경영학 박사 학위논문

Three Essays of Governance on Open Source Software Project:
–Individual, Project, and Organization Perspectives –

오픈 소스 소프트웨어 프로젝트 거버넌스: 개인, 프로젝트, 그리고 조직 차원의 관점으로

2016년 08월

서울대학교 대학원
경영학과 경영학 전공

이새롬

Abstract

Three Essays of Governance on Open Source Software Project: Individual, Project, and Organizational Perspectives

Saerom Lee
College of Business Administration
Seoul National University

Abstract

To foster innovation, many companies have been investing resources both outside and within companies. The recent developments in information and communication technologies (ICT) have also led various companies enhance their innovation performance through open collaborations that lead to similar and sometimes better than innovations developed by internal resources. In this study, factors affecting the innovation performance of open-source software (OSS) as a representative example of open collaboration are studied. Although OSS is developed by many, unspecified volunteers, its final goal is to develop software with intended functions. Therefore, governance mechanisms of OSS can affect the results of software development projects. A three-level analysis on the governance of OSS development project was conducted in this study. First, in Essay 1, when individual developers establish and manage an open-source project, they become its owner. We assumed that the role owners can affect project performance. In addition, when developers form an organization to manage and develop projects, governance styles to operate the project will differ according to organizations. Therefore,

organizational characteristics on innovation performance are the research objectives of Essay 2 in this paper. Finally, in Essay 3, the effects of inherent project characteristics on project innovation performance are also examined focusing on team compositions. For the three-point analysis, this study collected data by web crawling on GitHub (Github.com), which is the OSS development platform. The results of this study suggest directions for successful governance of OSS project, and provide theoretical contribution to the existing study as streams of innovation management.

Keyword : Open Innovation, Open Collaboration, Open Source Software, Project Management
Student Number : 2010-20504

TABLE OF CONTENTS

TABLES

FIGURES

CHAPTER 1 INTRODUCTION

1. 1. Research Background and Motivations

Various companies are investing in innovation to sustain their growth. Historically, companies have not paid much attention to the easy collapse of technical innovation development (Chesbrough, 2006a). In recent years, however, companies have looked into promoting innovations in various ways to prevent such tragedies. These efforts include research and development (R&D) investment, outsourcing, and joint development (Chesbrough, 2006a). The efforts are not limited to the investigation of resources within boundaries of firm but resources from outside of firms.

This study investigated successful innovation management through open collaboration, in which a large number of autonomous individuals participate, focusing on the Internet among other innovation methodologies. Rapid advancements in the field of information and communication technologies (ICT) have also made open collaboration easier (Hossain & Wigand, 2004). Various fields easily create communities for collaborating to accomplish certain purpose. These activities make possible to share resources of individual for free. Therefore, open collaboration is parts of activities for sharing economy which as newly added in Oxford dictionaries in 2015 (Heo, 2016). The Oxford dictionary defines sharing economy as “An economic system in which assets or services are shared between private individuals, either for free or for a fee, typically by means of the Internet”. Open collaboration as mechanisms of sharing economy is becoming more important as it can resolve many social problems.

Although open collaboration has been the focus of a number of studies, most of these studies have focused on individual motivation for voluntary contribution (Fang & Neufeld, 2009). Open collaboration does not provide physical rewards or authorities, unlike traditional companies; hence, understanding the motivation of its participants is important. In this regards, encouraging volunteers is crucial to the success of an open collaboration. However, although with contributions of volunteers, still there are large numbers of failures of open

collaboration exist. For successful open collaboration, the core of this study determining how development projects are managed in the field of open-source software (OSS) development which is representative examples of open collaboration.

Many studies that concentrate on OSS focus on the internal motivation of voluntary OSS contributors (Fang & Neufeld, 2009). Majority of studies also deal with the degree of openness of licenses to provide systematic rewards or to protect rights (Sen, Subramaniam, & Nelson, 2008). Such trends are caused by the voluntary-oriented development environment of OSS, which, unlike traditional companies, cannot impose responsibilities on developers. Recently, traditional companies have recognized that they cannot motivate intellectual workers by only providing more materialistic rewards or more responsibilities, and hence, these companies have been exerting efforts to provide an environment that will maximize voluntary contribution (Markus & Agres, 2000).

However, although various individuals participate in OSS development projects, large numbers of projects still fail (Hahn, Moon, & Zhang, 2008). In reality, many OSS projects cannot successfully complete the software (Chengalur-Smith & Sidorova, 2003), because a successful and innovative project development has two simultaneous goals: for developers to contribute voluntarily and to achieve their software's intended objectives. Therefore, for a more successful OSS project, the motivation of contributors should not only be achieved, it should also need the governance mechanisms of the project (Scacchi & Jensen, 2008).

1. 2. Research Goals and Research Questions

Based on the necessity of a deeper consideration on the limitations of existing studies, as well as governance mechanisms of OSS projects, this study determined the effects of OSS project governance on innovation performance from three different perspectives: individual, project, and organization levels. Scacchi and Jensen (2008) categorized the analytics levels into three: micro-, meso-, and macro-levels. Micro-level includes actions, beliefs, and social and technical resources. Meso-level analysis refers to the patterns of cooperation, coordination,

control, leadership, and project alliance. Finally, macro-level refers to multi-project OSS ecosystems and OSS as social movements. McGrath (1964) also classified the analysis perspectives into three level factors:1) individual, which refers to the patterns of member skills, attitudes, and personalities; 2) group, which refers to the structure, cohesiveness, and group size; and 3) environment, which refers to the group task characteristics, rewards structure, and environmental stress level. Our study focused on the three analysis levels based on project owners (individual level) and governance characteristics (project and organization levels).

In GitHub, project owners can be individual developers, and individual developers can form an organization to establish an OSS project (Bonaccorsi & Rossi, 2003). Therefore, we considered the aspects of individual owners that can affect project innovation performance when they have established the project. The developer participating in the OSS development collect information on the project they would participate in and attempts to predict possibilities to ensure that the effort and time invested will be injected into the project successfully. However, most OSS is not secure, which causes cooperators to stop developing. Hence, information provided for developers who want to participate remains limited. Conversely, developers can form an impression on their desired project through the nature of project owner. Therefore, the first study verified the effects of project owners' profile information on the project performance, focusing on impression formation theory. Factors affecting project performance were verified, focusing on subscriptions of developers (following), subscriptions in other projects (starred), knowledge contribution and the numbers of follows. These factors can be found on the development activity of developers and personal information they disclosed.

The second study verified the effects of the governance mechanisms for each organization on the project performance, in case the project managed by an organization. Each organization is a group of more than two developers. GitHub provides its organization members the right to operate a project by forming a group. The number of projects have numbers of governance mechanisms. Especially, we focus on the structure of organization and participation patterns of each organization members as indicators of organizational characteristics. The effects of these characteristics on project performance of each organization are also analyzed.

Finally, the effects of characteristics of each project operated by organizations on the project performance are also analyzed. When each project is divided into internal and external developers, their effects on the results are verified and the effects of their compositions on the results are suggested through response surface model. The study attempted to answer the following research questions:

- How do individual owner characteristics affect project innovation performance?
- What are the effects of characteristics of OSS project organizations on innovation performance of each organization?
- What are the effects of characteristics of team compositions in each project affecting project performance?

This study collected repository data on GitHub, individual data of owners, and organizations' project participation data. GitHub was opened in 2008 and was chosen as the most popular OSS development platform by developers in 2011, surpassing the usability of Sourceforge and Googlecode's OSS development platforms (Vasilescu, Filkov, & Serebrenik, 2015). GitHub provides features that are not provided by existing OSS development platforms, such as individual pages where developers' personal information is accumulated. GitHub also provides social functions, such as follow and following. Therefore, analyzing open collaboration in a diversified way is possible because users can obtain much information on the network among developers or individual characteristics of developers.

This paper is organized as follows. First, the theoretical background of this study is discussed in three sections of the second chapter. The third chapter provides Essay 1 on individual perspective on OSS project governance based on impression formation in CMC environment. Essay 2, which relates to organizational perspectives on OSS project governance, can be found in the fourth chapter. Essay 3 is introduced in the fifth chapter. Finally, the expected theoretical and empirical contribution with limitations and future plans are presented in the last chapter.

CHAPTER 2 THEORETICAL BACKGROUND

2. 1. Open Collaboration and Open Source Software

Since 1990s, the ICT industry has moved aggressively toward firm innovation and new product development, which has a direct effect on its continued survival and performance (Brown & Eisenhardt, 1997; Rothaermel & Deeds, 2004). However, although the area of information system (IS) has devoted efforts on process innovation through information systems, research on digital innovation, with novel value of digital product and changed structure of industries, remain lacking (Yoo, Henfridsson, & Lyytinen, 2010).

Many companies have established innovative strategies to avoid being left behind and to survive in the market (Chesbrough, 2006a). Technology-intensive industries with high uncertainty that have rapidly changing needs should develop new and lasting technologies (Sears & Hoetker, 2014). Innovations for long-term success require new ideas, insights, and expertise which can be gained from external knowledge (Rosenkopf & Almeida, 2003). The key point of innovation is to find new ideas, and combine such ideas with existing knowledge or commercialize them, which costs a significant amount of time and capital (Laursen & Salter, 2006). Investing in innovative ideas can be very difficult, particularly in the ICT field where technology changes quickly. Thus, many corporations are pursuing innovation through cooperation with other companies or by inviting external resources. However, forming new ideas using external resources can be difficult because of the limitations of firms' experience and learning skills (Nelson & Winter, 1982).

Open strategy is when a corporate body or an individual adopts not only internal, but also external resources. According to Chesbrough (2006a), an increasing number of companies look forward to achieving innovation by accepting a wide variety of external resources, while spending less amount of money on R&D. This approach is called the open-style cooperation, which considers that such idea changes company boundaries and that new ideas can be created through cooperation among external resources, in which the connection is weaker than the

relationships among existing corporations (Chesbrough, 2006a). Results from open collaboration can have equal or better performance than that developed by existing corporations. Their economic values are also significantly higher. The main feature of open collaboration is its openness to external resources, which is important to understand. Chesbrough (2006) introduced the most commonly used definition of open collaboration: a paradigm that assumes that firms could and should use internal and external ideas and paths to market, as firms look to advance their technology. In addition, Cosentino, Izquierdo, & Cabot (2014) defined the open collaboration as the tendency to of accepting new thoughts, methods, or changes.

Cooperation through the Internet is an extreme example of open collaboration. Since the development of the Internet, a platform has been established not only for companies, but also for individuals to facilitate cooperation for innovation, which is performed mainly by voluntary participants (Dahlander & Magnusson, 2008). Open collaboration comes in various ways. For instance, InnoCentive supports scientists and major companies to address R&D tasks by providing a connection between them. Conversely, KickStarter.com helps entrepreneurs to raise capital through crowd funding.

Among the various areas that actively implement open collaboration of external knowledge, the field of open-source software (OSS) one of the promising fields (Choi, Kim, Ferwerda, Moon, & Hahn, 2013). OSS has been consistently studied for about twenty years, and has been found to have caused a significant economic ripple effect (Hippel & Krogh, 2003). Gartner Group forecasted that most IT corporations will implement OSS for key elements in IT solutions beginning 2016 (Driver, 2012). Although OSS does not have one clear definition (Aksulu & Wade, 2010), it is commonly defined as "the software that is capable of modifying the source code, free to distribute, technically neutral, and is given with an autonomous license right" (Perens, 1999). OSS differs from traditional technological development methods of corporations, because OSS involves autonomous participation. In addition, OSS products are as excellent as those made by corporations in terms of performance or economic values.

OSS as a means for open collaboration was initiated by Richard Stallman in 1983. Stallman introduced the General Public License, where every part of the program can be modified or shared. In February 1998, Netscape opened their

source code online for free, thereby ushering the beginning of OSS (Aksulu & Wade, 2010; Raymond & Young, 2001). The Linux operating system is a typical example of an OSS-developed system. According to McPherson, Proffitt, and Hale-Evans (2008), the economic value of Linux is approximately $10.8 billion in 2008.

Recently, websites supporting OSS development have diversified functions to facilitate cooperative works, creating an environment where anyone can readily develop and download OSS, and share newly modified versions of the software. Insufficient active participation from internal experts have caused various companies to exert efforts in implementing OSS management methods to motivate employees, because the key concept of OSS management is voluntary participation (Markus & Agres, 2000).

