer relies on archetypal hacking-for-fun, but it also has entered a stage in which many projects require maintenance and stability, and are much less reliant on frequent communication and commits. If this is the case, the FOSS model is not going to disappear any time soon, but it is still not going to make any radical organizational difference, and will remain a temporary fad in the organization of work.

Finally, we cannot exclude the possibility that the larger FOSS organization are all falling prey to the “rise and decline” phenomenon observed in Wikipedia and some other peer production projects [101,102], and rooted in the fossilization of procedures, and the growth of quality control systems.

We believe that in the near future we may observe a steady decline in the role of the large and formal organizations as large independent FOSS organizations are replaced by corporate-driven FOSS foundations. Perhaps the free software-oriented movement will reorganize itself into smaller, dynamic, tools-oriented networks. FOSS will probably not die, but it may not really live.

Supporting information

S1 Dataset.
(XLSX)

Author Contributions

Conceptualization: Tadeusz Chełkowski.

Data curation: Tadeusz Chełkowski, Kacper Macikowski.

Formal analysis: Tadeusz Chełkowski, Kacper Macikowski.

Investigation: Tadeusz Chełkowski, Dariusz Jemielniak.

Methodology: Tadeusz Chełkowski.

Software: Kacper Macikowski.

Writing – original draft: Tadeusz Chełkowski, Dariusz Jemielniak.

Writing – review & editing: Tadeusz Chełkowski, Dariusz Jemielniak.

References

1. SL Daniel LM, Maruping L, Cataldo M, Herbsleb J. The impact of ideology misfit on open source software communities and companies. *Management information systems quarterly*. 2018; 42. Available: <https://par.nsf.gov/biblio/10108572>.
2. Levy S. *Hackers: Heroes of the computer revolution*. New York: Penguin Books; 2001.
3. Shaikh M, Vaast E. Folding and Unfolding: Balancing Openness and Transparency in Open Source Communities. *Information Systems Research*. 2016; 27: 813–833.
4. Hahn J, Moon JY, Zhang C. Emergence of New Project Teams from Open Source Software Developer Networks: Impact of Prior Collaboration Ties. *Information Systems Research*. 2008; 19: 369–391.
5. Barrett M, Oborn E, Orlikowski W. Creating value in online communities: The sociomaterial configuring of strategy, platform, and stakeholder engagement. *Inf Syst Res*. 2016; 27: 704–723.
6. Feller J, Finnegan P, Fitzgerald B, Hayes J. From Peer Production to Productization: A Study of Socially Enabled Business Exchanges in Open Source Service Networks. *Information Systems Research*. 2008; 19: 475–493.
7. Butler BS. Membership Size, Communication Activity, and Sustainability: A Resource-Based Model of Online Social Structures. *Information Systems Research*. 2001; 12: 346–362.
8. Rishika, Rishika R, Ramaprasad J. The Effects of Asymmetric Social Ties, Structural Embeddedness, and Tie Strength on Online Content Contribution Behavior. *Management Science*. 2019. pp. 3398–3422. <https://doi.org/10.1287/mnsc.2018.3087>
