

ScholarWorks@GSU

The Impact of Corporate Engagement in Open-Source Enterprise Systems Community on Release Performance

Authors	Li, Peiwei
Citation	Li, Peiwei. "The Impact of Corporate Engagement in Open-Source Enterprise Systems Community on Release Performance." 2022. Dissertation, Georgia State University. https://doi.org/10.31922/24py-4250
DOI	https://doi.org/10.31922/24py-4250
Download date	2026-05-13 16:40:58
Link to Item	https://hdl.handle.net/20.500.14694/3152

The Impact of Corporate Engagement in Open-Source Enterprise Systems Community on Release Performance

by

PEIWEI LI

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree

of

Doctor of Philosophy

In the Robinson College of Business

of

Georgia State University

CENTER FOR DIGITAL INNOVATION
GEORGIA STATE UNIVERSITY
ROBINSON COLLEGE OF BUSINESS
2022

Copyright by
Peiwei Li
2022

ACCEPTANCE

This dissertation was prepared under the direction of the Peiwei Li's Dissertation Committee. It has been approved and accepted by all members of that committee, and it has been accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Business Administration in the J. Mack Robinson College of Business of Georgia State University.

Richard Phillips, Dean

DISSERTATION COMMITTEE

Chair: Dr. Likoebe M. Maruping
(Center for Digital Innovation & Computer Information Systems Department, Georgia State University)

Co-chair: Dr. Lars Mathiassen
(Center for Digital Innovation & Computer Information Systems Department, Georgia State University)

Dr. Ling Xue
(Computer Information Systems Department, Georgia State University)

Dr. Sherae L. Daniel
(Carl H. Lindner College of Business, University of Cincinnati)

ABSTRACT

The Impact of Corporate Engagement in Open-Source Enterprise Systems Community on Release Performance

BY

Peiwei Li

June 23, 2022

Committee Chairs: Likoebe M. Maruping & Lars Mathiassen

Major Academic Unit: Center for Digital Innovation
& Department of Computer Information Systems

With the rise of corporate-sponsored open-source software (OSS) projects in the software industry, open-source enterprise systems (OS-ES) have become essential alternatives for small businesses to adopt and use the advanced business software packages. With a longitudinal study of a mature, collectively developed open source software project, we examine how corporate-communal engagement affects OS-ES performance through the theoretical perspective of group faultlines. Further, we propose that various release types can moderate the relationship between corporate-communal engagement and OS-ES release performance. Using ordinary least squares (OLS) regression with a final data set consisting of 124 data points (i.e., releases periods), we find that the relationship between corporate-communal engagement and OS-ES release performance is best characterized as a curvilinear relationship (U-shape relationship). That is, the evenness of corporate-communal engagement results in a reduced OS-ES release performance, and the unevenness of corporate-communal engagement can increase the OS-ES release performance in the forms of improved quality and innovativeness. Moreover, this curvilinear relationship is likely to be weaker in consolidating releases than in expanding releases. We find that our propositions are supported by the data. This dissertation provides various theoretical and practical contributions. Theoretically, we advance a theoretical framework to understand the effects and outcomes of corporate-communal engagement and release type contingencies by applying group faultlines theory to explain our research model. Further, we propose an alternative perspective on understanding software releases by distinguishing OS-ES releases into consolidating and expanding releases. Practically, this study provides suggestions and insights for corporate managers, open-source leaders, and small businesses to better engage in OS-ES development and adopt proper OS-ES products.

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank all of the professors who have taught me. I am very grateful and forever indebted to them.

First and foremost, I would like to express my gratitude to Dr. Likoebe M. Maruping for mentoring me through my MBA and Ph.D. journey. My first major in Material Science & Engineering provided me with sufficient knowledge of natural sciences. After getting my master's degree in that discipline, my interests changed—I was eager for social science knowledge. I then start with my MBA journey at the University of Louisville. During the first semester of my MBA, I took Dr. Maruping's class—Technology Management (MBA 608) and got my first A in the MBA program. Two years later, I looked for Dr. Maruping's advice on my Ph.D. application. The phone call from Dr. Maruping changed my life. After our conversation, I knew in my deepest heart that I wanted to study in the Ph.D. program at Georgia State University. My Ph.D. journey, unofficially, started with the Academy of Management (AOM) conference in 2016. I was so excited when Dr. Maruping took me around the conference to expand my network and gave me the initial impression of conducting IS research by talking with the senior Ph.D. students from all over the world. I know for sure Dr. Maruping enlightened my life, and I appreciate this great opportunity Dr. Maruping provided me to study in the Ph.D. program at GSU. During my Ph.D. study, the encouragement and guidance from Dr. Maruping, serving as a compass, always support me in overcoming difficulties in my Ph.D. journey. I would like to thank you for being so supportive of my academic pursuits and for being able to push me to go even further. Also, I would like to thank Dr. Maruping for introducing me to Dr. Lars Mathiassen, GRA Eminent Scholar and great Professor at the Center for Digital Innovation (the name was Center for Process Innovation—CEPRIN—at that time).

I know that Dr. Maruping would not have admitted me to the program without the blessing of his close colleague—Dr. Lars Mathiassen. I am very thankful to Dr. Mathiassen for selflessly looking after me for all these years in my Ph.D. Journey as my advisor and mentor. The first time we met was via Skype. I still remember I was extremely nervous, and I prepared many MISQ papers published by Dr. Mathiassen and described how I enjoyed reading those papers. Then, your smile calmed me down and made me feel more confident. With my engineering background, Dr. Mathiassen took a chance on me, providing me the opportunity to advance my Ph.D. journey in Information Systems. I knew for sure that I would have one of the most incredible opportunities in the world to have a fulfilling life when Dr. Mathiassen told me that I got the offer. The first day I entered my office, Dr. Mathiasen welcomed me warmly by showing me around the building and telling me the history of Atlanta. Working with Dr. Mathiassen was an incredible experience. His support, guidance, and training laid a solid foundation for me to become a qualified scholar. I learned so much from Dr. Mathiassen on becoming a rigorous researcher, and I am so inspired by how you conduct research projects. Again, I was very fortunate to take my Ph.D. adventure with Dr. Mathiassen.

I know I would not complete my Ph.D. journey without the tremendous support, help, and guidance from Dr. Arun Rai. Dr. Rai always believed in me and encouraged me throughout my Ph.D. study. Sometimes, only a small gesture/action will grant us indefinite power and energy, and that is what Dr. Rai offered me. His energized emotion empowered me whenever he shook my hand and started with: "Hello Peiwei, My Friend!" Having worked mostly with Dr. Maruping and Dr. Mathiassen, I got closer to Dr. Rai in his class—Theory Development. I discovered Dr.

Arun Rai from the researcher's perspective—brilliant, energized, and supportive. The way he talks about research projects in various situations always provides us with valuable insights and inspiration. I learned so much from Dr. Rai, including how to keep myself passionate about what I am doing and effectively express my research idea to the audience. I also get familiar with Dr. Rai from the party perspective—he likes a classical Chinese dish called “Ma Po Tofu,” and he always has a good sense of humor to make everyone feel so happy.

I would not have successfully completed and defended my dissertation without my beloved committee members—Dr. Ling Xue and Dr. Sherae L. Daniel. Dr. Xue, I am so grateful for the opportunity to work with you, and your valuable comments significantly improve the quality of my dissertation. Also, I am very appreciated your great support in my job hunting. And thank you so much for arranging those great CIS seminars. I learned so much from those seminars—what high-quality research looks like, how to present effectively, how to communicate with the audience politely, etc. Dr. Daniel, I appreciate your great support of my dissertation development. Without your valuable suggestions and comments, my dissertation would not turn out as it is now. As I am about to relocate to Cincinnati, we will have plenty of opportunities to collaborate on future research projects. I very much look forward to meeting you in Cincinnati.

To Dr. J.J. Po-An Hsieh, thank you for providing me with so many valuable insights from your class and all the support in my job hunting. To Dr. Yu-Kai Lin, I am very grateful to have met you during my Ph.D. study. I still remember we talked a lot on the way to the airport after you finished your visit to GSU. To Dr. Mark Keil, I made my first Ph.D. presentation in your seminar, and I gained not only valuable feedback but also confidence from that presentation.

I would express my gratitude for doing my Ph.D. at the Center for Digital Innovation. Everyone I met during my Ph.D. is wonderful. I want to thank my fellow Ph.D. students and post-doc scholars. Zhitao, I appreciate your friendship and all your support for me. Weifang, I appreciate your guidance on various matters. Youyou, thank you for helping me with all the information and sharing class materials with me, and I appreciate your great help when I joined this program. Yanran, I appreciate our discussions and conversations on research and day-to-day life. Yi, I appreciate our virtual interactions. An and Sudeep, although I did not get many chances to talk with you like some other students, I appreciate our conversations about our Ph.D. experience. I also want to appreciate all the support and help from Vanessa Browne. You are the one who is always taking care of me with various matters. Thank you!

Finally, I want to acknowledge Adenike Brewington. Although our interactions are mainly through emails, your effectiveness and efficiency make every step of my Ph.D. journey easier. I appreciate all your time and patience.

Table of Contents

ABSTRACT.....	4
List of Tables	9
List of Figures.....	10
Chapter 1. Introduction.....	11
1.1 Background	11
1.2 Research Motivation and Research Question.....	13
Chapter 2. Theoretical Background	19
2.1 ES Development in Open-source Context	19
2.2 Corporate-Communal Engagement in Open-source Software Development	22
2.3 Group Faultlines Theory	30
Chapter 3. Theoretical Framing	34
3.1 Corporate-communal Engagement in OS-ES Community	34
3.1.1 Corporate-communal Representation	38
3.1.2 Corporate-communal Contribution.....	39
3.1.3 Corporate-communal Responsiveness.....	41
3.2 Open-Source Enterprise Systems Release Types.....	43
Chapter 4. Research Model and Hypothesis Development	46
4.1 Research Model.....	46
4.2 Impact of Corporate-communal Representation on OS-ES Release Performance	47
4.3 Impact of Corporate-communal Contribution on OS-ES Release Performance.....	50
4.4 Impact of Corporate-communal responsiveness on OS-ES Release Performance	54
4.5 Moderating Role of OS-ES Release Type.....	57
Chapter 5. Research Methods	60
5.1 Research Context.....	60
5.2 Case Selection	61
5.3 Data and Sample.....	61
5.3.1 Data Collection and Coding – Pull Requests	63
5.3.2 Data Collection – Star.....	68
5.3.4 Data Collection and Coding – Releases	68
5.4 Construct Operationalization.....	69
5.4.1 Dependent Variable	69

5.4.2 Independent Variables	69
5.4.3 Moderator	70
5.4.4 Control Variables.....	70
Chapter 6. Results	72
6.1 Analysis and Results	72
6.1.1 Descriptive Statistics and Correlations.....	72
6.1.2 Hypothesis Testing	74
6.2 Supplementary Analyses for Robustness Tests.....	82
6.2.1 Test for Alternative Dependent Variable.....	82
6.2.2 Test for Early and Late Stage	83
6.2.3 Test for Alternative Independent Variable	84
Chapter 7. Discussion	86
7.1 Theoretical Contributions	88
7.2 Practical Contributions	92
7.3 Limitations and Future Research.....	93
Chapter 8. Conclusion.....	95
References.....	96
Appendix I: Tables of Robustness Tests.....	106
Appendix II: Intercoder Reliability.....	122

List of Tables

Table 2.1 Configurations of Corporate-communal Engagement.....	29
Table 3.1 Corporate-communal Engagement in OS-ES Community	37
Table 3.2 Concepts and Operationalizations of OS-ES Releases	45
Table 5.1 Data Collection & Coding of Pull Requests	66
Table 6.1 Descriptive Statistics and Correlations	73
Table 6.2 Hypotheses Testing.....	74
Table A.1 (Alternative DV) Descriptive Statistics and Correlations.....	106
Table A.2 (Alternative DV) Hypotheses Testing.....	107
Table A.3 (Early Stage) Descriptive Statistics and Correlations	110
Table A.4 (Early Stage) Hypotheses Testing	111
Table A.5 (Late Stage) Descriptive Statistics and Correlations	112
Table A.6 (Late Stage) Hypotheses Testing.....	113
Table A.7 (Alternative IV) Descriptive Statistics and Correlations	117
Table A.8 (Alternative IV) Hypotheses Testing	118

List of Figures

Figure 4. 1 Research Model.....47

Figure 5.1 Task and Developer Information Collected from Pull Requests of SuiteCRM 64

Figure 5.2 Response Information Collected from Pull Requests of SuiteCRM.....65

Figure 5.3 Coding Process.....67

Figure 6.1 Relationship between Corporate-communal Representation and Release Performance (Model 1)76

Figure 6.2 Relationship between Corporate-communal Contribution and Release Performance (Model 2).....77

Figure 6.3 Relationship between Corporate-communal Responsiveness and Release Performance (Model 3).....78

Figure 6.4 Interaction between Corporate-communal Representation and Release Type in Predicting Release Performance (Model 4)..... 80

Figure 6.5 Interaction between Corporate-communal Responsiveness and Release Type in Predicting Release Performance (Model 6)..... 82

Figure A.1 Relationship between Corporate-communal Representation and Release Performance (Model 1 for Alternative DV) 108

Figure A.2 Interaction between Corporate-communal Representation and Release Type in Predicting Release Performance (Model 4 for Alternative DV)..... 109

Figure A.3 Relationship between Corporate-communal Representation and Release Performance (Late-Stage Model 1)..... 114

Figure A.4 Relationship between Corporate-communal Contribution and Release Performance (Late-Stage Model 2) 115

Figure A.5 Relationship between Corporate-communal Responsiveness and Release Performance (Late-Stage Model 3)..... 116

Figure A.6 Relationship between Corporate-communal Representation and Release Performance (Alternative IV Model 1) 119

Figure A.7 Relationship between Corporate-communal Contribution and Release Performance (Alternative IV Model 2) 120

Figure A.8 Relationship between Corporate-communal Responsiveness and Release Performance (Alternative IV Model 3) 121

Chapter 1. Introduction

1.1 Background

Enterprise systems (ES) are complex commercial software packages that streamline the flow of information and transaction data to support diverse company functions and business activities (Davenport 1998; Markus and Tanis 2000). ES promise seamless information integration through various enterprise software applications such as enterprise resource planning (ERP), customer relationship management (CRM), and supply chain management (SCM) (Davenport 1998).

Despite the apparent advantages of ES, many small businesses still cannot afford to adopt it due to its high cost of maintenance and upfront fees (Lee et al. 2009; Olson et al. 2018). According to a report released by the Office of Advocacy, small businesses contribute over two-thirds of new jobs in the US and account for over 44% of the country's economic activity. To effectively compete in the market, the leading enterprise system vendors have been increasingly focused on developing effective marketing strategies that can help small businesses (Markus and Tanis 2000; Olson et al. 2018). According to SAP, the company has stated that it will provide a simplified version of its software to help small businesses. Microsoft, on the other hand, has committed to providing its customers with an affordable price range (Lee et al. 2009). As the demand for ES grows among small businesses, the emergence of open-source software (OSS) technology provides a viable alternative for small businesses—open-source enterprise systems (OS-ES; Boulanger 2005).

Traditionally, ES are developed out in-house by large organizations such as SAP, Microsoft, and Oracle (Davenport 1998). These ES vendors have strict control over development processes and are responsible for regular ES upgrades and maintenance (Lindberg et al. 2016; Olson et al.

2018). Raymond (1999) refers to this form of organizing as the cathedral model of development. The cathedral model involves establishing an embedded hierarchy that enables developers to manage and coordinate their work across various departments and regions (Lindberg et al. 2016). Large organizations usually have to pay substantial costs to hire skilled developers to successfully develop large-scale and complex ES (Curtis et al. 1998). In such a way, these ES vendors offer expensive closed-source systems and a limited scope of ES functionalities to be extended or modified (Dörner et al. 2009).

In contrast to the cathedral model, OSS development refers to the bazaar model (Raymond 1999). OSS development has fundamentally changed the way software is designed by utilizing a community approach to software development and providing external developers with broader access and authority (Daniel et al. 2013; Setia et al. 2012). Notably, OSS development has presented an alternative to the oligopolistic control of the ES market by large organizations, such as SAP, Microsoft, and Oracle. Considering the open nature of OSS, OS-ES development provides opportunities for external developers to contribute freely and offers small businesses free access to ES products (Mockus et al. 2002; Setia et al. 2012).

OS-ES development shares similar advantages as OSS development in general: capability to rapidly identify and resolve defects (Raymond 1999), capability to produce high-quality software (Mockus et al. 2002), capability to infuse new ideas, and capability to reduce development time (Chesbrough 2003). However, despite these advantages, most OS-ES are not as successful as they could be. Previous research has noted that open source is not a mature area when it comes to developing large-scale and complex applications, such as CRM, ERP, content management systems, and business intelligence products (Bruce et al. 2006; Johansson and Sudzina 2008; Olson et al. 2018). There are various reasons for this. One of the most significant reasons is that

the continuous development of OS-ES cannot satisfy the requirements of small businesses (Johansson and Sudzina 2008), such as adding new features, eliminating maintenance obligations, and providing professional customer support. Because most small businesses do not possess the necessary IT expertise to resolve issues or customize features by themselves, they rely entirely on the OS-ES community. Consequently, the lack of the requisite continuous development of OS-ES can affect their critical business activities and erode their engagements with OS-ES (Hauge et al. 2010; Li et al. 2020). Thus, we seek to explore a sustainable and effective way to develop and manage OS-ES.

1.2 Research Motivation and Research Question

As the market demand for OS-ES among small businesses increases (Olson et al. 2018), developers are trying to capture more market share by improving the quality and innovativeness of OS-ES. Thus, OS-ES success becomes increasingly essential to both the OS-ES developers and small businesses. On the one hand, small businesses rely on the performance (e.g., sustainability and effectiveness) of OS-ES to achieve their business objectives. Thus, the success of their organizations is largely dependent on the effectiveness of their implemented OS-ES. On the other hand, OS-ES developers can reap considerable benefits from a successful OS-ES in terms of reputation, career opportunities, and monetary benefits (Roberts et al. 2006). So, a better understanding of OS-ES development holds important practical implications for both the OS-ES developers and small businesses. As noted in the previous research, OS-ES development inherits various advantages and challenges of traditional OSS development. Similar to OSS development, globally distributed developers grant many advantages to OS-ES development through their diversified expertise and voluntary contributions (Giuri et al. 2010; Von Krogh et al. 2012; Daniel et al. 2013). However, the large number of these OS-ES developers may be problematic

and cause difficulties in leveraging their contributions due to the responsiveness is time-consuming (Lakhani and Von Hippel 2003; Markus 2007). For example, it is challenging for both corporations and OSS leaders to screen and coordinate such a huge number of developers and their contributions to reach the goal of effective OSS development. Thus, the success of OS-ES development would become difficult to attain without effectively leveraging these voluntary developers and their contributions. From the perspective of OS-ES developers, it is also critical for them to effectively engage in OS-ES development activities and sustainably provide necessary service, maintenance, and updates for the software.

Successful OS-ES projects on GitHub always need a group of developers who have the requisite skills and necessary knowledge base to deal with user-raised issues and deliver various releases. Integrating the insights of previous literature and our observations of successful OS-ES projects, we note that successful OS-ES projects are often backed by a sponsoring organization (Eilhard 2009; Germonprez et al. 2012; Giovacchini 2017; Priharsari et al. 2020). The success of sponsored OSS (e.g., OS-ES) projects hinges on a symbiotic community (Giovacchini 2017; Priharsari et al. 2020) combining the merits of both sponsored developers (SD)—who represent the interests of the sponsoring organization—and volunteer developers (VD; Setia et al. 2012; Spaeth et al. 2015)—who are not affiliated with the sponsor and freely contribute their time and expertise. We believe that the collaboration between SD and VD in OS-ES projects provides an alternative approach to developing large-scale and complex OSS projects. In such a way, we describe this collaboration between SD and VD as corporate-communal engagement and define it as *symbiotic ecosystem enabling both community volunteer developers (VD) and corporation sponsored developers (SD) to continuously participate, commit, and coordinate for OS-ES value co-creation* (Dahlander and Magnusson 2005; Stewart et al. 2006; Germonprez et al. 2012; West

and Sims 2016; Germonprez et al. 2017; Priharsari et al. 2020). The SD are the ones who represent the sponsoring organization, while the VD are the ones who act as the communal engagement (Mockus et al. 2002; West and O'Mahony 2008; Priharsari et al. 2020). Previous literature has identified mixed findings of corporate-communal engagement on software development performance (See Table 2.1 for the summary of previous findings; Dahlander and Wallin 2006; West and O'Mahony 2008; West and Lakhani 2008; Dahlander and Magnusson 2008; Stuermer et al. 2009; Teigland et al. 2014; Germonprez et al. 2017). For example, corporate-communal engagement can be mutually beneficial such that both the corporation and the open source community can benefit from the continuous exchange of knowledge and expertise, which can result in diffusion of innovation (Dahlander and Wallin 2006; West and O'Mahony 2008) and improvement of system quality (Shah 2006; Murray and O'Mahony 2007). In contrast, corporate-communal engagement can cause competition between SD and VD as both groups of developers simultaneously seek to prioritize their own interests and goals (West and O'Mahony 2008; Giovacchini 2017; Priharsari et al. 2020). The proportion of representation of such competing interests can affect the likelihood of achieving success. Consequently, we introduce *group faultline* (Lau and Murnighan 1998; Thatcher and Patel 2012) as our backbone theory to explain how corporate-communal engagement can influence OS-ES development, guided by our first research question: *what is the impact of corporate-communal engagement on OS-ES release performance?* As all these activities and contributions of OS-ES are finally reflected through a delivered release, we believe the characteristics of OS-ES releases hold important implications in OS-ES development. Inspired by the prior research on digital debt and digital options (Rolland et al. 2018), we distinguish OS-ES releases into consolidating and expanding releases based on the understanding of release notes. Thus far, we propose that

various release types may have different impacts on the relationship between corporate-communal engagement and OS-ES release performance, guided by our second research question: *how does the release type influence the relationship between corporate-communal engagement and OS-ES release performance?* Next, we elaborate on our research questions by introducing the group faultlines theory and our key concepts.

Building upon Lau and Murnighan's (1998) seminal conceptual work on demographic faultlines, Thatcher and Patel (2012) defined group faultlines as "hypothetical dividing lines that split a group or a team into two or more subgroups based on one or more individual attributes." The demographic attributes of a group can lead to the creation of hypothetical dividing lines. These lines can be caused by the alignment of various factors such as race, sex, and educational background (Bezrukova et al. 2009). We extend the application of this idea beyond demographics to include views and interests that are fundamental to the groups' objectives. Developers of OS-ES can also create a strong faultline by aligning their functional background, experience, and strategic objectives with the demographic attributes of the group. This can result in the creation of two homogenous subgroups—SD and VD. In such a way, the tension between VD and SD will increase when a strong faultline is activated. The intense conflict between groups will negatively affect the group outcomes (Thatcher and Patel 2012). Particularly, a strong faultline can be enabled by the evenness—i.e., equality—of subgroup size (Lau and Murnighan 1998; Thatcher and Patel 2012). Thatcher and Patel (2012) explain that the evenness of subgroup size could be linked to the presence of stronger faultlines. As the preceding discussion suggests, the evenness of SD and VD (e.g., in terms of their group size) can form a strong faultline and cause intensive competition between the sponsoring corporation and the

open source community. In turn, these conflicts will result in a reduced OS-ES release performance.

In this dissertation, we conceptualize corporate-communal engagement into three types: corporate-communal representation, corporate-communal contribution, and corporate-communal responsiveness. First, corporate-communal representation indicates the group size of corporation sponsored developers (i.e., SD) and volunteer developers emerging from the open source community (i.e., VD). Second, corporate-communal contribution reflects the number of committed tasks by both sponsoring corporation (i.e., SD committed tasks) and the open source community (i.e., VD committed tasks). Finally, corporate-communal responsiveness spots the amount of responsiveness offered by both sponsoring corporation (i.e., SD offered responses) and the open source community (i.e., VD offered responses).

Further, inspired by the research on digital debt and digital options (Rolland et al. 2018) and the merits of previous research on release notes, we distinguish the OS-ES releases into two types: expanding and consolidating releases (See the summary in Table 3.2; Huang et al. 2016; Bi et al. 2020). The main goal of the OS-ES releases is to improve the overall experience of users by adopting new features and eliminating existing bugs. According to the different information carried by OS-ES releases, expanding releases are likely to focus on knowledge addition in the form of realizing option-oriented tasks such as function expansion and improvement (Benaroch et al. 2006; Lankton and Luft 2008). In contrast, consolidating releases are likely to focus on knowledge modification in the form of resolving debt-oriented tasks such as fixing bugs, refactoring weak code, and upgrading system security (Singh et al. 2011; Huang et al. 2016; Bi et al. 2020). Therefore, we propose that corporate-communal engagement (i.e., corporate-

communal representation, contribution, and responsiveness) has a significantly lower influence on the OS-ES release performance for consolidating releases than for expanding releases.