Chengalur-Smith and Sidorova (2003) announced that about 80% of the projects through SourceForge, one of the major online open-source platforms, are currently inactive. Moreover, only 62.9% of the total projects in GitHub have had at least one string of code within the last 18 months, and only 74.22% of them involve two or more developers (Lima, Rossi, & Musolesi, 2014). Based on this phenomena, Stewart, Ammeter, and Maruping (2006) is interested in determining the reason for the success of some projects, while others fail. Fang and Neufeld (2009) found that most OSS projects that failed also lacked voluntary participation.

Crowston, Annabi, and Howison (2003) pointed out that voluntary participants in an OSS project are not bound to a contract or incentive. Thus, their participation is not expected as that of common employees, and OSS projects without such participation hardly have a possibility to succeed. The success of OSS development also requires active participant contributions. Many ongoing studies aim to identify individual contributions and the factors affecting project performance. Many studies have been conducted to identify the factors affecting OSS success. The internal motivation of project participants to develop an OSS differs from a traditional community aiming to create profit, and hence, they have different characteristics from organizations studied in the field of management.

Riehle et al. (2009) defined the characteristics of OSS projects as egalitarian, meritocratic, and voluntary. In OSS projects, anyone can access and contribute (egalitarian), participants can be evaluated transparently based on their performance (meritocratic), and participants can determine methods to establish the

organization and ways to achieve their goals (voluntary). In addition, voluntary participants also aim to form an egalitarian organization, with respect to communications made within the project in the process of achieving goals. The overall management method to operate OSS is affected by OSS project characteristics.

In particular, OSS communities operate differently from existing manufacturing firms, and their governance also differs from conventional firms, such that the purposes and methods of participation in OSS are open. However, despite the importance of OSS projects, studies on OSS governance remain insufficient. In previous studies, OSS communities are classified as loosely coordinated, self-organizing, and voluntary communities (Moon & Sproull, 2002), and their relationship is found to be a flat hierarchy that lacks formal structure (Dahlander & O'Mahony, 2011). However, in contrast with simple online communities that share knowledge or common interests, OSS projects have an explicit leader who initiated the project, and they have a purpose, which is to develop software according to the project goal (Giuri, Rullani, & Torrisi, 2008).

2. 2. Previous Research of Open Source Software

OSS studies have been conducted since the 1980s, when OSS was first introduced. OSS project developers voluntarily contributed their time and effort, which rarely occurred in existing companies. Motivation factors are shown in Table 1. In Table 1, we re-summarized the research purpose of each articles based on the categorizations of Fang and Neufeld (2009). Most studies about this issue are focused on the reasons behind voluntary contribution (Von Krogh, Haefliger, Spaeth, & Wallin, 2012). Von Krogh et al. (2012) classified developers' motivation into internal and external motivations, and analyzed a total of 214 studies from 2009. The results of the analysis showed that internal motivation involved ideology, altruism, kinship, and fun, whereas external motivation involved reputation, reciprocity, learning, own-use, career, and pay. Aksulu and Wade (2010) reviewed 618 OSS studies that used the macroscopic approach and determined that initial studies on OSS involved mainly analyses of OSS project characteristics in a

descriptive manner or OSS project advantages. The trend later shifted to analyzing the areas (licensing, motivation, governance, etc.) differentiated from the existing OSS or companies. Among these studies with various focuses, we focused on studies pertaining to governance, which affects the success of an OSS project (Bonaccorsi & Rossi, 2003; Franck & Jungwirth, 2003; Shah, 2006).

Table 1. Previous Research on Motivation of OSS Development

Reference	Software Use Value	Learning & Enjoyment	Reputation	Ownership	Career
Franke and Von Hippel (2003)	○				
Hertel, Niedner, and Herrmann (2003)	○	○	○		○
Lakhani and Wolf (2003)		○	○		
Lerner and Triole (2000)			○		○
Moon and Sproull (2002)				○	
Roberts, Hann, and Slaughter (2006)			○		
Shah (2006)	○	○	○		○
Von Hippel (2001)	○				
Von Krogh, Spaeth, and Lakhani (2003)	○		○	○	
Wu, Gerlach, and Young (2007)					○

Considering that studies on motivations developers are micro-scale - individual-level studies - macro-level studies focus on characteristics or patterns of entire projects. Many of the macro-level studies are exploratory because of the lack of data or other similar reasons. Macro-level studies also analyze the effects of OSS project characteristics or the developers' individual properties on productivity or collaboration of the project. Various previous studies have discussed the OSS

governance mechanisms. Franck and Jungwirth (2003) divided the participation factors of developers involved in OSS development into internal and external to explain the governance mechanisms. Various implications in the governance dimension on nonsense situations that OSS development project working for free should aim for success and support the reward were also considered (Bonaccorsi & Rossi, 2003). Shah (2006) also analyzed OSS development motivations related to governance mechanisms, which deal with rights on codes and licensing. The research suggested the proper strategies for handling each right to encourage motivation. However, these research only focus on the causality of internal and external rewards with free knowledge contributions.

To reduce the failure of OSS projects, it is important to understand a circumstance of OSS governance and roles of strategies on project performance (Hahn et al., 2008). More specifically, previous research investigate the roles of project leaders with various characteristics (Comino, Manenti, & Parisi, 2005; Giuri, Ploner, Rullani, & Torrisi, 2010; Giuri et al. 2008; Lerner & Tirole, 2005). Especially, during the communication in internet environment, developers recognize emergent leaders with certain characteristics such as attitudes toward communications, amount of knowledge contributions, or positions of communities considering the communication patterns (Faraj, Kudaravalli, & Wasko, 2015).

With macro level, structures of communities with different roles of developers also described (Crowston & Howison, 2006; Krishnamurthy, 2002; Mockus, Fielding, & Herbsleb, 2002). OSS projects are conducted by self-organizing, the role of developers are characterized by following how they contribute or what they contribute. In this case, the importance of each developers are different for effective project performance (Hahn et al., 2008; Singh & Tan, 2010). To specify the complex structures of OSS development communities, various research use social network analysis as analytical tools with social capital theory as theoretical lens (Baek & Oh, 2015; Crowston & Howison, 2005; Hahn et al., 2008; Singh & Tan, 2010). The research verify the relationship between positions of developers in OSS development communities and individual performances (Baek & Oh, 2015). Other research asserts factors affecting the decision making on newly established project with previous cooperation relationships (Hahn et al., 2008).

2. 3. Github

OSS dates back to 1980s, and its environment has been advancing continuously as the Internet develops. Various services have been providing an environment for OSS development. Assembly and SourceForge are among the major websites that provide such service. GitHub is a web-based code hosting service that supports projects using git (Kalliamvakou, Damian, Singer, & German, 2014). Compared to existing web services, GitHub has various useful functions that facilitate cooperative tasks. In contrast with SourceForge or Google Code, GitHub focuses on source code and patch developments. GitHub also facilitated the function to notify individual development status through social networking, and enhanced cooperative convenience by making connections readily available among developers (Lima et al., 2014). GitHub's unique differentiation provides an efficient developmental environment for many developers. Consequently, by February 2016, GitHub has invited 120 thousand developers and have set up 310,000 repositories. (Wikipedia, 2016.01.14). According to Fortune's report, GitHub has become a mecca for software developers (Fortune, 2015.09.29).

GitHub provided new functions that offer various information on open collaboration, which cannot be found in existing OSS development environment. Thus, researchers are actively participating in the in-depth application of understanding on GitHub and previous theories. Various studies have also been conducted to analyze GitHub functions, which can be classified briefly into functions and information provided in personal pages and those provided in repositories. Repositories are classified into private and public ones (Kalliamvakou et al., 2014). By providing accumulated and detailed information on each developer on activities that occurred within a repository, they can readily gain information on the repository. Dabbish, Stuart, Tsay, and Herbsleb (2012) stated that GitHub is a platform, making it available to observe information on OSS project activities, and provide various information to developers who are willing to participate in the project. Among various the GitHub functions, Feliciano (2015) described the interface that emphasizes its openness. The information given upon login shows the cooperative work status. According to Tsay, Dabbish, and Herbsleb (2014),

developers participating in the OSS project use information provided by the website to understand project norms and their work history through prior contribution information. Tsay et al. (2014) explained the significant values of social features of GitHub as OSS development platform, and those values can be achieved from the transparent and open user-to-user relationship or project information.

Davis (2015) explained the private user-perspective interface, with the aim of analyzing the information that private developers can gain when they use GitHub. When developers log into GitHub, the news feed called timeline (See Figure 1), which contains their starred repository activities and real-time information on those activities, including project initiation, commitment, and stare of a private user whom the developer is following, appears.

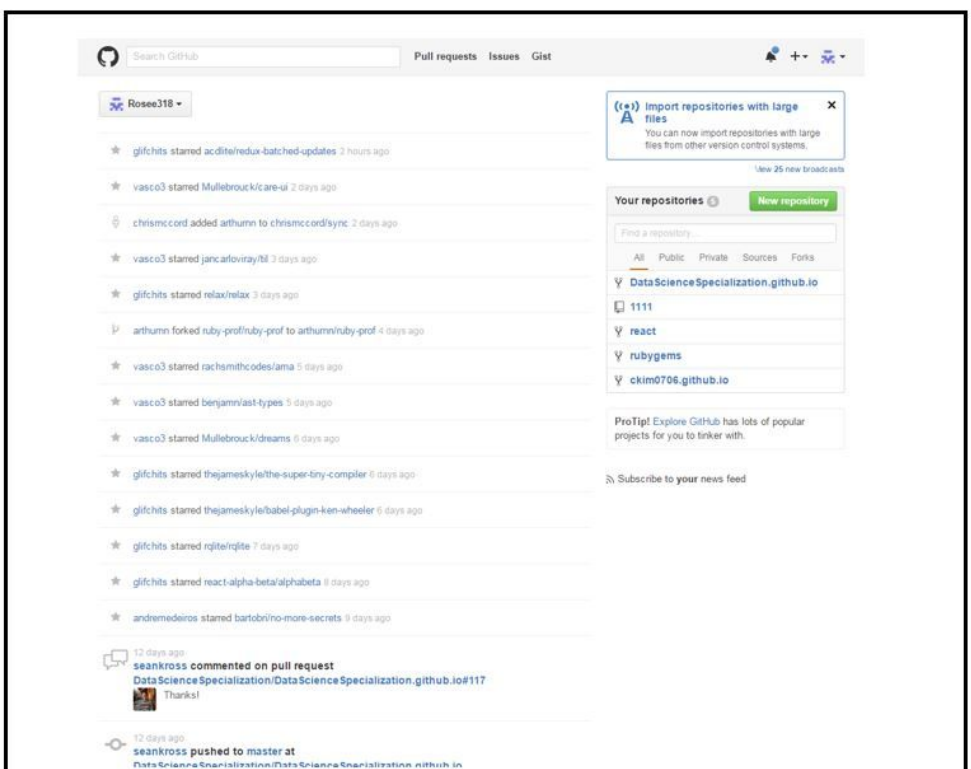


Figure 1. Timeline of Github

Users can take three different types of actions regarding the repository (See Figure 2). First, users can choose to watch, comments on the code changes, new pull requests, new issues, issues, and commit codes come up. When users do not

watch, they receive a notification when a specific mention is made (Onoue, Hata, & Matsumoto, 2013). The repository activities include comment, push to code, pull request, and new issue. GitHub provides a function for users to discuss the solutions or share ideas on the project by presenting various problems on issue, and adding comments.

