Corporate Dominance in Open Source Ecosystems: A Case Study of OpenStack

Yuxia Zhang*
Beijing Institute of Technology
Beijing, China
yuxiazh@bit.edu.cn

Klaas-Jan Stol
Lero, the SFI Research Centre for Software
University College Cork
Cork, Ireland
k.stol@ucc.ie

Hui Liu*
Beijing Institute of Technology
Beijing, China
liuhui08@bit.edu.cn

Minghui Zhou
School of Computer Science, Peking University
Key Laboratory of High Confidence Software Technologies, Ministry of Education
Beijing, China
zhmh@pku.edu.cn

ABSTRACT

Corporate participation plays an increasing role in Open Source Software (OSS) development. Unlike volunteers in OSS projects, companies are driven by business objectives. To pursue corporate interests, companies may try to dominate the development direction of OSS projects. One company's domination in OSS may 'crowd out' other contributors, changing the nature of the project, and jeopardizing the sustainability of the OSS ecosystem. Prior studies of corporate involvement in OSS have primarily focused on predominately positive aspects such as business strategies, contribution models, and collaboration patterns. However, there is a scarcity of research on the potential drawbacks of corporate engagement. In this paper, we investigate corporate dominance in OSS ecosystems. We draw on the field of Economics and quantify company domination using a dominance measure; we investigate the prevalence, patterns, and impact of domination in the evolution of the OpenStack ecosystem. We find evidence of company domination in over 73% of the repositories in OpenStack, and approximately 25% of companies dominate one or more repositories per version. We identify five patterns of corporate dominance: Early incubation, Full-time hosting, Growing domination, Occasional domination, and Last remaining. We find that domination has a significantly negative relationship with the survival probability of OSS projects. This study provides insights for building sustainable relationships between companies and the OSS ecosystems in which they seek to get involved.

*Corresponding authors

ESEC/FSE '22, November 14–18, 2022, Singapore, Singapore

ACM ISBN 978-1-4503-9413-0/22/11…$15.00
https://doi.org/10.1145/3540250.3549117

CCS CONCEPTS

• **Software and its engineering → Open source model**; **Programming teams**; • **Human-centered computing → Empirical studies in collaborative and social computing**.

KEYWORDS

Open source ecosystem, software development, corporate participation, company domination, survival analysis

ACM Reference Format:
Yuxia Zhang, Klaas-Jan Stol, Hui Liu, and Minghui Zhou. 2022. Corporate Dominance in Open Source Ecosystems: A Case Study of OpenStack. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '22), November 14–18, 2022, Singapore, Singapore.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3540250.3549117

1 INTRODUCTION

Open Source Software (OSS) has become the bedrock of information technology and even our society [1]; it has been cited as a multi-billion dollar business [72]. Over 80% of organizations rely on OSS [30]. Software organizations are increasingly aware that they must invest their resources into value-adding parts of their overall software stack, rather than commodity and infrastructure components, such as databases and other off-the-shelf functionality [21, 68]. Therefore, a growing number of companies participate in OSS communities and ecosystems, and some are becoming major, and sometimes primary, contributors in OSS communities [26, 58, 78]. In contrast to volunteer contributors who may have a variety of motivations, companies are always driven by business goals [24, 42], even when they are making contributions to OSS [78, 84]. Prior literature suggested a number of theories on how companies leverage OSS [15, 16, 35, 69, 78].

As highlighted in the field of economics, the tendency toward capturing a dominant share of a market has always been an incentive to companies who wish to win market leadership and thus capture higher profits [17, 39, 65]. Similarly, to pursue business objectives, companies involved in OSS may seek to control the

development direction of OSS projects by having their in-house developers work on these projects and make contributions to achieve their goals. Such intense involvement from companies is likely to change the nature and community of an OSS project, because such domination may crowd out volunteers and other contributing companies [84]. Features of the open source development model that have traditionally been considered strengths are a diversity of contributors, leveraging 'open innovation,' and an open development model where anybody can contribute [48, 59, 83]. These strengths are threatened when a single organization can exert exclusive control over the future evolution of an OSS project, which in turn can jeopardize the future sustainability of OSS projects and ecosystems. Indeed, some studies suggest very high rates of project failure or abandonment [8, 57, 72], and so it is important to better understand the relationship between corporate dominance and the sustainability of OSS projects.

A first stream of related research has studied corporate participation in OSS. These studies have mainly focused on understanding why companies participate in OSS projects [71], how companies make profits through OSS products or services [27], and how companies contribute to and influence OSS projects [77–79, 84]. These studies assume that contributing companies collaborate with others (both companies and volunteers); very few studies have explored the impact of control, or indeed, domination of companies on OSS ecosystems. Prior research provides evidence of tensions in OSS communities due to conflicting interests, for example, and such tensions may result in open source developers (whether employed or not) leaving, or companies abandoning a project [54, 75]. Further, many prior studies have focused on individual projects, but not on OSS ecosystems, which are larger-scale ensembles of repositories, involving many different companies.

A second stream of research has investigated the sustainability of OSS projects [3, 46, 53, 67], while the effects of corporate participation on the duration of OSS projects are rarely considered (cf. [67]). None of these studies have focused on corporate domination in OSS ecosystems. Understanding the prevalence of company domination in OSS ecosystems and its impact on the sustainability of projects is of prime importance because better insights into domination dynamics can help to take appropriate action. For example, OSS projects and ecosystems can shape community governance guidelines and policies to monitor and manage corporate involvement, and take action when corporate domination jeopardizes the sustainability of these projects. Hence, the goal of this study is to shed light on corporate dominance in open source ecosystems. Given the scarcity of research on this topic, we do this through a case study of OpenStack, an OSS ecosystem that incorporates thousands of repositories involving hundreds of companies to contribute.

To address this goal, we conducted a mixed-methods study following a sequential explanatory strategy [12], addressing three specific research questions. First, as there is little empirical evidence that goes beyond the anecdotal, our first question is:

RQ1: How prevalent is corporate dominance in OpenStack?

We define a dominant company in an OSS project as one that makes the majority of contributions and thus has a strong influence on the development roadmap of the project. We draw on Market Dominance theory from the field of economics to define a measure for corporate dominance in OSS. We find that domination commonly exists in the development of most (more than 70%) repositories.

In order to develop a better understanding of the characteristics of corporate dominance, we pose the following question:

RQ2: What are patterns of corporate dominance in OpenStack?

We select a random sample of 120 repositories and manually analyze the widely available online records about the dominant companies' participation in these repositories of OpenStack. We identify five distinct domination patterns: early incubation, full-time hosting, growing domination, occasional domination, and last remaining. Each pattern has unique characteristics, and multiple patterns may occur in the evolution of an OSS repository.

Finally, we seek to understand the relationship between corporate dominance and survival of repositories within OpenStack:

RQ3: Does corporate dominance correlate with the survival of OSS repositories?

Using survival analysis, we find statistically significant evidence that corporate dominance correlates negatively with the survival probability of OSS repositories.

This paper makes three specific contributions to the literature on OSS ecosystems. First, we unveil the prevalence of company domination in the development of a large-scale OSS ecosystem. Second, we categorize a set of domination patterns, which help to understand the nature of the development of a complex OSS ecosystem. Third, we show a relationship exists between corporate dominance and the sustainability of OSS projects, and highlight a number of implications for OSS community governance.

The remainder of this paper is structured as follows. We review related work in Sec. 2 and introduce our method in Sec. 3. Sec. 4 presents the results of this study. We discuss the implications and limitations of our study in Sec. 5. Sec. 6 concludes with a brief summary of the key findings.

2 RELATED WORK

Recent years have seen considerable research on so-called hybrid OSS projects involving corporate contributors [26, 44, 78, 84]. Early research on corporate participation in open source focused on company motivation [6, 7, 13, 33, 51]. For example, Bonaccorsi and Rossi [6] found that companies participate in OSS because it allows smaller companies to innovate and affords them "many eyes" to assist them in software development, whereas idealism is of least importance to companies. Similarly, Henkel's case study of Linux [33] emphasized the importance of receiving outside technical support as a motivator for companies to participate in OSS. There is a general consensus in the literature that companies focus on economic and technological reasons for engaging with open source communities, rather than social motivations such as developer reputation and learning benefits. Based on the understanding of companies' motivation towards OSS, several business strategies were proposed [15, 31, 49].

To understand the nature of OSS development with intensive corporate participation, researchers have conducted a series of case studies on OSS-Commercial hybrid projects. For example, Wagstrom et al. [43] conducted an in-depth analysis of companies' practice in Gnome and Eclipse and identified two types of corporate

participation. Zhou et al. [84] studied three projects producing the same software and identified three corporate participation models, and they found that full control mechanisms and high intensity of corporate participation are associated with a decrease of external inflow and with improved retention. Valiev et al. [67] found that the involvement of companies has a significant effect on the sustainability of projects in PyPI ecosystem.