We structure the rest of the dissertation by introducing the theoretical background of OS-ES development, corporate-communal engagement in OSS development, and group faultlines theory. Then, we elaborate on our theoretical framing with corporate-communal engagement and OS-ES release types. Next, we present our research model and associated hypotheses followed by the research methods, including the research context, case selection, data coding, and data analysis. Further, I conduct a series of additional analyses to assess the robustness of my results. Finally, we discuss theoretical and practical implications with limitations and future research directions.

Chapter 2. Theoretical Background

2.1 ES Development in Open-source Context

Generally, there are two fundamentally different models for developing large-scale and complex software (e.g., ES): the cathedral and the bazaar models (Raymond 1999). The traditional development model was referred to by the term “cathedral,” as a cathedral is closely controlled and operated by an organization (Raymond 1999; Dörner et al. 2009; Olson et al. 2018). For example, the embedded control and rules of the “cathedral” model provide required responsiveness within and between various teams in large-scale software development (Christopher et al. 2009; Olson et al. 2018). The “bazaar” model implies a free market of ideas without traditional organizational hierarchies and associated institutionalized responsiveness mechanisms (Raymond 1999; Winter et al. 2014; Lindberg et al. 2016; Olson et al. 2018). For instance, any developer can access the source code and join the bazaar by requesting new features and reporting bugs (Dörner et al. 2009).

ES has traditionally been developed under the cathedral model, giving rise to famous corporations such as SAP, Azure (by Microsoft), and Oracle (Davenport 1998; Olson et al. 2018). These organizations closely hold their intellectual property rights and strictly control the ES development processes with rules and norms (Lindberg et al. 2016; Olson et al. 2018). Further, the embedded hierarchical organization structure provides governance and responsiveness mechanisms that enable large-scale and complex ES development (Lindberg et al. 2016). For example, ES development requires intensive cooperation within and between engineering and management teams located in different departments to make sense of the extant functions and envision ES development directions (Poile et al. 2009; Lindberg et al. 2016). The

“cathedral” ES development model offers robust protections on their source code by hiring contract software developers and limited capability to customize or extend ES functionalities (Dörner et al. 2009; Olson et al. 2018). For example, in traditional ES development, it is impossible for business users to develop add-ons freely (Olson et al. 2018).

Recently, there has been a rise in ES that are developed under the bazaar model—a viable alternative to traditional software development. As opposed to the “cathedral” model, OSS development relies on code modularization and superposition of tasks (Howison and Crowston 2014; Lindberg et al. 2016; Medappa and Srivastava 2019). For example, most OSS tasks are accomplished through the incremental layering of contributions rather than co-work or concurrent task development (Lindberg et al. 2016; Medappa and Srivastava 2019). This “bazaar” development 1) allows developers and users to have full access to the source code and freedom to customize the software; 2) allows distributed decision-making power; 3) offers a community version of OS-ES with reduced costs (Serrano and Sarriegi 2006; Sen 2007; Dörner et al. 2009).

Recently, OSS technology has gradually shifted its focus from low-level infrastructure to business applications and attracted increased attention in the ES industry (Johansson and Sudzina 2008; Dörner et al. 2009; Li et al. 2020). The emergent OS-ES development represents a unique OSS development context and inherits many advantages of OSS development. First, large-scale OS-ES development is built upon collaboration through open superposition. Open superposition is distinct from organizing via hierarchies (Benkler 2002; Powell 1990; Howison and Crowston 2014). Howison and Crowston (2014) defined it as “the process of depositing motivationally independent layers of work on top of each other over time.” In other words, distributed developers contribute to a functionally interdependent IT artifact through the autonomous

production of independent layers (Howison and Crowston 2014; Medappa and Srivastava 2019). Open superposition through superposed tasks and incremental layering has been found as an effective way to accomplish complex tasks in open source communities (Howison and Crowston 2014; Medappa and Srivastava 2019). This unique work breakdown structure enabled by open superposition can enhance the motivation of OS-ES developers (Medappa and Srivastava 2019). For example, this work breakdown structure satisfies OS-ES developers' autonomy by minimizing developers' interdependencies and addressing the need for relatedness by layering tasks on others' work (Medappa and Srivastava 2019).

Second, on top of collaborations through open superposition, OS-ES development requires a governance mechanism to address interdependencies among developers and tasks (Howison and Crowston 2014; Lindberg et al. 2016). For example, essential OS-ES tasks require governance from core developers (e.g., merging a pull request) because of the need to integrate knowledge and align project goals into development tasks. In the absence of traditional organizational hierarchies, OS-ES development can benefit from the contributions of community volunteer developers (VD) and sponsored developers (SD). The VD who emerge from the community possess localized knowledge and communal resource for further development (Von Krogh et al. 2013; Lindberg et al. 2016; Chen et al. 2021). The SD hired by the sponsoring organizations have the authority to prioritize their own interests (Chen et al. 2021) and sufficient knowledge to maintain the OSS development (Mockus et al. 2002). Previous literature has identified that OSS projects are increasingly affiliated with formal organizations such as for-profit corporations (Stewart et al. 2006) and that such projects are characterized by developers from both the sponsoring organization (i.e., sponsored developers) and open source community (i.e., volunteer

developers; Daniel et al .2018). That is, the work they perform reflects the interests of the corporation and the community simultaneously (Daniel et al .2018).

As a notable example of large-scale OSS, OS-ES has received considerable industry attention. For instance, it has become common for firms—such as Home Depot, Toyota, and Fidelity—to invest in OS-ES projects (Weber 2005; Olson et al. 2018). To better understand ES development in the open-source context, this dissertation investigates how to deliver successful OS-ES systems through the collaboration of sponsored and volunteer developers—corporate-communal engagement. Next, we present an overview of the literature related to corporate-communal engagement in open source development.

2.2 Corporate-Communal Engagement in Open-source Software Development

Corporate-communal engagement has transformed the open source landscape, and in doing so, has created a managed and stabilized development environment and supported structured practices in OS-ES development (Kelty 2013; Germonprez et al. 2017). Initially, open source was a philosophy that utilized distributed voluntary talents and contributions to realize the objective of free and open software development (Coleman 2012; Germonprez et al. 2017). The past 15 years have seen a remarkable rise in corporations joining open source communities, which may enable and constrain communal engagements (Germonprez et al. 2017). Thus far, open source software development has moved beyond the basement hacker image to include sponsored developers who leverage, differentiate, and contribute to corporate value (Jeppesen and Frederiksen 2006; Dahlander et al. 2008; Germonprez et al. 2012).

Corporate-communal engagement has become a viable way to develop open source software that incorporates both corporate and communal characteristics (Eilhard 2009; Germonprez et al.

2012; Giovacchini 2017; Maruping et al. 2019). For example, it is much more common in complex open source software development (e.g., OS-ES development). Based on our observations of nearly 1000 OS-ES projects hosted on GitHub, we observe that majority of successful OS-ES projects involve developers coming from sponsoring corporations. With corporate-communal engagement, OS-ES are developing and evolving through both corporate and community ideals (Germonprez et al. 2012).

Corporate-communal engagement has been validated by previous literature in the expression of corporate engagement in open source communities (Dahlander and Wallin 2006; West and O'Mahony 2008; Teigland et al. 2014; Spaeth et al. 2015; Germonprez et al. 2017; Priharsari et al. 2020). West and Mahoney (2008, p. 149) defined sponsored open source community as "one where one (or more) corporate entities control the community's short- or long-term activities." In this arrangement, the sponsoring corporation and the open source community together establish a symbiotic relationship to fulfill the objective of system co-development and value co-creation (Priharsari et al. 2020). Building upon this, prior research has identified mixed results on how corporate-communal engagement impacts OSS projects' success (Dahlander and Magnusson 2005; Dahlander and Magnusson 2008; West and O'Mahony 2008; Germonprez et al. 2017). Specifically, corporate-communal engagement in the form of mutual dependence can result in high performance, such as innovation and improved quality (Dahlander and Wallin 2006; West and O'Mahony 2008). In contrast, corporate-communal engagement in the form of competition engenders lower performance due to the conflicts of goals, interests, and benefits between the sponsoring corporation and the community (West and O'Mahony 2008; Giovacchini 2017; Priharsari et al. 2020). We summarize three configurations of corporate-communal engagement

(see Table 2.1). Next, we will introduce the details of how these different configurations of corporate-communal engagement can impact OSS success.

Corporate-communal engagement reveals two key groups of developers—sponsored developers and volunteer developers (West and O’Mahony 2008; Priharsari et al. 2020). The sponsored developers represent the corporate engagement, while the volunteer developers act as the communal engagement. Both of these groups can be either control-focused—to affect the project content and decision-making (West and O’Mahony 2008), or openness-focused—to attract external developers (i.e., sponsored or community developers) to contribute to the project and appropriate value from it (Dahlander and Magnusson 2005). For example, control-focused corporate engagement occurs when the corporate entity controls the project’s short- and long-term activities, while openness-focused corporate engagement occurs when the corporation opens up access to the external developers (i.e., VD) in order to attract greater growth of participants (West and O’Mahony 2008). Particularly, Spaeth et al. (2015) observed that community volunteer developers are more likely to contribute to an OSS project when they identify that the sponsoring corporation is openness-focused—facilitating knowledge sharing and adoption between sponsoring corporation and open source community. Configuration 1 indicates the competition between corporate engagement (i.e., sponsored developers) and communal engagement (i.e., volunteer developers). Within this configuration, both groups are control-focused. Sponsored and volunteer developers simultaneously seek to control the open source project to realize their own primary goals. For example, the primary goal of corporations is to reap the return of their corporate engagement in the open source community (Eilhard 2009), while the goal of the open source community would be to improve the capabilities of shared technology (West and O’Mahony 2008). Related studies have recognized this competition in

corporate-communal engagement. For example, sponsored and volunteer developers are competing to control the OSS project content and goals (West and O'Mahony 2008; Giovacchini 2017; Priharsari et al. 2020), and their ultimate objective is to compete to control the values and outcomes of the OSS project (West and O'Mahony 2008). Prior research has further explored approaches to mitigate this conflict by leveraging the access to resources, aligning corporate and communal strategies, and assimilating corporate and communal knowledge (Dahlander and Magnusson 2005; Dahlander and Magnusson 2008). For instance, Chen et al. (2021) proposed the semi-decentralized governance structure where combining both the merits of SD and VD can result in higher performance in platform management.

Corporate-communal engagement can be mutually beneficial in the form of diffusion of innovation, learning of shared knowledge, and improvement of system quality (Dahlander and Wallin 2006; West and O'Mahony 2008; Stuermer et al. 2009; Germonprez et al. 2017). Configurations 2 and 3 present the mutual dependence between corporate engagement (i.e., sponsored developers) and communal engagement (i.e., volunteer developers). With configuration 2, the mutual dependence is realized through an openness-focused corporate engagement and a control-focused communal engagement. In other words, open source development is more aligned with and incentivized by communal engagement. This configuration is widely reported and extensively explored in the previous literature in the expression of sponsored developers embracing volunteer developers, and volunteer developers increasing their influence in open source development. For example, sponsored developers seek to attract volunteer developers' innovative contributions by opening up access to the open source community (West and O'Mahony 2008; West and Lakhani 2008; Dahlander and Magnusson 2008; Teigland et al. 2014), while volunteer developers look for opportunities to represent their

perspectives, leverage their local information, and improve their reputation through influencing the decision-making process (Ostrom 1990; Robert et al. 2006; Dahlander 2007; Von Krogh et al. 2012; Chen et al. 2021). With this mutual dependence ecosystem (i.e., community-driven), open source development is expected to 1) create commitment incentives for volunteer developers and result in more innovative contributions from communal engagement (Dahlander and Wallin 2006); 2) enable path-breaking innovations when volunteer developers can freely express their perspectives and leverage their local information with open access provided by sponsoring corporations (Dahlander and Wallin 2006; West and O'Mahony 2008; West and Lakhani 2008; Setia et al. 2012; O'Mahony and Karp 2022); and 3) establish new development standards as volunteer developers can promote their domain-specific knowledge to support the establishment of new standards (Fleming and Waguespack 2007; West and O'Mahony 2008).

The other mutual dependence configuration (i.e., configuration 3) of corporate-communal engagement is recognized through a control-focused corporate engagement and an openness-focused communal engagement. Specifically, in this configuration, open source development is regulated and driven by corporation engagement. It has been discussed in the previous literature in the expression of sponsored developers participating in attracting greater development activities and reaping the benefits of volunteer developer efforts by applying regulatory and governance structure, while the open source community keeps adopting the knowledge and standards of corporations (West and O'Mahony 2008; Stürmer 2009; Von Krogh et al. 2012; Chen et al. 2021). On the one hand, corporations can prioritize their own interests, in corporation-driven open source communities, over the community and align the decision-making process with their strategy goals (Stürmer 2009; Chen et al. 2021). On the other hand, the open source community enjoys the expertise and professional leadership that comes with corporations

(Eilhard 2009; Germonprez et al. 2013). With this mutual dependence ecosystem (i.e., corporation-driven), open source development is expected to 1) create commitment incentives for corporate developers to contribute and embed corporate views and standards in the open source community for reaping the returns of volunteer developer efforts via streamlining the development processes (Dahlander and Wallin 2006; Germonprez et al. 2013); 2) enable improved system quality and sustainability with long-term project affiliations (Dahlander and Wallin 2006; Stürmer 2009); and 3) enable further development by refining development iterations (Shah 2006; Murray and O'Mahony 2007; West and O'Mahony 2008; Stürmer 2009). For example, Shah (2006) found that volunteer developers who are motivated by reciprocity rarely spent time refining their code before contributing it to the community. With the development standards integrated by corporate engagement, developers have to refine the code to improve the code quality (Eilhard 2009).

Similar evidence has been found in OSS governance literature—a separate form of corporate-communal engagement. For example, the SD seek to govern the OSS project exclusively, allowing them to shape the decision-making process and project outcomes (Chen et al. 2021), while the VD look for leveraging their influence on OSS project and intent to collectively govern the OSS activities (De Noni et al. 2011; Chen et al. 2021). Still, these three configurations can be used well in the governance context. However, it is unclear on how to achieve mutual dependence instead of competition with the engagement of both SD and VD.

As these earlier studies have recognized the paradoxical nature of corporate-communal engagement (i.e., competition vs. mutual dependence), there have been only limited attempts to understand how to benefit from this symbiotic relationship, particularly in the context of OS-ES development. This dissertation seeks to investigate the ideal level of corporate-communal

engagement (i.e., the ideal composition of SD and VD) in terms of achieving the optimal OS-ES release performance.

Table 2.1 Configurations of Corporate-communal Engagement

	Corporate Engagement (Control-Focused)	Corporate Engagement (Openness-Focused)
Communal Engagement (Control-Focused)	<p>Configuration 1. Competition (Low Performance)</p> <ul style="list-style-type: none"> • Compete for controlling the OSS project content and decision-making (West & O’Mahonny 2008; Giovacchini 2017; Priharsari et al. 2020). • Compete for controlling the values and outcomes of the OSS project (West & O’Mahonny 2008). 	<p>Configuration 2. Mutual Dependence with Community-driven Community (Innovation-oriented High Performance)</p> <ul style="list-style-type: none"> • Create commitment incentives for communal developers (Dahlander and Wallin 2006). • Enable path-breaking innovations (West & O’Mahonny 2008; West and Lakhani 2008). • Establish new development standards (West & O’Mahonny 2008).
Communal Engagement (Openness-Focused)	<p>Configuration 3. Mutual Dependence with Corporation-driven Community (Quality-oriented High Performance)</p> <ul style="list-style-type: none"> • Create commitment incentives for corporate developers (Dahlander and Wallin 2006). • Enable improved system’s quality (Dahlander and Wallin 2006; Stürmer 2009). • Enable further development by refining design iterations (Shah 2006; Murray and O’Mahony 2007; West & O’Mahonny 2008; Stürmer 2009). 	<p>Configuration 4. Mutual Dependence Ideally Driven by Both Corporation and Community (Highest Performance)</p> <ul style="list-style-type: none"> • Open for the other group to leverage the collaboration and gain ambidexterity in OS-ES development—balancing innovation and quality. • Combining the merits of corporation and community engagement.

2.3 Group Faultlines Theory

Diversity within organizational groups is a challenge for organizations. To harness the benefits of groups and minimize the process losses associated with these groups, organizations have focused on group composition and individual attribute alignment (Thatcher and Patel 2012). For example, Lau and Murnighan (1998) address these issues by introducing an interesting concept of group faultlines. They noted that the alignment of individual demographic attributes (e.g., age) within a group tends to divide a larger group into subgroups of different ages (Lau and Murnighan 1998). Building upon Lau and Murnighan's (1998) research on group faultlines, Thatcher and Patel (2012, p. 970) defined group faultlines as "hypothetical dividing lines that split a group or a team into two or more subgroups based on one or more individual attributes." These hypothetical dividing lines—group faultlines—can be formed by aligning various individual demographic attributes (e.g., race, sex, nationality, age, and educational background; Bezrukova et al. 2009). Aside from their demographic characteristics, other factors such as personal values or personality may also lead to active subgroups within a larger group (Lau and Murnighan 1998).

Previous research on group faultlines has focused on finding appropriate ways of measuring group faultlines (Thatcher and Patel 2012). For example, Shaw (2004) described a methodology for measuring group faultlines and their relative strength by considering the following five steps: selection of proper attributes, identification of discrete categories within each attribute, development of measures based on subgroup internal alignment, calculation of an index of the "cross-subgroup alignment" of attributes, and combination of the measures of internal and cross-subgroup alignment into an overall index. Considering the relationship between faultline strength and distance, Zanutto et al. (2011) developed a measurement of group faultlines that reflects how

far apart the emerging subgroups are on demographic characteristics and captures how many demographic attributes align within a group. Taking into account that organizations may deal with multiple subgroups, Meyer and Glenz (2013) proposed a new cluster-based approach—average silhouette width (ASW)—that identifies the number of subgroups and subgroup membership. ASW approach is described as a clustering algorithm for identifying the number of subgroups and for reducing the calculation effort (Thatcher et al. 2003; Meyer and Glenz 2013). Further, the authors claimed that the ASW measure provides precise measurement of group faultlines in the presence of multiple subgroups (i.e., more than two subgroups can be identified in a larger group; Meyer and Glenz 2013).

Building upon the studies on group faultlines measurement, researchers acquire the capabilities to examine the potential impacts of faultline strength on group-level outcomes. Previous research has noted that the strength of group faultlines can influence the team processes and outcomes (e.g., Lau and Murnighan 2005; Bezrukova et al. 2007; Thatcher and Patel 2012). Particularly, Thatcher and Patel (2012) have identified that the theoretical development of the faultline concept has led to debates on the effects of faultlines on group-level outcomes. For example, in a qualitative field study, Earley and Mosakowski (2000) proposed a curvilinear relationship (i.e., an upright U-shaped relationship) between team heterogeneity on nationality and team performance in terms of effectiveness. Bezrukova et al. (2009) posited that not all faultline compositions (i.e., information-based faultlines—education and work experience) are necessarily negative for group performance through a moderated model of group faultlines, team identification, and group performance outcomes. Jehn and Bezrukova (2010) distinguish between dormant faultlines and activated group faultlines and found that activated faultlines are more likely to be associated with low levels of satisfaction and group performance than dormant

faultline groups. Vora and Markoczy (2012) further hypothesized that group faultlines play a moderating role in influencing the relationship between communication and group performance.

Inspired by the preceding discussions, we believe that group faultline holds important theoretical implications in the OS-ES development. As the OS-ES are developed based on the collaboration of corporations and open source communities, two subgroups of developers can easily form strong group faultlines. For example, in the symbiotic ecosystem of OS-ES development, SD and VD naturally form two subgroups as they work collaboratively. We hypothesize that the faultlines between SD and VD can significantly impact the group outcomes (i.e., OS-ES release performance) through corporate-communal representation, contribution, and responsiveness.

Additionally, previous research has pointed out that the group context (e.g., group size, the evenness of subgroup size, and the number of subgroups) may affect the extent to which a faultline influences outcomes (Lau and Murnighan 1998; O'Leary and Mortensen 2010; Thatcher and Patel 2011; Thatcher and Patel 2012). For example, Thatcher and Patel (2011) found that group size had an inverted-U effect on faultline strength, confirming that it is unlikely that subgroups can form strong faultlines when the overall group size is very large (Hart and Van Vugt 2006). Although relatively few empirical studies have investigated the evenness of subgroup size, the evenness of subgroups could have important implications for inter-subgroup dynamics (Lau and Murnighan 1998; Thatcher and Patel 2012). Lau and Murnighan (1998) noted that subgroups with an uneven number of members could have led to an imbalance in the distribution of power, resources, and abilities. Further, Thatcher and Patel (2012) examined the evenness of subgroups and found it was correlated with both stronger and more distant faultlines. O'Leary and Mortensen (2010) systematically examined the subgroup size. They found that an imbalance in the size of subgroups may associate with significantly poorer scores on

responsiveness problems, which could turn into a coalitional mentality. In this dissertation, we believe the evenness of subgroup size plays a similar role in influencing the group process and group outcomes. Particularly, we hypothesized that the evenness of subgroup size positively influences the group Faultline strength. As the evenness of subgroup size activates stronger group faultlines, team members are likely to be involved in a relationship conflict, task conflict, and lower team cohesion (Thatcher and Patel 2011), resulting in low OS-ES release performance. Next, we introduce the definitions and operationalizations of the key concepts and constructs.

Chapter 3. Theoretical Framing

3.1 Corporate-communal Engagement in OS-ES Community

Previous research has explored corporate-communal engagement to understand the phenomenon of for-profit corporations increasingly seeking to leverage this development model (Dahlander and Magnusson 2005; Stewart et al. 2006; Dahlander and Magnusson 2008; Von Krogh et al. 2012; Germonprez et al. 2017; Daniel et al. 2018; Priharsari et al. 2020). The concept of corporate-communal engagement has been developed in various ways. Stewart et al. (2006) used the term organizational sponsorship to indicate a publicly displayed affiliation between the corporation and the open source community. They define the corporate-communal engagement in the expression of organization sponsorship as “the fact that some but not all OSS projects are affiliated with a formal organization such as a for-profit company or a university” (Stewart et al. 2006, p. 128). Building upon the affiliation between corporation and community, researchers further conceptualize the corporate-communal engagement as an emerging value co-creation business model that can help promote risk mitigation and open innovation (Germonprez et al. 2012; West and Sims 2016; Germonprez et al. 2017). For instance, Germonprez et al. (2017) found that corporate-communal engagement can help companies develop effective risk management strategies. Previous research also defined corporate-communal engagement as a collaborative ecosystem involving both corporation sponsored developers and community volunteer developers (Dahlander and Magnusson 2005; Priharsari et al. 2020). Specifically, the relationships between sponsored developers and volunteer developers can be characterized as one of three forms: symbiotic, commensalistic, and parasitic ecosystems (Dahlander and Magnusson 2005). The symbiotic ecosystem implies that the corporation tries to co-develop the product with the open source community (Dahlander and Magnusson 2005). Specifically,

corporation and community both have the opportunity to gain benefits in the symbiotic ecosystem via effective collaboration and knowledge sharing. The commensalistic ecosystem is an intermediate way to inter-relate to the open source community and it allows corporations to benefit from the co-existence of the open source community while leaving it without harm (Dahlander and Magnusson 2005). The corporation will benefit from the commensalistic ecosystem, as open source developers will be indifferent to it. In contrast, the parasitic ecosystem implies that the corporation only prioritizes its own benefits, without taking into account that its actions might sabotage the open source community (e.g., alienate the community developers; Dahlander and Magnusson 2005). Typically, the firm will be the only winner in parasitic ecosystem. However, the existing research has been silent on defining corporate-communal engagement in large-scale OSS projects, such as OS-ES. Considering the large-scale OS-ES development requires continuous technical support, upgrades, and other resources that consumers of software products may need, in this study we focused on the symbiotic ecosystems—the high possibility of influencing the community (Dahlander and Magnusson 2005).

Recognizing the merits of previous research and the uniqueness of OS-ES development, we define corporate-communal engagement in the context of OS-ES development as *symbiotic ecosystem enabling both community volunteer developers (VD) and corporation sponsored developers (SD) to continuously participate, commit, and coordinate for OS-ES value co-creation*. Essentially, corporate-communal engagement provides opportunities for VD to increase community influence via representing their perspectives and leveraging their local information (Germonprez et al. 2017; Chen et al. 2021) and allows SD to align corporate strategies with communal activities through regulating and governing the OS-ES development process

(Dahlander and Magnusson 2008; Chen et al. 2021). Building upon our definition and the observations of OS-ES projects hosted on GitHub, we identify corporate-communal engagement in OS-ES community as being of three types: 1) corporate-communal representation, 2) corporate-communal contribution, and 3) corporate-communal responsiveness (The details are summarized in Table 3.1).