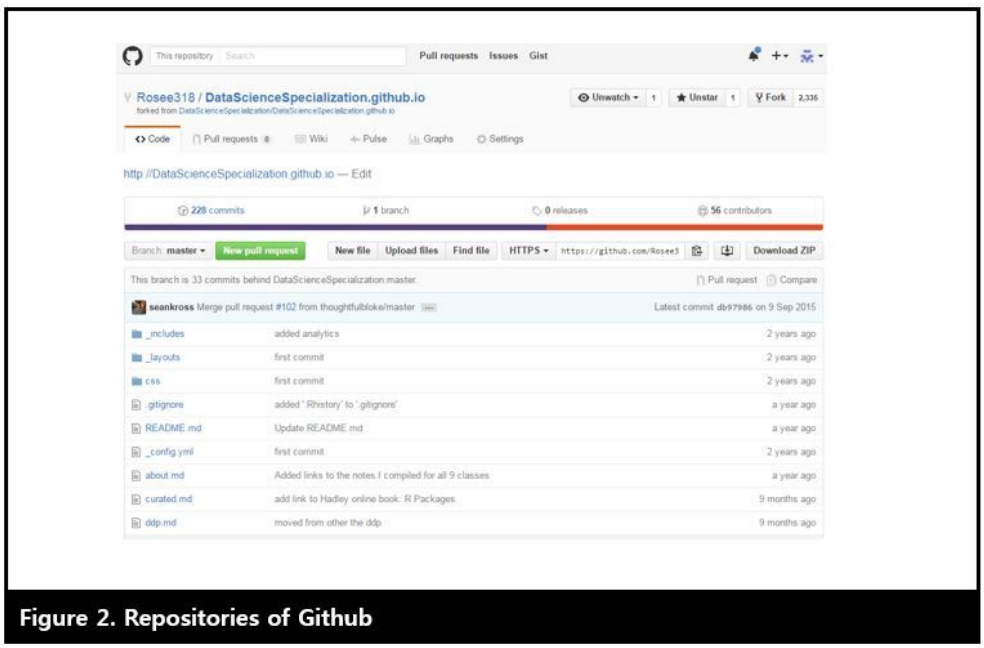


Figure 2. Repositories of Github

Users can also find contributor information or a graphic expression of the number of comments, and see information on the line of codes added by individuals. However, information on the number of users who starred the repository or the operation method of branches is not included. Such information can be regarded as informative transparency about how the repository is managed (Kalliamvakou, Damian, Blincoe, Singer, & German, 2015).

GitHub developers can create individual profiles and their development history or relevant project information can be checked. They can add pictures to their profiles, as well as personal information, such as name, email address, affiliated organization, location, and operating blog or website (Davis, 2015). Because GitHub is a platform that provides opened pages containing individual developer information, developers can collect data on profile information or career history of the project owner before they decide to participate in the project. Information on other developers, not the project owner, might be insufficient,

particularly at the beginning of development. Therefore, they will actively collect information on the owner to facilitate correct reasoning and decrease uncertainty regarding decision making of whether to participate in the project (Dabbish et al., 2012; Marlow, Dabbish, & Herbsleb, 2013).

Longo and Kelley (2015) explained the social networking function implemented site-specifically by GitHub. This function not only makes it possible to check participant information on their individual pages, but also identifies relationships among developers or their reputation through followed-follower relationship. It also provides quality information, including projects' fork, stare, pull request, and developers' information and their network, to analyze the factors that affect open collaboration. GitHub is also equipped with social functionality, in which developers can be connected to a network where they can check their interrelationship or reputation through followed-follower relationship (Longo & Kelley, 2015). When they follow one another, developers can check a series of activities, including newly initiated or participated project and bookmarked project status on their personal timeline. Within the project, they may propose codes, as well as share knowledge through certain issues. GitHub's provision of information about the OSS development process, cooperation relationship, or individual characteristics is unique because this information is not found in conventional OSS platforms. Therefore, it is helpful in broadening of new study prospects (Dabbish et al., 2012).

CHAPTER 3: [ESSAY 1] ROLE OF LEADERS IN OPEN COLLABORATION: FOCUS ON CHARACTERISTICS OF OWNER ON OSS PROJECT PERFORMANCE *

3. 1. Introduction

Unlike traditional companies, OSS development is created by significantly different employees, called voluntary contributors, through open collaboration. For the past 20 years, many studies have considered various aspects of OSS. However, few of these studies have been successful in developing OSS, and a significant number of projects have failed. As reported in Lima et al. (2014), such phenomenon is well described based on the results. These results have shown that even among the projects older than 18 months, only 62.9% have at least one line of code, and 46.7% have at least two or more developers including the project owner. Lima et al. (2014) indicates that those phenomena happen when projects do not appear promising to developers. However, the number of studies available cannot verify why developers choose specific projects to contribute their knowledge. This information can enhance the understanding of project success that can be achieved by active participation of contributors (Hahn et al., 2008).

Most studies on OSS focus only on the reasons why developers participate voluntarily in the projects (Hars & Ou, 2001; Hertel et al., 2003; Ke & Zhang, 2010; Von Krogh et al. 2012), and not on the factors affecting the success of a specific project (Hahn et al., 2008). In addition, most studies on OSS leaders focus mainly on implicit leaders, instead of explicit leaders who established the project and are named as its project owner.

Most projects fail in the initial stage when they lose opportunities to obtain support from developers other than the project owner (an explicit leader) (Lima et al., 2014). Therefore, the role of a project owner is important for a successful project in the initial stage. However, although various studies have

* Part of database in this research previously published in Lee, Baek, and Jahng (2016)

verified the roles of the leader in an OSS project (Giuri et al., 2008; Lerner & Tirole, 2001; O'Mahony & Ferraro, 2004; Raymond & Young, 2001; Yoo & Alavi, 2004), most of them focus on the emergent leader who comes out as the project goes on because in a consistently developed OSS, an important issue to study is the succession of roles of the initial project owner. The main goal of these studies has been to estimate the characteristics of potentially emergent leaders (Faraj et al., 2015; O'Mahony & Ferraro, 2004; Yoo & Alavi, 2004).

In this study, we focused on the roles of a leader among various factors affecting project performance. Developers who actively contribute to OSS usually make a complex inference and explore the project before they decide to put their time and effort in it. Such inference process is similar to the formation of personal impression to reduce uncertainties when trying to communicate with another person. However, most of the projects do not have sufficient information for developers to know them well enough and be targeted. Thus, to secure sufficient information, a theory-based verification can be made to estimate possibilities of a potential project through analysis of the characteristics of the owner (project founder). Therefore, to generalize research results, this paper examined the effects of owner characteristics on innovation performance of projects with a theoretical lens.

Therefore, in this study, we discussed project owner characteristics affecting innovation performance of the projects in GitHub, an OSS development platform, based on the impression formation theory. Although various studies have discussed open collaboration in OSS by exploring the project development process, (Dabbish et al., 2012; Dabbish, Stuart, Tsay, & Herbsleb, 2013), in-depth consideration on the advantages of information on the project owner remains insufficient (Stol & Fitzgerald, 2014).

In this study, we identified the effects of roles and characteristics of the owner on project performance using GitHub empirical data. We used project information, as well as personal data posted on individual websites of project owners on GitHub. GitHub is an internet platform where developers can collaborate with one another. Unlike other OSS development platforms, GitHub provides social features that enable followed-follower relationship among developers (Lima et al., 2014). GitHub also visualizes information on each

individual developer, and supports a bulletin board for developers that can be used to share knowledge within a project. Therefore, researchers can examine factors for collaboration that were not readily possible in previous studies on OSS development platforms.

With the aim of determining the effects of project owner characteristics on innovation performance, this study is structured as follows. In the second chapter, previous research on the roles of a leader is organized based on impression formation theory. In the third chapter, a research model and hypotheses are suggested. In the fourth chapter, we described GitHub data, which is an OSS development platform for testing hypotheses. In the fifth chapter, analytical results are suggested. Finally, in the sixth chapter, theoretical and practical contributions, as well as limitations of this study are explained.

3. 2. Theoretical Background

3. 2. 1 Role of OSS Project Leader

In particular, OSS developments do not require responsibilities nor have explicit rewards, which make voluntary collaborations necessary. OSS development naturally rely on the capability of leaders because they are operated in a way that skilled people contribute their knowledge, and maintaining such contribution until OSS completion is essential for the project. The motivation of contributors to offer codes relies on the governance mechanisms of the leader. The OSS development environment makes it easy for non-contributors, and hence, encouraging various people to contribute actively with their skills is an essential factor for a successful OSS project.

Previous studies have emphasized the role of a leader for a successful OSS development (Giuri et al., 2008; Mockus et al., 2002; O'Mahony & Ferraro, 2004; Raymond & Young, 2001). Unlike other online communities where people simply gather to share knowledge or interests, an OSS project has an project owner who establishes the project. The goal of the project is to develop software with intended purpose of the project. At the initial stages where characteristics of future

OSS are not yet settled, the project owner can set up license limits, purpose, or language of the OSS. In OSS development, the roles of the owner are as follows. The repository is where an OSS development project takes place. If developers suggest a new form or a changed code on the software to be developed, the project owner has the authority to approve such suggestion. Then, when the code is approved, as knowledge, the code can be contributed successfully to the repository. In such a way, the explicit authority of the owner is to approve codes (Cosentino, Izquierdo, & Cabot, 2014).

Various studies emphasize the importance of project owner, and their abilities on OSS development. Through the Linux case, Moon and Sproull (2002) argued the necessity of one excellent leader. According to Ye and Kishida (2003), developers can be classified according to their roles and influence, and the explicit leader has the most significant leverage and authority to manage the overall project and to complete and distribute the software. Lerner and Tirole (2001) suggested that an esteemed leader can contribute significantly to ensuring the success of the OSS. The leader does not need to be a technically skillful developer or the most skillful developer in the project; instead, the leader should suggest a goal to motivate collaborators and encourage developers with active communications with one another.

An OSS project under computer-mediated communication (CMC) environment has a different development condition from face-to-face (FTF) based projects (Faraj et al., 2015). CMC environment has an emergent leader in the collaboration process. The emergent leader comes out as developers communicate during the progress of the project (Faraj et al., 2015; O'Mahony & Ferraro, 2004; Yoo & Alavi, 2004). Emergent leaders do not pose explicit authority, but they are recognized as leaders among developers with significant activities, such as prominent contribution of codes or knowledge in the project or communication skills to encourage collaboration of other developers. In this study, emergent leaders are classified as implicit leaders. Many studies suggest the roles of a leader, but only a few studies have discussed the role of project owner and the practical influence of a leader on OSS performance. Lee, Baek, & Jahng, (2016) provides a summary of previous research on OSS project leaders (See Table 2)

.

Table 2. Previous Research on OSS Project Leaders

Reference	Types of Leader	Contributions	Limitations
Faraj et al. (2015)	Emergent Leader	· Identify impacts of three characteristics of emergent leader such as knowledge contribution, sociability, and structural social capital	· Limited setting for generalization (conducting the study based on three specialized technical discussion forums)
Giuri et al. (2008)	Emergent Leader	· Project leaders possess diversified skill sets to motivate contributors, and to coordinate contributors efforts	· Use secondary data which has limitation to interpretation, Survey 33 types of skills that may not enough to reflect real skills of developers
O'Mahony and Ferraro (2004)	Emergent Leader	· Important factors affecting emergent leaders (actively participated in a project, and knowledge sharing)	· Examine only one huge project
Yoo and Alavi (2004)	Emergent Leader	· Divide types message among developers into three categories (task, relationship, and technology-oriented) · Emergent leaders are related to the task-oriented message (not expertise but logistics-oriented) · Demographic variables are not important for emergent leaders	· Focus on e-mail communication behavior – partial picture of behaviors of developers in OSS project · Too small number of messages to generalize the results of study
Lerner and Tirole (2001)	Project Owner	· With high reputation, leaders could lead success of OSS performance. · Successful leader require not high technical skills but coordinator roles to provide motivation and effective communication	· No empirical test is conducted
Moon and Sproull (2002)	Project Owner	· Based on a Linux OSS project, the research identify that one skillful and brilliant leader is needed for successful project	· Focusing on one famous project, it is difficult to generalize to other OSS project

Source: (Lee et al., 2016)

3. 2. 2 Impression Formation

Open collaboration through the Internet is a CMC collaboration environment, which has many discrepancies from FTF communication environment occurring in the real world. Developers make complex inferences on their potential co-workers or the project before they decide to contribute with codes (Dabbish et al., 2012; Marlow et al., 2013), because information on other parties is crucial in cooperative

works (Gutwin, Schneider, Paquette, & Penner, 2004). In addition, uncertainties on a new project or a new developer can significantly affect OSS development, because participants have diverse skills, and they contribute to the project in many different ways (Marlow et al., 2013).

Impression formation is a behavior that aims to reduce the uncertainty and to improve the understanding and behaviors of other people for efficient communications (Gibbs, Ellison, & Lai, 2010). Walther (1992) suggested that people form an impression under online CMC situations from the given information on others, which is similar in FTF environments. Moore (2007) defined impression formation as a process through which individuals gather information and form an integrated impression on others by recognizing them. When people meet a stranger, they establish a mental model with various cues to learn of the other people's emotions and intentions (Antheunis, Valkenburg, & Peter, 2010). Sproull and Kiesler (1986) suggested that the biggest difference between FTF and CMC communications is the existence of the social context cue, which is delivered through physical circumstances or non-verbal factors. Under the CMC environment, individuals fill out the mental model with other information that can be attained from the interface or conversation, because of lack of non-verbal cues.