We conducted a series of studies [77–79] around company participation in OpenStack, the same OSS ecosystem that we selected in this study. We first investigate whether or not one company dominates a project and the impact of such domination based on only four repositories in OpenStack [77]. We defined dominant companies as those that submit the most commits to the projects together with the companies who makes more than 80% of the commits contributed by the largest companies. We found that one company often leads a project and this commercial domination is negatively associated with participation of other companies and contributors, while it is positively associated with the productivity of contributors and the quality of issue reports. This definition of domination enables the existence of dominant companies once the project has been contributed by companies, no matter how many contributions companies have made, which is different from the domination investigated in this paper.

In follow-up work, we empirically identified eight contribution models from the 124 involved companies based on their commercial objectives and examined their contribution intensity, extent, and focus. This study also found a significantly positive association between the number of volunteers and the diversity of contribution models [78]. In another study, we identified three collaboration patterns of companies: intentional collaborations (including supply and consumption, distribution-oriented ally, and service delegation), passive collaborations, and isolated fashion. We also found a positive association between a company's position in the collaboration network and its productivity [79]. In a more recent study, we found that conflicts arising from diverging roadmaps of companies and OSS projects is one reason that companies were withdrawing as a contributing organization from OpenStack [75].

Despite considerable research on corporate participation in OSS, corporate domination in open source has received little attention. This paper bridges this gap and complements the literature by conducting the first study of investigating the prevalence, patterns, and impact of company domination in OSS.

3 STUDY DESIGN

We applied a mixed-method approach [12] that combines a quantitative analysis of the commit history data with a qualitative examination of online records. This section describes the dataset (Sec. 3.1); the methods used for identifying company domination (Sec. 3.2) and qualitatively characterizing corporate dominance to extract patterns (Sec. 3.3); and to quantitatively measure the correlation of company domination and the sustainability of OSS repositories (Sec. 3.4). A replication package is available [76].

3.1 Dataset

OpenStack is an open source ecosystem of cloud computing infrastructure, including a set of software components that provide common services and allow users to plug and play components depending on their own needs [18]. We select the OpenStack ecosystem for this study for several reasons. First, having been founded in July 2010, it has a very extensive development history, which ensures availability of sufficient commit meta-data that allows a meaningful analysis. Second, hundreds of companies have been participating in the development of more than one thousand repositories within the OpenStack ecosystem; this provides a wealth of data for studying domination patterns, and offers ample data for a detailed survival analysis. Third, OpenStack has been subject of several prior scientific investigations, which not only allows us to develop a deeper understanding of this complex ecosystem with extensive corporate participation based on rigorous prior studies, but also to contribute detailed insights to this ecosystem which has seen very considerable uptake in the software industry.

We use the dataset provided by our previous studies. [78, 79], which is a processed dataset of OpenStack that covers its development history from July 2010 to January 2019. The dataset was carefully preprocessed, including removing commits submitted by bot accounts, merging multiple identities of developers, and identifying affiliations of developers and commits. We also validated the accuracy of the affiliation of developers and commits by conducting a survey with the corresponding developers directly and archived a high accuracy (93%) [78], which gives us confidence in the correctness of this dataset. This dataset involves 490 companies contributing to 1,216 repositories of OpenStack during the evolution of 18 complete versions, resulting in 338,035 commits. Each record in the dataset represents a commit, capturing its creation time, the name of the repository it belongs to, and the author's name, email, and affiliation when contributing this commit.

3.2 Measuring Company Domination

We build on Market Dominance theory, which is a classic theory from economics [29]. This theory is concerned with companies, brands, products, or services that control a large portion of the power in a particular market, relative to competing offerings [2, 29]. A dominant company in a specific market has a superior market position in relation to its competitors [29, 32, 39]. The assessment of dominance in a market is usually characterized by market share [32]. One of the most commonly used measures, proposed by Melnik et al. [39], is computed as follows:

$$s^D = \frac{1}{2}\left[1 - \gamma\left(s_1^2 - s_2^2\right)\right] \tag{1}$$

In this formula, s_1 and s_2 represent the market share of the top two companies in a specific market ($s_1 > s_2$). The parameter γ captures the industry-specific and time-dependent significance of potential competition. $\gamma = 1$ is commonly used to play a benchmark measure [32, 39]. s^D denotes the threshold market share beyond which Company 1 (i.e., the top one company) will be considered as having the dominant position in a market. This dominance measure mainly focuses on the market shares of the two largest companies in the market. The rationale of this measure is that, whether or not a company can be considered as dominant in a market, depends on the size of their closest competitor. This measure is easy to apply, which is why it can be used widely [32].

The contributions a developer has made to an open source repository can strengthen their reputation and influence in the community [41, 56, 81]. The same principle applies in OSS repositories that have intense corporate participation [78, 84]. Companies pursue their commercial objectives by making contributions to OSS repositories that align with their business objectives [35, 78, 79], which may result in a situation where they dominate a repository. Inspired by Melnik et al's [39] market dominance measure, we propose the calculation of company dominance by replacing market share with code authorship. It has been widely used in many studies for identifying developers who are responsible for maintaining modules using the authorship of code changes [5, 41, 66]. We assume that any version of an OSS repository is composed of a finite set of code changes that are developed by a finite set of companies and volunteers who submit their code modifications by sending contributions (i.e., commits) to a shared code repository. Therefore, for a company c, its Code Authorship (CA) of an OSS repository r in a specific version v is computed as follows:

$$CA(c, r, v) = \frac{\#commits_{c,r,v}}{\#sum\_commits_{r,v}} \quad (2)$$

where the numerator $\#commits_{c,r,v}$ represents the number of commits contributed by company c to repository r in the development of version v, and the denominator $\#sum\_commits_{r,v}$ represents the number of total commits of the repository r that are contained in version v.

After determining the code authorship of each company, we assess company dominance in an open source repository by calculating the threshold of dominance (cf. Melnik et al. [39]) in terms of code authorship as:

$$CA^D(r, v) = \frac{1}{2}\left[1 - \left(CA^2_{1,r,v} - CA^2_{2,r,v}\right)\right] \quad (3)$$

where $CA^2_{1,r,v}$ represents the code authorship of the top-contributing company to repository r in version v, and $CA^2_{2,r,v}$ represent the second largest company's code authorship, which poses the most significant obstacle to the dominance of Company 1.

We then compare $CA^2_{1,r,v}$ with $CA^D(r, v)$ to determine the existence of company dominance. More specifically, if $CA^2_{1,r,v}$ is greater than $CA^D(r, v)$, then we deem Company 1 to have a dominant position in repository r in the development of version v. Else, if $CA^2_{1,r,v} < CA^D(r, v)$, then no company dominates this repository. We group all individual, unaffiliated developers into a single 'organization' when calculating CA—that is, this 'organization' represents the "rest of the community," consisting of unrelated individuals. When this organization of unaffiliated individuals dominates a project as measured by the above formulas, then no single organization dominates this project, because it is the individual contributors that are the primary developers, similar to the 'traditional' notion of a volunteer-driven OSS community [40, 82].

To validate this measurement of company domination, we explored the perspective of developers in the OpenStack ecosystem using a questionnaire. We presented the percentage of commits that a repository had received from the top-contributing company and asked developers: "*Do you think <company name> has played a dominant role in the development of <repository name> in <version number>?*"; respondents were asked to select one of two options: 'Yes' (representing the company did dominate the repository) and 'No.'

We randomly selected 345 experienced developers who made contributions to the repositories that were dominated by companies in OpenStack (3,349 in total), with a confidence level of 95% and a margin of error of 5%. We then sent them the questionnaire (see the appendix [76]) to solicit their opinions. 146 emails could not be delivered. We received 35 valid responses, resulting in a response rate of 17.6% (= $\frac{35}{345-146}$). Of those 35, 29 (83%) developers agreed with our measurement, offering a sense of face validity of this measure.

3.3 Extracting Domination Patterns

Company domination may occur when a company has made a majority of commits to a repository within a specific timespan. After identifying corporate dominance (see Sec. 3.2 for details), we sought to extract patterns of company dominance from two perspectives: the functionality of the company-dominated repositories, and the dominant companies' objectives towards the repositories.

3.3.1 Selecting Repositories based on their Categories. Prior studies [78, 79] observed that companies' contributions to OSS vary by component functionality. For example, the plugin repositories in OpenStack are mainly contributed and maintained by the companies that created these repositories [79]. Thus, there is reason to believe that the patterns of company dominance relate to the types of functionality.