Table 3.1 Corporate-communal Engagement in OS-ES Community

Concept	Definition		
Corporate-communal Engagement	Symbiotic ecosystem enabling both community volunteer developers (VD) and corporation sponsored developers (SD) to continuously participate, commit, and coordinate for OS-ES value co-creation (Dahlander and Magnusson 2005; Stewart et al. 2006; Germonprez et al. 2012; West and Sims 2016; Germonprez et al. 2017; Priharsari et al. 2020).		
Engagement Types	Definitions	Examples	Operationalizations
Corporate-communal Representation	The evenness in representation of OS-ES developers participation between corporation and community (Eilhard 2009; Germonprez et al. 2012; Thatcher and Patel 2012; Germonprez et al. 2017).	# of Participating Sponsored Developer (SD), # of Participating Volunteer Developer (VD).	$\frac{\# \text{ of } SD}{\# \text{ of } (SD+VD)} \sim (0,1)$
Corporate-communal Contribution	The evenness in representation of OS-ES tasks committed by corporation and community developers (Di Tullio and Staples 2013; Germonprez et al. 2017; Maruping et al. 2019).	# of committed tasks by SD, # of committed tasks by VD.	$\frac{\# \text{ of Committed Tasks by } SD}{\# \text{ of Committed Tasks by } (SD+VD)} \sim (0,1)$
Corporate-communal Responsiveness	The evenness in representation of OS-ES responses offered by corporation and community developers (Zhang et al. 2013; Steinmacher et al. 2015; Zhou and Mockus 2015).	# of responses offered by SD, # of responses offered by VD.	$\frac{\# \text{ of Responses Offered by } SD}{\# \text{ of Responses Offered by } (SD+VD)} \sim (0,1)$

3.1.1 Corporate-communal Representation

Corporate-communal engagement is reflected by corporate-communal representation in OS-ES community (Eilhard 2009; Germonprez et al. 2012; Germonprez et al. 2017). Conventionally, the open source model was developed with the intention to avoid corporate participation, and the challenges involved in contributing to the open source community were significant for corporations due to the open source community being opposed to transforming voluntary efforts into commercial products (Dahlander and Magnusson 2005; Germonprez et al. 2013). However, a particular feature of OS-ES is that its development is increasingly affiliated with for-profit corporations (Stewart et al. 2006). The success of OS-ES development depends on the collaboration of both corporations and the open source community via the established symbiotic ecosystem (Priharsari et al. 2020). Additionally, corporate participation not only creates a positive reinforcement for corporations to further participate in the open source community, but also improves the capability of the geographically distributed open-source community to produce commercial-grade code for OS-ES through centered innovation, knowledge, and technologies (Dahlander and Magnusson 2005; Germonprez et al. 2017). In other words, these participating SD and VD establish a symbiotic ecosystem for OS-ES value co-creation. Thus far, we propose that OS-ES developers participation between corporation and community can reflect corporate-communal engagement significantly. Building upon the co-existence of the corporation (i.e., sponsored developers - SD) and the open source community (i.e., volunteer developers - VD) in OS-ES development, we then define *corporate-communal representation* as the evenness in representation of OS-ES developers participation between corporation and community (Eilhard 2009; Germonprez et al. 2012; Thatcher and Patel 2012; Germonprez et al. 2017). To operationalize the corporate-communal representation, we use the following equation:

$$\frac{\# \text{ of } SD}{\# \text{ of } (SD+VD)} \sim (0,1)$$

The above equation describes the evenness of the corporation and community representation in OS-ES development. With a range from 0 to 1, the larger the value, the higher the level of corporate engagement in the open source community in terms of corporation representation. For example, an OS-ES development consists of 10 developers covering the lifespan of a particular release period. Among these 10 developers, 5 of them are SD, and the rest are VD. The level of the corporate representation in this release period is 0.5—balanced representation between SD and VD. The unit of analysis is at the release level of an OS-ES project hosted on GitHub—SuiteCRM. The SD is labeled as “member” on GitHub, which refers to the developers who come from the sponsoring corporation; the VD is labeled as “contributor”, which refers to the developers who emerge from the open source community.

3.1.2 Corporate-communal Contribution

Corporate-communal engagement is reflected by corporate-communal contribution in OS-ES community (Maruping et al. 2019). As Maruping et al. (2019) posited that developers’ contribution grows as their commitment rises and sufficient contribution is fundamental for OS-ES development. The corporate-communal engagement of OS-ES development allows value co-creation through the contributions of both SD and VD. To develop OS-ES that is at least as robust and creative as traditional ES, it is always critical for OS-ES projects to attract sufficient developer contributions, given the lack of contractual obligation tying them to a certain project (Robles and Gonzalez-Barahona 2006; Seidel and Stewart 2011; Maruping et al. 2019). For example, SD will feel a strong commitment and contribute more when they embrace the OSS ideology (Eilhard 2009; Daniel et al. 2018); VD will have a strong commitment and higher level

of contribution to OS-ES projects when they experience a higher contribution level, which can also result in higher performance (Von Krogh and Hippel 2006). In other words, the symbiotic ecosystem of OS-ES development requires the contribution of both SD and VD collaboratively to reach OS-ES project goals by presenting correction of the software defects and improvement of software functionalities (Markus 2007; Di Tullio and Staples 2013). Hence, we define the *corporate-communal contribution* as the evenness in representation of OS-ES tasks committed by corporation and community developers (Di Tullio and Staples 2013; Germonprez et al. 2017; Maruping et al. 2019). Particularly, the SD who are committed and making contribution to OS-ES projects represent the desires of sponsoring organization. In contrast, the VD who are committed and making contribution to OS-ES projects represent the common value and interests of the open source community (Tiwana 2010). Previous research has measured developer contribution with the number of their inputs—commits (Maruping et al. 2019). In such a way, contribution is likely to be measured by developers committed tasks—a particularly effortful form of OS-ES developers engagement. Specifically, the OS-ES tasks committed by SD are more likely to be consistent with the sponsoring organization’s objectives, and the OS-ES tasks contributed by VD reflect the shared ideologies of the community. Recognizing the potential difference between SD and VD committed tasks in OS-ES projects and relationship between developer contribution and their committed tasks, the level of corporate-communal contribution can be reflected by the distribution of tasks committed by both developers from corporation and community. We operationalize the corporate-communal contribution with the following equation:

$$\frac{\# \text{ of Committed Tasks by SD}}{\# \text{ of Committed Tasks by (SD+VD)}} \sim (0,1)$$

The above equation describes the evenness of the corporation-community contribution in OS-ES development. The number of tasks committed by SD and VD is calculated within each release period. Taking the value between 0 and 1, the larger the value, the higher the level of corporate engagement in the open source community in terms of corporate contribution. For example, the tasks of Suite CRM are committed by 5 SD and 5 VD collaboratively in a particular release period. Within this period, 50 out of 100 OS-ES tasks are committed by VD to represent volunteer developers' perspectives, and the rest are fulfilled by SD for aligning to the sponsoring organization's objectives. The level of corporate contribution is 0.5—which indicates SD and VD are performing the same amount of tasks in a particular release period.

3.1.3 Corporate-communal Responsiveness

Corporate-communal engagement is reflected by corporate-communal responsiveness in the OS-ES community. The success of OS-ES development depends on the arrangement of and responsiveness between people, organizations, and technology (Howison et al .2012; Germonprez et al. 2017). The effective approach to coordinating OS-ES activities is to respond to the OS-ES community in a timely manner. Previous research has identified that a lack of community responsiveness can result in negative outcomes such as weak code and low developer retention rate and engagement (Daniel et al. 2013; Zhang et al. 2013). An effective OS-ES responsiveness in the form of providing sufficient responses to the OS-ES community can leverage developer motivation to further contribute and engage (Markus 2007). The symbiotic ecosystem of OS-ES development provides opportunities for both SD and VD to coordinate the development activities by providing high-quality responses for OS-ES community. To summarize the prior understandings of the incentive system and adapt them to our existing research context, we define the *corporate-communal responsiveness* as the evenness in

representation of OS-ES responses offered by corporate and OS-ES community developers (Zhang et al. 2013; Steinmacher et al. 2015; Zhou and Mockus 2015). In OS-ES projects, corporate-communal engagement establishes a solid foundation for both SD and VD to provide responses to the OS-ES community. With the responses offered by SD, OS-ES developers may be motivated through standard software solutions and potential career opportunities. However, the responses offered by SD may alienate voluntary OS-ES developers and decrease their motivations, as the SD should prioritize the sponsoring organizations' interests (Shah 2006; Eilhard 2009; Chen et al. 2021). With the responses offered by VD, OS-ES developers are motivated by the opportunities to freely represent their perspectives and partially control the project (Chen et al. 2021). However, the responses offered by VD may hinder OS-ES developers' motivations because of the reduced speed of decision-making and the likelihood of collective action (Shah 2006; Chen et al. 2021). To better leverage OS-ES developers' motivations, a combined responsiveness from both SD and VD is desired. We then operationalize the corporate-communal responsiveness with the following equation:

$$\frac{\# \text{ of Responses Offered by SD}}{\# \text{ of Responses Offered by (SD+VD)}} \sim (0,1)$$

The above equation describes the evenness of corporation and community responsiveness in OS-ES development. The number of responses offered by SD and VD is calculated within each release period. With a range from 0 to 1, the larger the value, the higher the level of corporate engagement in the open source community in terms of corporation responsiveness. For example, the 5 SD and 5 VD of SuiteCRM collaboratively provide responses to OS-ES developers in a particular release period. Within this period, 50 percent of responses are offered by VD, and the

rest of the responses are provided by SD. The level of corporation responsiveness is 0.5—which indicates SD and VD provide an equal number of responses in this release period.

3.2 Open-Source Enterprise Systems Release Types

As Eric Raymond (1999) phrased the release style of OSS development with the notion of “Release early. Release often”, it is common to observe that there are far more releases in OS-ES projects compared with traditional ES projects. In general, a large number of OS-ES releases are delivered through two common release strategies: time-based and content-based release strategies. Time-based release strategy means that releases are planned for a specific date (Michlmayr et al. 2015). A time-based strategy provides clear signals on the timing of system updates. As users and developers become aware of the release dates, OSS community activities will increase as the release dates approach (Rossi et al. 2009). Prior research also suggests that a time-based release strategy can increase the users’ synchronization, as it allows them to update their system at a regular frequency (Michlmayr et al. 2015). Content-based release strategy describes an approach where OSS projects issue a new release after implementing a certain set of features or defect fixes (Michlmayr et al. 2015). A content-based release strategy offers flexibility for both developers and users. For example, once there is a need to add an important new function or fix a critical defect, the content-based release strategy allows OSS developers to realize the new function or resolve the defect by releasing a new system version. In our case, the majority of OS-ES projects on GitHub employ a hybrid release strategy combining the time-based and content-based release strategies. For example, Suite CRM releases the major versions regularly and releases its minor versions after implementing important features or resolving critical defects. The time-based perspective provides limited information on OS-ES releases. Thus, to better understand OS-ES development, we seek to investigate OS-ES releases through a

content-based perspective. The content-based perspective allows me to classify OS-ES releases into different release types, which is helpful to indicate the associated software activities by granting each OS-ES release a label. By extracting useful software maintenance and evolution information from the release notes (Yu 2009), these release labels can guide more effective production and utilization by offering developers and users basic information about system activities spanning the whole OS-ES development cycle.

To date, a systematic understanding of the approach to classifying software releases, particularly OS-ES releases, into different types is still lacking. In the software development industry, releases are commonly distinguished into three types: major, minor, and patch releases. The classification criteria are based on the scale of the upgrades. For example, on GitHub, the major releases are large-scale upgrades, the minor releases are built upon the major releases and have a smaller scale, and the patch releases are the smallest upgrades. This distinction provides a vague and imprecise release classification and does not clearly describe the characteristics of each release. Developers and users cannot effectively and efficiently take advantage of the information provided by this classification. Thus, we propose to classify the OS-ES releases through the release notes. As a critical software artifact that serves as a communication bridge between developers and users, release notes contain a set of crucial information, such as enhancement of system quality and advancement of system functionality (Bi et al. 2020). Prior study on release notes indicates that six different information types could be identified in a release note: title, system overview, resource requirement, installation, address, issues, and caveats (Abebe et al. 2016). Further, the information in release notes can be used as signals to indicate different release types, such as feature-addition, future improvement, bug-fixing, feature-removing, and common operations (Huang et al. 2016). Based on the various information

carried by release notes, it is conceivable to classify OS-ES releases on GitHub into two types: expanding releases and consolidating releases. *Expanding releases* focus on the advancement of OS-ES system functionality through knowledge addition in the forms of feature addition and improvement (Singh et al. 2011; Huang et al. 2016; Bi et al. 2020). Expanding releases target improvements in OS-ES users' experience by adopting new features and improving existing features. *Consolidating releases* focus on the enhancement of OS-ES system quality through knowledge modification in the forms of bug fixes, code refactoring and reuse, documentation updates, and security improvement (Singh et al. 2011; Huang et al. 2016; Bi et al. 2020). Consolidating releases prioritize the OS-ES system stability by fixing defects, refactoring the weak code, and upgrading the system security. Table 3.2 summarizes the definitions of different OS-ES release types.

Table 3.2 Concepts and Operationalizations of OS-ES Releases		
Construct	Definition	Examples
Expanding releases	The advancement of OS-ES system functionality through knowledge addition in the form of feature addition and improvement (Singh et al. 2011; Huang et al. 2016; Bi et al. 2020).	Release 7.1 Beta Try 7.1 Beta to preview the new features coming in 7.1 including enhanced search.
Consolidating releases	The enhancement of OS-ES system quality through knowledge modification in the form of bug fixes, code refactoring and reuse, documentation update, and security improvement (Singh et al. 2011; Huang et al. 2016; Bi et al. 2020).	Release 7.1.7 This is a bug fix release which addresses Post-Auth RCE vulnerabilities in SuiteCRM.

Chapter 4. Research Model and Hypothesis Development

4.1 Research Model

The discussions have thus far focused on the conceptualizations and operationalizations of corporate-communal engagement and OS-ES releases. As an attractive alternative to traditional closed source ES development, OS-ES development is a prominent example of the community-based development model that leverages the contributions of both SD and VD to produce commercial-grade software (Dahlander and Magnusson 2008; West and Sims 2016). Corporation engagement (i.e., SD) allows the sponsoring organization and developers to prioritize their interests over those of the OS-ES community (Chen et al. 2021). Community engagement (i.e., VD) provides voluntary developers opportunities to represent their perspectives and leverage their local information (Chen et al. 2021). Together, SD and VD establish this symbiotic ecosystem to develop OS-ES collaboratively. Although our general knowledge about this collaboration (i.e., corporate-communal engagement) is extensive, we need to elaborate on the conceptualization and measurement of corporate-communal engagement and better understand how the collaboration between SD and VD can impact the OS-ES project performance (e.g., when the team conflict or cohesion exist between SD and VD). In response, we conceptualize corporate-communal engagement based on existing research, develop the operationalizations, and seek to examine the relationship between corporate-communal engagement and OS-ES performance through three types of engagement: corporate-communal representation, corporate-communal contribution, and corporate-communal responsiveness. Further, we test the moderation effects of OS-ES releases on the relationships between the three types of corporate-communal engagement and OS-ES release performance. To distinguish the OS-ES releases, we

categorize them into expansion and consolidating releases. The proposed research model is shown in Figure 4.1.

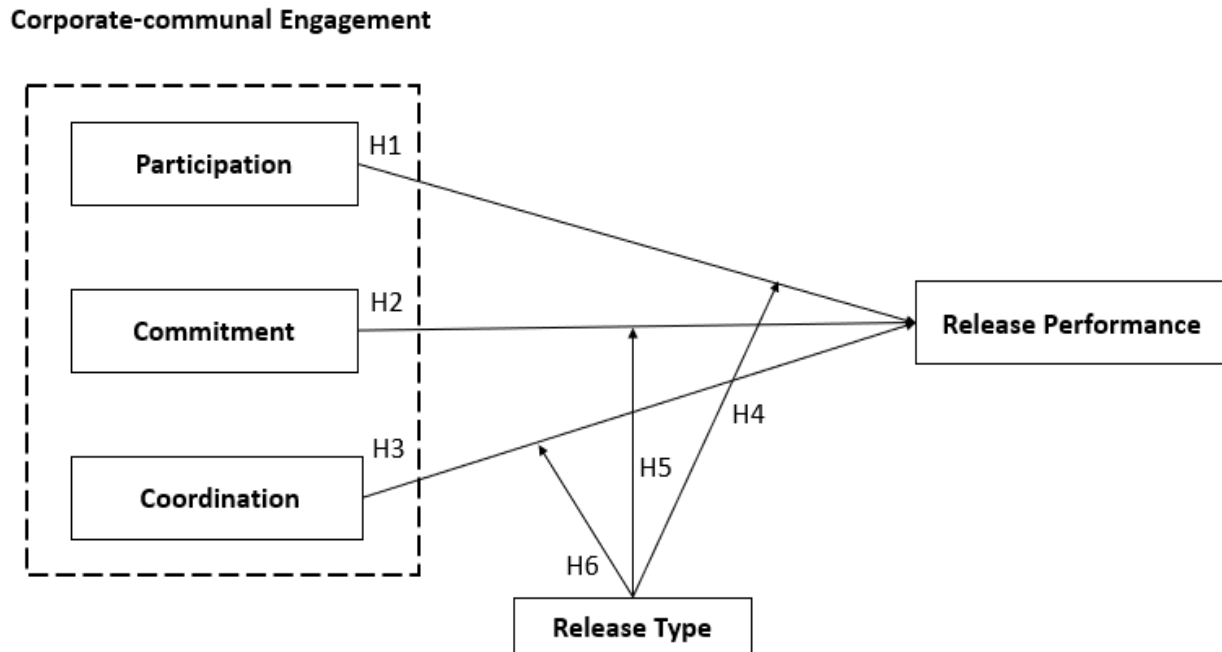


Figure 4.1 Research Model

4.2 Impact of Corporate-communal Representation on OS-ES Release Performance

Corporate-communal representation reflects the evenness in representation of OS-ES developers' distribution between corporation and community (Eilhard 2009; Germonprez et al. 2012; Thatcher and Patel 2012; Germonprez et al. 2017). The symbiotic ecosystem of OS-ES development consists of two sub-groups of developers (i.e., SD and VD) who are significantly distinct from each other. Previous research has explicitly identified their differences (Stewart et al. 2006; Germonprez et al. 2017; Daniel et al. 2018; Chen et al. 2021). For example, SD who are

sponsored by corporations always prioritize the corporate strategies over community interests, and VD who emerge from open source community consistently seek to increase their community influence via representing their perspectives and leveraging their local information (Dahlander and Magnusson 2008; Germonprez et al. 2017; Chen et al. 2021). What effects do such differences have on OS-ES release performance? Noting that these two groups of OS-ES developers both compete and collaborate with each other as they carry different knowledge bases and have divergent developing interests (see Table 2.1 for details of ways in which they compete and collaborate), we propose that group faultlines—the hypothetical dividing lines in the composition of a group (Lau and Murnighan 1998; Thatcher and Patel 2012)—can be formed between the subgroups of SD and VD.

Specifically, the tension between SD and VD can rise and may turn into conflicts and competition when a strong faultline is formed and activated, which significantly reduces the performance of OS-ES projects. Researchers have found that group faultlines can affect the group process (e.g., team conflict or cohesion), performance (e.g., OS-ES release performance), and its affective outcomes (e.g., popularity among developers on GitHub; Lau and Murnighan 2005; Thatcher and Patel 2011; Thatcher and Patel 2012). For instance, Thatcher and Patel (2011) identified that the stronger the group faultlines, the more likely the team members would be involved in relationship conflict, task conflict, and lower team cohesion. More importantly, they noted that strong and active group faultlines directly reduce team performance and team satisfaction (Thatcher and Patel 2011). Therefore, corporate-communal representation enabled group faultline holds important implications on OS-ES release performance.

Additionally, the evenness of corporate-communal representation (i.e., subgroup size of VD and SD) is positively correlated with both stronger and more distant faultlines (Lau and Murnighan

1998; Thatcher and Patel 2012). Given the inherent tension between SD and VD within the OS-ES development (Teigland et al. 2014), it is important to consider the implications of group faultlines in the form of corporate-communal representation, as the evenness of subgroups is likely to reflect the level of representation between corporation (i.e., SD) and community (i.e., VD) and have a direct influence on the potential faultline strength (Hart and Van Vugt 2006; Thatcher and Patel 2012). In turn, strong faultlines enabled by the evenness of corporate-communal representation in OS-ES development would significantly reduce OS-ES performance (Thatcher and Patel 2011). For example, there would be less time and effort spent on meeting the OS-ES goals if these critical resources were allocated to bridge the chasm created by a strong faultline between SD and VD (Li and Hambrick 2005). The SD and VD become competitive with one another, and communications hindrances prevent essential knowledge transformation and assimilation (Lau and Murnighan 2005).

As the preceding discussion suggests, we anticipate that the relationship between corporate-communal representation and OS-ES release performance would be U-shaped. Although studies on faultlines have not directly hypothesized that the evenness of subgroups would affect the release performance in the context of OS-ES development, the evenness of subgroups has been used as a control variable (Thatcher et al. 2003). Based on the above discussion, the evenness between SD and VD would create a strong and active group faultline and result in intensified confictions between groups (Thatcher and Patel 2012). As stronger group faultlines are associated with a higher likelihood of relationship conflict, task conflict, and lower team cohesion (Lau and Murnighan 2005; Barkema and Shvyrkov 2007; Thatcher and Patel 2012), the negative effects of group faultlines on performance of SD and VD and satisfaction can be much stronger. Thus, we expect that a stronger group faultline between SD and VD can result in low

levels of OS-ES release performance. In contrast, the unevenness between SD and VD would be less likely to form a strong group faultline because of the imbalance in the distribution of power, resources, and abilities (Lau and Murnighan 1998). For example, the collaborations between a few SD and considerable VD would form a group cohesion instead of a faultline because the SD has less power and abilities to address the issues of OS-ES development and would align with VD's strategies and goals. Consequently, the OS-ES release performance will be high because of the added innovative design and new features. Thus, we hypothesize:

Hypothesis 1: Corporate-communal representation will have a U-shape relationship to OS-ES release performance.

4.3 Impact of Corporate-communal Contribution on OS-ES Release Performance

Corporate-communal contribution indicates the evenness in representation of OS-ES tasks committed by corporation and community developers (Di Tullio and Staples 2013; Germonprez et al. 2017; Maruping et al. 2019). As developers often make one-off contributions to the open source community (Pham et al. 2013; Pinto et al. 2016; Zhou and Mockus 2012), developer contribution becomes critical in large-scale and complex OSS development, and it is because these large-scale and complex OSS projects (e.g., OS-ES) require a continuous contribution enabled by robust and reliable developer contribution. Previous research has identified that developer contribution is positively related to OSS project performance (Maruping et al. 2019). However, it is common for many OSS developers to leave the community once their needs are met (Shah 2006; Von Krogh et al. 2012). Previous research has noted that the open source community always faces significant challenges in attracting sufficient developer contributions (Maruping et al. 2019). Previous literature identifies three main reasons for this situation. First, the low exit barrier makes it relatively easy for members to leave or stop contributing to the

community (Ren et al. 2016). Second, the wide range of available OSS projects makes it difficult to retain developers' interests over time (Robles and Gonzalez-Barahona 2006). It might not be an issue for those uncomplicated OSS projects; however, it becomes critical for OS-ES projects because the large-scale and complex OS-ES development requires developers to contribute consistently to the OS-ES community. Third, lack of contractual employment relationships between the projects and the OSS developers (Seidel and Stewart 2011). OSS developers constitute an unstable development resource in such an environment as they can stop contributing or leave the project anytime (Von Krogh et al. 2012). Thus far, extant studies focus on attracting developer contribution and how the developer contribution can influence their contributions (Maruping et al. 2019). Building upon the previous research on developer contribution, we elaborate on the differences between corporate and communal contributions in OS-ES development and examine the various impacts of corporate-communal contribution on OS-ES release performance.

As noted earlier, SD and VD collaboratively construct the OS-ES systems. This collaboration implies, on the one hand, that OS-ES projects have the resources and capabilities to successfully complete a wide range of tasks; on the other hand, both SD and VD should be committed to and responsible for a certain amount of activities in OS-ES development. As developer contributions are distributed between SD and VD, it is vital to recognize the differences between corporate (i.e., SD committed tasks) and communal contributions (i.e., VD committed tasks). First, corporate contribution reflects that SD are committing tasks for OS-ES development by acting in the corporation's interests rather than in the best interests of the open source community (Hurwicz 2008; Chen et al. 2021). From time to time, corporate contribution has increased development effectiveness and system quality through established development processes and

standards (Germonprez et al. 2012; Crowston et al. 2008). However, the alignment of the corporation's strategy may disadvantage other participants, alienate those who are not being consulted, and erode the sustainability of the OS-ES community (Crowston et al. 2008; Chen et al. 2021). Second, communal contribution indicates that the VD are taking responsibility for OS-ES development by acting in the open source community's interests (Cheibub et al. 2010; Chen et al. 2021). The communal contribution creates an environment for innovation where VD can better represent their perspectives, enhance their influences, and protect their interests in OS-ES project. However, this friendly development environment can also distract developers' focus and reduce the overall effectiveness (Chen et al. 2021).