9. Ren Y, Chen J, Riedl J. The Impact and Evolution of Group Diversity in Online Open Collaboration. *Manage Sci*. 2016; 62: 1668–1686.
10. Jemielniak D. Naturally emerging regulation and the danger of delegitimizing conventional leadership: Drawing on the example of Wikipedia. In: Bradbury H, editor. *The SAGE Handbook of Action Research*. London, UK—New Delhi, India—Thousand Oaks, CA: Sage; 2015.
11. Johnson SL, Safadi H, Faraj S. The Emergence of Online Community Leadership. *Information Systems Research*. 2015; 26: 165–187.
12. Lindberg A, Berente N, Gaskin J, Lyytinen K. Coordinating Interdependencies in Online Communities: A Study of an Open Source Software Project. *Information Systems Research*. 2016; 27: 751–772.
13. Lee JY-H, Yang C-S, Hsu C, Wang J-H. A longitudinal study of leader influence in sustaining an online community. *Information & Management*. 2019; 56: 306–316.
14. Peng G, Wan Y, Woodlock P. Network ties and the success of open source software development. *The Journal of Strategic Information Systems*. 2013; 22: 269–281.
15. Wen W, Ceccagnoli M, Forman C. Opening up IP Strategy: Implications for Open Source Software Entry by Start-Up Firms. 2015. Available: <https://papers.ssrn.com/abstract=2590198>.
16. Lakka S, Michalakelis C, Varoutas D, Martakos D. Exploring the determinants of the OSS market potential: The case of the Apache web server. *Telecomm Policy*. 2012; 36: 51–68.
17. Fernandez-Ramil J, Lozano A, Wermelinger M, Capiluppi A. Empirical Studies of Open Source Evolution. In: Mens T, Demeyer S, editors. *Software Evolution*. Berlin, Heidelberg: Springer Berlin Heidelberg; 2008. pp. 263–288.
18. Söderberg J. *Hacking Capitalism: The Free and Open Source Software Movement*. Routledge; 2015.
19. Paulson JW, Succi G, Eberlein A. An empirical study of open-source and closed-source software products. *IEEE Trans Software Eng*. 2004; 30: 246–256.
20. Deshpande A, Riehle D. The total growth of open source. *Open Source Development, Communities and Quality*, 275: 197–209. 2008.
21. Schrape J-F. Open-source projects as incubators of innovation: From niche phenomenon to integral part of the industry. *Convergence: The International Journal of Research into New Media Technologies*. 2019. pp. 409–427. <https://doi.org/10.1177/1354856517735795>
22. Davis GF. Can an Economy Survive Without Corporations? *Technology and Robust Organizational Alternatives*. AMP. 2016; 30: 129–140.
23. Colford S. Explaining Free and Open Source Software Open Source Software in Libraries. *Bulletin of the American Society for Information Science and Technology*. 1986; 35: 10–14.
24. Stallman R, Free Software Foundation (Cambridge M). *Free software, free society: selected essays of Richard M. Stallman*. dl.acm.org; 2002. pp. 72–90.
25. Tsai J. For Better or Worse: Introducing the GNU General Public License Version 3. *Berkeley Technol Law J*. 2008; 23: 547–581.