The OpenStack community categorizes its repositories in release into three types: components, client tools, and deployment tools [19]. However, not all OpenStack repositories are included in a distributed release [64], such as the repositories for building the OpenStack communities. Zhang et al. [78, 79] proposed more than ten categories of OpenStack repositories based on their functionality, but most of these are sub-categories of "component", such as 'computing,' 'storage,' and 'networking.' We ignored these sub-categories of 'component' because these are too limited to OpenStack and may reduce the generalizability of our results. As a result, we retained six general repository categories: components, client tools, deployment tools, infrastructure tools, community build, and plugin/driver, and revisited the categories of the 1,216 repositories, based on the definition of the six general categories proposed in [19] and [78, 79] together. We first searched the official documents (i.e., created and maintained by the OpenStack Foundation) and the README file of each repository. We then determined the category of each repository by deductive thematic analysis [47], using themes from prior research as seed categories. Specifically, two authors independently cross-checked the set of categories and descriptions that were derived from the homepage of OpenStack [19] and Zhang et al. [78, 79] as a starting point, and applied them to the 1,216 repositories. We obtained high consistency (Cohen's kappa coefficient = 0.86) between the two coders. Then, the two authors discussed to address any disagreements. Table 1 shows the distribution of the 1,216 repositories in the six categories.

As the qualitative analysis is an extremely time-consuming and intense effort, we randomly selected 20 repositories per category (column 1 in Table 1) for analysis, with a total of 120 repositories.

3.3.2 Extracting Domination Patterns. We used data collection and analysis methods previously used by Zhang et al. [78, 79], i.e., collecting related data by search engine and conducting thematic analysis, to extract domination patterns. We first measured whether a repository had been dominated in each version (i.e., approximately six months in OpenStack) based on the domination measurement described in Section 3.2. Then, for each dominated repository, we conducted web searches using its name plus the name of the dominant company and the term 'OpenStack' using the Google search engine, and visited the top ten links (if available) to understand company dominance, and collected related online records; 269 online records were collected (links are included in the online appendix [76]). We used thematic analysis [14] to characterize patterns of company dominance following a series of steps. (1) We first read and analyzed all online records to understand why a dominant company participated in OpenStack and dominated a repository, and marked key phrases of interest. (2) We revisited the online records and relevant phrases carefully to generate initial codes and organized them in a systematic way. (3) After initial codes were identified and assigned, we aggregated related codes into a cluster, and identified a theme representing that cluster. Having identified and labeled all clusters helped in identifying any emergent patterns that characterized the occurrence of company dominance. (4) We then revisited the initial set of themes to identify opportunities to merge similar themes. (5) Finally, we defined the final set of themes. To reduce potential researcher bias [52], the first two authors independently performed steps (1) to (4) described above. After this, a sequence of meetings was held to resolve conflicts and assign the final themes (Step 5). We reached theoretical saturation when analyzed a half of the targeted repositories, meaning roughly that subsequent data all fit within the domination patterns derived from the first 60 repositories. The results of our coding are available in the online appendix [76].

3.4 Survival Analysis

Survival analysis is a cluster of statistical approaches where the time of occurrence of an event of interest is modeled to understand survival probabilities [36]. It is widely used in a variety of fields such as epidemiology, biology, and demographic studies [70]. In the software engineering literature, survival analysis has also been used to study software project duration [53, 67] and developer turnover [37, 55]. The most common tool for modeling survival analysis data is the Cox proportional-hazards regression model [22]. It can be used to study the dependency of survival time on both continuous variables and categorical variables [22]; it allows us to examine any given factor's influence on the rate of the event happening at a particular point in time [22]. Thus, in this study we adopt the Cox proportional-hazards regression model to investigate the impact of company dominance on the survival of OSS repositories.

3.4.1 Basics of the Cox proportional hazards model. In this study we use the Cox proportional-hazards model to evaluate the effect of company dominance, and several other control variables, on the survival status of an OSS repository. Our event of interest is a repository becoming 'dormant'; that is, when a repository does not receive any contributions for a significant period of time, the repository is treated as being dormant, in line with prior work [8, 67]. This may happen when it is no longer maintained, abandoned, but also when it is considered 'feature complete.' Following prior studies [34], we consider the event of interest, i.e., a repository becoming 'dormant,' that occurred when a repository did not receive any commit within more than six months (i.e., a version in OpenStack) after its most recent commit, specifically.

Since our dataset contains 18 versions of OpenStack's development history, we cannot observe repositories going dormant (the event of interest in our survival analysis) in version 18, because that version is the latest in the dataset. In survival analysis, these cases are called *right censoring* and can be well handled by the Cox model [22]. The survival probability (denoted as S(t)) refers to the *probability* that a 'live' entity survives from the time origin (i.e. the creation of an OSS repository) to a specified future time t [36]. In the Cox model, the *probability* expressed by the hazard function (denoted by h(t)), which is the probability that a repository under observation at time t undergoes an event (i.e., becomes dormant). Simply stated, the hazard function h(t) can be treated as the risk of becoming dormant at time t. The hazard function h(t) is defined as follows:

$$h(t) = h_0(t) \times exp(b_1x_1 + b_2x_2 + ... + b_nx_n)$$

where: t represents the survival versions of a repository; the coefficients $(b_1, b_2, ..., b_n)$ measure the effects of factors $(x_1, x_2, ..., x_n$, resp.); h_0 is the baseline hazard, corresponding to the value of the hazard if all factors x_i are equal to zero. Factors with a statistically significant coefficient are relevant to the survival probability.

The quantities $exp(b_i)$ are called hazard ratios (HR), which present the effect of factors on the survival of a repository, especially: 1) HR = 1: no effect; 2) HR < 1: reduced hazard; and 3) HR > 1: increase in hazard.

3.4.2 Control Variables. Software project 'survival' has been widely investigated [3, 46, 53, 67]. Prior studies identified several factors that impact survival probabilities of projects or repositories. To reduce potential bias caused by only considering whether a repository is dominated by a company, we also include other related factors in our model as control variables. The related definitions and proposed metrics are as follows:

- `Num_Dev`: the number of developers who have contributed at least one commit to the observed repository r in the development timespan of version v;
- `Num_Commits`: the number of commits that were received by repository r in the development of version v;
- `Num_Corp`: the number of corporations that participated in the development of repository r in version v. Considering the characteristics of extensive corporate participation in OpenStack, we also include this factor in our model;
- `Repo_Category`: a category factor referring the function type of repository r. Table 1 shows the six categories of the 1,216 repositories in OpenStack.

The first three factors, i.e., `Num_Dev`, `Num_Commits`, and `Num_Corp`, are used to measure the active degree of an OSS repository. The last factor (`Repo_Category`) is mainly for examining the difference of the survival probability of repositories belonging to different categories (see Sec. 3.3 for details of repository categories). We log-transformed the first three skewed variables to stabilize the variance

Table 1: Six categories of the 1,216 repositories in OpenStack

Category	Description	# Repositories	Example
Components	To set up OpenStack's cloud computing services and the management of these services.	400	Nova
Client Tools	Client-side tools and libraries for interacting with OpenStack APIs.	126	Cl-openstack-client
Deployment Tools	Tools and recipes for deploying and maintaining the life cycle of OpenStack deployment [78].	547	Fuel-main
Infrastructure Tools	To provide infrastructure tools for the development of OpenStack [78].	44	Gertty
Community Build	To record, manage, and optimize the participation activities and code architecture for sustainable development.	73	Stackalytics
Plugin/Driver	To facilitate integration of OpenStack and other hardware or software infrastructure [79].	26	Compute-hyperv

and improve model fit [23]. Because of the time-varying factors we identified (such as `Num_Dev`), we used the time-dependent Cox model [22, 80] in this study. We used the `survival` package in R [63] to fit the model. Sec. 4.3 presents the results.

4 RESULTS

In this section, we present the quantitative and qualitative results of the prevalence, patterns, and possible impact of company domination, combining with the corresponding discussion.

4.1 RQ1: How Prevalent is Corporate Domination in OpenStack?

The first research question is concerned with establishing company dominance as a phenomenon. To answer this question, we first calculated code authorship of all contributing companies, following the formula defined in Sec. 3.2. Specifically, the parameter v for each of the 1,216 repositories is assigned with their overall time span, i.e., from their creation to 30 August 2018 (the release date of the most recent version in our dataset). Figure 1 shows the histogram of the distribution of the largest code authorship in all OpenStack repositories. The vertical axis represents the number of repositories, and the horizontal axis represents the value intervals of the 1,216 repositories' largest code authorship. The figure shows that most repositories' largest code authorship (i.e. max code authorship) is greater than 0.50. Specifically, there are 888 repositories (accounting for 73%) that are dominated by a single company with a median code authorship of 70.6%. This indicates that company domination is a very common phenomenon when taking the entire development history of all the repositories in OpenStack. Among the 888 dominated repositories, 163 repositories are contributed by only one company. This observation is consistent with recent literature that some companies work in an isolated fashion in an OSS ecosystem [79].