Given the inherent tension between the committed tasks by SD and VD in OS-ES development, we believe that the evenness of corporate-communal contribution holds important implications for OS-ES development. We may spot a weak group faultline between SD and VD according to corporate-communal representation (e.g., one SD and ten VD will form a weak group faultline). However, we may identify a strong group faultline in the same situation based on the evenness of corporate-communal contribution (e.g., one SD and ten VD committed the same amount of contributions, respectively). Broadly, the group faultlines theory posits the hypothetical dividing lines in the composition of a group (Lau and Murnighan 1998; Thatcher and Patel 2012). As developers generally do not interact face-to-face in OS-ES development (Seidel and Stewart 2011; Stewart and Gosain 2006), the strength of the group faultline between SD and VD can be better reflected through the evenness of corporate-communal contribution. Consistent with group faultlines theory (Lau and Murnighan 1998), we believe that the evenness of corporate-communal contribution plays a similar role in influencing the strength of group faultlines (Hart and Van Vugt 2006; Thatcher and Patel 2012). In such a way, the evenness of corporate-

communal contribution is correlated with more distant and stronger faultlines between SD and VD, which would reduce OS-ES performance significantly. For example, the competition between prioritizing a corporation's strategy and embracing the open source community's innovative insights would be intensive, with a strong faultline activated by the evenness of corporate-communal contribution. The competition will consume lots of critical resources such as time and effort and result in low OS-ES release performance.

The preceding arguments suggest that the relationship between corporate-communal contribution and OS-ES release performance would be U-shaped. The evenness of corporate-communal contribution would create a strong group faultline and result in strategic competition and conflicts between corporation and community. Consistent with the previous research on the evenness of sub-group size (Thatcher and Patel 2012), the strong group faultline formed and activated by the evenness of corporate-communal contribution will cause strategic conflicts in the form of opposition and debates regarding the OS-ES specific task-related goals. Thus, we expect that the evenness of contribution is likely to cause low levels of OS-ES release performance. On the contrary, the unevenness of corporate-communal contribution would be less likely to form a strong group faultline because of the imbalance in the distribution of power and abilities (Lau and Murnighan 1998). As such, SD's power and ability to influence the strategic goal of the OS-ES project (e.g., align the OS-ES strategic goal with the sponsoring corporation) decreases with fewer SD contributions. There is a higher likelihood that the objectives of the OS-ES project will be aligned with the open source community's interests due to VD having greater power and ability to influence the project via performing a higher number of contributions. Thus, the unevenness of contribution (i.e., few corporate contributions, many communal contributions) can result in a higher OS-ES release performance driven by innovative functions. Similarly,

another form of the unevenness of contribution (i.e., few communal contributions, many corporate contributions) can result in a higher OS-ES release performance driven by improved system quality (e.g., sustainability and reliability) as the objectives of the OS-ES project will be aligned with corporation's interests. Therefore, we predicted as follows:

Hypothesis 2: Corporate-communal contribution will have a U-shape relationship to OS-ES release performance.

4.4 Impact of Corporate-communal responsiveness on OS-ES Release Performance

Corporate-communal responsiveness is defined as the evenness in representation of OS-ES responses offered by corporation and community developers (Zhang et al. 2013; Steinmacher et al. 2015; Zhou and Mockus 2015). Different from the traditional closed source ES development, it is important for OS-ES projects to successfully coordinate the contributions of thousands of distributed developers to maintain and further evolve their software (Shaikh and Henfridsson 2017). Previous research has so far largely focused on how the major tasks of OSS development are coordinated and identified various coordination mechanisms such as hierarchical and horizontal coordination mechanisms (Barley and Kunda 2001; Lakhani and Von Hippel 2003; Von Hippel and Von Krogh 2003; Fleming and Waguespack 2007; O'Mahony and Ferraro 2007). The sponsoring corporation applies the hierarchical coordination mechanism through an established structure (Lakhani and Von Hippel 2003; Dahlander and O'Mahony 2011; Lindberg et al. 2016). In contrast, the horizontal coordination mechanism is realized by the open source community members via appointing a subset of their peers to serve in coordinating or governing roles that offer horizontal authority over tasks, but not necessarily over individuals (Hall 1996). In OS-ES development, we refer to hierarchical coordination as OS-ES corporate responsiveness and horizontal coordination as OS-ES communal responsiveness. Considering the involvement

of the sponsoring organization (Daniel et al. 2018; Maruping et al. 2019), corporate-communal responsiveness is critical to the success of OS-ES development.

Prior literature has paid relatively little attention to the differences between corporate and communal responsiveness (De Noni et al. 2011; Markus, 2007; O'Mahony, 2007). In this study, we elaborate on the distinction between OS-ES corporation and community responses and test the impacts of different levels of corporate-communal responsiveness on OS-ES release performance. In OS-ES development, the corporate responses are offered by SD, and the communal responses are realized through VD. On the one hand, corporate responses are efficiency-oriented (Xue et al. 2011; Chen et al. 2021). For example, corporate responses are informationally efficient—they economize on the costs of communication and allow an efficient decision-making process through exclusive control of sponsoring corporations (Chen et al. 2021). On the other hand, communal responses are compatibility-oriented (Chen et al. 2021). For instance, communal responses are incentive-compatible—they enable all stakeholders to realize their own interests and preferred outcomes (Chen et al. 2021) through improved responses (Xue et al. 2011), which can boost innovative outcomes.

Considering the significant distinction between corporation and community responses, we believe that the evenness of corporate-communal responsiveness holds important implications in OS-ES development. Draw on group faultlines theory, we propose that the strength of group faultlines between SD and VD can be also reflected through the evenness of corporate-communal responsiveness. Similarly, we believe that the evenness of corporate-communal responsiveness positively influences the strength of the group faultline (Hart and Van Vugt 2006; Thatcher and Patel 2012). In such a way, the strong faultline reflected by the evenness of corporate-communal responsiveness would significantly reduce OS-ES release performance. To

elaborate the theoretical mechanisms linking the evenness of corporate-communal responsiveness and performance, we leverage the previous research on developer uncertainty (Maruping et al. 2019). Our explanation is that as the strength of the faultline increases (i.e., the evenness of responsiveness increases), developer uncertainty rises with the intensified conflicts between SD and VD (e.g., developers have high uncertainties about the objective of OS-ES project) and, consequently, the OS-ES release performance decreases because of a lower level of developers code input.

Taken together, we believe that the relationship between corporate-communal responsiveness and OS-ES release performance would be U-shaped. Considering, once again, the research on the evenness of sub-group size (Thatcher and Patel 2012), the strong faultline formed and activated by the evenness of corporate-communal responsiveness will cause informational conflicts between corporation and open-source community (i.e., efficiency-oriented vs. compatibility-oriented coordination) and result in low OS-ES release performance. In contrast, the unevenness of corporate-communal responsiveness would be less likely to form a strong group faultline because of the imbalance in the distribution of influences and result in high OS-ES release performance. For example, the faultline between corporation and open source community will be weaker as SD perform the majority of responses of OS-ES project because the corporation's influence on OS-ES project is dominant through responses by SD. Thus, the unevenness of corporate-communal responsiveness will reduce developer uncertainty and align their inputs to the efficiency-oriented strategic goals. In turn, there is a higher likelihood of gaining better OS-ES release performance (e.g., system quality) driven by efficient responses. Hence, we predicted as follows:

Hypothesis 3: Corporate-communal responsiveness will have a U-shape relationship to OS-ES release performance.

4.5 Moderating Role of OS-ES Release Type

This study proposes that release types also influence OS-ES release performance. Classification of various releases is carried out based on the primary nature of their specific tasks. Previous research has suggested that OS-ES development mainly deals with two types of tasks: technical debt resolution and digital option realization (Rolland et al. 2018). Integrating the insights of prior research on digital debt and digital options and our understanding of OS-ES release notes, we distinguish OS-ES releases into consolidating and expanding releases. Previous research has suggested that release notes serve as a communication bridge among developers by carrying crucial information on software development (Bi et al. 2020). Through the release notes, we can determine that consolidating releases are mainly focused on knowledge modification in the form of resolving debt-oriented tasks such as fixing bugs, refactoring weak code, and upgrading system security (Singh et al. 2011; Huang et al. 2016; Bi et al. 2020). As debt-oriented tasks require a lower level of expertise and developers can use the fragmentary time to resolve most of the technical debt independently, developers typically require a relatively low level of knowledge sharing and goal alignment to deliver these consolidating releases. Similarly, we can also spot the expanding releases that primarily focus on knowledge addition in the form of realizing option-oriented tasks such as function expansion and improvement (Benaroch et al. 2006; Lankton and Luft 2008). As option-oriented tasks require a higher level of expertise and developers need to work on the task and communicate with others consistently, the involved developers need a relatively high level of knowledge sharing and goal alignment to deliver these

expanding releases. In short, the level of knowledge sharing and goal alignment that is required in expanding releases is higher than that required in consolidating releases.

As the preceding discussion suggests, we expected that the group faultline would not cause intensive conflict between SD and VD in delivering consolidating releases. Since the majority of tasks in consolidating releases are debt-oriented and their resolutions require a relatively low level of knowledge sharing and goal alignment, developers' interdependencies are minimized. In other words, developers generally can complete the assigned tasks independently without much communication and alignment in their interests. Thus, the group faultline will have a limited negative influence on OS-ES release performance with consolidating releases. This suggests that the u-shaped relationship between corporate-communal representation and OS-ES release performance is likely to be weaker with consolidating releases.

In contrast to consolidating releases, we expected that the group faultline would further escalate the conflict between SD and VD in delivering expanding releases. Since the majority of tasks in expanding releases are option-oriented and their realizations require a relatively high level of knowledge sharing and goal alignment, developers' interdependencies are maximized.

Specifically, developers need to consistently communicate with each other and, simultaneously, they should be motivated by shared goals and interests. So, the group faultline will have a fortified negative influence on OS-ES release performance with expanding releases. As a result, the u-shaped relationship between corporate-communal representation and OS-ES release performance is likely to be stronger with expanding releases. Therefore, we propose the following hypotheses:

Hypothesis 4: Release type moderates the U-shape relationship between corporate-communal representation and OS-ES release performance such that the corporate-communal representation has a significantly lower influence on the OS-ES release performance for consolidating releases than for expanding releases.

Hypothesis 5: Release type moderates the U-shape relationship between corporate-communal contribution and OS-ES release performance such that the corporate-communal contribution has a significantly lower influence on the OS-ES release performance for consolidating releases than for expanding releases.

Hypothesis 6: Release type moderates the U-shape relationship between corporate-communal responsiveness and OS-ES release performance such that the corporate-communal responsiveness has a significantly lower influence on the OS-ES release performance for consolidating releases than for expanding releases.

Chapter 5. Research Methods

5.1 Research Context

We seek to advance the theorizing of corporate-communal engagement and examine its impact on release performance in the context of OS-ES. As a leading example of complex OSS, OS-ES have become more prevalent in the industry and raised considerable academic research interest (Johansson and Sudzina 2008; Olson et al. 2018). For instance, many firms such as Home Depot and Toyota have started investing in OS-ES projects. Developing the software through voluntary contributions and supporting critical business activities in organizations, OS-ES represents large and complex OSS and a new solution for small businesses to perform their business operations. Furthermore, the OS-ES context is a joint research context characterizing the merits of both ES and OSS. First, open superposition is the central work mechanism of OS-ES development. It allows developers to collaborate independently through superposed tasks and incremental layering (Howison and Crowston 2014). Second, the complex nature of OS-ES requires a practical governance approach to ensure that the various developers are working together seamlessly. Third, OS-ES development relies on the collaboration of both VD and SD. For instance, the VD will support the OS-ES development with their innovative insights and potential voluntary contribution. Meanwhile, the SD drive the effective decision-making process through their exclusive control power over the OS-ES. Finally, the OS-ES development should be continuous. Many OSS projects, such as game apps, are not required for continuous development. However, the OS-ES development must continuously add new features and eliminate defects as the organizations essentially run their business on the OS-ES systems. In short, the OS-ES research context is a unique context that has not been thoroughly investigated.

5.2 Case Selection

Case selection is of utmost importance for examining a theory in a new context. Drawing on Yin (2009), the selected case should most likely illuminate our research questions. As such, the OS-ES project—SuiteCRM—was selected among several alternatives listed on GitHub, such as Vtiger, Yetiforce, Really Simple Systems, Odoo, OroCRM, X2CRM, and EspoCRM. To choose the Suite CRM, we compared several attributes of these OS-ES projects. Including the number of stars on GitHub, functionality similarity, lifecycle on GitHub, subscription approach and price, suitable for small businesses or not, and all the related information listed on GitHub (e.g., number of issues, commits, and pull requests). The case is especially attractive and representative (Yin 2009) because SuiteCRM is the #1 OS-CRM on the market and got the most stars on GitHub among similar projects. As the most successful OS-CRM, the selected case provides us the opportunity to draw valuable conclusions. Also, our unit of analysis is on the minor release level. Suite CRM provides us with sufficient data points, which will increase the likelihood of gaining rich insights and enough variance for data analysis. As such, the characteristics of the selected case positioned our research well for examining our research model and associated hypotheses.

5.3 Data and Sample

We conducted an exploratory study based on the top-performing OS-CRM project on the GitHub platform—SuiteCRM (<https://github.com/salesagility/SuiteCRM>) to examine the ways in which an OS-ES community coordinates voluntary developers and their contributions. The SuiteCRM is a corporate sponsored OS-ES project, and its sponsor is SalesAgility. Established in 2009, SalesAgility is a leading open-source software consultancy that focuses on providing exceptional customer relationship management solutions to organizations around the world (<https://salesagility.com/>). Since its first release in 2009, SuiteCRM has been downloaded over

800,000 times and its community has attracted over 90,000 developers and collaborators contribute to the project (Wikipedia of SuiteCRM). Building upon this, successful cases of small businesses using SuiteCRM across different industries, including Oneworld Accuracy (i.e., an independent external quality assessment provider for medical laboratories), Humanetics Innovative Solutions (i.e., a leading provider of precision test systems and sensor solutions), and Scottish Book Trust (i.e., a national charity that believes in reading and writing for pleasure has the power to change lives).

To test our hypotheses in the research model, we focused on three different datasets: 1) digital traces of "Pull Requests"; 2) historical dataset of "Star"; 3) digital traces of "Releases." In the following, we briefly introduce the relationships between our key concepts and the above three datasets. Firstly, we capture the corporate-communal engagement by targeting digital traces of "Pull Requests." The GitHub platform organizes development work around its "Pull Requests" (Lindberg et al. 2016). Between the opening of a pull request and a decision (e.g., merge or close) is made, there are a variety of activities that may occur, and the majority of activities (around 60%) are associated with engagement (e.g., core developers making decisions and explaining the problem by providing comments; Lindberg et al. 2016). Therefore, "Pull Requests" allow us to trace how the core developers and voluntary developers participate, commit, and coordinate the activities in OS-ES developments (See the operationalization details in section 6.2).

Secondly, from the three datasets described above, we also capture the *performance level* of a particular release by retrieving the historical dataset of "star." The number of stars is a classical measure of performance in a particular release period. Thirdly, we capture the *release types* (i.e., expanding and consolidating releases) by targeting digital traces of "Releases" of SuiteCRM. In

the following subsections, we elaborate on how we collected and coded the "Pull Requests," "Star," and "Releases" datasets. Finally, we capture the *task types* by digital traces of "Pull Requests." Each pull request is a distinct event that may focus on bug reports, code refactoring, feature implementation and enhancement, and question discussions. Combing the insights of previous research and our observations on the current dataset, we then distinguished the pull requests into two types: technical debt resolution and digital option realization (Rolland et al. 2018).

5.3.1 Data Collection and Coding – Pull Requests

We collected the digital traces of "Pull Requests" through the GitHub platform from October 16, 2013 (when the SuiteCRM Started on GitHub) until March 02, 2022. We collected multiple attributes of each pull request for retrieving key information in three perspectives: 1) Task information—the pull request content including pull request number, title, and description; 2) Developer information—the pull request participants including pull request author, decision-maker, and general participant; 3) Response information—the pull request conversations including action responses and comment responses made by SD and VD (See examples in Figure 5.1 & 5.2).

The screenshot displays a GitHub pull request interface for the SuiteCRM repository. At the top, the repository name 'salesagility/SuiteCRM' is visible, along with navigation links for Code, Issues (1.2k), Pull requests (301), Projects, Wiki, Security, and Insights. The pull request title is 'Fix Issue #6994: Update pollMonitoredInboxesAOP to double check that ...'. A 'Merged' badge is present, indicating the pull request has been accepted. The pull request was merged by 'mattlorimer' into the 'salesagility:hotfix-7.10.x' branch from the 'MikeyJC:fix/16994' branch on November 18, 2021. The pull request statistics show 0 conversations, 1 commit, 2 checks, and 1 file changed. The description section is highlighted with a red box and contains the following text:

...SugarFolder has been retrieved correctly.

Description

Fixing issue #6994 by updating the pollMonitoredInboxesAOP scheduled task (lines 605-611) to make sure it gets the SugarFolder correctly. First attempts using the InboundEmail->groupfolder_id value as normal, if that fails, then attempts using InboundEmail->id instead. If that fails as well, then the \$isGroupFolderExists value is not set to true.

Motivation and Context

There is a bug where if the value in the groupfolder_id does not match with a SugarFolder id, then the logs will end up being filled

The right sidebar shows the 'Contributor' information, including reviewers 'mattlorimer' and 'jack7anderson7', both with green checkmarks. The 'Assignees' section shows 'No one assigned'. The 'Labels' section shows 'Status:Assessed'. The 'Projects' section is empty.

Figure 5.1 Task and Developer Information Collected from Pull Requests of SuiteCRM

The screenshot shows a GitHub pull request titled "FIX #8873 - Silent Install - Load custom data #8878" which is marked as "Closed". It features three comments:

- Abuelodelanada** (Contributor, Author) commented on Oct 23, 2020: "Hi @Dillon-Brown @mattlorimer @cameronblaikie I changed my email address in github because I am working in other company, so I need to sign the CLA again, but when I got to the link the site auto-complete my personal data with my old email, and I cannot edit it.... Can you solve it?"
- mattlorimer** (Member) commented on Oct 26, 2020: "Hi @Abuelodelanada I think the issue is the commits themselves are linked to your old email address. The only solution, is to add you old email address to you account as an extra email address, or redo/edit the commits to link to your new email"
- Abuelodelanada** (Contributor, Author) commented on Oct 27, 2020: "Hi @Abuelodelanada I think the issue is the commits themselves are linked to your old email address. The only solution, is to add you old email address to you account as an extra email address, or redo/edit the commits to link to your new email"

Figure 5.2 Response Information Collected from Pull Requests of SuiteCRM

With the collected data, we 1) coded each pull request into different task types, such as resolving technical debt or realizing digital option; 2) distinguished core developers into SD or VD; 3) counted the number of responses made by each core developer (See Table 5 for a summary of data collection and coding).

Table 5.1 Data Collection & Coding of Pull Requests		
Key Information	Collected Data	Data Coding
Task Information	Title, Description, Pull Request Status and Label, Code Content, # of Commits, and # Files Changed.	Task Types: <ul style="list-style-type: none"> • Digital Option Realization • Technical Debt Resolution
Developer Information	Name and Role of Author, General Participants, and Core Developers.	Core Developer Types: <ul style="list-style-type: none"> • SD • VD
Response Information	Contents of Actions and Conversations, Action and Comment Time.	Responses of SD and VD: <ul style="list-style-type: none"> • # of SD Responses • # of VD Responses

To code the collected "Pull Requests" data, we follow our established coding process. First, we coded the task type based on *descriptive content* (e.g., pull request title, description, label), *code content*, and *numerical indicators* (e.g., number of commits and files changed). The *descriptive content* will directly indicate the pull request is either debt or option oriented. For example, the title of pull request # 34—*Fixed "main menu (Tab All) not with 2 column layout"*—indicates it focuses on reporting and fixing a bug, so we code it as *defect code technical debt*. By combining the insights of these descriptive content in pull request title, description, and label, we can distinguish most pull requests. However, some pull requests do not provide explicit and sufficient descriptive content to be distinguishable. With such pull requests, we need to integrate our understanding of the code content and numerical indicators to make final decisions. For

example, the title of pull request # 21 is “*QuickCRM 3.5*” and there is no additional description. Fortunately, the *code content* indicates that this pull request is upgrading SuiteCRM into a new version by improving many functions, such as Admin function improvement, label tab improvement, and security improvement. The *numerical indicators* also indicate the same conclusion. For instance, this pull request includes 34 commits, which means the workload is hefty (most pull requests only have one or two commits) and is more likely to be a digital option-oriented task. Similarly, this pull request imposes a vast influence on SuiteCRM indicated by 487 “file changed.” Combining the insights of descriptive content, code content, and numerical indicators, we, therefore, conclude that pull request # 21 is a “digital option” (see Figure 2 for the coding process).

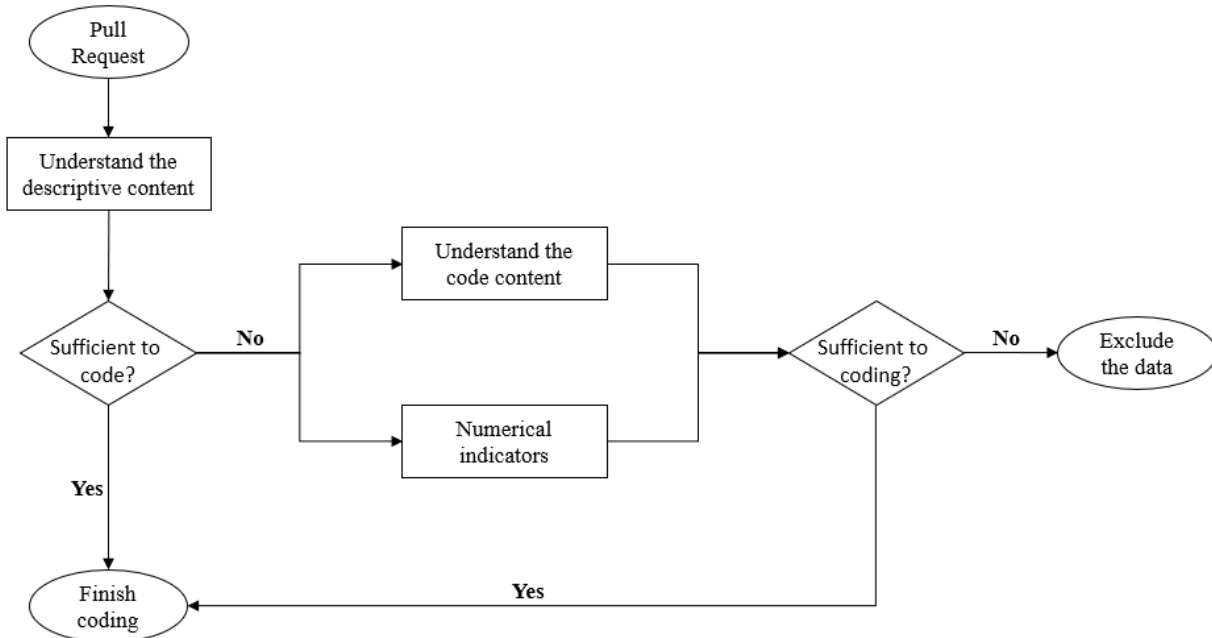


Figure 5.3 Coding Process

Second, we coded the developer types based on the assigned roles within a pull request. We can identify core developers in a pull request based on their activities. For example, core developers have the authority to close or merge a pull request. To further distinguish developers into SD and VD, we rely on their assigned roles. For example, SD are always labeled as “member,” while VD are labeled as “contributor,” “collaborator,” or not labeled (See examples in Figure 5.1 & 5.2). Finally, we coded the responses of SD and VD. In each pull request, we counted the number of actions and comments offered by each SD and VD as their total responses.

5.3.2 Data Collection – Star

We collected the historical dataset of “Star” through the GraphQL API embedded in GitHub. On GitHub, users and developers can star a project to indicate their satisfaction with the work (Chen et al. 2021). Most developers typically consider the number of stars before committing to a project (Borges and Valente 2018). Compared to REST API v3, the GraphQL API offers flexibility and the ability to define precisely the data you want to fetch (GitHub Docs). The GitHub platform archives the author and timestamp of each star since February 12, 2011. Using GraphQL query, we used the GraphQL API to retrieve the historical dataset of “Star” ordered by timestamp.

5.3.4 Data Collection and Coding – Releases

We collected the digital traces of “Releases” through the REST API v3 embedded in GitHub. Using a Python script, we retrieve the complete digital traces of all “Releases” for SuiteCRM. Based on the release notes within this dataset, we coded the releases into expanding and consolidating releases. Furthermore, we collected the commits data upon each release period. As

each release is built upon a certain amount of commits, we may code the releases according to their associated commits types in future work.

5.4 Construct Operationalization

5.4.1 Dependent Variable

Release Performance. On GitHub, developers and users can “star” a project to indicate their interests and satisfaction with this project (Chen et al. 2021). “Three out of four developers consider the number of stars before using or contributing to a GitHub project” (Borges and Valente 2018, p112). Therefore, we can capture the project performance through the number of the project’s stars. We collect the total stars accumulated between the two releases. The longer time between two releases, the more stars the release period earns. Thus, we used the total stars divided by the dates between two releases to get the average number of stars gained per day for one release as the measurement of the performance of this release.