26. Tozzi C, Zittrain J. *For Fun and Profit: A History of the Free and Open Source Software Revolution*. MIT Press; 2017.
27. Coleman B, Hill M. How Free Became Open and Everything Else under the Sun: Introduction. *M/C Journal*. 2004. Available: <https://journal.media-culture.org.au/index.php/mcjournal/article/view/2352>.
28. Brethauer D. Open source software: A history. 2001. Available: http://opencommons.uconn.edu/libr_pubs/7/. PMID: 11926296
29. Raymond ES. *The cathedral and the bazaar*. Beijing-Cambridge: O'Reilly; 1999.
30. Jemielniak D. *Common knowledge?: An ethnography of Wikipedia*. Stanford: Stanford University Press; 2014.
31. Casadesus-Masanell R, Ghemawat P. Dynamic Mixed Duopoly: A Model Motivated by Linux vs. Windows. *Manage Sci*. 2006; 52: 1072–1084.
32. Casalo LV, Cisneros J, Flavián C, Guinalíu M. Determinants of success in open source software networks. *Industrial Management & Data Systems*. 2009. pp. 532–549. <https://doi.org/10.1108/02635570910948650>
33. Benkler Y. Freedom in the commons: Towards a political economy of information. *Duke Law J*. 2003; 1245–1276.
34. Benkler Y. Sharing nicely: On shareable goods and the emergence of sharing as a modality of economic production. *Yale Law J*. 2004; 273–358.
35. Benkler Y. *The Wealth of Networks: How Social Production Transforms Markets and Freedom*. Yale University Press; 2006.
36. Romer PM. Endogenous Technological Change. *J Polit Econ*. 1990; 98: S71.
37. Romer PM. The Origins of Endogenous Growth. *J Econ Perspect*. 1994; 8: 3–22. <https://doi.org/10.1257/jep.8.3.3> PMID: 10136763
38. Schoonmaker S. HACKING THE GLOBAL. *Information, Communication & Society*. 2012. pp. 502–518. <https://doi.org/10.1080/1369118x.2012.665938>
39. Von Hippel E, Von Krogh G. Open source software and the “private-collective” innovation model: Issues for organization science. *Organization Science*. 2003; 14: 209–223.
40. von Krogh G. & von Hippel E. The Promise of Research on Open Source Software. *Manage Sci*. 2006; 52: 975–983.
41. Khondhu J, Capiluppi A, Stol K-J. Is It All Lost? A Study of Inactive Open Source Projects. *Open Source Software: Quality Verification*. 2013. pp. 61–79. https://doi.org/10.1007/978-3-642-38928-3_5.
42. Schweik CM. Sustainability in Open Source Software Commons: Lessons Learned from an Empirical Study of SourceForge Projects. *Technology Innovation Management Review*. 2013. pp. 13–19. <https://doi.org/10.22215/timreview645>
43. Benkler Y, Shaw A, Hill BM. *Peer production: A form of collective intelligence*. Cambridge, MA: MIT Press; 2015.
44. David M. Sharing: post-scarcity beyond capitalism? *Cambridge Journal of Regions, Economy and Society*. 2017. pp. 311–325. <https://doi.org/10.1093/cjres/rsx003>.
45. Margan D, Candric S. The success of open source software: A review. 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). 2015. <https://doi.org/10.1109/mipro.2015.7160503>
46. Lee J-A. *Nonprofit Organizations and the Intellectual Commons*. Edward Elgar Publishing; 2012.
47. Vetter GR. The Collaborative Integrity of Open-Source Software. *Utah Law Rev*. 2004; 563.
48. Kavanagh P. How Open Source Software Is Developed. *Open Source Software*. 2004. pp. 203–219. <https://doi.org/10.1016/b978-155558320-0/50010-6>
49. Kostakis V. In defense of digital commoning. *Organization*. 2018; 1350508417749887. <https://doi.org/10.1177/1350508417726547> PMID: 30369827
50. Kostakis V, Bauwens M. *Network Society and Future Scenarios for a Collaborative Economy*. New York: Springer; 2014.
51. Benkler Y. Peer production, the commons, and the future of the firm. *Strategic Organization*. 2017; 15: 264–274.
52. Beverungen A, Böhm S, Land C. Free Labour, Social Media, Management: Challenging Marxist Organization Studies. *Organization studies*. 2015; 36: 473–489.
53. Tucci CL, Afuah A, Viscusi G. *Creating and Capturing Value through Crowdsourcing*. Oxford University Press; 2018.