Further, we studied company domination at the level of versions, which represents a finer granularity. For each version, we determined whether any repository was dominated by calculating the threshold code authorship CA^D (using the formula defined in Sec. 3.2) and comparing this with code authorship of the company who contributed the most commits to the repository in this version. Different from the granularity of the *entire* history of one repository, we only considered its commits contributed in each version. As shown in Fig. 2, the blue bars represent the number of repositories dominated by companies, and the green bars represent the total number of repositories per version. The left y-axis corresponds to the blue/green bars and represents the number of repositories, and the right y-axis corresponds to the black plot and represents the percentage of dominant repositories to the total repositories in each version. For this analysis, we discarded the first five versions due to the instability in the initial phase of OpenStack [77]. The horizontal axis represents the versions from six to 18. As the number of repositories increases per version, the number of dominated projects also increases. The percentages of dominated repositories per version range from ca. 61.7% to ca. 78.5%, with a median of 71.3%. It indicates that company domination consistently existed during the development OpenStack.

Finally, we calculated the distribution of companies that dominate different numbers of repositories. The results are shown in Fig. 3, where the horizontal axis represents the 18 versions included

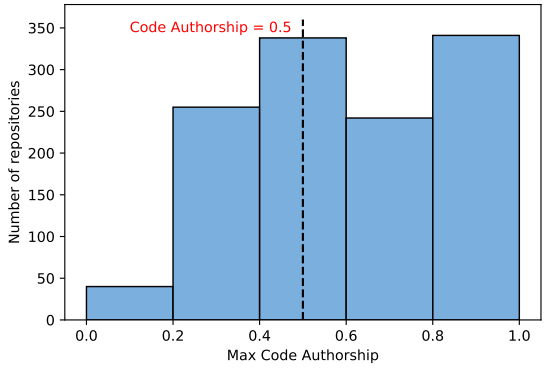

Figure 1: Distribution of max code authorship.

in our analysis; the green bar ('= 0') in the legend represents the portion of companies that did not dominate any repository; blue ('= 1') represents the portion of companies that only dominated one repository; and yellow ('> 1') represents the portion of companies that dominated more than one repository. We can see that the dominant companies account for a relatively small proportion, i.e., approximately 25.2% of companies have dominated one or more repositories in each version on average, almost meeting the Pareto distribution [25]. This proportion has been slowly increasing with version evolution despite the first five unstable versions [79]: ranging from 15.8% to 36.7%. Further, the proportions of companies that dominated one repository and the proportions of companies that dominated more than one repository are similar (median: 11.2% and 13.7%). This indicates that nearly half (56.0%) of the dominant companies dominate more than one repository.

Summary for RQ1: Company domination is common in the development of OpenStack's repositories, both during its entire development history, and within specific versions of OpenStack. Over 70% of repositories are dominated by a quarter of companies, a half of which dominated more than one repository.

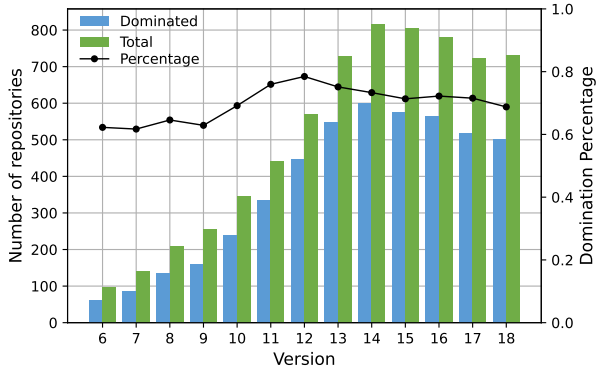

Figure 2: Percentage of dominated repositories per version.

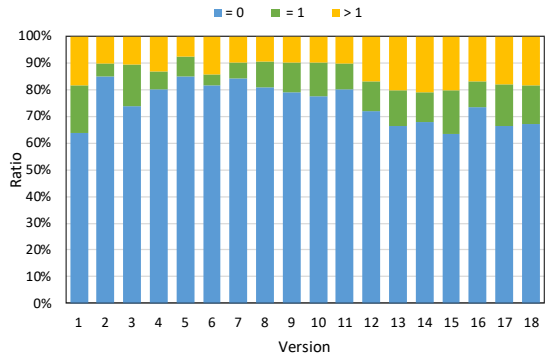

Figure 3: Distribution of companies dominating 0, 1, or more than one repositories.

4.2 RQ2: What are Patterns of Corporate Dominance in OpenStack?

The second research question is concerned with deepening our understanding of patterns of corporate dominance in OSS projects. Using the method described in Sec. 3.3, we manually analyzed the development history of 120 randomly selected repositories. The 120 repositories cover the six categories of OpenStack, specifically, 20 for each category. The development history of the 120 repositories varies from two versions to 18 versions, with a median of eight versions. For each version of each repository, we explored whether company domination occurred and why, using the approach described in Sec. 3.3.[1] We identified five types of company domination: Early incubation, Full-time hosting, Growing domination, Occasional domination, and Last remaining. The numbers of observations of these five patterns in different versions of the 120 sampled repositories are shown in Fig. 4. As a repository evolves, we may be able to observe different domination patterns over time; we then speak of a 'pattern sequence.' The two patterns 'Full-time hosting' and 'Early incubation' account for the majority of cases of company domination, and 'Growing domination' has the minimum proportion. Next we describe each of these patterns.

4.2.1 Early Incubation. The Early Incubation pattern refers to a company's domination of a repository because that company created the repository, and it was the dominating contributor. When creating a new repository in OpenStack, the company tends to make many contributions to that repository in the initial incubation process. New repositories are usually of limited interest to other companies [79]. For example, to manage cloud applications Red Hat created the repository `Heat` [11]; Heat is an OpenStack service that provides a framework to organize the configuration and deployment of cloud applications [19]. Red Hat contributed the most commits to Heat in its first four versions, and the corresponding code authorship CA ranged from 100% to 59%. Red Hat reduced its contributions (from around 400 commits to 100 commits in each subsequent version); at the same time other companies were slowly ramping up their involvement in further versions of Heat. In the 120 analyzed repositories, we observed *Early incubation* in

[1] It is feasible to determine the domination patterns at the version level because the number of dominant companies in most analyzed repositories is no more than two.

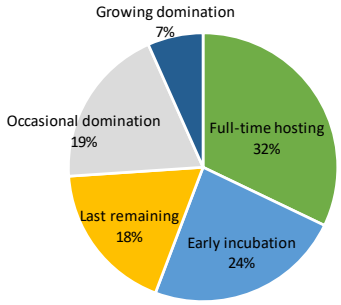

Figure 4: Distribution of the five domination patterns.

39 repositories across all six repository categories. By definition, Early Incubation always appears first in a pattern sequence.

4.2.2 Full-time Hosting. The Full-time Hosting pattern is similar to Early Incubation, but unlike Early Incubation, companies following this pattern remain the dominant contributor throughout the complete development history of the repository (or until they quit OpenStack). We distinguish between Full-time Hosting and Early Incubation by looking at the company's duration of involvement; we consider a company dominating a repository for up to two years (four 6-month versions) as Early Incubation. Longer-term domination (i.e., five or more versions) is considered as Full-time Hosting. A recent study [72] has found that the average time of incubation an OSS project in the Apache Foundation is 23.9 months. Therefore, we deem the choice of two years is, in our view, a reasonable balance between characterizing an initial, short-term involvement to incubate a repository to a stage where it could survive by itself, and a longer-term involvement that is more strategic. In other words, there is a phenomenon in the evolution of the OpenStack ecosystem that a company initiates a repository and continues to play a dominant role in a continued series of subsequent versions. Among the 120 repositories, we found that 53 repositories (44%) have experienced this form of domination at some point, and indeed this is the most common pattern we observed. For these repositories, a high survival risk exists once the dominant companies reduce their involvement from OpenStack, or abandon these repositories altogether. For example, Rackspace, an established IT web hosting company and one of the initiators of OpenStack [78], launched the repository `quark` in version 7 to achieve a scalable networking service, and continued to be the dominant contributing company in the seven versions that followed with an average CA of 88%. However, `quark` has been inactive since Rackspace stopped contributing in version 15. When a dominant contributing company stops contributing, it may happen that a different company steps in to become the next dominating contributor. In that case, a pattern sequence emerges; we discuss this in more detail later.