5.4.2 Independent Variables

There are two types of developers in the OS-ES project development team: developers who come from the sponsoring corporation and developers who emerge from the open source community. Our unit of analysis is on the release level. We aggregate our data for each release. We count the number of SD and VD involved in each release. *Corporate-communal Representation* is measured by the percentage of corporation-sponsored developers in total developers.

$$\frac{\# \text{ of } SD}{\# \text{ of } (SD+VD)} \sim (0,1)$$

In addition to counting the number of various types of developers for a release, we also tracked the tasks each developer performed. *Corporate-communal Contribution* is measured by the percentage of committed tasks by corporation-sponsored developers.

$$\frac{\# \text{ of Comitted Tasks by } SD}{\# \text{ of Comitted Tasks by } (SD+VD)} \sim (0,1)$$

Our last independent variable is *Corporate-communal responsiveness*. This variable is related to community responsiveness. We tracked the contributor of each response associated with all the pull requests within each release. *Corporate-communal responsiveness* is measured by the percentage of responses offered by corporation-sponsored developers.

$$\frac{\# \text{ of Responses Offered by } SD}{\# \text{ of Responses Offered by } (SD+VD)} \sim (0,1)$$

5.4.3 Moderator

Release Type. In this study, we categorized each OS-ES release based on their release notes. A release is considered as expanding release if it focuses on knowledge addition, such as adding new features or improving existing features. On the contrary, we consider it a consolidating release if a release is more about knowledge modification, such as bug fixing, code refactoring, etc. Release type is a dummy variable, and the value is equal to 1 for an expanding release and 0 for a consolidating release.

5.4.4 Control Variables

Merge Status. Based on the quality of the pull request, it will be either merged into the new release or closed, which means the code won't be integrated into the next release. The merge

status is a dummy variable. Merge status equals 1 when the pull request is merged to the upstream branch and 0 when the pull request is closed without integrating into the branch.

Task Type. The OS-ES development is mainly concerned with obtaining an idealized balance between technical debt resolution and digital option realization. We also categorized each pull request into two task types: resolving technical debt or realizing digital options. The type of tasks may also impact the release performance, and thus during our analysis, we include the percentage of digital options in a certain release period as one control.

Added Code. We also control for how many new lines of code have been added in a certain release period.

Chapter 6. Results

6.1 Analysis and Results

6.1.1 Descriptive Statistics and Correlations

Descriptive Statistics. The descriptive statistics and correlations of the key variables are presented in Table 6.1. As shown in Table 6.1, SuiteCRM release performance as measured by GitHub stars is negatively correlated with corporate-communal engagement across its three types: corporate-communal representation ($r = -0.46, p < 0.001$), corporate-communal contribution ($r = -0.45, p < 0.001$), and corporate-communal responsiveness ($r = -0.31, p < 0.001$).

The mean values shown in Table 6.1 provide additional insight into our dataset. Within each release period, on average, the number of “stars” increases by 0.85 per day. In other words, SuiteCRM received an average of 0.85 stars within each release period. The mean values of three types of corporate-communal engagement range from 0.18 to 0.39. For corporate-communal representation (i.e., mean = 0.18), the mean value indicates that the average sub-group size of SD and VD is uneven in each release period (i.e., the average ratio of SD and VD is 18:82). For corporate-communal contribution (i.e., mean = 0.39), the mean value indicates that the average committed tasks by SD and VD is relatively even compared to corporate-communal representation in each release period (i.e., the average ratio of the committed tasks by SD and VD is 39:61). For corporate-communal responsiveness (i.e., mean = 0.23), the mean value indicates that the average responsiveness fulfilled by SD and VD is relatively even compared to corporate-communal representation in each release period (i.e., the average ratio of the responsiveness fulfilled by SD and VD is 23:77). Another insight is that each SD contributes

more than each VD in the form of contribution and responsiveness. On average, 18% of total developers (i.e., SD) have committed 39% of total contribution and 23% of total responses in each release period.

For the moderator, we coded releases into two types: consolidating releases = 0 and expanding releases = 1. The mean value of release type (i.e., mean = 0.39) indicates that 39% of the total releases are expanding releases. For the control variables, the mean value of merge status is 0.6, which indicates 60% of the total release has been merged to the main branch; the mean value of task types is 0.11, which indicates, on average, 11% of tasks are option-oriented for each release period. Finally, the average lines of code added in each release period is 2712 (i.e., mean = 2712.06).

Table 6.1 Descriptive Statistics and Correlations

	1	2	3	4	5	6	7	8
1. Release Performance	1.00							
2. Representation	-.46***	1.00						
3. Contribution	-.45***	.54***	1.00					
4. Responsiveness	-.31***	.58***	.74***	1.00				
5. Release Types	-.09	-.03	-.11	-.18 ⁺	1.00			
6. Merge Status	.46***	-.45***	-.44***	-.37***	-.13	1.00		
7. Task Type	.08	-.07	-.02	-.19*	.10	-.04	1.00	
8. Added Code	-.21*	.02	.02	-.04	.05	-.29**	.37***	1.00
Mean	.85	.18	.39	.23	.39	.60	.11	2712.06
Min	.00	.00	.00	.00	.00	.00	.00	1.00
Max	4.00	1.00	1.00	1.00	1.00	1.00	.67	30265.25
Standard Deviation	.62	.15	.16	.18	.49	.28	.12	5629.74

Note: $N = 124$ releases.

Standard errors are shown in parentheses.

*** Correlation is significant at the 0.001 level.

** Correlation is significant at the 0.01 level.

* Correlation is significant at the 0.05 level.

+ Correlation is significant at the 0.10 level.

6.1.2 Hypothesis Testing

To test our hypotheses, we performed several regression analyses. Additionally, the selected project—SuiteCRM—started in 2013 (i.e., nine years until now). Thus, we further include the fixed effects of the month to control the influence of time on the hypothesized relationships in this dissertation. All the results are shown in Table 6.2. The main effects of corporate-communal engagement are tested with Model 1, 2, and 3 for corporate-communal representation, corporate-communal contribution, and corporate-communal responsiveness. Model 4, 5, and 6 test the moderation effects of release type on the relationships between corporate-communal representation, contribution, and responsiveness on release performance. All three independent variables are mean-centered before computing the interactions to reduce multicollinearity (Aiken and West, 1991).

Table 6.2 Hypotheses Testing

	Project Release Performance (Star)					
	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6
Merge Status	.60** (.19)	.65** (.19)	.81*** (.20)	.57** (.19)	.60** (.20)	.72*** (.20)
Task Type	.65 (.41)	.62 (.42)	.39 (.44)	.80+ (.41)	.70 (.43)	.48 (.44)
Added Code	.00* (.00)	.00* (.00)	.00+ (.00)	.00* (.00)	.00* (.00)	.00* (.00)
Representation	-2.39*** (.33)			-1.79** (.61)		
Representation-Squared	3.53** (1.02)			2.03+ (1.17)		
Contribution	.60** (.19)	-1.04** (.33)			-1.23** (.42)	
Contribution-Squared		2.51* (1.06)			2.58* (1.21)	
Responsiveness			-.97* (.37)			-.82+ (.46)
Responsiveness-Squared			2.15* (.97)			1.45 (1.08)
Release Types				-.34** (.13)	-.15 (.13)	-.33* (.16)

Release Type × Representation							-2.28*	
							(.97)	
Release Type × Representation-Squared							8.69**	
							(3.08)	
Release Type × Contribution							.29	
							(.74)	
Release Type × Contribution-Squared							.48	
							(3.18)	
Release Type × Responsiveness								-.38
								(.74)
Release Type × Responsiveness-Squared								10.00*
								(4.91)
Fixed Effects of Month of Creation	YES	YES	YES	YES	YES	YES	YES	
Intercept	.26	.30	.20	.25	.36	.29	.29	
	(.22)	(.23)	(.23)	(.23)	(.23)	(.23)	(.23)	
Lowest VIF				1.38	1.34	1.35		
Highest VIF				4.70	3.31	3.06		
R ²	.45	.43	.37	.50	.44	.40		
Adjusted R ²	.37	.34	.28	.41	.33	.29		

Note: $N = 124$ releases.

Standard errors are shown in parentheses.

*** Correlation is significant at the 0.001 level.

** Correlation is significant at the 0.01 level.

* Correlation is significant at the 0.05 level.

+ Correlation is significant at the 0.10 level.

Hypothesis 1 predicted that corporate-communal representation has a U-shaped relationship with OS-ES release performance. We used Model 1 to test this relationship. The results in Table 6.2 (Model 1) indicate that the overall model explains 37% of the variance of OS-ES release performance. The coefficient of representation is negative and significant ($\beta = -2.39, p < 0.001$) and representation-squared is positive and significant ($\beta = 3.53, p < 0.01$). This indicates our Hypothesis 1 is supported. To visualize this relationship, we also plot the relationship between corporate-communal representation and release performance, further confirming a U-shaped relationship. As shown in Figure 6.1, low OS-ES release performance is attributed to the evenness of corporate-communal representation (i.e., the evenness of sub-group size, the middle part of Figure 6.1) because a strong group faultline is formed, and it can cause intensive conflicts

between the corporation and the open source community (Lau and Murnighan 1998; Thatcher and Patel 2012).

In contrast, OS-ES release performance is high with the unevenness of corporate-communal representation (i.e., the unevenness of sub-group size, left or right side of Figure 6.1). In such situations, the strength of the group faultline is weak, and conflicts between the corporation and the open source community are reduced, resulting in a higher OS-ES release performance. The figure below provides additional support for Hypothesis 1.

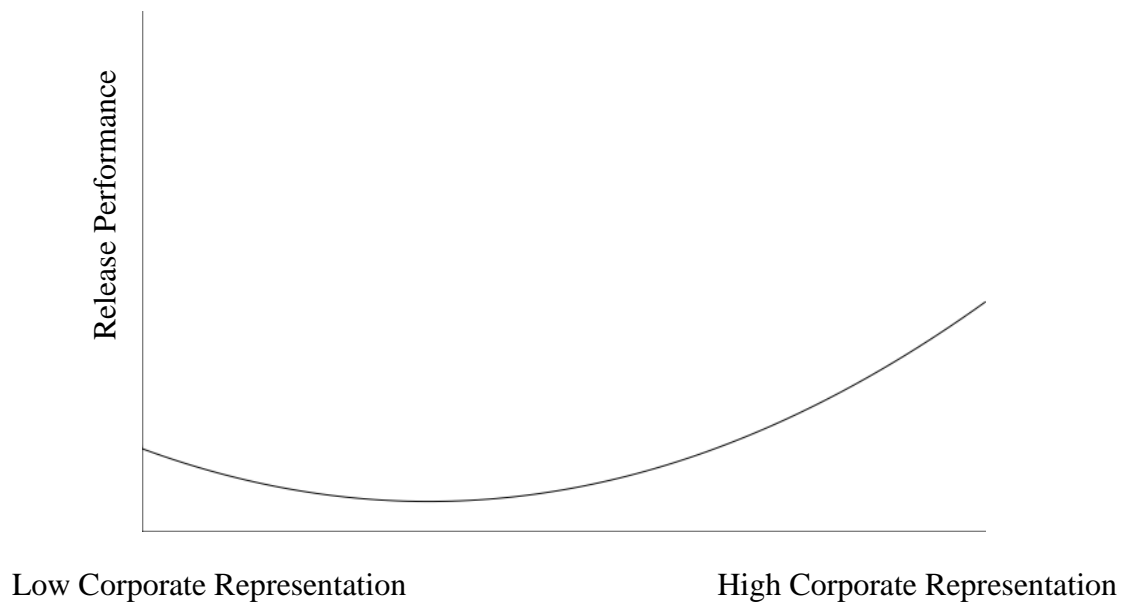


Figure 6.1 Relationship between Corporate-communal Representation and Release Performance (Model 1)

Hypothesis 2 predicted that corporate-communal contribution has a U-shaped relationship with OS-ES release performance. This hypothesis is tested with Model 2. The results in Table 6.2 (Model 2) indicate that the overall model explains 34% of the variance of release performance. The coefficient of contribution is negative and significant ($\beta = -1.04, p < 0.01$) and contribution-

squared is positive and significant ($\beta = 2.51, p < 0.05$). The results support Hypothesis 2. We also plot the relationship between corporate-communal contribution and release performance to visualize this relationship, further confirming a U-shaped relationship. As shown in Figure 6.2, a low OS-ES release performance is attributed to the evenness of corporate-communal contribution (i.e., the evenness of committed tasks by SD and VD, the middle part of Figure 6.2) because the conflicts between the corporation and the open source community rise as a strong group faultline is formed by the evenness of contribution. In contrast, OS-ES release performance is high with the unevenness of corporate and communal contribution (i.e., the unevenness of committed tasks by SD and VD, left or right side of Figure 6.2). In such situations, the strength of the group faultline is weak, and conflicts between the corporation and the open source community are reduced, resulting in a higher OS-ES release performance. The figure below provides additional support for Hypothesis 2.

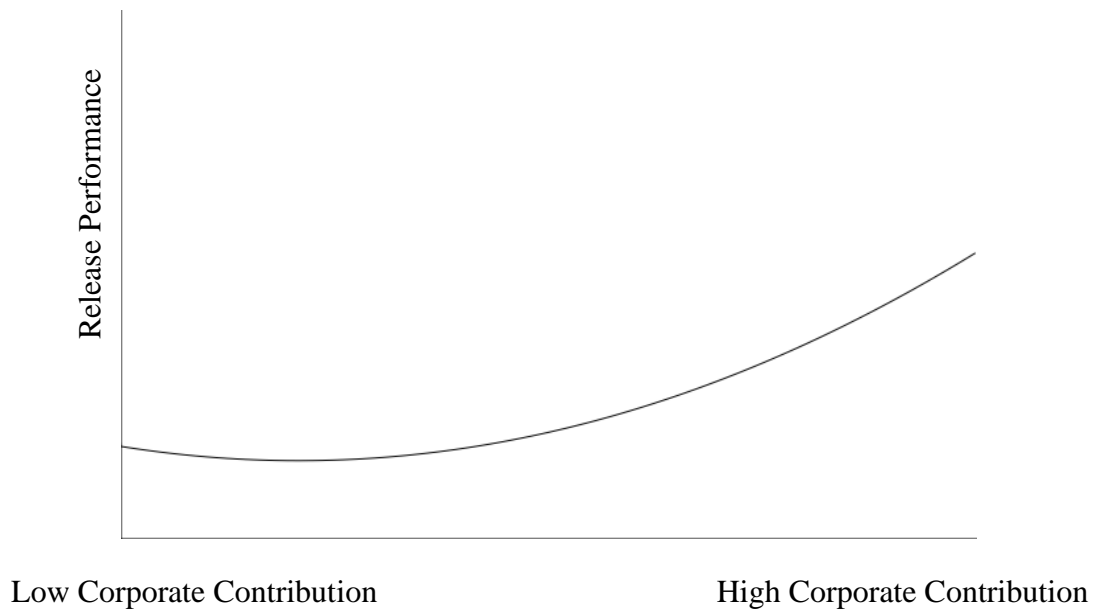


Figure 6.2 Relationship between Corporate-communal Contribution and Release Performance (Model 2)

Hypothesis 3 predicted that corporate-communal responsiveness has a U-shaped relationship with OS-ES release performance. We used Model 3 to test this relationship. The results in Table 6.2 (Model 3) indicate that the overall model explains 28% of the variance of release performance. The coefficient of responsiveness is negative and significant ($\beta = -0.97, p < 0.05$) and responsiveness-squared is positive and significant ($\beta = 2.15, p < 0.10$). This indicates our Hypothesis 3 is supported. To visualize this relationship, we plot the relationship between corporate-communal responsiveness and release performance, further confirming a U-shaped relationship. Consistent with previous explanations, the evenness of corporate-communal responsiveness leads to a lower OS-ES release performance and the unevenness of responsiveness fulfilled by SD and VD can gain higher OS-ES release performance as the conflicts reduced. The figure below provides additional support for Hypothesis 3.

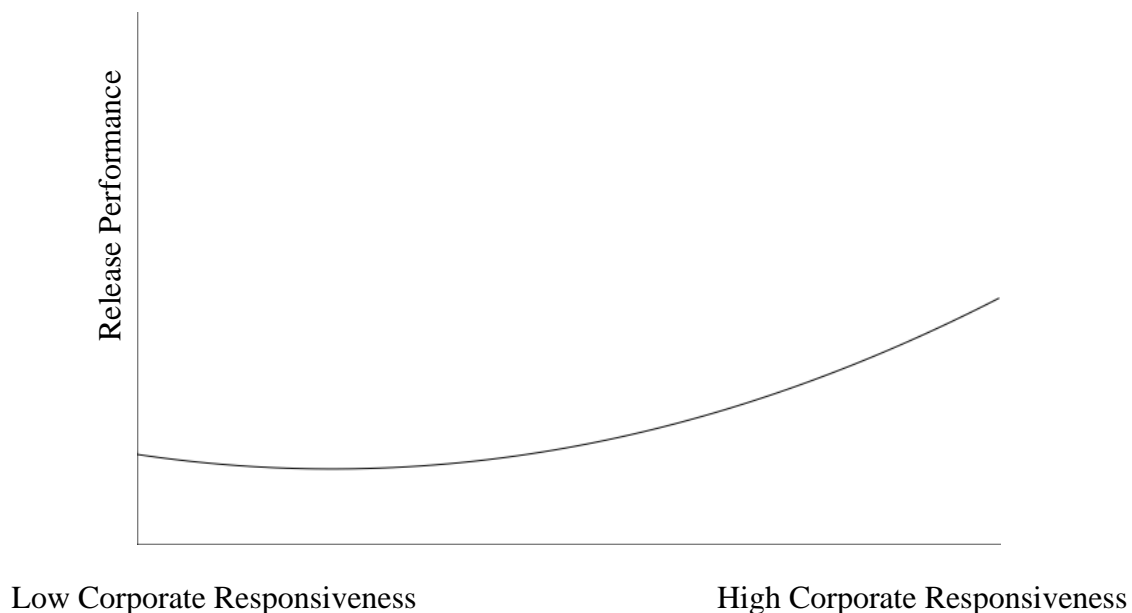


Figure 6.3 Relationship between Corporate-communal Responsiveness and Release Performance (Model 3)

Hypothesis 4 predicted that release type moderates the U-shaped relationship between corporate-communal representation and OS-ES release performance such that the corporate-communal representation has a significantly lower influence on the OS-ES performance for a consolidating release than an expanding release. This hypothesis is tested with Model 4. The results in Table 6.2 (Model 4) indicate that the overall model explains 41% of the variance of release performance. The coefficient of representation is negative and significant ($\beta = -1.79, p < 0.01$) and representation-squared is positive and significant ($\beta = 2.03, p < 0.10$). The coefficient of the interaction of release type and representation is negative and significant ($\beta = -2.28, p < 0.05$). The coefficient of the interaction of release type and representation-squared is positive and significant ($\beta = 8.69, p < 0.01$). The results indicate a significant moderation effect (Haans et al. 2015). This indicates that when the release is an expanding release, the curvilinear relationship between corporate-communal representation and release performance is stronger. These results provide support for Hypothesis 4 and suggest that release type moderated the relationship between corporate-communal representation and release performance. Figure 6.4 graphically illustrates the curvilinear relationship between corporate-communal representation and release performance and the moderating effect of release type. The moderating effect of release type on the curvilinear relationship is illustrated by the different levels of flatness of those curves. The curve for a consolidating release is flatter than the curve for an expanding release, indicating that the U-shaped relationship between corporate-communal representation and release performance is stronger for an expanding release than for a consolidating release. Figure 6.4 also support Hypothesis 4.

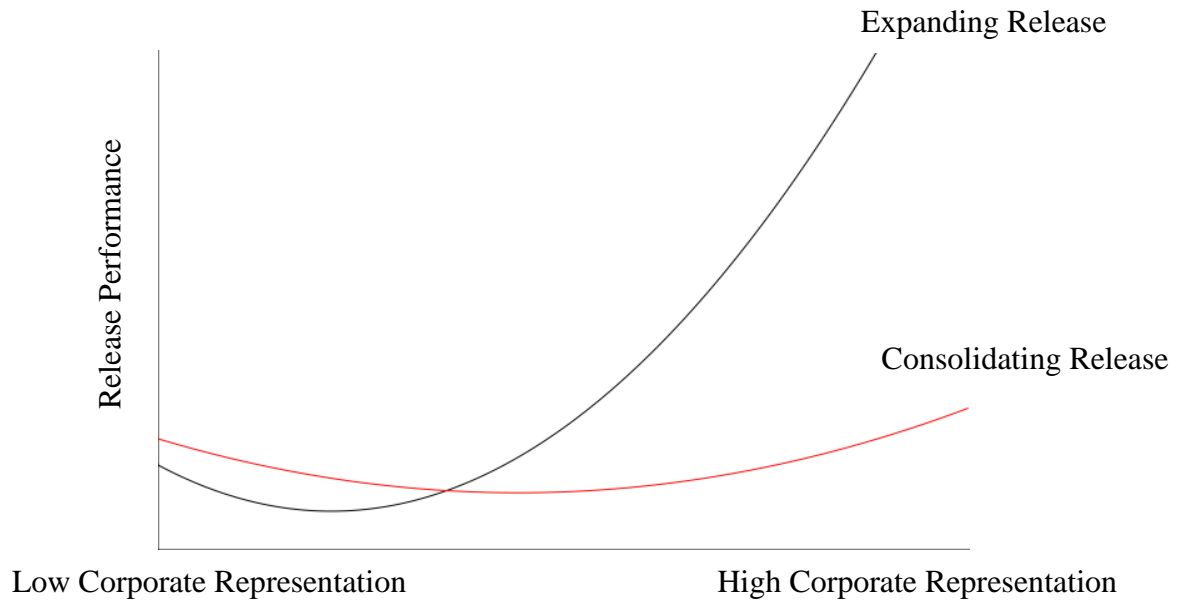


Figure 6.4 Interaction between Corporate-communal Representation and Release Type in Predicting Release Performance (Model 4)

Hypothesis 5 predicted that release type moderates the U-shaped relationship between corporate-communal contribution and OS-ES release performance such that the corporate-communal contribution has a significantly lower influence on the OS-ES performance for a consolidating release than an expanding release. This hypothesis is tested with Model 5. The results in Table 6.2 (Model 5) indicate that the overall model explains 33% of the variance in release performance. The coefficient of representation is negative and significant ($\beta = -1.23, p < 0.01$) and representation-squared is positive and significant ($\beta = 2.58, p < 0.05$). Although the main effect remains significant in Model 5, the coefficients of the interactions of release type, contribution, and contribution-squared are not significant. These results do not provide sufficient support for Hypothesis 5.

Hypothesis 6 predicted that release type moderates the U-shaped relationship between corporate-communal responsiveness and OS-ES release performance such that the corporate-communal

responsiveness has a significantly lower influence on the OS-ES performance for a consolidating release than an expanding release. This hypothesis is tested with Model 6. The results in Table 6.2 (Model 6) indicate that the overall model explains 29% of the variance in release performance. The coefficient of responsiveness is negative and significant ($\beta = -0.82, p < 0.10$). The coefficient of the responsiveness-squared is positive but not significant. The coefficient of the interaction of release type and responsiveness-squared is negative but not significant. The coefficient of the interaction of release type and responsiveness-squared is positive and significant ($\beta = 10.00, p < 0.05$). These results provide partial support for Hypothesis 6 and suggest that release type moderated the relationship between corporate-communal responsiveness and release performance. This indicates that when the release is an expanding release, the curvilinear relationship between corporate-communal responsiveness and release performance is stronger. Figure 6.5 graphically illustrates the curvilinear relationship between corporate-communal responsiveness and release performance and the moderating effect of release type. The moderating effect of release type on the curvilinear relationship is illustrated by the different levels of flatness of those curves. The curve for an expanding release is steeper than the curve for a consolidating release, indicating that the U-shaped relationship between corporate-communal representation and release performance is weaker for a consolidating release than for an expanding release. Figure 6.5 also support Hypothesis 6.

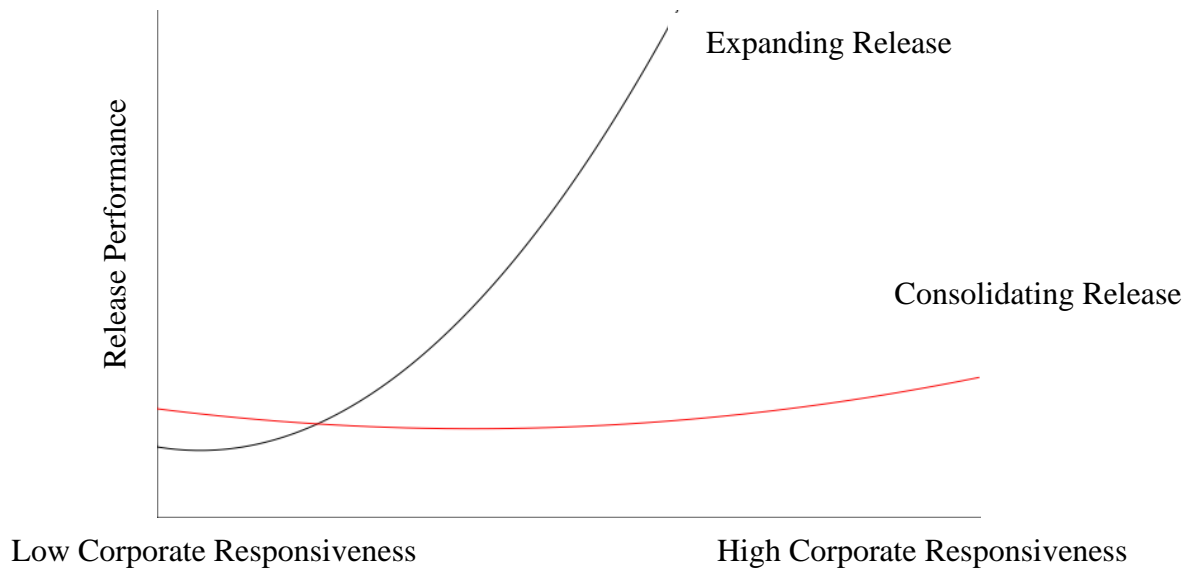


Figure 6.5 Interaction between Corporate-communal Responsiveness and Release Type in Predicting Release Performance (Model 6)

6.2 Supplementary Analyses for Robustness Tests

The results of the analysis provide support for the majority of the hypotheses. We conducted several additional analyses to ensure the robustness of our results. Firstly, we used an alternative measure for release performance (total number of downloads) as the dependent variable for the six models. Secondly, we separated the data based on release time into two sub-groups: early stages and late stages. We run six models using the two subsets of data. Finally, we used an alternative measure for developers as the independent variable for our six models. All the tables of the robustness test results are listed in the Appendix.