3. 2. 3 Impression Formation and OSS Projects

In case of OSS development, according to Lima et al. (2014), developers need to understand fully the goals and structure of the software to allow them to access, determine, and contribute to a project. Therefore, recognizing the atmosphere or structure of repositories is complicated, but necessary to developers. Forming impressions by collecting developer and project information has a significant role in enhancing the efficiency of cooperative work and in understanding the possibilities or goals regarding the projects. However, only a few studies on the impression formation on OSS projects have been made (Bosu et al., 2014; Marlow et al., 2013; Tsay et al., 2014).

Bosu et al. (2014) conducted a survey to investigate the impression of

developers on their co-workers within an OSS project. They verified the differences in the ways that impressions are formed within the OSS communities from that in traditional communities, factors affecting the impression formation, and the transition process of impression formation. In particular, they classified the factors with several items, including productivity, capability, ease in cooperation, recognized expertise, and credibility. Marlow et al. (2013) conducted a survey to investigate the impression of developers on their co-workers within an OSS project. They verified the differences in the ways that impressions are formed within the OSS communities from that in traditional communities, factors affecting the impression formation, and the transition process of impression formation. In particular, they classified the factors with several items, including productivity, capability, ease in cooperation, recognized expertise, and credibility. Marlow et al. (2013) analyzed the impression formation on developers who submit codes from the perspective of an project owner. In that study, they used the interview method and spoke with 18 developers. They also showed the interviewees the profiles of those who submitted codes, and made them form an impression. The interviewees were then asked if they would accept the codes. The study was aimed at determining the effects of open information of contributors on the evaluation of the codes they suggested. Considering that a code submitted by a professional or experienced developer has a high possibility of being useful, expertise factor positively affected the possibility of code acceptance. On the contrary, developers who submit codes also considered the expertise of the project owner who examines the code with strict evaluation standards.

3. 3. Research Model and Hypotheses

3. 3. 1 Research Model

In this study, an analysis of the factors affecting OSS project performance was conducted, with particular focus on the personal characteristics of the owner. The commit number, which refers to the number of codes contributed by the developers, is used as an index of project performance. Owner characteristics consisted of the

information accessible through GitHub interface: the openness of personal information, number of followers, number of project subscriptions, knowledge contribution (number of code contributions within one year), and the number of following (See Figure 3).

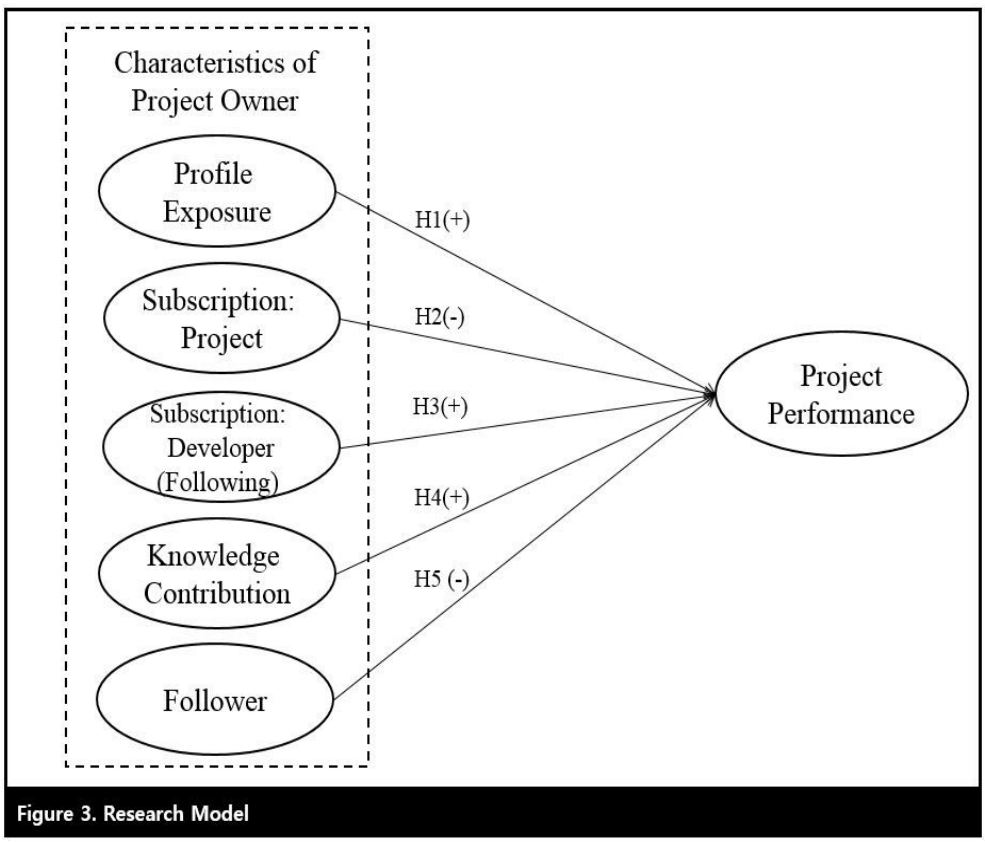


Figure 3. Research Model

3. 3. 2 Hypotheses

Providing information for impression formation before developers participate in certain OSS projects has two main advantages. First, abundant information improves the credibility of other developers under CMC environment, which is in the same context as to verify the information source when evaluating the credibility of information in an online community environment (Briggs, Burford, De Angeli, & Lynch, 2002). Baek, Ahn, and Ha (2011) also found that the helpfulness of their posts increases when online reviewers reveal their real name. The meaning of helpfulness is that the review of a certain products is useful to purchase decision makings. Therefore, the credibility of a review is strongly associated with the

helpfulness of the review. With same context, Ma and Agarwal (2007) suggested that publicizing user information also affected the reputation and credibility of the information source.

The efficiency of collaboration can be boosted through the information provided by the owners. Previous studies identified that developers collect information on other developers to facilitate efficient collaboration. Consequently, possibilities of a successful collaboration increase as more information is opened (Dabbish et al., 2012; Dourish & Bellotti, 1992; Gutwin et al., 2004). Dabbish et al. (2012) argued that people tend to decide their next behavior based on their estimation of the behavior of others that are revealed online. Gutwin et al. (2004) revealed that when people participate in cooperative work for the first time, they observe who their opponents are and who are working in which project. The study also showed that people tended to collect information on potential co-workers before beginning to work with them. Such functionality is useful in estimating the behavior of developers (Gutwin et al., 2004). Such information can be used by developers to understand who works with them, what they are doing, and how their behaviors can affect others (Dourish & Bellotti, 1992).

Previous OSS development platforms did not provide separate personal information, but only granted access to text-based information. Conversely, GitHub is a platform that provides a page where individual developers' information is open to the public. Thus, developers can check the profile or activity history of the project owner before they participate in a project. GitHub contains personal information, such as e-mail address, location, or companies, which allows developer to collect information on the project owner to reduce uncertainties and form accurate inferences (Dabbish et al., 2012; Marlow et al., 2013). In this study, regarding the personal information publicized by a project owner in their personal page, we hypothesized that the reliability of the owner can be improved and efficient collaboration can be enhanced as more information is publicized.

Hypothesis 1. The amount of information in the profile of the project owner will positively influence on OSS project performance

In GitHub, each developer can subscribe to various functions. Through the

functions, developers can subscribe to information for changes of repositories and activities of other developers. The purpose of subscription can differ depending on the information type. Subscribing to information on repositories means that subscribers are interested in the project, and they want to learn specific changes or issues. Similarly, subscribing to information on other developers means that the developers are technically skillful or subscribers prefer to communicate with them as the function makes it possible for them to exchange messages. Therefore, in this research, we divided the roles of subscription into two: subscribing to repositories called starred, and subscribing to other developers called following.

By subscribing to repositories, developers can acquire new knowledge from other developers. Therefore, the main purpose of subscribing to repositories is to learn. Subscribing to many repositories serves as proof that the developer is interested in many artifacts and information Dabbish et al. (2012). However, as argued by the attention-based theory, excessive task information can bring negative results. From the perspective of learning, Koput (1997) explained the disadvantages of exploring an excessive amount of information. When an excessive amount of information is arranged, people who have to accept it will be required to make many choices and management issues. Excessive information would not be helpful to pertinent organizations, but its value depends on the situation. Therefore, forming a good decision after accepting much information to reflect on would not be an easy task (Ocasio, 1997). In accordance with attention-based theory, organizational learning theory also states that excessive information is not helpful for acceptance and reflection of such information. In instances where there are excessive channels, people become too distracted to concentrate on certain information (Tseng & Chen, 2009). Therefore, the developer who receives much information on a project can form an impression that the concern can be dispersed rather than accepting and reflecting on the refined information with specific concern. The hypothesis for receiving the owner's project is as follows.

Hypothesis 2. Subscription (projects) of project owner will negatively influence on OSS project performance

Faraj et al. (2015) explained that a leader should have an active attitude in helping

others or in facilitating communications. Under CMC environment, interest in others and efforts to communicate are important factors to motivate new developers to join in the project (Moon & Sproull, 2002). Lima et al. (2014) asserted that unlike other social media service users chasing hedonic pursuits, GitHub developers do not want a busy timeline to prevent confusion in the development process, as their purpose in using the service is to develop an OSS. Therefore, they answered that most OSS developers form following relationships only with those who need communication, instead of forming networks reciprocally with various people. Nevertheless, when developers have a large following relationship, it can imply that they have higher interest than others, and have higher preference in forming correlations with others. Therefore, in this study, an assumption was made that an owner with a large number of followings will have a higher interest in others.

Hypothesis 3. Subscription (developers) of project owner will positively influence on OSS project performance

GitHub provides automatically formed information, involving activities of each developer within GitHub, along with the information that individuals entered themselves intentionally. Such automatically formed information is helpful in facilitating efficient cooperation. GitHub provides accumulated numbers of knowledge contributions of developers within the last year, even when developers are not required do so. Therefore, others can easily recognize whether a developer is an active volunteer.

Marlow et al. (2013) analyzed the impression formation process regarding developers who submit codes from the perspective of project owners. Considering that a code submitted by a professional or a well-experienced developer has high possibility of being useful, the expertise factor positively affects the possibility of accepting codes. Faraj et al. (2015) also indicated that knowledge contribution is one of the three major factors affecting leaders' abilities. Therefore, leaders can achieve recognition within a project when they actively participate in the knowledge contribution process. Dahlander and O'Mahony (2011) pointed out that the technical contribution is important in authority role. Technical contribution

within an OSS development project can be the production of codes. Spending time and effort in technical contributions leads to achieving goals. When developers contribute to a specific project more than others, they will gain more credibility as a reliable decision maker (Baldwin & Clark, 2006; MacCormack, Rusnak, & Baldwin, 2006). Finally, Von Krogh et al. (2003) suggested that a person with a greater amount of knowledge is believed to have better competencies in addressing more complex problems. Therefore, in this study, a hypothesis was established as follows, regarding the effects of the number of owner's activities within the past year on OSS project performance.

Hypothesis 4. Knowledge contribution of project owner will positively influence OSS project performance

Blincoe, Sheoran, Goggins, Petakovic, and Damian (2016) used the survey method to identify the perspectives of users who follow famous people who have a significant number of followers. Findings showed that about 40% of users recognized the popular user as an expert. Blincoe et al. (2016) verified the reasons of GitHub users to follow others, and the influence of popular users on their followers. Research involving 199 popular users showed that popular users act as messengers of a newly opened project to many other developers. However, although the range of influence increases with the number of followers, the number of followers does not have a significant relation with OSS development performance. Based on Blincoe et al. (2016) we assumed that a large number of followers indicate a highly expert developer. Many studies have focused on the technical abilities of leaders to enable a successful OSS development. Giuri et al. (2008) explained that capabilities of leaders should involve not only technical abilities, but also the abilities to discover useful codes from various participants, motivate external developers, and encourage cooperation. Lerner and Triole (2000) emphasized that a leader does not need to be someone with the best skills, nor someone who has special abilities. In addition, when the owner is recognized as an expert, division of responsibility (Darley & Latane, 1968) can occur, which in turn causes inactive participation among external developers, which consequently has negative effects on the result. This recognition can be associated with

discouragement of developers. Because the expertise of the owner is outstanding, external developers might feel that they do not need to contribute actively to the project. Lerner and Triole (2000) argued that a leader should avoid contributing too many codes to ensure that other developers can accumulate experience and feel self-satisfaction. Based on the assumption, a hypothesis is derived.