4.2.3 Growing Domination. This type refers to a pattern of corporate dominance whereby a company is increasing its contributions to a repository. Different from Early Incubation and Full-time Hosting, this kind of company domination tends to occur in the middle or later stages of a repository's evolution. Therefore, this pattern is always a part of a pattern sequence, and does not appear as a stand-alone pattern in Table 2. In the 120 analyzed repositories, we identified this pattern in 11 repositories (9%). For example, Rackspace started the development of Swift, which is a distributed object storage software providing data store and retrieval service [19]. This repository was added to OpenStack in August 2009 and Rackspace was the dominant contributor in the first five versions. At that point, SwiftStack, an object storage software company [62], gradually replaced Rackspace as the leading contributing company, and began to dominate Swift from version 16 onwards.

4.2.4 Occasional Domination. Occasional Domination usually occurs 'by chance' in a couple of versions during a repository's evolution, and the dominant companies always play a key role in the development of OpenStack; that is, these companies, such as Red Hat, rank in the top contributors by commit count to OpenStack. We observed this pattern in 32 repositories (27%): 15 of them are Client Tools and 12 of them belong to the Community Build category. Client Tools and Community-Build repositories represent functionality that is more general (i.e., support/utilities) than other categories and unlikely to be dominated by any business-oriented company. Therefore, the Occasional Domination pattern is more likely to occur in these Client Tools and Community-build repositories compared to other domination patterns, due to the 'incidental' nature of this pattern. For example, in version 18, Red Hat contributed eight commits (CA=50%) to `oslo.db`, a database connectivity library handling different database back-ends, and various other utilities [20].

4.2.5 Last Remaining. When a repository becomes inactive (i.e., receiving less than ten commits each version), company domination may happen because of a few commits contributed by the last remaining company; hence, we name this pattern of domination "Last Remaining." By definition, this pattern *always* occurs at the end of the development history of repositories, after a sequence of one or more other patterns—hence the label 'last.' For example, Mirantis only contributed four commits (CA=50%) to `docker-machine-openstack` [61] in version 16. We observed this pattern appeared in 30 repositories (25%), suggesting that around a quarter of cases of corporate dominance may be because these repositories are no longer actively maintained, rather than due to active company domination. The reasons why some repositories are no longer actively maintained are complex, and an exploration for this is beyond the scope of this paper, but is an avenue for future research.

4.2.6 Sequences of Patterns. Our analysis also revealed that domination patterns evolved over the history of repositories; we refer to this as a 'pattern sequence,' as briefly introduced above. A pattern sequence happens when one domination pattern is replaced by a different one over time as contributing companies change their involvement. Table 2 shows all the patterns (and their sequences) of domination in the development history of the 120 analyzed repositories. The table shows, for example, that within the Components repository category, the Early Incubation pattern was observed in 12 repositories (7+4+1): in 7 cases, Early Incubation was the only pattern observed throughout the lifetime of a repository, but in 4 cases this pattern was replaced by the Growing Domination pattern (E–G), and in one case it was replaced by the Last Remaining pattern (E–L). If the same domination pattern appears in continuous versions, we only counted it once, rather than counting that pattern for each of the consecutive versions. For example, the combination E–G indicates that Early incubation was observed for one or more versions during the initial development of a repository, and was then replaced by Growing Domination; this means that the repository was later dominated by a different company that increased its contributions to that repository.

Table 2 shows that the most common pattern of company dominance varies by repository category; the most frequent pattern per category is encircled. Component repository domination often occurred in the initial incubation stage (any pattern of sequence with 'E': 60%, 12=7+4+1 out of 20), then disappeared because of other companies' increased contributions (58%: 7 out of 12 cases), or was replaced by a growing dominant player (E–G, 33%: 4 out of 12

Table 2: Frequency of observed domination patterns and pattern sequences, organized by repository category
E: Early incubation; F: Full-time Hosting; G: Growing Domination; O: Occasional Domination; L: Last Remaining

Repository Category (20 repositories per category)	E pattern and sequences					F pattern and sequences				O pattern and sequence	
	E	E – G	E – L	E - O	E-G-L	F	F – L	F - O	L	O	O - L
Components	7	4	1			5	1			2	
Client Tools			1	3		2	2	1		10	1
Deployment Tools	1	2	1		1	14	1				
Infrastructure Tools	3	1	2			5	7			2	
Community Build	3		3				1	1	1	9	2
Plugin/Driver	2	3		1		9	4		1		

cases), or becoming inactive with the 'Last remaining' dominated (only 1 out of 12 cases).

For Client Tools, the most common domination is 'Occasional' (50%, n=10). We observe the same pattern in the Community Building repositories in OpenStack (45%, n=9). The reason might be that these kinds of repositories, aiming to improve the soft power of an OSS ecosystem, i.e., community building or improving code maintainability, may have limited long-term attraction to companies that pursue business goals [78].

For the remaining three categories, i.e., deployment tools, infrastructure tools, and plugin/driver, the most common patterns are all related to Full-time hosting (75%, 60%, and 65% respectively). This means that most of these repositories are mainly dominated by the corresponding sponsoring company. This also indicates that the repositories in these three categories have high 'survival risks' if the founding companies of these repositories reduce their contributions, or withdraw altogether. While our results do not suggest a causal link, we can observe that this risk is realistic in infrastructure tools, where the most common domination is F–L, meaning that most of these repositories are mainly dominated by the corresponding sponsoring company in their initial and intermediate stages, and then became inactive when the sponsoring company withdrew. Being dominated by one company for a long time may be a reason of real concern in an OSS ecosystem of which the wider ecosystem should be aware. OSS ecosystems thrive best when there is diversity in contributing actors, and reducing this as well as withdrawal of key contributors pose a major risk to survival.

Summary for RQ2: We identified five patterns of corporate domination in the OpenStack ecosystem. A repository may see multiple domination patterns over its lifetime, as repository founders and contributors may change their involvement. Some domination patterns correlate by the type of repositories: Early incubation is more common in Components; Occasional domination is more common in Community-build repositories and Client tools, and Full-time hosting is more common in the remaining three categories.

4.3 RQ3: Does Corporate Dominance Correlate with the Survival of OSS Repositories?

The third and last research question focuses on a key outcome of interest for open source communities, namely, is corporate domination linked to survival of open source repositories? To answer this question, we calculated the domination state (i.e., `Is_Dominated`) and values of other control variables described in Sec. 3.4 for each repository in each survived version. We obtained 6,808 observations, including 675 events (i.e., repositories that went dormant). As laid out in Sec. 3.4, we used the Cox proportional-hazards regression model. Before fitting the Cox model, we calculated the variance inflation factors (VIF) to detect multicollinearity problem [38]. Due to moderately high collinearity (VIF > 3) [9], we removed `Num_Dev` and `Num_Corp` from the model so as to secure the reliability and stability of the regression coefficients. Therefore, the final regression equation is:

$$\text{Surv}(t_{start}, t_{stop}, \text{Status}) \sim \textit{Is\_Dominated} + \textit{Log(Num\_Commits)} + \textit{Repo\_Category}$$

where t_{start} and t_{stop} in the response are used to set the time interval for observing each repository, and we treat each version as a time interval. `Status` in the response is the survival status (i.e., dormant or not) of a repository in the time range. `Is_Dominated`, `Repo_Category`, and `Num_Commits` are predictors. Table 3 presents the results of the fitted model.

The table shows that the p-values of three repository categories (i.e., Client Tools, Infrastructure Tools, and Plugin/Driver) are non-significant (p > .05), which means we found no evidence of the association between the three repository categories and the repository survival probability. Two repository categories, Components and Community Build, on the other hand, are significant, suggesting that repositories that provide services, or serve to build the OSS community, are less likely to become dormant over time, when comparing with deployment tools. The p-value for $Log(Num\_Commits)$ is less than 2e-16, with a hazard ratio $HR = exp(coef) = 0.30$, indicating a strong relationship between the number of commits and a repository's decreased risk of becoming dormant. This finding is in line with prior work [53, 67]

Table 3: Coefficients of the model
(n=6,808, #events=675)

	coef	exp(coef)	p-value
Is_Dominated	0.38	1.46	<0.001***
Log(Num_Commits)	−1.20	0.30	<0.001***
Repository Categories:			
Components	−0.21	0.81	0.023*
Community Build	−0.51	0.60	0.006**
Client Tools	−0.23	0.79	0.077
Infrastructure Tools	−0.19	0.83	0.354
Plugin/Driver	−0.22	0.80	0.381

The variable `Is_Dominated` is a Boolean variable indicating whether a repository is dominated by a company. After taking these existing factors into consideration, the factor `Is_Dominated` still shows a statistically significant impact on the survival of repositories. As Table 3 shows, the p-value for `Is_Dominated` is less than 0.001, with a hazard ratio exp(coef) = 1.46, indicating a strong relationship between a repository's state of being dominated and an increased risk of it becoming dormant. Specifically, with other factors constant, being dominated (`Is_Dominated` = 1) increases the hazard of a repository becoming dormant by a factor of exp(coef) = 1.46, or 46%, which is a significant effect. This may suggest that although having a strong supporter or advocate (i.e. the dominant company) may benefit the evolution of an OSS project [77, 84], the degree of contribution of such a supporter should be monitored, and perhaps appropriate governance mechanisms should be put in place to ensure future sustainability.