6.2.1 Test for Alternative Dependent Variable

In this robustness analysis, we tracked the number of total downloads for the SuiteCRM from sourceforge.net. Download numbers have commonly been used as a measure of a software's

performance. Additionally, download numbers can reflect release success from the perspectives of small businesses. As we use the number of “Stars” to reflect the popularity of OS-ES among developers, the download numbers can be used as an alternative measure of release success to reflect the OS-ES market performance. Considering the platform differences between Sourceforge and GitHub (e.g., the different sources of the data may generate some inconsistencies in the results), we did not use it in the primary analysis. However, we still find similar results for Model 1 and Model 4. This indicates the support of Hypotheses 1 and 4. The descriptive statistics and correlations of variables are listed in Table A.1. The regression analysis results using the alternative dependent variables are shown in Table A.2. The results of Model 1 indicate that the overall model explains 14% of the variance of release performance. The coefficient of representation is negative and significant ($\beta = -132.37, p < 0.01$) and representation-squared is positive and significant ($\beta = 225.89, p < 0.01$). This provides additional support for Hypothesis 1. The results of Model 4 show that the overall model explains 14% of the variance of release performance. The coefficient of interaction between release type and representation is negative and significant ($\beta = -97.43, p < 0.10$), and the coefficient of the interaction between release type and representation-squared is positive and significant ($\beta = 368.13, p < 0.10$). This indicates Hypothesis 4 is supported.

6.2.2 Test for Early and Late Stage

To understand how the different maturity stages of OS-ES impact our research analysis and results, we further test our research model by distinguishing the early and late development stages. In our database, the earliest release was launched on Oct 21, 2013, and the latest release was launched on March 02, 2022. The release time was calculated by counting the days from the earliest launch day. We took the median value of the release time (1183 days) and separated the

dataset into two subgroups: early stage and late stage. Each sub-sample includes 62 observations. The descriptive statistics and correlations of variables are listed in Table A.3 and Table A.4. Then, we ran our six models using the two datasets. The results are shown in Table A.5 and Table A.6. None of the hypotheses are supported while using the early-stage data. While using the late-stage data, we found similar results for all the main effects. The regression results support Hypothesis 1, 2, and 3. The results of Model 1 indicate that the overall model explains 25% of the variance of release performance. The coefficient of representation is negative and significant ($\beta = -2.52, p < 0.05$) and representation-squared is positive and significant ($\beta = 20.04, p < 0.01$). This provides additional support for Hypothesis 1. The results of Model 2 indicate that the overall model explains 32% of the variance of release performance. The coefficient of contribution is negative but not significant, and the contribution-squared is positive and significant ($\beta = 1.44, p < 0.10$). This indicates our Hypothesis 2 is supported. The results of Model 3 indicate that the overall model explains 10% of the variance of release performance. The coefficient of responsiveness is negative but not significant. The responsiveness-squared is positive and significant ($\beta = 7.54, p < 0.05$). This indicates Hypothesis 3 is partially supported.

6.2.3 Test for Alternative Independent Variable

Our primary analysis set the VD as voluntary core developers. Those developers have a clear label in the system, such as VD are labeled as “contributor” in GitHub. In this robustness analysis, we expand the boundary of VD by including peripheral developers as part of VD. Previous research has noted that peripheral developers hold important theoretical implications for OSS development (Ren et al. 2006). In this dissertation, we are interested in how peripheral developers can impact our research results. Thus, we added all the peripheral developers to the previous VD group in this robustness test. Using the new dataset, we ran our six models again.

We found the support for all of our main effects. Hypotheses 1, 2, and 3 are supported. The descriptive statistics and correlations of variables are listed in Table A.7. The regression analysis results using the alternative dependent variables are shown in Table A.8. The results of Model 1 indicate that the overall model explains 41% of the variance of release performance. The coefficient of representation is negative and significant ($\beta = -3.70, p < 0.001$) and representation-squared is positive and significant ($\beta = 11.24, p < 0.001$). This provides additional evidence for Hypothesis 1. The results of Model 2 indicate that the overall model explains 37% of the variance of release performance. The coefficient of contribution is negative ($\beta = -0.44$) but not significant, and the contribution-squared is positive and significant ($\beta = 7.61, p < 0.01$). This indicates Hypothesis 2 is partially supported. The results of Model 3 indicate that the overall model explains 29% of the variance of release performance. The coefficient of responsiveness is negative and significant ($\beta = -1.18, p < 0.01$), and the responsiveness-squared is positive and significant ($\beta = 4.58, p < 0.01$). This indicates Hypothesis 3 is supported.

Chapter 7. Discussion

This dissertation sought to examine corporate-communal engagement in the context of OS-ES. We were particularly interested in OS-ES because it represents the intersection of OSS and business, such as a business that wants to utilize OSS products. OS-ES combines the essential attributes of both OSS and ES. On the one hand, OS-ES possess the open nature of OSS that allows voluntary contributions and enables high visibility of peer communication among developers and users (Raymond 1999; Fitzgerald 2006). On the other hand, OS-ES inherit the large-scale and complex nature of ES that requires continuous development and management. Considering the uniqueness of OS-ES and the insights of literature on corporate-communal engagement, we proposed that corporate-communal engagement affects OS-ES release performance. Further, we proposed that the curvilinear relationship between corporate-communal engagement and OS-ES release performance would be stronger when the developers prepare to deliver an expanding release rather than a consolidating one. To accomplish this, we draw on group faultlines literature. We tested the proposed research model and found support for most hypotheses, as summarized in Table 7.1.

This empirical study found that the relationship between corporate-communal engagement and OS-ES release performance is curvilinear (i.e., u-shaped) based on the data collected from one of the best performing OS-ES on GitHub - SuiteCRM. Specifically, we found that the relationships between three types of corporate-communal engagement—representation, contribution, and responsiveness—and OS-ES release performance are u-shaped (See Figures 6.1, 6.2, and 6.3). We found that the evenness of corporate-communal representation—represented by the group size of SD and VD—resulted in lower OS-ES release performance because a strong group faultline exists (Lau and Murnighan 1998; Thatcher and Patel 2012). It could cause intensive

conflicts between the corporation and the open source community. At the same time, the unevenness of corporate-communal representation resulted in higher OS-ES release performance. We found similar results with corporate-communal contribution and corporate-communal responsiveness. The evenness of contribution and responsiveness resulted in lower performance, and the unevenness of contribution and responsiveness resulted in higher performance.

Further, we found that various release types moderate this curvilinear relationship. The u-shaped relationship is likely to be stronger in expanding releases than in consolidating releases (See Figures 6.4 and 6.5). Drawing on group faultline theory, we expected that the group faultline would not cause intensive conflict between the corporation and the open source community in delivering consolidating releases. When it comes to consolidating releases, the tasks are usually focused on debt-oriented and require minimal knowledge sharing. This minimizes the need for interdependencies. In contrast, we believed that the group faultline would further escalate the conflict between SD and VD in delivering expanding releases because developers should be motivated by shared goals and common interests as option-oriented tasks require a relatively high level of knowledge sharing and goal alignment. Contrary to expectations, H5—posited the relationship between corporate-communal contribution and OS-ES release performance—did not receive support from our hypothesis testing results (See Table 7.1). According to the results of H5, the impact of release type on the relationship between corporate-communal contribution and OS-ES release performance is insignificant. Possibly, the insignificant results are due to our relatively small sample size (i.e., 124 releases are included in the analysis). In the following, we discuss the theoretical and practical implications of our research while also acknowledging some limitations and outlining directions for future research.

Table 7.1 Summary of Hypotheses and Results

	Hypotheses	Results
H1	<i>Corporate-communal representation will have a U-shape relationship to OS-ES release performance.</i>	Supported
H2	<i>Corporate-communal contribution will have a U-shape relationship to OS-ES release performance.</i>	Supported
H3	<i>Corporate-communal responsiveness will have a U-shape relationship to OS-ES release performance.</i>	Supported
H4	<i>Release type moderates the U-shaped relationship between corporate-communal representation and OS-ES release performance such that the corporate-communal representation has a significantly lower influence on the OS-ES release performance for consolidating releases than for expanding releases.</i>	Supported
H5	<i>Release type moderates the U-shaped relationship between corporate-communal contribution and OS-ES release performance such that the corporate-communal contribution has a significantly lower influence on the OS-ES release performance for consolidating releases than for expanding releases.</i>	Not Supported
H6	<i>Release type moderates the U-shaped relationship between corporate-communal responsiveness and OS-ES release performance such that the corporate-communal responsiveness has a significantly lower influence on the OS-ES release performance for consolidating releases than for expanding releases.</i>	Supported

7.1 Theoretical Contributions

This dissertation aims to provide an essential contribution to the ES literature by advancing a theoretical framework of how corporate-communal engagement (Kelty 2013; Germonprez et al. 2017; Table 3.1) and release types (Table 3.2) are implicated in ES development in relation to the open-source business model. Raymond's (1999) research on software development with the bazaar model provides the foundation for developing large-scale ES by utilizing a community approach—OS-ES development. Although previous literature has shown the feasibility of the

OS-ES development model (Olson et al. 2018), limited research has focused on how to effectively develop ES with the OSS model. Considering the large market potential of OS-ES and insufficient knowledge of OS-ES development, it is necessary for both software engineers and IS scholars to obtain a better understanding of OS-ES development. Inspired by Germonprez et al.'s (2017) research on corporate-communal engagement and our observations on over 1000 OS-ES projects—most successful OS-ES projects are sponsored and supported by one or more corporations, we sought to understand the OS-ES development from the theoretical framework of corporate-communal engagement in the OS-ES development. The framework focuses on how corporate-communal engagement can either enable or inhibit the success of OS-ES development. Further, this framework identifies the moderation effect of release type on the above relationship. As such, the framework is premised on the need to challenge and go beyond a simplistic positive relationship between developers' collaboration and software development performance. This relationship is complicated by the fact that the evenness of engagement between SD and VD can result in lower OS-ES release performance, and the unevenness of engagement between these two groups can lead to higher OS-ES release performance.

This dissertation also contributes to the OSS literature in two ways. First, our study advances the existing literature on OSS development (Howison and Crowston 2014; Lindberg et al. 2016) because it takes a significant step in establishing the role of sponsoring organization and open-source community in the large-scale OSS development model (i.e., OS-ES development). A fundamental motivating factor for this research was that, increasingly, small businesses seek to adopt these free and open OS-ES to support their day-to-day business activities (Boulanger 2005; Olson et al. 2018) and those small businesses face challenges in implementing the OS-ES due to insufficient maintenance and technical support. Yet the OSS literature lacked the theoretical and

empirical insight into the OS-ES development. Thus, we contribute to the OSS literature by providing insight into the large-scale OS-ES development. As such, the engagement level of both corporation and open-source community plays an important role in ensuring sustainability and innovation of OS-ES. Second, our research advances the literature surrounding sponsoring organization in OSS development (Fitzgerald 2006, Stewart et al. 2006, Spaeth et al. 2015) by enhancing our understanding of the impact of corporate-communal engagement on OS-ES release performance. Particularly, we develop the measurements of corporate-communal engagement based on its three types (i.e., corporate-communal representation, contribution, and responsiveness) and the data retrieved from a selected case—SuiteCRM—hosted on Github. With the measurements of corporate-communal engagement, we found that the relationship between corporate-communal engagement and OS-ES release performance is curvilinear (i.e., U-shaped). These findings add insight to the inconsistent findings in prior research on corporate involvement in OSS. Previous literature revealed a complex relationship between the sponsoring corporation and the open source community (Mockus et al. 2002; West and O’Mahony 2008; Priharsari et al. 2020). Their relationship could be either beneficial or detrimental. For example, corporate-communal engagements can be beneficial as SD and VD share common goals and communicate sufficiently (Dahlander and Wallin 2006; Shah 2006; Murray and O’Mahony 2007; West and O’Mahony 2008). However, in contrast to the positive effects of corporate-communal engagements, it can cause competition between the corporation of the open source community (West and O’Mahony 2008; Giovacchini 2017; Priharsari et al. 2020). This is because both SD and VD are focused on their own goals and benefits. Thus far, we have clarified the relationship between corporate-communal engagement and OS-ES release performance based on the established measurement of corporate-communal engagement. For example, our

results show that the evenness of corporate-communal engagement resulted in lower OS-ES release performance, and this curvilinear relationship will be more salient for a consolidating release than for an expanding release.

Further, this dissertation applies the well-defined theory—group faultlines theory—in explaining how to successfully develop OS-ES with corporate-communal engagement (i.e., representation, contribution, and responsiveness) and release type contingencies. As a step toward this goal, we explain that stronger group faultlines between SD and VD can be formed and activated by the evenness of corporate-communal engagement in the form of representation, contribution, and responsiveness. With these stronger group faultlines enabled by the evenness of corporate-communal engagement, the group outcomes (i.e., OS-ES release performance) will decline significantly because of the intensified conflicts between SD and VD. Additionally, we identify the group faultlines between SD and VD will become more active for an expanding release than for a consolidating release—the relationship between corporate-communal engagement and OS-ES release performance is more salient for an expanding release than for a consolidating release.

Finally, our work proposed a new perspective on understanding the software releases. Inspired by previous research on digital options and debt (Rolland et al. 2018), we identify the possibility of categorizing various releases into different types, such as consolidating and expanding releases. Specifically, we map out the relationship between release type and group faultlines between SD and VD by identifying the moderation role of release type. For instance, we find that corporate-communal engagement (i.e., corporate-communal representation, contribution, and responsiveness) has a significantly lower influence on the OS-ES release performance for consolidating releases than for expanding releases.

7.2 Practical Contributions

This research holds important practical implications for corporation managers, OSS community leaders, and small businesses. First, previous research identified the feasibility of the OS-ES development model (Olson et al. 2018) and provided insight into the corporate-communal engagement (Germonprez et al. 2017), leaving the impacts of corporate-communal engagement on OS-ES release performance unclear. Our results suggest that the evenness of corporate-communal engagement undermines the OS-ES development represented by OS-ES release performance due to the conflict raised by the strong group faultline. Thus, corporation managers may facilitate their engagement with the open source community in two ways: 1) control the OS-ES project by dominating engagement; 2) let the voluntary developers control the OS-ES project by allowing them to influence the project objectives and strategies.

Second, the results of this dissertation also provide insights for OSS developers and leaders who seek to contribute to and influence the OS-ES project. While many studies attempt to understand the motivation of OSS voluntary developers (Roberts et al. 2006; Von Krogh et al. 2012), few studies consider the engagement level of OSS developers can impact the OSS performance, especially in the context of OS-ES. Our results suggest that open source developers and leaders may engage more in the OS-ES project when the development environment is relatively open and friendly. In contrast, developers from the OSS community may want to follow the sponsoring corporation and provide complementary support for development when the sponsoring corporation applies many restrictions and established standards to the OS-ES development.

Finally, small businesses may find the results of this research helpful when they seek to adopt OS-ES. Previous research has noted that the OS-ES market is not mature enough (Olson et al.

2018), leaving the question of how to choose a proper OS-ES unresolved. With our findings, small businesses may want to select those OS-ES supported by one or more sponsoring corporations. Building upon this, small businesses may prefer OS-ES projects mainly controlled by sponsoring corporations if they need high quality in the form of sustainability and better maintenance. Small businesses may also adopt OS-ES projects driven by OSS voluntary developers if they require innovative functions and customization flexibilities.

7.3 Limitations and Future Research

We acknowledge some limitations of the study that also offer useful opportunities for future work. First, we included one dependent variable (i.e., OS-ES release performance) in our study. Although we measured the OS-ES release performance through two perspectives: number of star and number of download, our understanding on the effects and outcomes of corporate-communal engagement and release type contingencies is still limited with one dependent variable. Future research may consider including other dependent variables to improve our understanding on how corporate-communal engagement and release type impact the OS-ES development, such as developers input and contribution level. The contributions and inputs from OS-ES developers can reflect the value, influence, and success of OS-ES project. As developers input and contribution are critical to OSS project success (Maruping et al. 2019), it is necessary to test whether corporate-communal engagement can attract more developers contribution with different release types.

Second, our conclusion is drawn on a longitudinal dataset across nine years of a single OS-ES project—SuiteCRM . Using a single case study raises the question of generalizability. Also, we selected the best OS-ES hosted on GitHub. The survival bias also limits the generalizability of

our theoretical explanation for the phenomenon. Thus, a potential direction for future research is to test our theory with a broader dataset that includes multiple OS-ES projects.

Third, the SuiteCRM is sponsored by a single company—Salesagility. However, there are some OS-ES projects are sponsored by multiple corporations. In the future study, we may also address the difference between the sponsoring power in the form of the number of sponsoring corporations.

Fourth, the corporate-communal engagement in this study is primarily about the pull requests. The technic of “Pull Requests” is an important approach for developers to collaborate. There are alternative ways for developers to collaborate, such as the technic of “Issues” and the corporation’s forum. Future research may want to examine how OS-ES developers collaborate through “Issues” and the corporation’s forum (e.g., our pilot analysis on the Issue dataset indicates a linear relationship between corporate-communal engagement and OS-ES release performance).

Finally, we run a series of OLS regressions with our established research model. Our three types of corporate-communal engagement may have endogeneity issues. To avoid these issues and test the mutual influence of our variables, future research may consider using the panel vector autoregression (PVAR) model to test the mutual relationships between our variables, as all variables are assumed to be endogenous and interdependent with the PVAR model.

Chapter 8. Conclusion

The OS-ES development presents both benefits and challenges for both corporations and open source communities. This dissertation takes preliminary steps to understand how corporate-communal engagement impacts OS-ES development in the form of release performance.

Building upon the measurement of corporate-communal engagement and group faultline theory, this dissertation suggests that the evenness of corporate-communal engagement would reduce the OS-ES release performance significantly. Our work also suggests that the expanding release would strengthen the curvilinear relationship (i.e., U-shaped), while the consolidating release would attenuate this relationship. This dissertation holds important implications for both the academic and software development industry. It aims to understand the OS-ES development through the perspective of the collaboration between corporations and open source communities.

References

- Abebe, S.L., Ali, N. and Hassan, A.E., 2016. "An empirical study of software release notes," *Empirical Software Engineering*, 21(3), pp.1107-1142.
- Aiken, L., and West, S. 1991. "Multiple regression: Testing and interpreting interactions." *Newbury Park, CA: Sage Publications*
- Barkema, H. G., and Shvyrkov, O., 2007. "Does top management team diversity promote or hamper foreign expansion?" *Strategic Management Journal*, 28(7), pp.663-680.
- Benaroch, M., 2002. "Managing information technology investment risk: A real options perspective," *Journal of Management Information Systems*, 19(2), pp.43-84.
- Benaroch, M., Lichtenstein, Y. and Robinson, K., 2006. "Real options in information technology risk management: An empirical validation of risk-option relationships," *MIS Quarterly*, pp.827-864.
- Bezrukova, K., Thatcher, S. M. B., and Jehn, K. A., 2007. "Group heterogeneity and faultlines: Comparing alignment and dispersion theories of group composition." In *K. J. Behfar & L. L. Thompson (Eds.), Conflict in organizational groups: New directions in theory and practice* (pp. 57–92). Evanston, IL: The Northwestern University Press.
- Bezrukova, K., Jehn, K. A., Zanutto, E. L., and Thatcher, S. M., 2009. "Do workgroup faultlines help or hurt? A moderated model of faultlines, team identification, and group performance." *Organization science*, 20(1), pp.35-50.
- Bi, T., Xia, X., Lo, D., Grundy, J. and Zimmermann, T., 2020. "An empirical study of release note production and usage in practice," *IEEE Transactions on Software Engineering*.
- Black, F. and Scholes, M., 2019. "The pricing of options and corporate liabilities," In *World Scientific Reference on Contingent Claims Analysis in Corporate Finance: Volume 1: Foundations of CCA and Equity Valuation* (pp. 3-21).
- Bodin, Ö., 2017. "Collaborative environmental governance: achieving collective action in social-ecological systems," *Science*, 357(6352).
- Borges, H., and Valente, M. T. 2018. "What's in a GitHub star? Understanding repository starring practices in a social coding platform," *Journal of Systems and Software*, 146: 112-129.
- Boulanger, A. 2005. "Open-source versus proprietary software: is one more reliable and secure than the other?" *IBM Systems Journal*, Vol. 44 No. 2, pp. 239-48.
- Bruce, G., Robson, P. and Spaven, R. 2006. "OSS opportunities in open source software – CRM and OSS standards," *BT Technology Journal*, Vol. 24 No. 1, pp. 127-40.
- Chen, Y., Pereira, I. and Patel, P.C., 2021. "Decentralized governance of digital platforms," *Journal of Management*, 47(5), pp.1305-1337.

- Chesbrough, H. W. 2003. "Open innovation: The new imperative for creating and profiting from technology," Boston, MA: *Harvard Business Press*.
- Clark, P.B. and Wilson, J.Q., 1961. "Incentive systems: A theory of organizations," *Administrative Science Quarterly*, pp.129-166.
- Coleman EG., 2012. "Coding Freedom: The Ethics and Aesthetics of Hacking." *Princeton University Press*, Princeton, NJ.
- Crowston, K., Wei, K., Howison, J. and Wiggins, A., 2008. "Free/Libre open-source software development: What we know and what we do not know," *ACM Computing Surveys (CSUR)*, 44(2), pp.1-35.
- Curtis, B., Krasner, H. and Iscoe, N., 1988. "A field study of the software design process for large systems," *Communications of the ACM*, 31(11), pp.1268-1287.
- Dahlander, L., and Magnusson, M. G., 2005. "Relationships between open source software companies and communities: Observations from Nordic firms." *Research policy*, 34(4), pp.481-493.
- Dahlander, L., and Wallin, M. W., 2006. "A man on the inside: Unlocking communities as complementary assets." *Research policy*, 35(8), pp.1243-1259.
- Dahlander, L., 2007. "Penguin in a new suit: a tale of how de novo entrants emerged to harness free and open source software communities." *Industrial and corporate change*, 16(5), pp.913-943.
- Dahlander, L., Frederiksen, L. and Rullani, F., 2008. "Online communities and open innovation," *Industry and innovation*, 15(2), pp.115-123.
- Dahlander, L., & Magnusson, M., 2008. "How do firms make use of open source communities?" *Long range planning*, 41(6), pp.629-649.
- Davenport, T. H. 1998. "Putting the enterprise into the enterprise system," *Harvard business review*, 76(4).
- Daniel, S., Agarwal, R. and Stewart, K.J., 2013. "The effects of diversity in global, distributed collectives: A study of open source project success," *Information Systems Research*, 24(2), pp.312-333.
- Daniel, S, Maruping, L.M, Cataldo, M., and Herbsleb, J., 2018. "The impact of ideology misfit on open source software communities and companies," *Management Information Systems Quarterly*, 42(4).
- De Laat, P.B., 2007. "Governance of open source software: state of the art," *Journal of Management & Governance*, 11(2), pp.165-177.
- Denison, D. R. (1996). "What is the difference between organizational culture and organizational climate? A native's point of view on a decade of paradigm wars," *Academy of Management Review*, 21(3), 619-654.