Hypothesis 5. Followers of owner will negatively influence on OSS project performance

3. 4. Data Collection and Analysis

3. 4. 1 Data Collection

For this research, GitHub data regarding repositories created during the period from February 2014 to February 2015 were collected. Among them, a total of 1,850 repositories that gained five or more subscription (project) (bookmark function that developers use to show their interest and see the change log of a specific repository) weekly were first sorted. Then, 600 repositories were selected as final subjects of analysis under the following criteria: those with individual owners, and those with more than two contributors, considering the definition of cooperation. An activity period control was created on the 600 repositories by collecting data eight months after the creation of each repository. Eight months of the period was chosen because it shows a significant measurement of project performance. For data collection, web crawler was established and utilized based on Python.

3. 4. 2 Variables

In this study, the total number of commit of projects was used as a measurement tool to evaluate the dependent variable, which is the OSS project performance. Many existing studies use the commit number of a project or an individual to measure performance. Crowston et al. (2003) considered the commit number to

measure the success of an OSS project based on performance during the development process. Existing articles used the commit numbers of contributors as individual performance when measuring the level of contribution of each individual developer (Adams, Capiluppi, & Boldyreff, 2009), and Grewal, Lilien, and Mallapragada (2006) used commit numbers to measure technical success.

Considering the type of personal information, we divided the information into three types: 1) information that can be edited manually, such as bio information; 2) information that express activities of developers, which is automatically accumulated as a GitHub function, such as subscription activities or knowledge contribution; and 3) information of developers that does not reflect their activities directly, such as followers. Editable personal information is shown below the profile picture. In this study, we measured the number of open profile information from each individual project owner's personal website within GitHub. In GitHub, users can enter their email address, website, location, company, and a brief history manually. Among such data, brief history is text-based information, and hence was excluded from the open profile information measurement. Data are classified as numbers 0 and 1, where 1 is given when information exists, whereas 0

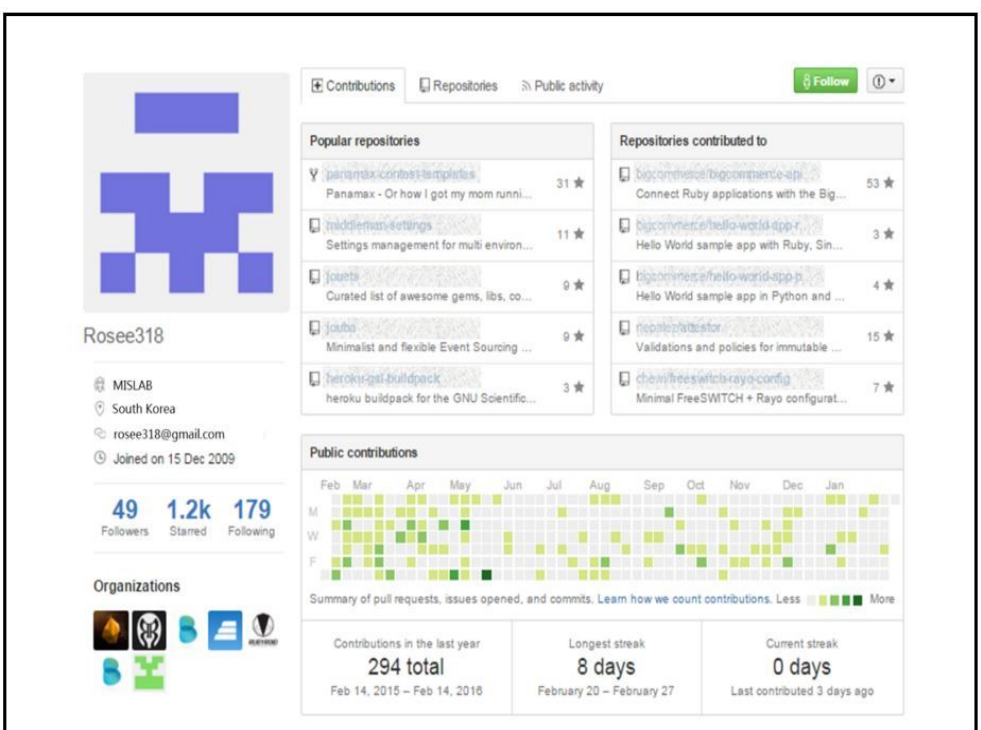


Figure 4. Individual Webpage of Github

is given when it does not. Finally, a sum of the score in each category (minimum=0, maximum=4) is used as measurement items.

GitHub also shows the numbers of followings or followers of each individual developer. In this study, the numbers of followers and followings found in project owners' profiles are used as a measurement tool. <Figure 4> shows the personal information of developers, as well as activity information within GitHub.

Social features among developers are created through the follow function located on the upper right end of the screen. In addition, accumulated social status is expressed in the numbers of followers and followings on the lower left corner automatically. In addition, if the meaning of following is subscribing to the developer, the function of subscribing to projects is starred. Therefore, we used starred as a measurement of subscribing projects. Finally, considering that the number of contributors is proportional to the commit number, the total number of contributors in a project is considered as its control value. <Table 3> shows the variables collected through GitHub for this study.

Table 3. Description of Variables

Variables		Measurement
Dependent Variable	*(Ln)Project Performance*$_i$	Total number of commits in repository i
Independent Variables	*Profile Exposure*$_i$	Number of profile Information of owner in repository i
	Subscription(Project)$_i$	Number of repositories that owner subscribe in repository i
	Subscription(Developers)$_i$	Number of following of owner in repository i
	Knowledge Contribution$_i$	Number of owner's last year contribution in repository i
	Follower$_i$	Number of followers of owner in repository i
Control Variable	*(Ln)Project Size*$_i$	Number of contributor in repository i

3. 5. Results

3. 5. 1 Descriptive Statistics

Table 4. describes the descriptive statistics of each variable. Analysis on the targeted 600 repositories showed that the average commit number of each repository was 3.69, and the average of the personal profile information is 2. The average of subscription (project) is 4.45, with minimum=0 and maximum=8.16. In terms of the number of followers (subscription-developers) and followings (i.e., social function), project owner with a large number of followers have 622.66 on average, but one owner had a maximum of 11,801 followers. On the contrary, the average number of followings is 48.66. Finally, regarding the number of the contributors (that is, the control variable), the average value is 1.29. We adopted log into commit number, starred, and contributors for normalization of data distributions.

Table 4. Descriptive Statistics

Variables		N	Mean	SD	Min	Max
Dependent Variable	*(Ln)Project Performance*$_i$	600	3.6860	1.0209	.0000	6.6619
Independent Variables	*Profile Exposure*$_i$	600	2.02	1.049	0	4
	(Ln)Subscription(Project)$_i$	600	4.4511	1.6798	.0000	8.1605
	Subscription(Developers)$_i$	600	48.66	119.193	0	1325
	Knowledge Contribution$_i$	600	658.93	1084.642	1	6718
	Follower$_i$	600	622.66	1444.009	0	11801
Control Variable	*(Ln)Project Size*$_i$	600	1.2919	.6853	.6931	4.6250

3. 5. 2 Results

The study used hierarchical regression analysis model to verify the hypothesis, as shown in Table 5. The hierarchical regression analysis is a method used to evaluate

the effects of the independent variable by explaining quantity, except for the covariance of the entire account quantity (R^2), by injecting variables that researchers will see in order. As a result of injecting in stages considering the covariance of main variables, the account quantity of OSS project size, which is also the control variable, is 27.1% (Model 1). Subsequently, the disclosure degree of user information (B=.097, p<.01), which is the individual characteristics of project owner, has a static effect on project performance, and the account quantity increased to 28.1% (Model 2). Then, the numbers of subscription (developers and projects) which are the information on attention for other developers and tasks, are injected as variables. As a result, the (Log) subscription (project) (B=-0.053, p<.05) is shown to have a negative effect on project performance; the subscription (developers) (B=.001, p<.1) has a static effect; and explanation ability is 28.9% (Model 3). Dabbish et al. (2012) argued that people tend to establish their behaviors based on the estimation of others' behaviors revealed online. GitHub provides automatically formed information involving the activities of each developer within GitHub, along with the information that individuals entered themselves intentionally. Such information is helpful in facilitating efficient cooperation. From this perspective, we assumed that contributions in a year and followings are automatically formed information through activities of developers or actions followed by other developers. Therefore, this study analyzed contributions in a year and followings as information for developers with relatively small intention, compared to the previous three kinds of information (profile exposure, subscription (developers), and subscription (projects)). Knowledge contribution (B=6.5E-5, p<.1), which contains information on code contribution activity for one year and items that the owner indirectly acts upon are accumulated as an indication of positive effect on project performance. The model explanation ability at this time is 29.3% (Model4). Finally, followings (B=-.0001, p<.05), which is the information created by other developers' activity regardless of the owner's intention, has a negative effect on the project performance. The explanation ability of Model 5 is 30.6%. As a result of hierarchical regression analysis, when the independent variable is injected to all five models, the increment quantity of explanation ability (F value variation) is significant. Therefore, among the individual features of the project owner, having many stars on other repositories

or having many followers has a negative effect (supports Hypothesis 2), while disclosure of profile information and followings has a positive effect (thereby supporting Hypotheses 1 and 3).

Table 5. Results of Hierarchical Regression

	Model 1		Model 2		Model 3		Model 4		Model 5	
	B	SE	B	SE	B	SE	B	SE	B	SE
(Ln)Project Size$_i$	.776***	.052	.776***	.052	.776***	.052	.771***	.051	.776***	.051
Profile Exposure$_i$			.097***	.034	.112***	.034	.132***	.035	.132***	.035
(Ln) Subscription (Project)$_i$					-.053**	.023	-.063**	.024	-.049**	.024
Subscription (Developers)$_i$					.001*	.000	.000	.000	.0003	.000
Knowledge Contribution$_i$							6.5E-5*	.000	.0001***	.000
Follower$_i$									-.0001***	.000
Cons	2.684	.076	2.488	.102	2.666***	.132	2.654***	.132	2.592***	.132
R^2	.271		.281		.289		.293		.306	
Adjusted R^2	.270		.279		.284		.287		.299	
ΔF	222.519***		8.203***		3.155**		3.363*		11.862***	
N	600		600		600		600		600	
VIF	1.00		1.00		1.10		1.14		1.23	

※ Dependent variable: *(Ln)Project Performance*$_i$

Note: * p < .1, ** p < .05, *** p < .01

3. 6. Conclusion

3. 6. 1 Discussion

This study confirmed that by disclosing more information on their individual

profile, project owners can gain more trust from potential developers, which consequently have a positive effect on the project result. The effect is verified using the social function (the follower and following information) that project owner having many followings had a no effect on the results, while having many followers has a negative effect. The former shows that owners are interested in other developers, while the latter implies that owners are considered as specialists and experienced in a certain field, but cannot motivate developers who to participate and cooperate. In addition, the results of our research indicate that the number of followings which means are attention towards other developers with highly willingness to communicate, have no impacts on project performance. From this results, we can assume that Github is not platforms for communications but collaborations through the knowledge contributions. Therefore, the impacts of following does not statistically supported. Furthermore, subscriptions of other project as attentions towards tasks negatively influence to projects success. This reflects that the attention of owner are scattered to various projects. Since it is impossible to implement every information from the subscription of projects, developers who make impression from the information of owners also understand the limitation of attentions towards information. Furthermore, developers who eager to learn from other project conversely represent that the developers are lack of knowledge in software development. Therefore, subscriptions of project negatively impacts on the impression of owners with high reliabilities and experts. In addition, knowledge contributions last one year also positively impacts on the project performance. This results indicate the passion or activation of owner on OSS development is important factors to determine the success and potential of projects.

3. 6. 2 Conclusion

This study verified the main factors for successful project development in developing OSS, which shows that open collaboration is being conducted. The theme of the study is to identify why some projects succeed and why some fail. This study focused on the role of owners, the explicit leaders who owns the

projects, in conducting the analysis. The effect of the owners' individual nature on the project result is considered in detail by using GitHub data, focusing on impression formation theory.

To understand the project they will contribute in, OSS project developers explore available information of the project itself and the project owners, during which they form an impression. The theoretical/empirical implications of this study are as follows. For theoretical implications, the study confirmed the application of impression formation process by developers for FTF and CMC situations in choosing the project through the owners' developer webpage information, which leads them to the results. Some past studies explained a cooperation process through impression formation, focusing on various activities in the OSS project (Bosu et al., 2014; Marlow et al., 2013). However, while most studies are limited to the impression formation on external developers suggesting new codes, this study explained the process of choosing a certain project in the position of the external developer.