54. Kreiss D, Finn M, Turner F. The limits of peer production: Some reminders from Max Weber for the network society. *New Media & Society*. 2011; 13: 243–259.
55. Crowston K, Howison J. Hierarchy and centralization in free and open source software team communications. *Knowledge, Technology & Policy*. 2006; 18: 65–85.
56. Crowston K, Shamshurin I. Core-periphery communication and the success of free/libre open source software projects. *Journal of Internet Services and Applications*. 2017; 8: 10.
57. Katz DS, McInnes LC, Bernholdt DE, Mayes AC, Chue Hong NP, Duckles J, et al. Community Organizations: Changing the Culture in Which Research Software Is Developed and Sustained. *Computing in Science & Engineering*. 2019. pp. 8–24. <https://doi.org/10.1109/mcse.2018.2883051>
58. Fielding RT. Shared leadership in the Apache project. *Commun ACM*. 1999; 42: 42–43.
59. Oliva GA, Santana FW. Characterizing key developers: a case study with apache ant. *Collaboration and ...* 2012. Available: http://link.springer.com/chapter/10.1007/978-3-642-33284-5_8.
60. Raymond ES. The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary. 119.93.23.123; 2001. p. 241.
61. Mockus A, Fielding RT, Herbsleb J. A case study of open source software development: the Apache server. *Proceedings of the 21nd International Conference on Software Engineering*. 2000; 263–272.
62. Mockus A, Fielding RT, Herbsleb JD. Two case studies of open source software development: Apache and Mozilla. *ACM Trans Softw Eng Methodol*. 2002; 11: 309–346.
63. Gala-Pérez S, Robles G, González-Barahona JM, Herraiz I. Intensive metrics for the study of the evolution of open source projects: Case studies from Apache Software Foundation projects. *MSR '13 Proceedings of the 10th Working Conference on Mining Software Repositories*. 2013; 159–168.
64. Guzzi A, Bacchelli A, Lanza M, Pinzger M, van Deursen A. Communication in open source software development mailing lists. 2013 10th Working Conference on Mining Software Repositories (MSR). 2013. pp. 277–286.
65. Foss N j., Frederiksen L, Rullani F. Problem-formulation and problem-solving in self-organized communities: How modes of communication shape project behaviors in the free open-source software community. *Strat Mgmt J*. 2016; 37: 2589–2610.
66. Arafat O, Riehle D. The Commit Size Distribution of Open Source Software. 2009 42nd Hawaii International Conference on System Sciences. 2009. <https://doi.org/10.1109/HICSS.2009.421>
67. Sbati N, Lenarduzzi V, Taibi D, Ben Sassi S, Ghezala HHB. Exploring information from OSS repositories and platforms to support OSS selection decisions. *Information and Software Technology*. 2018. pp. 104–108. <https://doi.org/10.1016/j.infsof.2018.07.009>
68. Song J, Kim C. What Is Needed for the Sustainable Success of OSS Projects: Efficiency Analysis of Commit Production Process via Git. *Sustainability*. 2018. p. 3001. <https://doi.org/10.3390/su10093001>
69. Fluri B, Würsch M, Giger E, Gall HC. Analyzing the co-evolution of comments and source code. *Software Quality Journal*. 2009; 17: 367–394.
70. Thompson SK. Adaptive sampling in behavioral surveys. *NIDA Res Monogr*. 1997; 167: 296–319. PMID: 9243567
71. Wackerly D, Mendenhall W, Scheaffer RL. *Mathematical Statistics with Applications*. Cengage Learning; 2014.
72. Thompson SK, editor. *Stratified Sampling: Thompson/Sampling 3E*. Sampling. Hoboken, NJ, USA: John Wiley & Sons, Inc.; 2012. pp. 139–156.
73. Demil B, Lecocq X. Neither market nor hierarchy nor network: The emergence of bazaar governance. *Organization Studies*. 2006; 27: 1447–1466.
74. Germonprez M, Lipps J, Goggins S. The rising tide: Open source's steady transformation. *First Monday*. 2019. <https://doi.org/10.5210/fm.v24i8.9297>
75. Karger T. The meaning of sharing in free software and beyond. *Information, Communication & Society*. 2019. pp. 1295–1309. <https://doi.org/10.1080/1369118x.2017.1418016>
76. Schor JB, Attwood-Charles W. The “sharing” economy: labor, inequality, and social connection on for-profit platforms. *Sociology Compass*. 2017; 11: e12493.
77. O'Neil M, Muselli L, Raissi M, Zacchioli S. “Open source has won and lost the war”: Legitimising commercial-communal hybridisation in a FOSS project. *New Media & Society*. 2020. p. 146144482090702. <https://doi.org/10.1177/1461444820907022>
78. Fitzgerald B. The transformation of open source software. *Miss Q*. 2006; 30: 587–598.
79. Arvidsson A. Value and virtue in the sharing economy. *Sociol Rev*. 2018; 66: 289–301.
80. Jemielniak D, Przeglasińska A. *Collaborative Society*. Cambridge, MA: MIT Press; 2020.