- De Noni, I., Ganzaroli, A. and Orsi, L., 2011. "The governance of open source software communities: An exploratory analysis," *Journal of Law and Governance*, 6(1).
- Di Tullio, D. and Staples, D.S., 2013. "The governance and control of open source software projects," *Journal of Management Information Systems*, 30(3), pp.49-80.
- Dorner, C., Draxler, S., Pipek, V. and Wulf, V., 2009. "End users at the bazaar: Designing next-generation enterprise resource planning systems," *IEEE software*, 26(5), pp.45-51.
- Earley, C. P., and Mosakowski, E., 2000. "Creating hybrid team cultures: An empirical test of transnational team functioning." *Academy of Management journal*, 43(1), pp.26-49.
- Eilhard, J. (2009). Open source incorporated. Available at SSRN 1360604.
- Fichman, R.G., 2004. "Real options and IT platform adoption: Implications for theory and practice," *Information Systems Research*, 15(2), pp.132-154.
- Fleming, L. and Waguespack, D. M., 2007. "Brokerage, boundary spanning, and leadership in open innovation communities." *Organization Science*, 18, pp.165–180.
- Gamalielsson, J. and Lundell, B., 2014. "Sustainability of Open Source software communities beyond a fork: How and why has the LibreOffice project evolved?" *Journal of Systems and Software*, 89, pp.128-145.
- Germonprez, M., Young, B., Mathiassen, L., Kendall, J. E., Kendall, K. E., & Warner, B., 2012. "Risk mitigation in corporate participation with open source communities: protection and compliance in an open source supply chain." Petter S, Mitchell A, eds. *AIS SIGIRWITPM Workshop. ICIS Conf. (AIS, Atlanta)*, pp.89–100.
- Germonprez, M., Kendall, J. E., Kendall, K. E., Mathiassen, L., Young, B., & Warner, B., 2017. "A theory of responsive design: A field study of corporate engagement with open source communities." *Information Systems Research*, 28(1), pp.64-83.
- Giovacchini, E., 2017. "Weaving the symbiotic relationship: A longitudinal study of the maintenance of a firm-sponsored open source community." (*Doctoral dissertation, Stockholm Business School, Stockholm University*).
- GitHub Docs. "GraphQL API," <https://docs.github.com/en/graphql>.
- Giuri, P., Ploner, M., Rullani, F. and Torrisi, S., 2010. "Skills, division of labor and performance in collective inventions: Evidence from open source software," *International Journal of Industrial Organization*, 28(1), pp.54-68.
- Giovacchini, E., 2017. "Weaving the symbiotic relationship: A longitudinal study of the maintenance of a firm-sponsored open source community," *Doctoral dissertation, Stockholm Business School, Stockholm University*.
- Greer, D. and Ruhe, G., 2004. "Software release planning: an evolutionary and iterative approach," *Information and software technology*, 46(4), pp.243-253.

- Guo, Y., Spínola, R.O. and Seaman, C., 2016. “Exploring the costs of technical debt management—a case study,” *Empirical Software Engineering*, 21(1), pp.159-182.
- Haans JFR, Pieters C, He ZL, 2015. “Thinking about U: Theorizing and testing U- and inverted U-shaped relationships in strategy research.” *Strategic Management J.* 37, pp.1177–1195.
- Hall, R. I. 1996. “Participation in Congress.” *Yale University Press*, New Haven, CT.
- Hart, C. M., & Van Vugt, M., 2006. “From fault line to group fission: Understanding membership changes in small groups.” *Personality and Social Psychology Bulletin*, 32(3), pp.392-404.
- Hauge, Ø., Ayala, C., and Conradi, R. 2010. “Adoption of Open Source Software in Software-Intensive Organizations--A Systematic Literature Review,” *Information and Software Technology*, 52(11), pp. 1133–1154.
- Howison, J. and Crowston, K., 2014. “Collaboration through open superposition: A theory of the open source way,” *MIS Quarterly*, 38(1), pp.29-50.
- Huang, Y., Liu, Z., Chen, X. and Luo, X., 2016, December. “Automatic Matching Release Notes and Source Code by Generating Summary for Software Change,” *In 2016 6th International Conference on Digital Home (ICDH)* (pp. 104-109). IEEE.
- Jehn, K. A., and Bezrukova, K., 2010. “The faultline activation process and the effects of activated faultlines on coalition formation, conflict, and group outcomes.” *Organizational Behavior and Human Decision Processes*, 112(1), pp.24-42.
- Jeppesen, L.B. and Frederiksen, L., 2006. “Why do users contribute to firm-hosted user communities? The case of computer-controlled music instruments,” *Organization Science*, 17(1), pp.45-63.
- Johansson, B. and Sudzina, F., 2008. “ERP systems and open source: an initial review and some implications for SMEs,” *Journal of Enterprise Information Management*.
- Kelly K., 2013. “Dreams.” *Wired* 21(5), pp.48–57.
- Kruchten, P., Nord, R.L. and Ozkaya, I., 2012. “Technical debt: From metaphor to theory and practice,” *IEEE software*, 29(6), pp.18-21.
- Kuan, J., 2002. “OSS as Lead User’s Make or By Decision: A Study of Open and Closed Source Quality,” *Open Source Software: Economics, Law and Policy*, Toulouse, France.
- Lau, D. C., & Murnighan, J. K., 1998. “Demographic diversity and faultlines: The compositional dynamics of organizational groups.” *Academy of management review*, 23(2), pp.325-340.
- Lau, D. C., & Murnighan, J. K., 2005. “Interactions within groups and subgroups: The effects of demographic faultlines.” *Academy of Management Journal*, 48(4), pp.645-659.
- Lakhani, K.R. and Von Hippel, E., 2003. “How open source software works: “free” user-to-user assistance,” *Research Policy*, 32(6), pp.923-943.

- Lankton, N. and Luft, J., 2008. "Uncertainty and industry structure effects on managerial intuition about information technology real options," *Journal of Management Information Systems*, 25(2), pp.203-240.
- Lee, S.M., Olson, D.L. and Lee, S.H., 2009. "Open process and open-source enterprise systems," *Enterprise Information Systems*, 3(2), pp.201-209.
- Li, J., and Hambrick, D. C., 2005. "Factional groups: A new vantage on demographic faultlines, conflict, and disintegration in work teams." *Academy of Management Journal*, 48(5), pp.794-813.
- Li, P., Maruping, L.M. and Mathiassen, L., 2020. "Developing and Managing Open Source Enterprise Systems through Open Superposition: A Digital Options and Technical Debt Perspective," *Proceedings of the Twenty-Fifth Americas Conference on Information Systems*, August 10-14, 2020. Virtual Conference.
- Lindberg, A., Berente, N., Gaskin, J. and Lyytinen, K., 2016. "Coordinating interdependencies in online communities: A study of an open source software project," *Information Systems Research*, 27(4), pp.751-772.
- Lombard, M., Snyder-Duch, J., and Bracken, C. C., 2002. "Content analysis in mass communication: Assessment and reporting of intercoder reliability," *Human communication research*, 28(4), pp.587-604.
- Kallinikos J, Aaltonen A, Marton A 2013. "The ambivalent ontology of digital artifacts," *MIS Quarterly*. 37(2):357–370.
- Markus, M.L., 2007. "The governance of free/open source software projects: monolithic, multidimensional, or configurational?" *Journal of Management & Governance*, 11(2), pp.151-163.
- Markus, M.L. and Tanis, C., 2000. "The enterprise systems experience-from adoption to success. *Framing the domains of IT research: Glimpsing the future through the past*," 173(2000), pp.207-173.
- Maruping, L.M., Daniel, S.L. and Cataldo, M., 2019. "Developer centrality and the impact of value congruence and incongruence on commitment and code contribution activity in open source software communities," *MIS Quarterly*, 43(3), pp.951-976.
- Medappa, P.K. and Srivastava, S.C., 2019. "Does superposition influence the success of FLOSS projects? An examination of open-source software development by organizations and individuals," *Information Systems Research*, 30(3), pp.764-786.
- Meyer, B., & Glenz, A., 2013. "Team faultline measures: A computational comparison and a new approach to multiple subgroups." *Organizational Research Methods*, 16(3), pp.393-424.
- Michlmayr, M. and Fitzgerald, B., 2012. "Time-based release management in free and open source (FOSS) projects," *International Journal of Open Source Software and Processes (IJOSSP)*, 4(1), pp.1-19.

- Michlmayr, M., Fitzgerald, B. and Stol, K.J., 2015. "Why and how should open source projects adopt time-based releases?" *IEEE Software*, 32(2), pp.55-63.
- Mockus, A., Fielding, R.T. and Herbsleb, J.D., 2002. "Two case studies of open source software development: Apache and Mozilla," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 11(3), pp.309-346.
- Murray, F., and O'Mahony, S., 2007. "Exploring the foundations of cumulative innovation: Implications for organization science." *Organization Science*, 18(6), pp.1006-1021.
- Olson, D.L., Johansson, B. and De Carvalho, R.A., 2018. "Open source ERP business model framework." *Robotics and Computer-Integrated Manufacturing*, 50, pp.30-36.
- O'Leary, M. B., and Mortensen, M., 2010. "Go (con) figure: Subgroups, imbalance, and isolates in geographically dispersed teams." *Organization science*, 21(1), pp.115-131.
- O'Mahony, S., 2003. "Guarding the commons: how community managed software projects protect their work," *Research policy*, 32(7), pp.1179-1198.
- O'Mahony, S., 2007. "The governance of open source initiatives: what does it mean to be community managed?" *Journal of Management & Governance*, 11(2), pp.139-150.
- O'Mahony, S. and Ferraro, F., 2007. "The emergence of governance in an open source community," *Academy of Management Journal*, 50(5), pp.1079-1106.
- O'Mahony, S., and Karp, R., 2022. "From proprietary to collective governance: How do platform participation strategies evolve?" *Strategic Management Journal*, 43(3), pp.530-562.
- Office Of Advocacy. 2019. "Small Businesses Generate 44 Percent Of U.S. Economic Activity," <https://advocacy.sba.gov/2019/01/30/small-businesses-generate-44-percent-of-u-s-economic-activity/>.
- Ostrom, E., 1990. "Governing the commons." *Cambridge, UK: Cambridge University Press*.
- Paulson, J.W., Succi, G. and Eberlein, A., 2004. "An empirical study of open-source and closed-source software products," *IEEE transactions on software engineering*, 30(4), pp.246-256.
- Poile, C., Begel, A., Nagappan, N. and Layman, L., 2009. "Coordination in large-scale software development: Helpful and unhelpful behaviors," *Dept. of Management Sciences University of Waterloo Waterloo, ON, Canada cpoile@uwaterloo. CA and Microsoft Research Redmond, WA, fandrew. begel, naching@ microsoft.com. and Dept. of Computer Science, North Carolina State University, Raleigh, NC, lucas. layman@ncsu.edu*.
- Priharsari, D., Abedin, B., & Mastio, E., 2020. "Value co-creation in firm sponsored online communities: what enables, constrains, and shapes value." *Internet research*, 30(3), 763-788.

- 3Qi Labs, 2020. "Why Software Release Cycle Frequency Continues to Increase," <http://3qilabs.com/why-software-release-cycle-frequency-continues-to-increase/>.
- Ramasubbu, N., Kemerer, C.F. and Woodard, C.J., 2015. "Managing technical debt: Insights from recent empirical evidence," *IEEE Software*, 32(2), pp.22-25.
- Ramasubbu, N. and Kemerer, C.F., 2016. "Technical debt and the reliability of enterprise software systems: A competing risks analysis," *Management Science*, 62(5), pp.1487-1510.
- Raymond, E., 1999. "The cathedral and the bazaar. Knowledge," *Technology & Policy*, 12(3), pp.23-49.
- Roberts, J.A., Hann, I.H. and Slaughter, S.A., 2006. "Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the Apache projects," *Management Science*, 52(7), pp.984-999.
- Robey, D., Ross, J. W., & Boudreau, M. C. 2002. "Learning to implement enterprise systems: An exploratory study of the dialectics of change," *Journal of Management Information Systems*, 19(1), 17-46.
- Robles, G., and Gonzalez-Barahona, J. 2006. "Contributor Turnover in Libre Software Projects," Boston: Springer.
- Rolland, K.H., Mathiassen, L. and Rai, A., 2018. "Managing digital platforms in user organizations: The interactions between digital options and digital debt," *Information Systems Research*, 29(2), pp.419-443.
- Sambamurthy, V., Bharadwaj, A. and Grover, V., 2003. "Shaping agility through digital options: Reconceptualizing the role of information technology in contemporary firms," *MIS Quarterly*, pp.237-263.
- Sandberg, J., Mathiassen, L. and Napier, N., 2014. "Digital options theory for IT capability investment," *Journal of the Association for Information Systems*, 15(7), p.1.
- Seidel, M.-D., and Stewart, K. J. 2011. "An Initial Description of the C-Form," *Research in the Sociology of Organizations* (33:November), pp. 37-72.
- Sen, R., 2007. "A strategic analysis of competition between open source and proprietary software," *Journal of Management Information Systems*, 24(1), pp.233-257.
- Serrano, N. and Sarriei, J.M., 2006. "Open source software ERPs: a new alternative for an old need," *IEEE software*, 23(3), pp.94-97.
- Setia, P., Rajagopalan, B., Sambamurthy, V. and Calantone, R., 2012. "How peripheral developers contribute to open-source software development," *Information Systems Research*, 23(1), pp.144-163.
- Shah, S.K., 2006. "Motivation, governance, and the viability of hybrid forms in open source software development," *Management Science*, 52(7), pp.1000-1014.

- Shaw, J. B., 2004. "The development and analysis of a measure of group faultlines." *Organizational Research Methods*, 7(1), pp.66-100.
- Shaikh, M. and Henfridsson, O., 2017. "Governing open source software through coordination processes," *Information and Organization*, 27(2), pp.116-135.
- Singh, P.V., Tan, Y. and Mookerjee, V., 2011. "Network effects: The influence of structural capital on open source project success," *MIS Quarterly*, pp.813-829.
- Spaeth, S., Von Krogh, G. and He, F., 2015. "Research note—Perceived firm attributes and intrinsic motivation in sponsored open source software projects," *Information Systems Research*, 26(1), pp.224-237.
- Steinmacher, I., Conte, T., Gerosa, M. A., and Redmiles, D., 2015, February. "Social barriers faced by newcomers placing their first contribution in open source software projects." *In Proceedings of the 18th ACM conference on Computer supported cooperative work & social computing*, pp.1379-1392.
- Stewart, K. J., Ammeter, A. P., and Maruping, L. M., 2006. "Impacts of license choice and organizational sponsorship on user interest and development activity in open source software projects." *Information Systems Research*, 17(2), 126-144.
- Stopford, A., Wallace, K. and Allspaw, J., 2011. "Technical Debt," *Capturado em: <http://www.weblogs.asp.net/astopford/archive/2010/07/19/technical-debt.asp>*.
- Stuermer, M. E., 2009. "How firms make friends: Communities in private-collective innovation." (*Doctoral dissertation, ETH*), 2009
- Teigland, R., Di Gangi, P. M., Flåten, B. T., Giovacchini, E., and Pastorino, N., 2014. "Balancing on a tightrope: Managing the boundaries of a firm-sponsored OSS community and its impact on innovation and absorptive capacity." *Information and Organization*, 24(1), pp.25-47.
- Thatcher, S., and Patel, P. C., 2011. "Demographic faultlines: A meta-analysis of the literature." *Journal of Applied Psychology*, 96(6), pp.1119.
- Thatcher, S. M., and Patel, P. C., 2012. "Group faultlines: A review, integration, and guide to future research." *Journal of Management*, 38(4), pp.969-1009.
- Tiwana, A., 2010. "Systems development ambidexterity: Explaining the complementary and substitutive roles of formal and informal controls," *Journal of Management Information Systems*, 27(2), pp.87-126.
- Tom, E., Aurum, A. and Vidgen, R., 2013. "An exploration of technical debt," *Journal of Systems and Software*, 86(6), pp.1498-1516.
- Von Krogh, G., Haefliger, S., Spaeth, S. and Wallin, M.W., 2012. "Carrots and rainbows: Motivation and social practice in open source software development," *MIS Quarterly*, pp.649-676.

- Volkoff, O., Elmes, M. B., & Strong, D. M. (2004). "Enterprise systems, knowledge transfer and power users," *Journal of Strategic Information Systems*, 13(4), 279-304.
- Von Krogh, G., Haefliger, S., Spaeth, S. and Cohen, J.L., 2003. "Collective action and communal resources in open source software development: the case of freenet," *Academy of Management*.
- Vora, D., and Markóczy, L., 2012. "Group learning and performance: The role of communication and faultlines." *The International Journal of Human Resource Management*, 23(11), pp.2374-2392.
- Weber, S., 2004. "The Success of Open Source Harvard University Press," Cambridge, MA.
- West, J., & Lakhani, K. R., 2008. "Getting clear about communities in open innovation." *Industry and Innovation*, 15(2), pp.223-231.
- West, J. and O'mahony, S., 2008. "The role of participation architecture in growing sponsored open source communities," *Industry and innovation*, 15(2), pp.145-168.
- West, J., and Sims, J., 2017. "How firms leverage crowds and communities for open innovation." (2017), pp.58-96.
- Wikipedia of SuiteCRM. <https://en.wikipedia.org/wiki/SuiteCRM>
- Winter S, Berente N, Howison J, Butler B (2014) "Beyond the organizational "container": Conceptualizing 21st century sociotechnical work," *Inform. Organ.* 24(4):250–269.
- Woodard, C.J., Ramasubbu, N., Tschang, F.T. and Sambamurthy, V., 2013. "Design capital and design moves: The logic of digital business strategy," *MIS Quarterly*, pp.537-564.
- Xue, L., Ray, G., and Gu, B., 2011. "Environmental uncertainty and IT infrastructure governance: A curvilinear relationship." *Information Systems Research*, 22(2), pp.389-399.
- Yamauchi, Y., Yokozawa, M., Shinohara, T. and Ishida, T., 2000, December. "Collaboration with Lean Media: how open-source software succeeds," *In Proceedings of the 2000 ACM conference on Computer supported cooperative work* (pp. 329-338).
- Yin RK 2009. "Case Study Research: Design and Methods (Sage, London)."
- Yin, X. and Zajac, E.J., 2004. "The strategy/governance structure fit relationship: Theory and evidence in franchising arrangements," *Strategic management journal*, 25(4), pp.365-383.
- Yoo Y, Henfridsson O, Lyytinen K 2010. "Research commentary—The new organizing logic of digital innovation: An agenda for information systems research," *Information Systems Research* 21(4):724–735.
- Yu, L., 2009. "Mining change logs and release notes to understand software maintenance and evolution," *CLEI Electron Journal*, 12(2), pp.1-10.

Zanutto, E. L., Bezrukova, K., and Jehn, K. A., 2011. “Revisiting faultline conceptualization: Measuring faultline strength and distance.” *Quality & Quantity*, 45(3), pp.701-714.

Zhang, C., Hahn, J., and De, P., 2013. “Research note—Continued participation in online innovation communities: does community response matter equally for everyone?” *Information Systems Research*, 24(4), pp.1112-1130.

Zhou, M., and Mockus, A., 2014. “Who will stay in the floss community? modeling participant’s initial behavior.” *IEEE Transactions on Software Engineering*, 41(1), pp.82-99.

Appendix I: Tables of Robustness Tests

Table A.1 (Alternative DV) Descriptive Statistics and Correlations

	1	2	3	4	5	6	7	8
1. Release Performance	1.00							
2. Representation	-.26***	1.00						
3. Contribution	-.21*	.54***	1.00					
4. Responsiveness	-.18*	.58***	.74***	1.00				
5. Release Types	.07	-.03	-.11	-.18 ⁺	1.00			
6. Merge Status	.24**	-.45***	-.44***	-.37***	-.13	1.00		
7. Task Type	.06	-.07	-.02	-.19*	.10	-.04	1.00	
8. Added Code	-.13	.02	.02	-.04	.05	-.29**	.37***	1.00
Mean	33.24	.18	.39	.23	.39	.60	.11	2712.06
Min	5.98	.00	.00	.00	.00	.00	.00	1.00
Max	336	1.00	1.00	1.00	1.00	1.00	.67	30265.25
Standard Deviation	43.73	.15	.16	.18	.49	.28	.12	5629.74

Note: $N = 124$ releases.

Standard errors are shown in parentheses.

*** Correlation is significant at the 0.001 level.

** Correlation is significant at the 0.01 level.

* Correlation is significant at the 0.05 level.

⁺ Correlation is significant at the 0.10 level.

Table A.2 (Alternative DV) Hypotheses Testing

	Project Release Performance (Download)					
	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6
Merge Status	22.74 (15.83)	27.01 (16.33)	29.97 ⁺ (15.84)	23.85 (15.90)	29.91 ⁺ (16.32)	35.42* (16.26)
Task Type	34.84 (34.16)	27.89 (35.34)	19.33 (35.58)	34.38 (34.79)	20.78 (35.26)	10.80 (35.91)
Added Code	.00 (.00)	.00 (.00)	.00 (.00)	.00 (.00)	.00 (.00)	.00 (.00)
Representation	-132.37** (39.71)			-99.57 ⁺ (51.55)		
Representation-Squared	225.89** (83.74)			156.71 (99.14)		
Contribution		-41.9 (27.57)			-3.56 (34.37)	
Contribution-Squared		-21.72 (88.47)			-27.76 (99.84)	
Responsiveness			-41.27 (29.78)			-13.44 (37.11)
Responsiveness-Squared			33.01 (77.43)			7.65 (87.58)
Release Types				-1.71 (10.68)	14.46 (10.49)	13.39 (12.70)
Release Type × Representation				-97.43 ⁺ (40.20)		
Release Type × Representation-Squared				368.13 ⁺ (180.79)		
Release Type × Contribution					-131.75* (60.80)	
Release Type × Contribution-Squared					-321.28 (261.81)	
Release Type × Responsiveness						-68.94 (59.84)
Release Type × Responsiveness-Squared						-322.26 (397.72)
Fixed Effects of Month of Creation	YES	YES	YES	YES	YES	YES
Intercept	.33 (18.19)	6.36 (18.90)	2.73 (18.44)	-4.63 (19.13)	6.19 (19.20)	-1.78 (18.90)
Lowest VIF				1.38	1.34	1.35
Highest VIF				4.70	3.31	3.06
R ²	.25	.19	.19	.27	.23	.21
Adjusted R ²	.14	.07	.07	.14	.09	.07

Note: $N = 124$ releases.

Standard errors are shown in parentheses.

*** Correlation is significant at the 0.001 level.

** Correlation is significant at the 0.01 level.

* Correlation is significant at the 0.05 level.

+ Correlation is significant at the 0.10 level.

Figure A.1 Relationship between Corporate-communal Representation and Release Performance (Model 1 for Alternative DV)

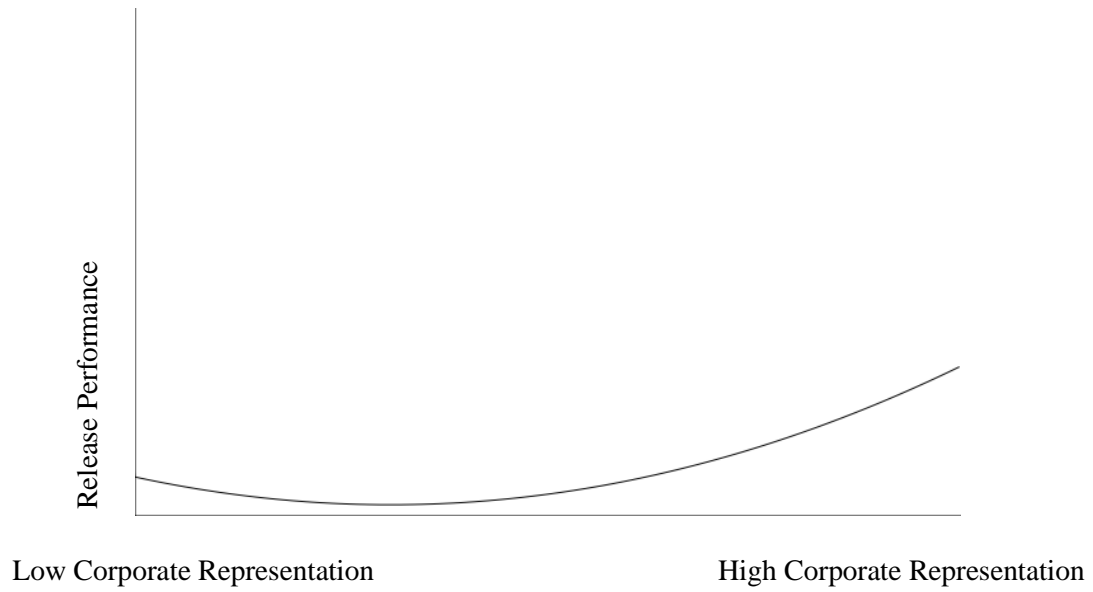


Figure A.2 Interaction between Corporate-communal Representation and Release Type in Predicting Release Performance (Model 4 for Alternative DV)

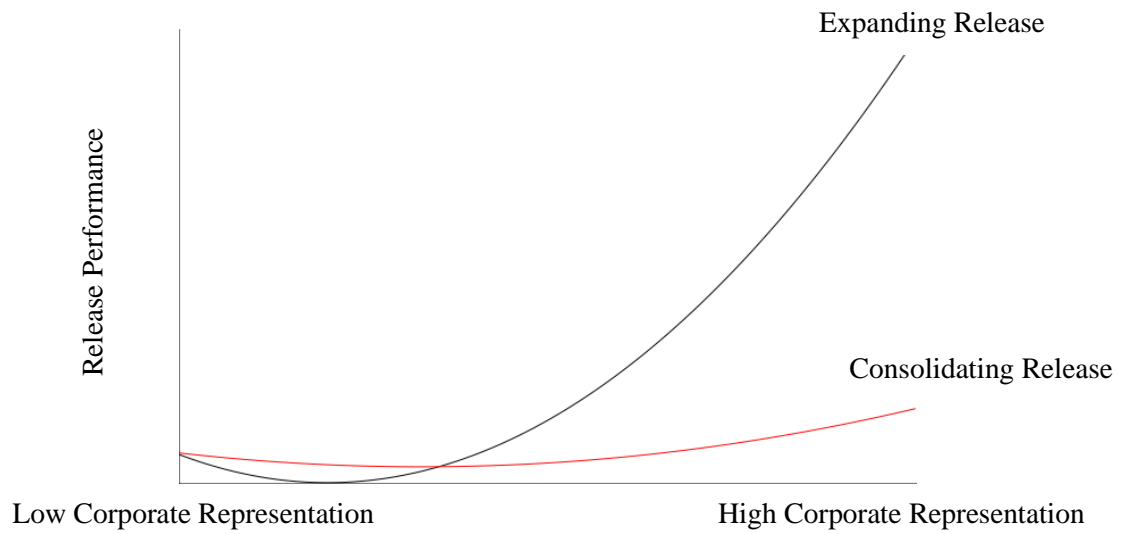


Table A.3 (Early Stage) Descriptive Statistics and Correlations

	1	2	3	4	5	6	7	8
1. Release Performance	1.00							
2. Representation	-.44***	1.00						
3. Contribution	-.10	.48***	1.00					
4. Responsiveness	-.04	.47***	.70***	1.00				
5. Release Types	-.06	-.09	-.23 ⁺	-.37**	1.00			
6. Merge Status	.24	-.36**	-.29*	-.23 ⁺	-.04	1.00		
7. Task Type	.14	-.20	-.13	-.25*	.22 ⁺	.03	1.00	
8. Added Code	-.14	-.04	-.13	-.16	.04	-.17	.48***	1.00
Mean	.46	.25	.47	.30	.45	.44	.11	3557.00
Min	.00	.07	.11	.01	.00	.00	.00	1.00
Max	1.22	1.00	1.00	1.00	1.00	1.00	.67	30265.25
Standard Deviation	.27	.17	.13	.19	.50	.27	.14	7134.23

Note: $N = 62$ releases.