Finally, the p-value of the Cox model for all three overall tests (likelihood, Wald, and score) are significant, indicating that the model is significant. Further, the *Concordance* statistic, which is the most common measure of goodness-of-fit in survival models [22, 28], of the model is 0.85, indicating that the model explains the observed data well and has a good predictive ability.

Summary for RQ3: After considering existing factors (the number of commits and repository categories), we still find a statistically significant association between company domination and the survival probability of a repository. That is to say, being dominated by one company might be harmful to the long-term development of OSS repositories. More careful monitoring and management of corporate participation in OSS is needed.

5 DISCUSSION

5.1 Implications for Practice

In this study, we find that the development and maintenance of most repositories in OpenStack mainly relies on a single company. In some cases, these companies play a dominant role throughout these repositories' development. These findings suggest that although large OSS ecosystems are developed in an open and transparent form, the advantages of open source have not been fully utilized, especially user innovations that are brought by contributors from diverse backgrounds. Despite that, we also find a statistically significant association between company domination and the survival probability of OSS repositories, raising an alarm about intense corporate participation. Our work sheds some light on the internal contribution mechanism of a large and popular OSS ecosystem with hundreds of companies involved. This study offers a detailed analysis of corporate dominance, and seeks to increase awareness of the significant implications that may have for the sustainability of OSS ecosystems.

OSS communities can use dashboards to monitor the health of an ecosystem, regarding company mobilization. Our method of measuring company domination can play as one key metric for the dashboard. Moreover, the survival model of OSS repositories can help to identify risky situations, as this study links corporate dominance to a negative survival probability.

Our findings highlight potential concerns for intense corporate participation in OSS projects. Although pursuing business objectives is a primary concern for companies, the decline or abandonment of an OSS project should be of equal concern to organizations as this jeopardizes the survival probability of open source ecosystems, and may render earlier investments in such ecosystems worthless. For any company intending to join OSS, our study emphasizes the importance of keeping a balance between their own business goals and the open source project's roadmap and long-term sustainability. They can draw on our findings (e.g. the survival model) and establish metrics to monitor this balance. Furthermore, dominant companies that consider leaving a repository, could identify companies (or individuals) who may be willing to take over the maintenance role of a repository, preventing it from abandonment. As corporate participation in OSS ecosystems has become the industry norm [26, 67, 84], these results may help companies to become more aware of the responsibility that comes with OSS contributorship, and seek a balance as they define an OSS strategy.

5.2 Implications for Research

As the importance of open source continues to grow in the software industry, corporate participation in OSS is an inevitable trend [26, 78]. Different from volunteers, companies are driven by business goals. Companies may maximize their benefits by dominating the development of a project or only contributing to directly related projects. The resulting corporate dominance may lead to a reduced survival probability of OSS projects. Therefore, it is imperative that companies balance their business goals and commitment and investment in open source ecosystems in a responsible manner.

On the other hand, an existing study has found that corporate participation (measured by the share of commits from companies) in PyPI ecosystem has a negative effect on the sustainability of the repositories in PyPI ecosystem [67]. However, our previous study found that corporate participation (represented by the proportion of the commits contributed by the company with the most contribution) has a positive association with the productivity of contributors and the quality of issue reports [77]. In this study, we found company domination tends to decrease the survival probability of these repositories. These diverse findings indicate that more delicate metrics to measure the impact of corporate participation on OSS projects and the characteristics of different OSS ecosystems are important. Future research may advance measurement framework towards company participation in OSS. A more challenging task is

to develop a model that predicts what percentage of the commits contributed by the company will have a negative impact on the survival of an OSS project. Investigating other possible impacts of company domination on the development of OSS ecosystems is also an avenue for future work.

5.3 Threats to Validity

We follow common guidelines for empirical studies [52, 73] to discuss the validity threats of our study.

Construct Validity. We are interested in investigating prevalence, patterns, and impact of company domination in OSS ecosystems. We believe that these questions have a high potential to provide unique insights and values for practitioners and researchers. To achieve this goal, we sought to measure company domination first. There are a variety of ways to make contributions to OSS projects, such as conducting code reviews, reporting issues, participating in governance processes, and even securing funding. In this study, we chose commits as the primary form of contributions because of the following reasons: 1) the number of commits is widely used in measuring OSS contributions in both literature [50, 67, 78] and OSS communities [10]; 2) the code changes submitted as commits are closer to software development. While other forms of contributions than code have recently attracted some attention in the SE literature [4], domination in terms of non-code contributions to OSS ecosystems remains an area for future research. We chose to ignore commit size for three reasons, because OpenStack requires contributors to create commits with only one 'logical change' [45]. Further, commit size distributions of companies do not show statistical differences; a recent study [74] has found that the majority of companies primarily fix bugs and implementing new features in OpenStack, both essential activities in open source development.

By drawing on Market Dominance theory [32], we adopted a measure of company domination. To validate its use, we surveyed the perspectives of developers from OpenStack; 83% (29 out of 35) of developers agreed with the proposed measure. This high level of agreement lends face validity to this measure, and also suggests that the measurements are conducted in a reasonable way. Besides, when determining a domination type is Full-time Hosting Early Incubation, we took the incubation time (i.e., 23.9 months on average) of OSS projects in the Apache Foundation [72] as a reference and considered the dominant company's duration of involvement: up to two years as Early Incubation; longer time domination as Full-time Hosting. The incubation time in different OSS foundations might be various, which may slightly affect the distribution of the two domination patterns and the corresponding pattern sequences. Future work should investigate the specific incubation time of OpenStack and revisit the two domination patterns.

Internal Validity. From the internal perspective, we checked the assumptions for the Cox proportional hazards model. Similar to other regression models, the Cox model also faces threats to multicollinearity [38] and outliers [23]. We investigated the distribution of the variables to fit the Cox model for RQ3 to detect outliers and multicollinearity. We log-transformed variables (i.e., `Num_Commits`) with highly skewed distributions to reduce heteroscedasticity and calculate Variance Inflation Factors. As a result, we removed two factors (i.e., `Num_Dev` and `Num_Corp`) to avoid the impact of multicollinearity on our model. Our findings show an association between the response and predictors, but that association may not be causal. There may be some aspects that we did not measure but that cause both the response and the predictors to behave in the observed pattern.

Further, we did not consider the type of corporate domination in the Cox model. The reason for this is that no matter the *type* of corporate domination, domination always threatens the open source community's diversity and potential for user innovation, which in turn jeopardize the sustainability of OSS projects and ecosystems. The impact of different domination type on the survival of OSS projects may vary, however, and this is a fruitful direction for future work.

External Validity. There is a scarcity of research on corporate dominance in OSS ecosystems, which is why we adopted the case study method to conduct an in-depth investigation. We selected one very popular and large ecosystem, OpenStack, which enjoys corporate participation from hundreds of companies. However, the case study method does impose limits to the generalizability of our findings, as case study findings tend to be tied to specific contextual factors [60]. However, we do not seek to establish statistical generalizability, but rather to offer insights into this unexplored phenomenon which can help to develop an initial understanding, which can provide a foundation for future studies [73]. The domination patterns categorized from diverse companies and six project categories may have the potential to generalize to other open source ecosystems where companies involved and similar projects exist. However, investigating corporate dominance in other open source ecosystems is an avenue for future research.

6 CONCLUSION

As corporate participation in OSS ecosystems is growing, their influence on the future evolution of these ecosystems also becomes stronger, which in turn can have significant consequences for the sustainability of these ecosystems. In this paper, we present a detailed study of one popular OSS ecosystem, OpenStack, to (a) establish corporate dominance as a prevalent issue, (b) identify five patterns of corporate dominance, and (c) apply survival analysis on a large set of repositories, and find that corporate dominance of repositories negatively associates with their survival chance. This paper is among the first to provide a detailed investigation of corporate dominance in OSS ecosystems, which we hope will draw more attention to this phenomenon which may have significant impact on the future of OSS ecosystems. The findings of this study may help OSS communities to increase their awareness of this issue and the associated risks, and adjust their governance mechanisms so as to ensure their future sustainability.