This study used GitHub, an OSS development platform, from which we academic contributions can be found. GitHub is equipped with a bulletin board that provides information on their developers and the social function among them, and enables developers to share their knowledge in the project, unlike in existing project-oriented OSS development platforms. Therefore, OSS study reports can verify the factors, which were not considered in past studies, more deeply. This study is significant in verifying the hypotheses by using actual data, such as the developers' information and social function, which were provided by GitHub.

Results of this study can be used as a guideline for operators who want to manage successful OSS projects. Although many people open OSS projects to achieve a certain goal, it is not easy to operate them successfully. Recently, traditional firms have experienced limitations in labor, which they solve by providing materialistic rewards for skilled laborers. They have made various efforts to use open collaboration model as a benchmark for voluntary contribution (Markus & Agres, 2000)(Markus & Agres, 2000). However, the leaders' role is important even in a voluntary contribution environment. The leaders' characteristics or operational methods can also affect project results.

The limitation of this study is in verifying individual subjective

psychological process, which is also called the impression formation, by means of actual data, which limits reinterpretation to be based only on existing theories and studies through GitHub's secondary data rather than analyzing the psychological process that individuals felt on the technique, such as through interviews or questionnaires. Therefore, future studies will be able to reveal more detailed impression formation processes through qualitative study method and conducting hypothesis verification using the actual data.

CHAPTER 4: [ESSAY 2] CHARACTERISTICS OF ORGANIZATION IN OPEN COLLABORATION: BASED ON RESOURCE ALLOCATION AND ORGANIZATIONAL STRUCTURE

4. 1. Introduction

Various studies have verified that factors that affect the innovation performance of OSS. Because of the inherent nature of OSS project, various companies try to introduce its atmosphere or mechanisms to encourage their employees to make profit for the firm (Sharma, Sugumaran, & Rajagopalan, 2002). Regardless of these enviable attempts, many OSS projects fail easily (Chengalur-Smith & Sidorova, 2003; Lima et al., 2014). Conversely, OSS project development communities need to adopt successful strategies from traditional firms to sustain and enhance project performance.

While some OSS projects are operated by individual developers, some are managed by multiple developers who formed the organization. A comprehensive project that needs to be developed consistently by various people should be operated by organization members. An organization is composed of people who share the same goal (Venkatraman & Henderson, 1998) or, have done various project activities together. Organizations that manage various projects have a scale similar to that of traditional companies, as well as the governance problems that managers face.

Therefore, we assumed that governance mechanisms will affect innovation performance of OSS development organizations (Scacchi & Jensen, 2008). In this study, we identified how organizational characteristics established by certain governance style influence innovation performance. The ratio of participation of individual developers on multiple projects and organizational structure is one of our research concerns.

In an OSS project, as control and governance mechanisms are dispersed in project level, making it difficult to operate with explicit centralized authorities.

Therefore, understanding intrinsic issues, such as development activities, project communities, surrounding organizations, and collective actives, is crucial (Scacchi & Jensen, 2008). Scacchi and Jensen (2008) explained that these intrinsic issues are related to decision-making authority, resource allocation, motivation theories, leadership, coordination mechanisms, and organizational structure. Ghosh and Prakash (2000) identified that 10% of developers produce around 70% of codes in the project, and 10 other developers make 25% of codes. In addition, only 25 developers participated on more than 25 projects. Most of the developers participated in only one project. Furthermore, from famous and large OSS projects, such as Apache or GNME, a small number of developers produce most of the codes (Bonaccorsi & Rossi, 2003). This phenomenon indicates opposite results of the purpose of open collaboration. Various researchers emphasize the decentralized and open communication of knowledge-sharing within OSS project development, such as open collaboration (Raymond, 1999). However, empirical data show a certain structure or hierarchy of OSS projects. Therefore, this research verified the structure of OSS communities on innovation performance. Although many studies analyze the structure of OSS project and its effects on performance, they only used one or two case studies or just focused on project-level analysis.

Previous studies on effective project management were analyzed (Barreto, Barros, & Werner, 2008; Cooper, Edgett, & Kleinschmidt, 2001; Turner & Speiser, 1992). Project management is a prominent field in organizational structure studies. With the rapid growth of ICT, companies can handle multiple projects simultaneously (Ahuja, Yang, & Shankar, 2009). In R&D environment, companies need to allocate limited resources on various projects efficiently. Technology-based environment, in particular, requires specialized knowledge. Human resource allocation issues are crucial for successful project management (Hendriks, Voeten, & Kroep, 1999). Therefore, in this research, we verified the efficient resource allocation strategies in OSS development communities that operate multiple projects.

In particular, this study analyzed the effects of human resource allocation as project participation and organization structures with organization-level analysis. Furthermore, to analyze project management mechanisms of organizations, we facilitated social network analysis (Kwak, 2014) to determine the interrelation

among members of an organization through developer-project relationship. Finally, a regression model analysis is conducted to evaluate the effects of organizational management characteristics on project performance.

To determine the effects of the characteristics of development organizations on OSS performance, this study is structured as follows. In the second chapter, previous studies with theoretical background, including project management and social capital, are reviewed. In the third chapter, a research model and relevant hypotheses to verify the model are suggested. The fourth chapter explains data from GitHub, the OSS development platform used for this analysis. In the fifth chapter, analytical results are suggested. Finally, the sixth chapter presents the conclusion of this study, as well as its limitations.

4. 2. Theoretical Background

4. 2. 1 Organizational Structure

In organizational structure, decentralization defined as decision-making discretion is pushed down to lower levels of the organization (Lin & Germain, 2003). For several decades, there are inconclusive results among the relationship of organizational structures on firm performance. Mintzberg (1979) assert the negative effects of formalization which refers to the presence of written rules and procedures when it leads the organizations to wrong directions and undesired conformity in planning. However, if the firm collect valuable information and conveys priorities, it may positively influence to firm performance. In contrast, decentralized organizations are more flexible, creative, and efficient than formalized organizations (Adler, 1999).

In an OSS project, a network is formed among developers when they share ideas and work collaboratively (Xu & Madey, 2004). The actors in this network can be regarded as the developers, and the link is the cooperative relation. Using the cooperative relation network, the network type of a specific project can be understood, and the position and influence of a specific developer within the network can be identified (Xu & Madey, 2004). Various studies on OSS have

focused mainly on the ways developers participate in OSS projects (Singh & Tan, 2010). They also verified the effects of developers' relationship on OSS project performance by analyzing it as a network.

In OSS development communities, a centralized structure indicates that few developers contribute most of the codes, while other developers write the remaining codes (Crowston & Howison, 2006)(Crowston & Howison, 2006). Conversely, in a decentralized structure, developers contribute to the OSS project evenly. Traditionally, a hierarchical structure describes authorities of managers. To adopt these perspectives to OSS development, a few developers have more power to codes compared to other developers. Crowston and Howison (2006) analyzed the hierarchical structure of OSS projects not only with code contributions, but also with communication structures. Weber (2004) determined the various organizational structures according to OSS project types. Large-scale projects often have an organization, instead of an individual as its owner, and thus, their organizational structure, including external developers, can be understood. Linux projects were identified to have a pyramid structure, whereas BSD projects have a centralized structure that resembles a concentric circle. However, such identification cannot be verified quantitatively. Therefore, one of the study's limitation that it is based on the direct observation of authors.

Grewal et al., (2006), Singh, (2010), and Singh, Tan, & Mookerjee, (2008) suggested that the network structure affects knowledge sharing, which consequently affects project performance. In the past, studies that sought to determine the communication structure among developers based on email lists were prevalent (Bird, Gourley, Devanbu, Gertz, & Swaminathan, 2006). One study found that power law exists within a project's communication structure, and that an active change of codes is performed according to developers when they pose the power law within communications. However, because of the limitations of data, the observation was conducted on only one project, and hence, could generalize the results. Since then, many studies have targeted a larger number of projects on communication network and code development. Bonaccorsi and Rossi (2003) also described a successful OSS project. As volunteers contribute to OSS projects and create organizational structure, every developer requires proper leadership and coordination mechanisms. In addition, the study asserted that most successful

projects have a hierarchical structure, which is not specific or well-structured at the initial stage. However, as long as communication and knowledge contributions are sustained, the organizational structure will come to the front.

Studies on OSS use Social network analysis (SNA) method as a tool to observe properties of OSS communities (Baek & Oh, 2015; Gregorian, 2014; Hahn et al. (2008); Singh & Tan, 2010; Xu & Madey, 2004). SNA can be used as a methodology to understand the relationship among society members and to measure social capital. To facilitate SNA, each member of the society is classified as an actor in the network, whereas the relationship among members is expressed as link. A number of studies have conducted empirical analyses on the relationships and patterns of actors (Freeman, 1978), and they observe social phenomena by analyzing actors as the factors in the network (Gregorian, 2014). The position and relationship of actors play an important role in network efficiency (Jackson, 2005; Jackson & Wolinsky, 1996). The network analysis method has been widely used in various studies to grasp and analyze meanings. For example, co-authorship analysis among researchers can be used to observe cooperative relations (Gregorian, 2014), and a similar method can be used in disease research to understand the diffusion paths of a disease (Jackson, Kirkland, Jackson, & Bimler, 2005).

Crowston and Howison (2005) formed a network within a project and focused on the mailing list of 52 different projects. The study found that large-scale projects tend to be modulated internally to form multiple small-scale networks. A verified the relationship between network property within a project and developers' performance. Through SNA, Hahn et al. (2008) showed that past cooperative relationships have a positive effect on the participation in newly established projects. Singh and Tan (2010) also revealed that developers with different skills and abilities form networks, and found a network structure especially effective for the performance. Howison, Inoue, and Crowston (2006) conducted a chronic observation of OSS development communities to verify the effects of the structure's centrality or the stable participation of developers on projects. Through SNA, researchers systematically verified the organizational structures. Organizational structures have long been studied in many areas, including personnel management, MIS, and strategy. However, with the recent advancements in IS, more types of organizational structures have become available. Developers of

OSS projects contribute by considering their needs and the needs of projects themselves, the structure of organization is informal and develops in an unplanned manner. Developers can form a network structure in a desirable manner (Singh & Tan, 2010)(Singh & Tan, 2010). In addition, project structures vary for each project. Madey (2005) also investigated the network structure of developers who participate in multiple projects. They claimed that both project size and number of projects can follow power-law relationships. They extended Barabasi's network model on data at SourceForge. Insights on the role of Linchpin developers linking two projects is provided in this research.

Various studies that target network concept to show the relevant positions of individuals or to understand whether they form social capital within a network or a project based on communication network have been conducted. However, studies analyzing organizational characteristics at the level of organizations that operate OSS projects remain insufficient. Rullani and Frederiksen (2010) also identify interactions between developers through multi-project. The paper asset that OSS projects are not isolated since developers are involved in multi-project simultaneously. Through the interconnection among the project, the developers could share the knowledge each other. Although the paper try to explain the interrelationship between developers the boundaries of networks are not limited to certain organizations.

4. 2. 2 Project Management

Recently, with the rapid development of the Internet and ICT, companies can manage multiple projects simultaneously. Each employee can also participate in multiple projects at the same time. Therefore, the importance of project management has become much more significant. Previous studies on project management focused on efficient and effective decision making for a successful project performance. Liberatore (1987) used the AHP model to analyze the effects of decision making on resource allocation in R&D project management environment. Hendriks et al. (1999) also considered how specialized human resources need to be allocated and scattered in multi-project environment in R&D

sectors. In software development, Barreto et al. (2008) claimed that allocating proper developers in proper projects can be a complex and difficult step, considering the developers' area of expertise and the needs of a company. Other studies have also studied how human resource involved in multiple projects can specialize, or how it can maximize the efficiency of performance with resource allocation perspectives (Certa, Enea, Galante, & Manuela La Fata, 2009; Engwall & Jerbrant, 2003).

Hyatt and Ruddy (1997) analyzed successful projects and their antecedents, such as work group morale, work group support, commitment to common goals, structure of group activities, and schedule. In this research, we regarded resource allocations that vary with organizational characteristics. From previous research, governance mechanisms for human resource management can be organizational characteristics because they are established by the firms' decision making.

Unlike traditional companies, organizational governance within the CMC environment is formed spontaneously, which is similar to the OSS development environment (Singh & Tan, 2010). The OSS project is a place where the position of developers is determined by their contributions because of the self-organizing property (Singh & Tan, 2010), and not by the owners who have the authority. Several studies have analyzed the various tasks of developers within an OSS. Ye and Kishida (2003) classified developers according to their roles and importance of their respective tasks within an OSS project. The classification involved the project leader, key member, active developer, peripheral developer, bug-fixing-oriented developer, bug reporter, subscriber, and passive user. Although such classification can be applied to large-scale projects, small projects hardly require such structure. The research also revealed chronological discrepancies in the key roles of developers. The role of the owner is especially important in the initial stage; but after the project is established, the project becomes modulated, placing more importance on the participation of external developers.