Standard errors are shown in parentheses.

*** Correlation is significant at the 0.001 level.

** Correlation is significant at the 0.01 level.

* Correlation is significant at the 0.05 level.

⁺ Correlation is significant at the 0.10 level.

Table A.4 (Early Stage) Hypotheses Testing

	Project Release Performance (Star)					
	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6
Merge Status	.02 (.14)	.15 (.15)	.13 (.15)	.04 (.13)	.14 (.14)	.12 (.15)
Task Type	.50 ⁺ (.28)	.68* (.30)	.70* (.31)	.64* (.27)	.81** (.30)	.76* (.30)
Added Code	.00 ⁺ (.00)	.00 (.00)	.00 (.00)	.00* (.00)	.00* (.00)	.00* (.00)
Representation	-.95* (.37)			-.88* (.43)		
Representation-Squared	.63 (.65)			.33 (.71)		
Contribution		.14 (.37)			-.29 (.58)	
Contribution-Squared		-.93 (.89)			-.40 (1.11)	
Responsiveness			.13 (.36)			.19 (.44)
Responsiveness-Squared			-.53 (.74)			-.87 (.83)
Release Types				-.21* (.08)	-.18 (.11)	-.14 (.10)
Release Type × Representation				-.43 (.82)		
Release Type × Representation-Squared				2.01 (2.07)		
Release Type × Contribution					.42 (.78)	
Release Type × Contribution-Squared					-1.97 (2.88)	
Release Type × Responsiveness						-.42 (.65)
Release Type × Responsiveness-Squared						-2.69 (4.02)
Fixed Effects of Month of Creation	YES	YES	YES	YES	YES	YES
Intercept	.74*** (.14)	.60*** (.15)	.59*** (.15)	.73*** (.15)	.70*** (.16)	.66*** (.15)
Lowest VIF				1.54	1.41	1.49
Highest VIF				7.16	5.34	6.35
R ²	.44	.33	.32	.53	.41	.42
Adjusted R ²	.25	.09	.08	.31	.14	.15

Note: $N = 62$ releases.

Standard errors are shown in parentheses.

*** Correlation is significant at the 0.001 level.

** Correlation is significant at the 0.01 level.

* Correlation is significant at the 0.05 level.

+ Correlation is significant at the 0.10 level.

Table A.5 (Late Stage) Descriptive Statistics and Correlations

	1	2	3	4	5	6	7	8
1. Release Performance	1.00							
2. Representation	-.19	1.00						
3. Contribution	-.28*	.41***	1.00					
4. Responsiveness	-.12	.54***	.68***	1.00				
5. Release Types	.01	-.17	-.18	-.09	1.00			
6. Merge Status	.13	.10	-.21	-.07	-.08	1.00		
7. Task Type	.15	.18	.05	-.16	-.08	-.14	1.00	
8. Added Code	-.24 ⁺	-.13	.03	.01	.00	-.49***	-.24	1.00
Mean	1.24	.11	.31	.15	.32	.77	.10	1867.11
Min	.00	.00	.00	.00	.00	.00	.00	3.00
Max	4.00	.50	.60	.58	1.00	1.00	.67	16000.42
Standard Deviation	.63	.08	.16	.14	.47	.19	.10	3399.86

Note: $N = 62$ releases.

Standard errors are shown in parentheses.

*** Correlation is significant at the 0.001 level.

** Correlation is significant at the 0.01 level.

* Correlation is significant at the 0.05 level.

⁺ Correlation is significant at the 0.10 level.

Table A.6 (Late Stage) Hypotheses Testing

	Project Release Performance (Star)					
	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6
Merge Status	.07 (.48)	-.17 (.47)	.10 (.53)	-.07 (.50)	-.27 (.47)	.13 (.56)
Task Type	1.15 (.79)	.43 (.73)	.67 (.83)	1.31 (.83)	.42 (.72)	.69 (.86)
Added Code	.00 ⁺ (.00)	.00 ⁺ (.00)	.00 (.00)	.00* (.00)	.00* (.00)	.00 (.00)
Representation	-2.52* (1.04)			-3.35* (1.47)		
Representation-Squared	20.04** (5.74)			23.09** (7.45)		
Contribution		1.44 ⁺ (.75)			1.58 ⁺ (.89)	
Contribution-Squared		12.36*** (3.11)			16.79*** (3.99)	
Responsiveness			-.41 (.61)			-.46 (.78)
Responsiveness-Squared			7.54* (3.63)			6.87 (4.29)
Release Types				-.01 (.44)	.35 ⁺ (.20)	.05 (.39)
Release Type × Representation				2.36 (2.60)		
Release Type × Representation-Squared				8.21 (21.69)		
Release Type × Contribution					.29 (1.91)	
Release Type × Contribution-Squared					-7.70 (7.01)	
Release Type × Responsiveness						.31 (1.40)
Release Type × Responsiveness-Squared						1.81 (9.96)
Fixed Effects of Month of Creation	YES	YES	YES	YES	YES	YES
Intercept	.36 (.54)	.77 (.51)	.51 (.58)	.41 (.56)	.80 (.51)	.48 (.60)
Lowest VIF				1.48	1.36	1.29
Highest VIF				8.46	12.83	5.40
R ²	.45	.50	.34	.46	.55	.34
Adjusted R ²	.25	.32	.10	.22	.35	.05

Note: $N = 62$ releases.

Standard errors are shown in parentheses.

*** Correlation is significant at the 0.001 level.

** Correlation is significant at the 0.01 level.

* Correlation is significant at the 0.05 level.

+ Correlation is significant at the 0.10 level.

Figure A.3 Relationship between Corporate-communal Representation and Release Performance (Late-Stage Model 1)

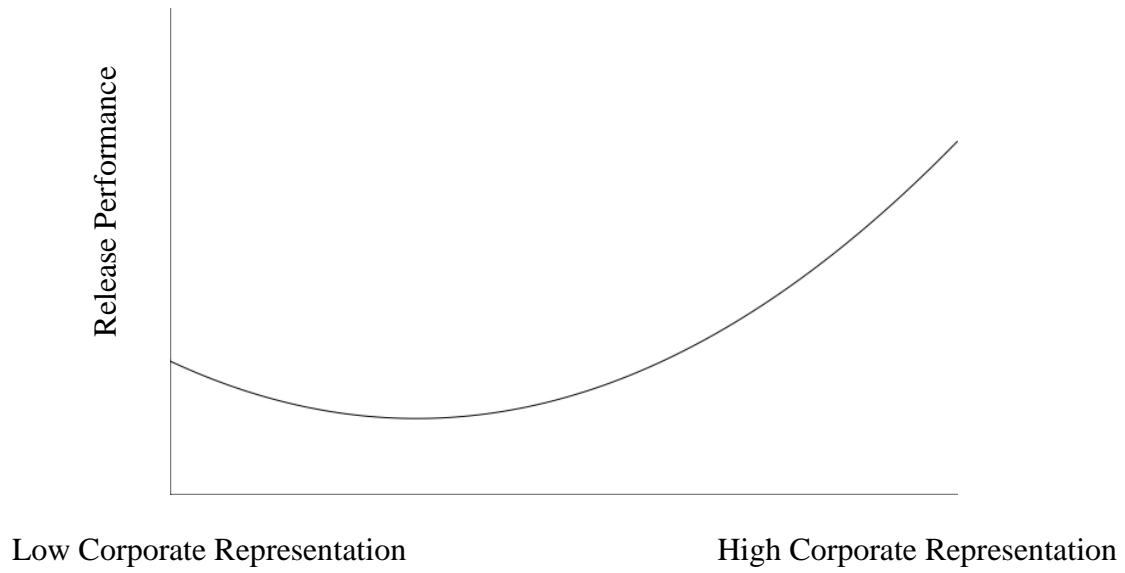


Figure A.4 Relationship between Corporate-communal Contribution and Release Performance (Late-Stage Model 2)

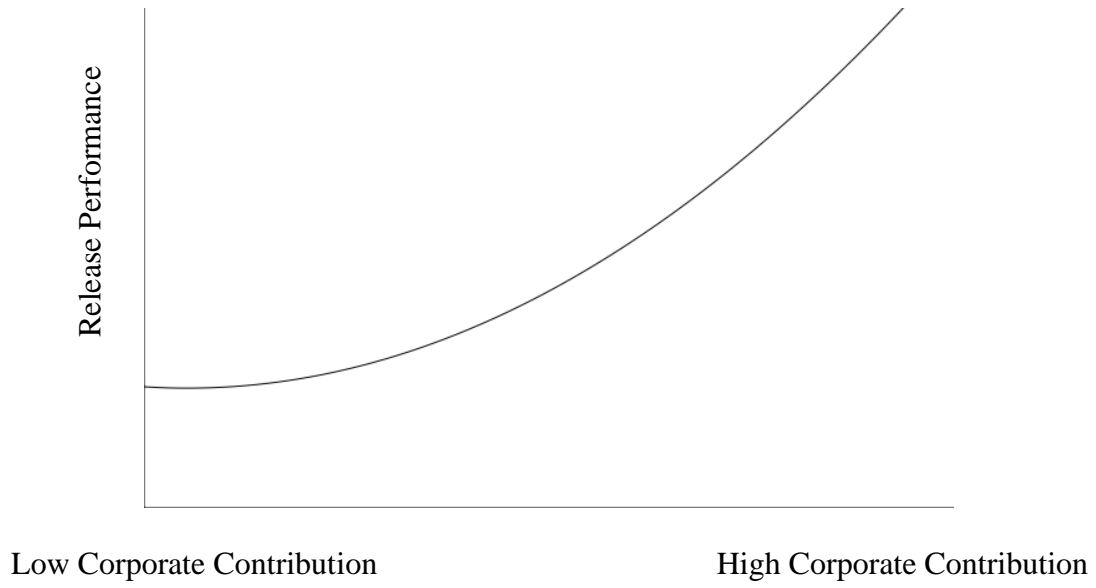


Figure A.5 Relationship between Corporate-communal Responsiveness and Release Performance (Late-Stage Model 3)

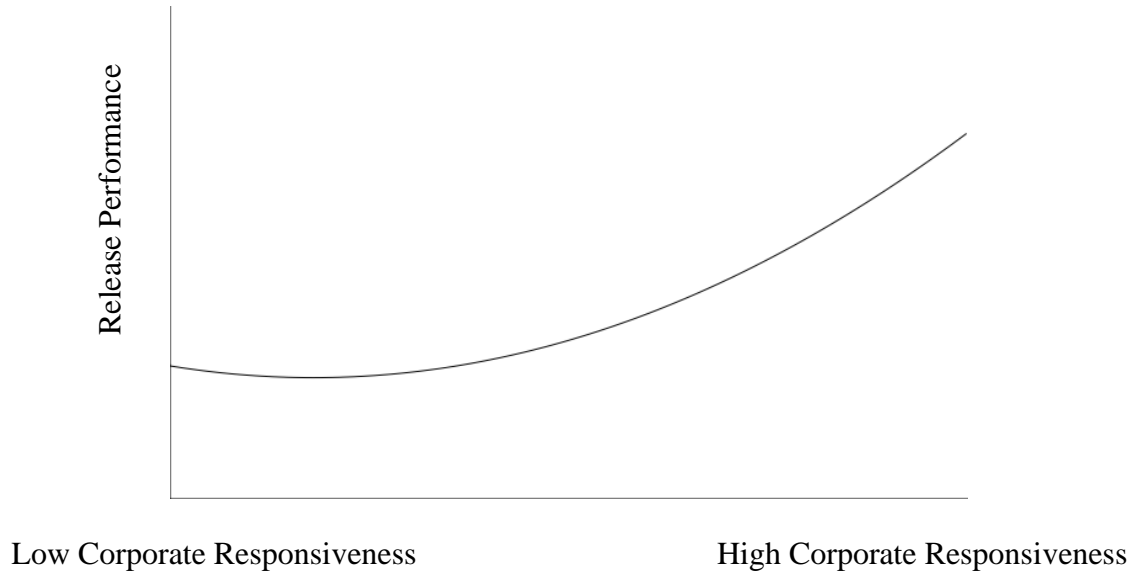


Table A.7 (Alternative IV) Descriptive Statistics and Correlations

	1	2	3	4	5	6	7	8
1. Release Performance	1.00							
2. Representation	-.46***	1.00						
3. Contribution	-.43***	.44***	1.00					
4. Responsiveness	-.29**	.49***	.67***	1.00				
5. Release Types	-.09	-.03	-.14	-.20*	1.00			
6. Merge Status	.46***	-.44***	-.42**	-.35***	-.12	1.00		
7. Task Type	.08	-.07	-.02	-.20*	.10	-.04	1.00	
8. Added Code	-.21*	.02	.01	-.04	.05	-.29**	.37***	1.00
Mean	.85	.15	.35	.21	.39	.60	.11	2712.06
Min	.00	.00	.00	.00	.00	.00	.00	1.00
Max	4.00	.50	.60	.68	1.00	1.00	.67	30265.25
Standard Deviation	.62	.12	.14	.16	.49	.28	.12	5629.74

Note: $N = 124$ releases.

Standard errors are shown in parentheses.

*** Correlation is significant at the 0.001 level.

** Correlation is significant at the 0.01 level.

* Correlation is significant at the 0.05 level.

+ Correlation is significant at the 0.10 level.

Table A.8 (Alternative IV) Hypotheses Testing

	Project Release Performance (Star)					
	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6
Merge Status	.63*** (.18)	.67*** (.19)	.80*** (.19)	.59** (.18)	.63** (.19)	.74*** (.20)
Task Type	.85*(.41)	.49 (.41)	.32 (.44)	.94* (.41)	.52 (.42)	.39 (.45)
Added Code	.00+ (.00)	.00* (.00)	.00* (.00)	.00* (.00)	.00*(.00)	.00* (.00)
Representation	-3.70*** (.64)			-3.22*** (.88)		
Representation-Squared	11.24*** (2.58)			9.39** (3.52)		
Contribution		-.44 (.45)			-.48 (.53)	
Contribution-Squared		7.62** (2.32)			10.79*** (2.90)	
Responsiveness			-1.14** (.41)			-1.06* (.51)
Responsiveness-Squared			4.58** (1.70)			3.59+ (1.92)
Release Types				-.25+ (.14)	.03 (.13)	-.30+ (.17)
Release Type × Representation				-1.49 (1.29)		
Release Type × Representation-Squared				5.19 (5.47)		
Release Type × Contribution					-.85 (.95)	
Release Type × Contribution -Squared					-9.02+ (4.99)	
Release Type × Responsiveness						-.13 (.82)
Release Type × Responsiveness-Squared						10.63 (6.67)
Fixed Effects of Month of Creation	YES	YES	YES	YES	YES	YES
Intercept	.12 (.22)	.23 (.22)	.18 (.23)	.22 (.22)	.26 (.22)	.25 (.23)
Lowest VIF				1.43	1.35	1.36
Highest VIF				6.35	5.47	2.94
R ²	.49	.45	.39	.51	.47	.41
Adjusted R ²	.41	.37	.29	.42	.38	.30

Note: $N = 124$ releases.

Standard errors are shown in parentheses.

*** Correlation is significant at the 0.001 level.

** Correlation is significant at the 0.01 level.

* Correlation is significant at the 0.05 level.

+ Correlation is significant at the 0.10 level.

Figure A.6 Relationship between Corporate-communal Representation and Release Performance (Alternative IV Model 1)

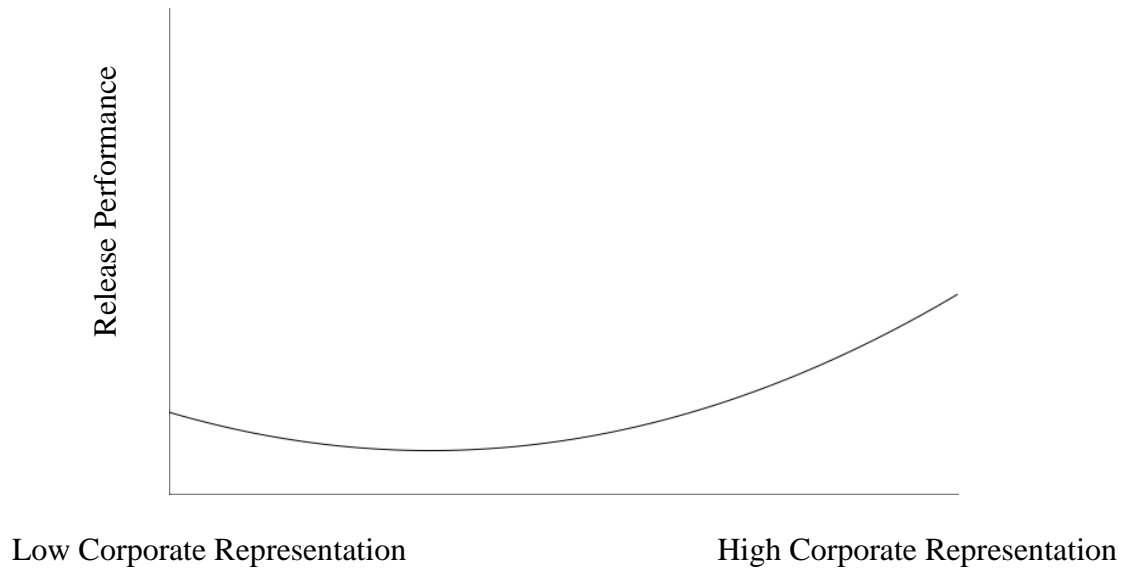


Figure A.7 Relationship between Corporate-communal Contribution and Release Performance (Alternative IV Model 2)

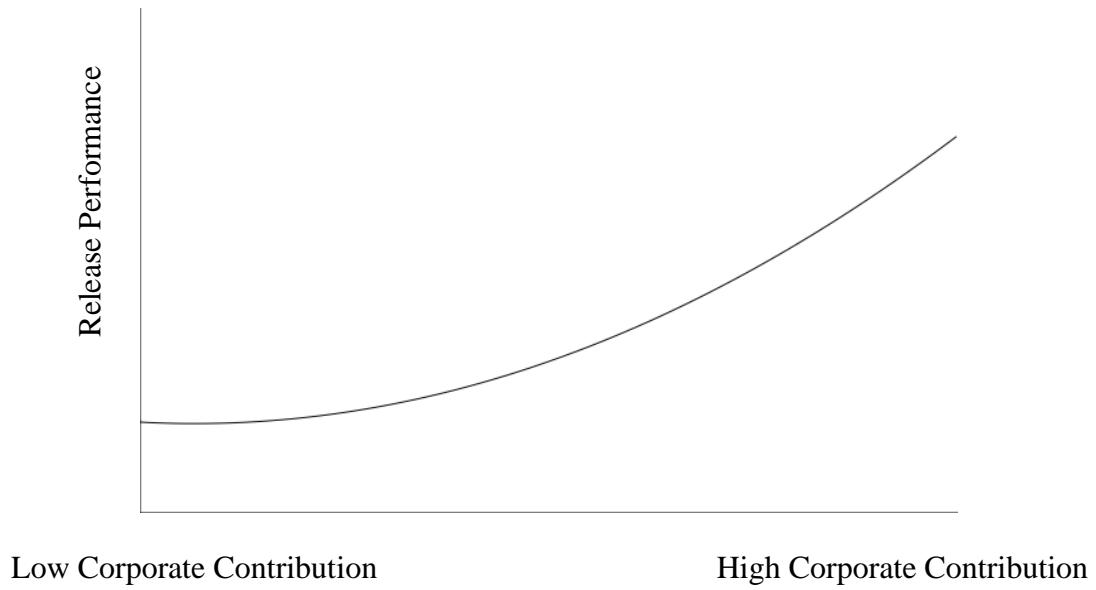
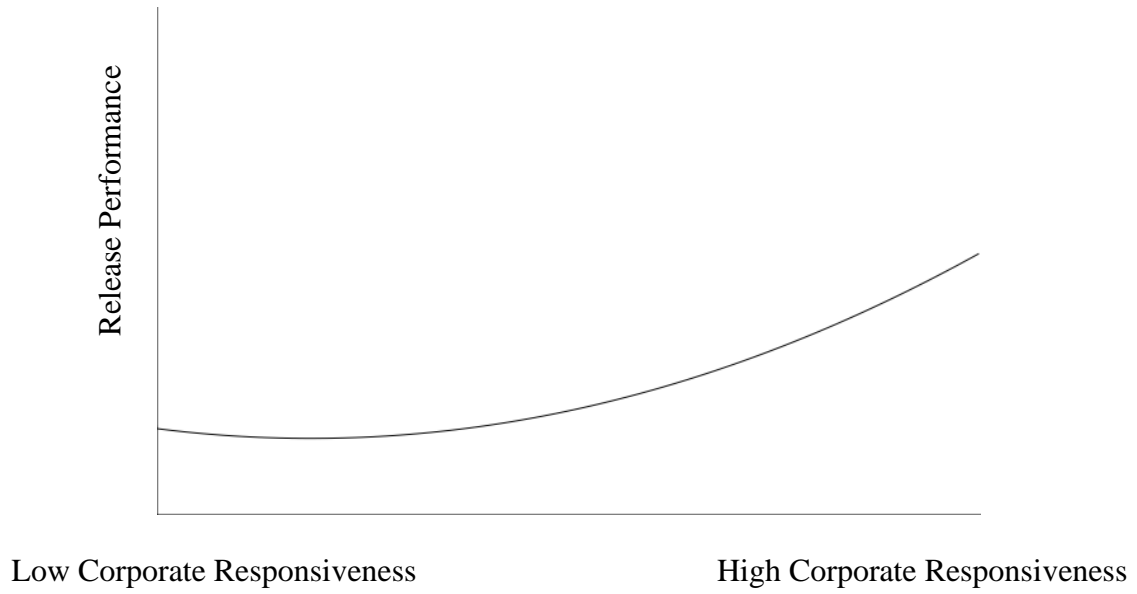


Figure A.8 Relationship between Corporate-communal Responsiveness and Release Performance (Alternative IV Model 3)



Appendix II: Intercoder Reliability

Building upon Lombard et al. (2002) research (p. 590), intercoder reliability “is assessed by having two or more coders categorize units (programs, scenes, articles, stories, words, etc.), and then using these categorizations to calculate a numerical index of the extent of agreement between or among the coders.” Thus, we hired two Ph.D.-level coders to code the entire releases into two types: consolidating and expanding releases. The inter-coder reliability reached 90% after two rounds of coding training. Finally, these two coders compare their results and reach agreements on the differences in the coding. The coding scheme and process are shown in Figure 5.3.

To calculate the intercoder reliability, we follow the insights of Lombard’s (2002) study on content analysis to conduct two rounds of training for these two coders. Previous research has shown researchers themselves can serve as coders (Lombard et al. 2002). Accordingly, one of the coders is the author of this dissertation, and the other holds a Ph.D. degree in Entrepreneurship. Before the training, the author of this dissertation introduces the general background and the overall idea of this dissertation. Additionally, both coders have a well-established understanding of the GitHub dataset because of their prior research on GitHub. Each training section includes ten randomly selected releases of SuiteCRM. Two coders coded all the selected releases according to the release notes by following the definitions established in this study (See Table 3.2 for the definitions of different types of releases). After the intercoder reliability reached 90% by comparing their coding results, two coders coded the rest of the releases. Finally, two coders compared their results and agreed on any existing differences.