81. Parker G, Van Alstyne M. Two-Sided Network Effects: A Theory of Information Product Design. *Manage Sci.* 2005; 51: 1494–1504.
82. Nagle A. Kill All Normies: Online Culture Wars From 4Chan And Tumblr To Trump And The Alt-Right. John Hunt Publishing; 2017.
83. Crowston K, Howison J. Assessing the health of open source communities. *Computer.* 2006; 39: 89–91.
84. Chelkowski T, Gloor P, Jemielniak D. Inequalities in Open Source Software Development: Analysis of Contributor's Commits in Apache Software Foundation Projects. *PLoS One.* 2016; 11: e0152976. <https://doi.org/10.1371/journal.pone.0152976> PMID: 27096157
85. Shaikh M, Henfridsson O. Governing open source software through coordination processes. *Information and Organization.* 2017. pp. 116–135. <https://doi.org/10.1016/j.infoandorg.2017.04.001>
86. Lee S, Baek H, Jahng J. Governance strategies for open collaboration: Focusing on resource allocation in open source software development organizations. *International Journal of Information Management.* 2017. pp. 431–437. <https://doi.org/10.1016/j.ijinfomgt.2017.05.006>
87. Lindman J, Hammouda I. Support mechanisms provided by FLOSS foundations and other entities. *Journal of Internet Services and Applications.* 2018. <https://doi.org/10.1186/s13174-018-0079-2>
88. Redlich T, Moritz M, Wulfsberg JP. Co-Creation: Reshaping Business and Society in the Era of Bottom-up Economics. Springer; 2018.
89. Mogri M, Mei X, Qiu L, Bandyopadhyay S. Effect of “Following” on Contributions to Open Source Communities. *Journal of Management Information Systems.* 2018. pp. 1188–1217. <https://doi.org/10.1080/07421222.2018.1523605>
90. Hergueux J, Jemielniak D. Should digital files be considered a commons? Copyright infringement in the eyes of lawyers. *The Information Society.* 2019; 35: 198–215.
91. Tkacz N. Wikipedia and the Politics of Openness. Chicago: University of Chicago Press; 2015.
92. Bauwens M, Kostakis V, Pazaitis A. Peer to Peer: The Commons Manifesto. University of Westminster Press; 2019.
93. Stewart KJ, . . . APA-. IS, 2006 U. Impacts of license choice and organizational sponsorship on user interest and development activity in open source software projects. *Information Systems Research.* Available: <https://pubsonline.informs.org/doi/abs/10.1287/isre.1060.0082>.
94. Bergquist M, Ljungberg J. The power of gifts: organizing social relationships in open source communities. *Information Systems Journal.* 2001; 11: 305–320.
95. Lakhani K, Wolf R. Does free software mean free labor? Characteristics of participants in open source communities. BCG Survey Report. 2001; BCG Survey Report.
96. Samoladas I, Angelis L, Stamelos I. Survival analysis on the duration of open source projects. *Information and Software Technology.* 2010; 52: 902–922.
97. Hars A. Working for Free? Motivations for Participating in Open-Source Projects. *International Journal of Electronic Commerce.* 2002; 6: 25. 15p. 3 Charts.
98. Carillo KDA, Huff S, Chawner B. It's Not Only about Writing Code: An Investigation of the Notion of Citizenship Behaviors in the Context of Free/Libre/Open Source Software Communities. 2014 47th Hawaii International Conference on System Sciences. 2014. pp. 3276–3285.
99. Fang Y, Neufeld D. Understanding Sustained Participation in Open Source Software Projects. *Journal of Management Information Systems.* 2009; 25: 9–50.
100. Koch S, Schneider G. Results from software engineering research into open source development projects using public data. *Diskussionspapiere zum Tätigkeitsfeld Informationsverarbeitung und Informationswirtschaft.* 2000.
101. Halfaker A, Stuart Geiger R, Morgan JT, Riedl J. The Rise and Decline of an Open Collaboration System. *American Behavioral Scientist.* 2013. pp. 664–688. <https://doi.org/10.1177/0002764212469365>
102. TeBlunthuis N. Density Dependence Without Resource Partitioning on an Online Petitioning Platform. University of Washington Libraries; 2017.