Consequently, in OSS projects, allocating developers with specific knowledge in the right time and the right project can be a difficult task, and so, voluntary participation is encouraged. Scacchi and Jensen (2008) asserted the importance of OSS governance in self-organizing manner. However, if projects are

modulated, developers with specific skills may hardly contribute fully to the project.

4. 3. Research Model and Hypotheses

4. 3. 1 Research Model

This research studied characteristics of organizational structures and their relevant effects on organizations' innovation performance by analyzing the types of participation ratio in the projects. First, we identified the effects of the types of participation within an organization-managed project on organizational innovation performance. Then, we considered the participation ratio; the average of project participation ratio of each individual developer among the projects they are involved in and managed by the organization. Second, organizational structures are verified based on centralization concepts. Organizational structure is the extent to which project participants are connected within an organization (See Figure 5).

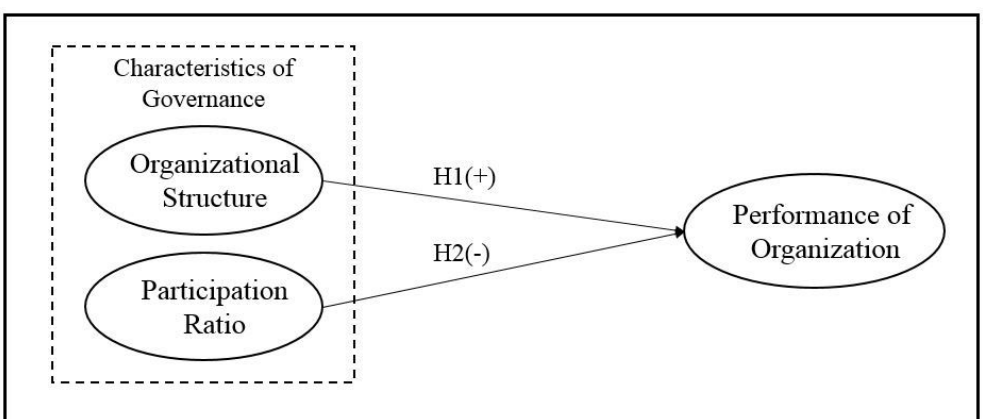


Figure 5. Research Model

4. 3. 2 Hypotheses

Our exploratory research describes the status of multi-project participation of

developers. Figure 6 shows the dispersion of developers in various projects. The data set focus on developers involved in OSS development organizations. Figure 6 presents that developers who participate in only one project number approximately 73044. By contrast, several developers simultaneously participate in over 100 projects. Accordingly, the number of developers involved in multiple projects is relatively small. In this case, the number of projects participated in indicate the scatter factor of developers. Evidently, developers involved in only one project are different from those who work on multiple projects in terms of effort, attention, or energy toward each project. Therefore, this study focuses on the number of projects that developers are involved in as an important factor affecting their performance, as well determines the tendency of centralization in project participation. Finally, only a few developers participate in multiple projects, whereas numerous developers participate in merely one or two projects.

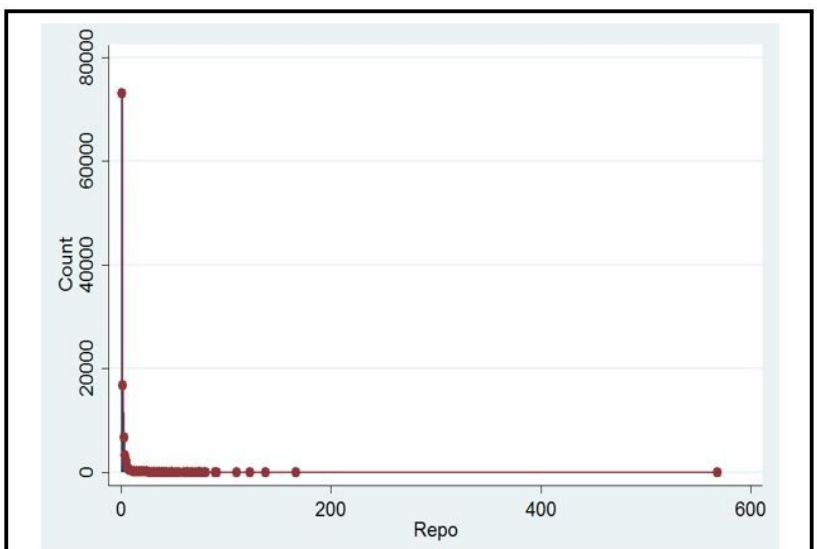


Figure 6. Distribution of Developers with Multi-project

Research streams on the effects of organizational structures on innovation performance remain inconclusive (Balkundi & Harrison, 2006; Crowston & Howison, 2006; Reagans & Zuckerman, 2001; Yang & Tang, 2004). Various researchers have emphasized that more innovation organizations result in a more decentralized communication structure, which indicates an active sharing of idea among employees (Crowston & Howison, 2005; Raymond, 1999). These assertions

reflect that OSS projects open themselves through communication and knowledge sharing, which results in positive effect on innovation performance. Especially, Raymond (1999) also insisted that developers may choose to participate in an OSS project, thinking that having many developers result in better projects. He also said that OSS projects prefer non-hierarchical and decentralized structures. From the network perspective, these assertions can be discussed again. Several researchers emphasized that the density of team network with informal social ties can affect team performance (Balkundi & Harrison, 2006; Reagans & Zuckerman, 2001), while others cannot (Sparrowe, Liden, Wayne, & Kraimer, 2001). Balkundi and Harrison (2006) identified that team members with large numbers of tie (high-density team) can actively share resources with high performance. In contrast, low-density teams with small numbers of tie share limited numbers of resources that result in negative effects on performance (Hansen, 1999). However, positive opinions on decentralized structures on successful innovation performance are rarely supported by multiple cases.

Various studies have emphasized the opposite opinion on organizational structure on innovation performance. Crowston and Howison (2006) insisted that hierarchical organization results in high performance. Bonaccorsi and Rossi (2003) also report that hierarchical structure is positively related to innovation performance. Centralization in particular reflects the dispersion of each member. Zero-value indicates that all members are equally important and have the same authorities. In contrast, a high value of centralization means few actors have important positions in the network. Therefore, centralization with the value of one expresses a star-shape network. From this perspective, Colazo (2010) concluded that the degree of centralization is positively related to the quality and productivity of OSS development.

Hypothesis 1. Organizational structure where few of core developers participating in multiple projects are present will have a positive effect on organization performance

In the case of OSS projects, researchers identified why developers continuously participate in certain projects (Fang & Neufeld, 2009), or why they

choose certain projects (Hahn et al., 2008). Through streams of research, we easily determined that more active developers with large amounts of contributions participate in projects continuously (Hippel & Krogh, 2003). They can also hold a more dominant position in the project (Barcellini, Détienne, & Burkhardt, 2008; Faraj et al., 2015; O'Mahony & Ferraro, 2004).

Various previous studies have argued for the determination of the position of developers according to the degree of participation. (Barcellini et al., 2008; O'Mahony & Ferraro, 2004) assumed that people who make a larger amount of more important contributions take a central position within a project, because most developers participate in projects with technical contribution in their mind. Therefore, past studies showed that the total amount of technical contributions plays an important role in determining a leader within a project (Barcellini et al., 2008; Faraj et al., 2015; O'Mahony & Ferraro, 2004).

However, this finding only indicates the roles of a leader from an individual perspective. From the perspectives of multi-project management, especially technology-oriented innovation, allocating specialists in projects can be difficult. Engwall and Jerbrant (2003) also analyzed the importance of resource allocation in multi-project management environment. (Hendriks et al., 1999) suggested allocating few people to a few projects. They defined the level of participation on projects as project-scatter factors, which describe the number of projects that employees are involved in, considering work time. A low degree of project-scatter factors indicate that employees participate in one project as full-time with fixed teams. This paper demonstrated that high project-scatter factors result in low project performance. In addition, according to Eskerod (1998), Rau and Hyland (2002), and Zika-Viktorsson, Sundström, and Engwall (2006), multi-resource allocations create role conflicts. Therefore, we assumed that when a significant number of developers participate in several projects at the same time, the innovation performance of the whole organization will be lower. The hypothesis of this assumption is as follows.

Hypothesis 2. Organizations with higher ratio of work on multi-projects will have a negative effect on organization performance

4. 4. Data Collection and Analysis

4. 4. 1 Data Collection

For this study, we gathered 116,644 active repositories or those that have more than five stars, which were created from January 2014 to December 2015 in GitHub. To investigate innovation performance of OSS development organization, we lowered the number of repositories to 34,163, focusing on those managed by individual organizations. Next, after eliminating duplicate organizations and considering cooperation, 20,696 repositories remained, those of which have more than two contributors. To analyze the organization of each project, we eliminated organizations with only one member, considering the definition of organization. Finally, 18,844 repositories remained as final data set. For social network analysis, we selected organizations with more than 10 projects. Because our study focused on the participation and structure of organizations, having a sufficient number of projects is important for investigation. The control on the repository acting period is conducted by collecting data eight months after its creation, which is enough time to measure project results, significantly targeting 251 organizations. Web crawler using Python is used for data collection.

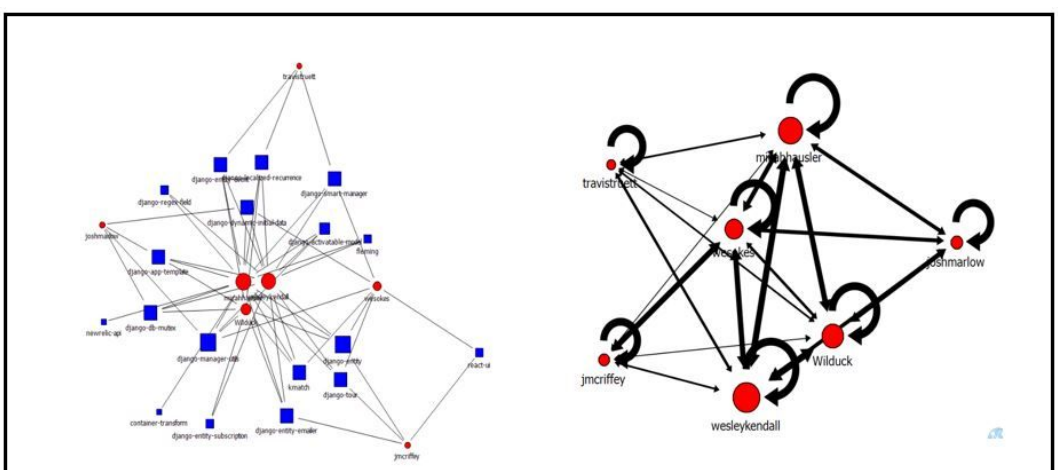


Figure 7. Social Network Analysis of OSS Development Organization

To measure group degree centralization, we conducted social network analysis through information of organization members and participation history of projects managed by the organization. Through this step we were able to create the networks of 251 organizations. Examples of network of each organization are shown in Figure 7. We created a two-mode network, which represents relationship between organization members and projects and change the two-mode network to one–mode.

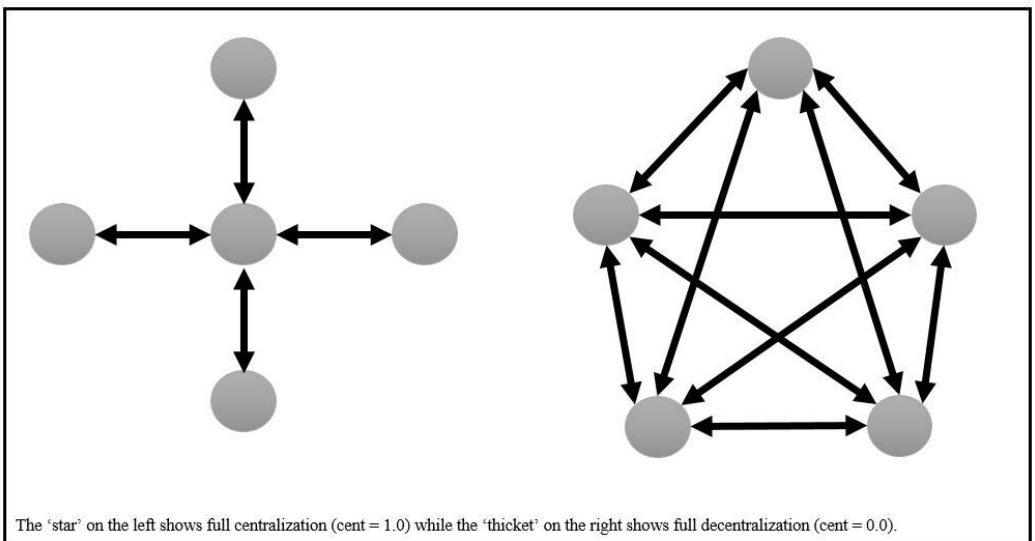
4. 4. 2 Variables

This study used the average of commit of each project managed by organizations to measure the organizational performance as dependent variable. Rullani and Frederiksen (2010) also generates indicators for organizational level analysis by sum the interactions among the developers. Previous studies measured commit of the project or individual commit as a project or individual result. According to Crowston et al. (2003), the success of OSS project is considered a project result, development process, and result of project participants. The result of the development process is measured as the number of commits. Previous studies measure the level of contribution of individual developers using contributors' commit as individual result (Adams et al., 2009). Grewal et al. (2006) used commit to measure technical success.