ACKNOWLEDGMENT

This work was sponsored by the National Natural Science Foundation of China (62141209, 61825201, 62172037), and Science Foundation Ireland grants 15/SIRG/3293 and 13/RC/2094-P2. For the purpose of Open Access, the authors have applied a CC-BY public copyright license to any Author Accepted Manuscript version arising from this submission.

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] Susan Athey and Armin Schmutzler. 2001. Investment and Market Dominance. *The RAND Journal of Economics* 32, 1 (2001), 1–26. http://www.jstor.org/stable/2696395

[3] Guilherme Avelino, Eleni Constantinou, Marco Tulio Valente, and Alexander Serebrenik. 2019. On the abandonment and survival of open source projects: An empirical investigation. In *2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. IEEE, 1–12.

[4] Ann Barcomb, Andreas Kaufmann, Dirk Riehle, Klaas-Jan Stol, and Brian Fitzgerald. 2020. Uncovering the periphery: A qualitative survey of episodic volunteering in free/libre and open source software communities. *IEEE Transactions on Software Engineering* 46, 9 (2020), 962–980.

[5] Christian Bird, Nachiappan Nagappan, Brendan Murphy, Harald Gall, and Premkumar Devanbu. 2011. Don't Touch My Code! Examining the Effects of Ownership on Software Quality. In *Proceedings of the 19th ACM SIGSOFT Symposium and the 13th European Conference on Foundations of Software Engineering* (Szeged, Hungary) *(ESEC/FSE '11)*. Association for Computing Machinery, New York, NY, USA, 4–14. https://doi.org/10.1145/2025113.2025119

[6] Andrea Bonaccorsi and Cristina Rossi. 2006. Comparing motivations of individual programmers and firms to take part in the open source movement: From community to business. *Knowledge, Technology & Policy* 18, 4 (2006), 40–64.

[7] Capek, G P., Frank, P S., and Gerdt. 2005. A history of IBM's open-source involvement and strategy. *Ibm Systems Journal* 44, 2 (2005), 249–257.

[8] Jailton Coelho and Marco Tulio Valente. 2017. Why modern open source projects fail. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*. ACM, 186–196.

[9] Patricia Cohen, Stephen G West, and Leona S Aiken. 2014. *Applied multiple regression/correlation analysis for the behavioral sciences*. Psychology press.

[10] OpenStack Community. 2021. Homepage of Stackalytics. https://www.stackalytics.io/?metric=commits.

[11] Red Hat community. 2021. Chapter 2. Understanding Heat Templates. https://access.redhat.com/documentation/en-us/red_hat_openstack_platform/13/html/advanced_overcloud_customization/sect-understanding_heat_templates.

[12] John W Creswell and J David Creswell. 2017. *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage publications.

[13] Kevin Crowston, Kangning Wei, James Howison, and Andrea Wiggins. 2012. Free/Libre open-source software development. *Comput. Surveys* 44, 2 (2012), 1–35.

[14] Daniela S Cruzes and Tore Dyba. 2011. Recommended steps for thematic synthesis in software engineering. In *2011 International Symposium on Empirical Software Engineering and Measurement*. IEEE, 275–284.

[15] Carlo Daffara. 2007. Business models in FLOSS-based companies. (2007).

[16] Linus Dahlander and Mats Magnusson. 2008. How do firms make use of open source communities? *Long range planning* 41, 6 (2008), 629–649.

[17] Mikkel Flyverbom, Ronald Deibert, and Dirk Matten. 2019. The governance of digital technology, big data, and the internet: New roles and responsibilities for business. *Business & Society* 58, 1 (2019), 3–19.

[18] OpenStack Foundation. 2021. OpenStack Homepage. https://www.openstack.org/.

[19] OpenStack Foundation. 2021. The OpenStack Landscape. https://www.openstack.org/software/.

[20] OpenStack Foundation. 2021. oslo.db – OpenStack Database Pattern Library. https://docs.openstack.org/oslo.db/latest/.

[21] The Linux Foundation. 2020. 2020 Linux Kernel History Report. https://www.linuxfoundation.org/wp-content/uploads/2020_kernel_history_report_082720.pdf.

[22] John Fox. 2002. Cox proportional-hazards regression for survival data. *An R and S-PLUS companion to applied regression* 2002 (2002).

[23] A. Gelman and Jennifer Hill. 2006. *Data Analysis Using Regression and Multilevel/Hierarchical Models: Simulation of probability models and statistical inferences*. Data analysis using regression and multilevel/hierarchical models /.

[24] Marco Gerosa, Igor Wiese, Bianca Trinkenreich, Georg Link, Gregorio Robles, Christoph Treude, Igor Steinmacher, and Anita Sarma. 2021. The Shifting Sands of Motivation: Revisiting What Drives Contributors in Open Source. In *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*. 1046–1058. https://doi.org/10.1109/ICSE43902.2021.00098

[25] Mathieu Goeminne and Tom Mens. 2011. Evidence for the pareto principle in open source software activity. In *the Joint Porceedings of the 1st International workshop on Model Driven Software Maintenance and 5th International Workshop on Software Quality and Maintainability*. Citeseer, 74–82.

[26] Jesus M Gonzalez-Barahona and Gregorio Robles. 2013. Trends in free, libre, open source software communities: From volunteers to companies. *it–Information Technology it–Information Technology* 55, 5 (2013), 173–180.

[27] Dietmar Harhoff, Joachim Henkel, and Eric Von Hippel. 2003. Profiting from voluntary information spillovers: how users benefit by freely revealing their innovations. *Research policy* 32, 10 (2003), 1753–1769.

[28] Frank E Harrell Jr, Kerry L Lee, and Daniel B Mark. 1996. Multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors. *Statistics in medicine* 15, 4 (1996), 361–387.

[29] Donald A Hay and John Vickers. 1987. *The economics of market dominance*. B. Blackwell.

[30] Lawrence E Hecht and Libby Clark. 2018. Survey: Open Source Programs Are a Best Practice Among Large Companies. https://thenewstack.io/survey-open-source-programs-are-a-best-practice-among-large-companies.

[31] Frank Hecker. 1999. Setting up shop: The business of open-source software. *IEEE software* 16, 1 (1999), 45–51.

[32] Stefan Hellmer and Linda Wårell. 2009. On the evaluation of market power and market dominance—The Nordic electricity market. *Energy policy* 37, 8 (2009), 3235–3241.

[33] Joachim Henkel. 2006. Selective revealing in open innovation processes: The case of embedded Linux. *Research Policy* 35, 7 (2006), 953–969.

[34] W. Jing. 2012. Survival factors for Free Open Source Software projects: A multi-stage perspective. *European Management Journal* 30, 4 (2012), 352–371.

[35] Nicolas Jullien, Klaas-Jan Stol, and James D. Herbsleb. 2019. A Preliminary Theory for Open-Source Ecosystem Microeconomics. In *Towards Engineering Free/Libre Open Source Software (FLOSS) Ecosystems for Impact and Sustainability*. Springer Singapore, 49–68.

[36] David G Kleinbaum and Mitchel Klein. 2010. *Survival analysis*. Vol. 3. Springer.

[37] Bin Lin, Gregorio Robles, and Alexander Serebrenik. 2017. Developer turnover in global, industrial open source projects: Insights from applying survival analysis. In *IEEE 12th International Conference on Global Software Engineering*. 66–75.

[38] Edward R Mansfield and Billy P Helms. 1982. Detecting multicollinearity. *The American Statistician* 36, 3a (1982), 158–160.

[39] Arie Melnik, Oz Shy, and Rune Stenbacka. 2008. Assessing market dominance. *Journal of Economic Behavior & Organization* 68, 1 (2008), 63–72.

[40] Audris Mockus, Roy T Fielding, and James Herbsleb. 2000. A case study of open source software development: the Apache server. In *International Conference on Software Engineering*. 263–272.

[41] Audris Mockus and James D Herbsleb. 2002. Expertise browser: a quantitative approach to identifying expertise. In *Proceedings of the 24th international conference on software engineering. icse 2002*. IEEE, 503–512.

[42] Dennis C Mueller and John Cubbin. 2005. *The dynamics of company profits*. Cambridge University Press.

[43] Claudia Mueller-Birn, Marcelo Cataldo, Patrick Wagstrom, and James D Herbsleb. 2010. *Success in Online Production Systems: A Longitudinal Analysis of the Socio-Technical Duality of Development Projects*. Technical Report. School of Computer Science, Carnegie Mellon University. CMU-ISR-10-129. ISR Technical Reports.

[44] W Oh. 2007. Membership herding and network stability in the open source community : The Ising perspective. *Management Science* 53, 7 (2007), 1086–1101.

[45] OpenStack Foundation. 2022. Git Commit Good Practice. https://wiki.openstack.org/wiki/GitCommitMessages.

[46] Felipe Ortega and Daniel Izquierdo-Cortazar. 2009. Survival analysis in open development projects. In *2009 ICSE Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development*. IEEE, 7–12.

[47] Noel Pearse. 2019. An illustration of deductive analysis in qualitative research. In *18th European Conference on Research Methodology for Business and Management Studies*. 264.

[48] Huilian Sophie Qiu, Alexander Nolte, Anita Brown, Alexander Serebrenik, and Bogdan Vasilescu. 2019. Going Farther Together: The Impact of Social Capital on Sustained Participation in Open Source. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. 688–699.

[49] Michael A. Rappa. 2004. The utility business model and the future of computing services. *IBM Systems Journal* 43, 1 (2004), 32–42.

[50] Gregorio Robles, Jesús M González-Barahona, Carlos Cervigón, Andrea Capiluppi, and Daniel Izquierdo-Cortázar. 2014. Estimating development effort in free/open source software projects by mining software repositories: a case study of openstack. In *Proceedings of the 11th Working Conference on Mining Software Repositories*. ACM, 222–231.

[51] Cristina Rossi and Andrea Bonaccorsi. 2006. *4 – Intrinsic Motivations and Profit-Oriented Firms in Open Source Software : Do Firms Practise What They Preach?* 83–109 pages.

[52] Per Runeson and Martin Höst. 2009. Guidelines for conducting and reporting case study research in software engineering. *Empir Softw Eng* 14, 2 (2009), 131.

[53] Ioannis Samoladas, Lefteris Angelis, and Ioannis Stamelos. 2010. Survival analysis on the duration of open source projects. *Information & Software Technology* 52, 9 (2010), 902–922.

[54] Mario Schaarschmidt and Klaas-Jan Stol. 2018. Company Soldiers and Gone-Natives: Role Conflict and Career Ambition Among Firm-Employed Open Source Developers. In *International Conference on Information Systems (ICIS)*.

[55] Andreas Schilling, Sven Laumer, and Tim Weitzel. 2013. Together but apart: How spatial, temporal and cultural distances affect FLOSS developers' project retention. In *Proceedings of the 2013 annual conference on Computers and people*

research. 167–172.
[56] David Schuler and Thomas Zimmermann. 2008. Mining usage expertise from version archives. In *International Conference on Mining software repositories*. 121–124.
[57] Charles M Schweik and Robert C English. 2012. *Internet success: a study of open-source software commons*. MIT Press.
[58] Diomidis Spinellis and Vaggelis Giannikas. 2012. Organizational adoption of open source software. *Journal of Systems and Software* 85, 3 (2012), 666–682. https://doi.org/10.1016/j.jss.2011.09.037 Novel approaches in the design and implementation of systems/software architecture.
[59] Diomidis Spinellis and Clemens Szyperski. 2004. How is open source affecting software development? *IEEE software* 21, 1 (2004), 28.
[60] Klaas-Jan Stol and Brian Fitzgerald. 2018. The ABC of software engineering research. *ACM Transactions on Software Engineering and Methodology (TOSEM)* 27, 3 (2018), 11.
[61] OpenStack Superuser. 2015. Using Docker Machine with OpenStack. https://superuser.openstack.org/articles/using-docker-machine-with-openstack/.
[62] SwiftStack. 2021. Multi-cloud data storage and management for data-driven applications and workflows. https://www.swiftstack.com/.
[63] R Core Team et al. 2013. R: A language and environment for statistical computing.
[64] José Teixeira. 2017. Release Early, Release Often and Release on Time. An Empirical Case Study of Release Management. In *Springer, Cham*.
[65] Peter A Thiel and Blake Masters. 2014. *Zero to one: Notes on startups, or how to build the future*. Currency.
[66] Patanamon Thongtanunam, Shane McIntosh, Ahmed E. Hassan, and Hajimu Iida. 2016. Revisiting Code Ownership and Its Relationship with Software Quality in the Scope of Modern Code Review. In *Proceedings of the 38th International Conference on Software Engineering*. ACM, New York, NY, USA, 1039–1050.
[67] Marat Valiev, Bogdan Vasilescu, and James Herbsleb. 2018. Ecosystem-Level Determinants of Sustained Activity in Open-Source Projects: A Case Study of the PyPI Ecosystem. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (Lake Buena Vista, FL, USA). ACM, New York, NY, USA, 644–655.
[68] Frank van der Linden, Björn Lundell, and Pentti Marttiin. 2009. Commodification of Industrial Software: A Case for Open Source. *IEEE Software* 26, 4 (2009), 77–83.
[69] Patrick Wagstrom, James D. Herbsleb, Robert E Kraut, and A Mockus. 2010. The impact of commercial organizations on volunteer participation in an online community. In *Academy of Management Annual Meeting*.
[70] Ping Wang, Yan Li, and Chandan K Reddy. 2019. Machine learning for survival analysis: A survey. *ACM Computing Surveys (CSUR)* 51, 6 (2019), 1–36.
[71] Joel West and Scott Gallagher. 2006. Challenges of open innovation: the paradox of firm investment in open-source software. *R&d Management* 36, 3 (2006), 319–331.
[72] Likang Yin, Zhiyuan Zhang, Qi Xuan, and Vladimir Filkov. 2021. Apache Software Foundation Incubator Project Sustainability Dataset. In *2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR)*. IEEE, 595–599.
[73] Robert K Yin. 2017. *Case study research and applications: Design and methods*. Sage publications.
[74] Yuxia Zhang, Hao He, and Minghui Zhou. 2022. Commercial participation in OpenStack: Two sides of a coin. *Computer* 55, 2 (2022), 78–84.
[75] Yuxia Zhang, Hui Liu, Xin Tan, Minghui Zhou, Zhi Jin, and Jiaxin Zhu. 2022. Turnover of Companies in OpenStack: Prevalence and Rationale. *ACM Trans. Softw. Eng. Methodol.* (jan 2022). https://doi.org/10.1145/3510849
[76] Yuxia Zhang, Klaas-Jan Stol, Hui Liu, and Minghui Zhou. 2022. Online appendix to corporate domination in OSS. https://github.com/anonymous-0203/online-appendix-to-corporate-domination-in-OSS.
[77] Yuxia Zhang, Xin Tan, Minghui Zhou, and Zhi Jin. 2018. Companies' Domination in FLOSS Development – An Empirical Study of OpenStack. In *ICSE '18 Companion: 40th International Conference on Software Engineering Companion, May 27-June 3, 2018, Gothenburg*. IEEE.
[78] Yuxia Zhang, Minghui Zhou, Audris Mockus, and Zhi Jin. 2021. Companies' Participation in OSS Development–An Empirical Study of OpenStack. *IEEE Transactions on Software Engineering* 47, 10 (2021), 2242–2259.
[79] Yuxia Zhang, Minghui Zhou, Klaas-Jan Stol, Jianyu Wu, and Zhi Jin. 2020. How Do Companies Collaborate in Open Source Ecosystems? An Empirical Study of OpenStack. In *International Conference on Software Engineering* (Seoul, South Korea). Association for Computing Machinery, New York, NY, USA, 1196–1208.
[80] Zhongheng Zhang, Jaakko Reinikainen, Kazeem Adedayo Adeleke, Marcel E Pieterse, and Catharina GM Groothuis-Oudshoorn. 2018. Time-varying covariates and coefficients in Cox regression models. *Annals of translational medicine* 6, 7 (2018).
[81] Minghui Zhou, Qingying Chen, Audris Mockus, and Fengguang Wu. 2017. On the Scalability of Linux Kernel Maintainers' Work. In *11th Joint Meeting on Foundations of Software Engineering*. ACM, 27–37.
[82] Minghui Zhou and Audris Mockus. 2012. What make long term contributors: Willingness and opportunity in OSS community. In *Software Engineering (ICSE), 2012 34th International Conference on*. IEEE, 518–528.
[83] Minghui Zhou and Audris Mockus. 2015. Who Will Stay in the FLOSS Community? Modeling Participant's Initial Behavior. *IEEE Transactions on Software Engineering* 41, 1 (Jan 2015), 82–99.
[84] Minghui Zhou, Audris Mockus, Xiujuan Ma, Lu Zhang, and Hong Mei. 2016. Inflow and retention in oss communities with commercial involvement: A case study of three hybrid projects. *ACM Trans Softw Eng Methodol* 25, 2 (2016), 13.