We also measured the participation ratio through the numbers of project that each developer is involved in among the total number of projects managed by the organization. The individual participation ratio is averaged as indicators of organizational information. Thus, when all members participate in the project operated inside the organization, the value is one; and when the average value is low, each member concentrated working on one or two projects.

The relationship among members in the organization and project participation pattern are made into figures through SNA. The participation of members comprised a two-mode network that focused on the project, and is converted into one-mode network again. SNA uses various figures to express relationship among actors. In particular, the position that the actors occupy in the

network can be considered as a concept of centrality (Kwak, 2014). The difference of actors within the social network in terms of centrality should be considered to understand the pattern of the network itself, rather than the position that development actors occupy in the network. Centrality can be divided into connection, medium, and closeness, according to the central position or the in-network role. The difference of each centrality distributed in terms of the entire network pattern is digitized. This study measured the differences in the connection level centrality of each actor in the OSS development organization, focusing on group degree centralization. This step is similar to the concept of dispersion in statistics.



The 'star' on the left shows full centralization (cent = 1.0) while the 'thicket' on the right shows full decentralization (cent = 0.0).

Figure 8. Concept of Degree Centralization

Group degree centralization can have a value between 0 and 1. Since it has a high value when the level of centralization is different, the sum of connection level centrality difference of maximum actors, which has a high value when the level of centralization differs, has the form of a star network, in which only one actor is connected with all the actors (Crowston & Howison, 2005; Ngamkajornwiwat, Zhang, Koru, Zhou, & Nolker, 2008). Therefore, group degree centralization of the star network is 1, and when all nodes are equal as the connection level centrality is distributed evenly, it has a value of 0. If it approaches to 1, it has unequal hierarchical distribution (See Figure 8).

Accordingly, in OSS development environment, digitizing the place where the developer of the coordinator role participates in all projects within the organization is possible to enable professional participation in the project each developer assumes, or all developers participate in all projects by becoming a coordinator. They can be digitized by the organizational nature to express.

This study covered the feature on how much the method of participating in the project is dispersed or professionalized, thus, the study no longer considered group closeness centralization and group betweenness centralization, which are analyzed focusing on the medium centrality level and closeness centrality as well as group degree centralization. Closeness centrality measures how close the actor is to other actors existing in the network (Singh et al., 2008). Calculating closeness centrality not only considers actors directly connected to one actor, but also all actors indirectly connected in the network (Gulati & Gargiulo, 1999). This centrality has the merit of being able to understand positions in the entire network and to emphasize the distance among actors in the entire network by considering even the indirect connection inside the network. Thus, although operators do not answer the issue directly in the OSS project, it can be used to understand how influential positions are to the developers of the entire project and the position of individual developers. As for medium centrality, actors located in the shortest route among others are in an advantageous position. Thus, group betweenness centralization measures how differently the actors in these advantageous positions are distributed. However, because this study focused on the dispersion degree, such as participating in the project that many developers attend or participating in many projects in an organization rather than a special position within the network similar to the closeness or medium centrality of each actor, only group degree centralization is used as a variable.

To calculate group closeness centralization, we adopt equation suggested by (Freeman, 1978).

$$C_D = \frac{\sum_{i=1}^{n}[C_D(p^*) - C_D(p_i)]}{(n-1)(n-2)}$$

Where p* is the most central point where the degree measurement C_D is the maximum; n is number of nodes in the network.

Hendriks et al. (1999) also assert that if organizations could not solve the

problems using knowledge of internal members, firm need to adopt the man power from external knowledge. Therefore, the role of external knowledge is important factors to performance of organizations. In order to explain the role of internal member and the governance strategies of OSS development communities, considerations of internal and external members are important. This research also generate variable related to the ratio of internal members among total number of members in each projects. We calculate average of ratio of internal members as control variable (See Table 6).

Table 6. Description of Variables

Variables		Measurement
Dependent Variable	*(Ln)Org_Performance*$_i$	Average number of commits of each repository$_h$ in organization$_i$
Independent Variables	*Organizational _Structure*$_i$	Degree of difference in degree centrality of each developer$_j$ in organization$_i$ Ex) High degree centralization: A few of the members of the organization$_i$ participate in most of the projects in organization$_i$ $C_D = \frac{\sum_{i=1}^{n}[C_D(p^*) - C_D(p_i)]}{(n-1)(n-2)}$ Where p* is the most central point where the degree measurement C_D is the maximum; n is number of nodes in the network.
	Participation_Ratio$_i$	Average of ratio of participating in organization$_i$ *Ratio of participation* $= \frac{Number\ of\ project\ that\ developer\ j\ is\ involved\ with}{Total\ number\ of\ project\ managed\ by\ orgnizationi}$
Control Variable	*Mean_InternalRatio*$_i$	Average of ratio of internal developer of each repository$_h$ in organization$_i$ *Ratio of internal developer* $= \frac{Number\ of\ internal\ developer\ j\ in\ organizationi}{Total\ number\ of\ developers\ in\ organizationi}$

4. 5. Results

4. 5. 1 Descriptive Statistics

Table 7. shows the technical statistics value of each variable. The results of the analysis on 251 organizations shows that the average commit of each organizations is 4.9, and participation ratio is 0.5156 on average. Degree centralization retains up to 0.24 and has an average of about 0.1024. Conversely, average internal member ratio in each project per organization is 0.5118.

Table 7. Descriptive Statistics

Variables		N	Mean	SD	Min	Max
Dependent Variables	$(Ln)Org\_Performance_i$	251	4.905	1.148	1.494	9.489
Independent Variables	$Organizational\_Structure_i$	251	.1024	.102	.00	.24
	$Participation\_Ratio_i$	251	.5156	.262	0.319	1.000
Control Variable	$Mean\_InternalRatio_i$	251	.5118	.031	.108	967

4. 5. 2 Results

The study used hierarchical regression analysis model to verify the hypothesis, as shown in <Table 8>. Hierarchical regression analysis evaluates the effects of the independent variable by explaining its quantity, except for the quantity of covariance at the entire account quantity (R^2), by injecting the variables in order. The average ratio of internal members on each project as the control variable is 20.7% (Model 1). Next, degree centralization, which is the information on organizational structure (B=3.96, p<.01), has a positive effect on organization performance. The explanation power is 24.2% (Model 2). Subsequently, the average of participation ratio as indicator of organization governance characteristics (B=-1.002, p<.01) increases to 28.5%, which has a static effect on organization performance. Results of the hierarchical regression analysis show that

when the independent variable is injected into all three models, the increment quantity of explanation ability (F value variation) is significant. In sum, having more external members will have a positive effect on innovation performance as control variable. In addition, organizations with a high degree of centralization perform better than those with a low degree of centralization (support Hypothesis 1). Finally, organizations with high participation ratio of developers on multiple projects will affect organization performance negatively (support Hypothesis 2).

Table 8. Results of Hierarchical Regression Model

	Model 1		Model 2		Model 3	
	B	SE	B	SE	B	SE
Mean_InternalRatio$_i$	-2.875***	.357	-2.853***	.349	-2.592***	.347
Organizational _Structure$_i$			3.957***	1.169	3.237	1.153
Participation_Ratio$_i$					-1.002***	.261
Cons	6.434***	.193	6.018***	.226	6.443***	.246
R^2	.207		.242		.285	
Adjusted R^2			.236		.276	
ΔF	65.007***		11.455***		14.774***	
N	251		251		251	
VIF	1.00		1.04		1.05	

※ Dependent variable: *(Ln)Org_Performance$_i$*
Note: * p < .1, ** p < .05, *** p < .01

[7]

4. 6. Conclusion

4. 6. 1 Discussion

This research verify the impacts of characteristics of OSS development organizations on innovation performance based on the organizational structures and resource allocation strategies. Especially, we analyze the structure of organizations using social network analysis and generate indicators of multi-project participation status of each organization. The results indicate with centralized organizations with a few core developers who involved in multi-projects have positive impacts on

organizational performance. In addition, from the perspectives of project management, the results of our study suggest that organizations that allocate developers with specialized project perform better than organization that scatter the developers in multiple project at the same time.

4. 6. 2 Conclusion

The contributions of this research mainly lay in level of analysis. Previous research focus on the interrelationship between developers who participated in multi-project at the same time. Others forms structures of network within a project based on a communication. Unlike previous research, this paper analyze the organization which operate various OSS projects simultaneously with numbers of developers. Considering the organizations as CMC communities with explicit purpose, we conclude the even open collaboration communities also have efficiency with hierarchical structures. In order to effective strategies in project management, the results indicate that developers need to focus on single project as much as they can, when they have a chance to involve in multi-project. In R&D environment, as each project required specialized knowledge, developers need to focus on certain projects within limited time space.

Therefore, we have three theoretical contributions. First, in social capital theory, with inconclusive results of previous research on structure of OSS development communities, we supports the effectiveness of centralized structures on innovation performances. Since OSS development strongly related to the open collaboration circumstances, researchers and developers asserts that the structures of communities are decentralized with highly innovative communications. However, there are numbers of evidence that more hierarchical and centralized structures are positively impacts on performance of projects. The results reflects that even the open collaboration aiming ‘going open’, also required to accomplish certain purpose by developing software, the projects are more efficient with centralized structures. Second, with various research or project management, this research add the stream of previous research on efficient project management literatures. Open collaboration environment containing differences from traditional

firms, developers with self-organizing in multiple projects could make decisions on time and efforts sharing. As positive results on ratio of particpations with organizational performance, our research could contributes the practical strategies for OSS development organizations, too.

However, the research face various limitations. First, we did not consider the environmental factors that main languages in OSS development organizations or experiences of the organizations. Considering programming language, language could be differed in complexities and modality etc. In addition, the experience of organizations also could influence on organizational performance. However, our data collect limited numbers of projects which are established during 8 months. In addition, collaboration experience of developers also could strongly influence to organization performance. Therefore, our future research need to consider those variables to control the performance of organizations more specifically.

.

CHAPTER 5: [ESSAY 3] CHARACTERISTICS OF PROJECT ON INNOVATION PERFORMANCE: BASED ON TEAM COMPOSITION AND ORGANIZATIONAL LEARNING THEORY

5. 1. Introduction

OSS development methodologies use open collaboration through the Internet and have separate classifications compared with the traditional R&D processes or product development methods (Levine & Prietula, 2013). Therefore, previous innovation-related research regards the OSS development as an approach that is outside the knowledge flow or exploration activity perspective (Chesbrough, 2006b). However, development mechanisms or team composition emphasizes that knowledge creation activities could be varied (Ferriani, Cattani, & Baden-Fuller, 2009). In addition, the method of managing knowledge creation may be an important issue for project owners. In particular, when large communities attempt to develop OSS, assigning team members and providing opportunities for external developers are difficult but vital decisions that should be made. Accordingly, decision-making guides leading developers in certain projects when they contribute the code that affects innovation performance.

When an OSS project owner is an organization, the members of such organization could be authorized to allow the use of codes from external members. Therefore, sharing the goal of software development within the organization by accepting the proper code from outside is important. In addition, an organization does not typically focus on merely one project but manages multiple projects simultaneously. In this case, constant collaboration experiences enable organizations to accumulate their own OSS knowledge, establish the development process norm, or share the efficient norm among members (Hahn et al., 2008). If an organization sets up a project as public, then this organization is allowed to develop such project by involving external developers. In the case of exclusively

developing OSS through organizational members, organizations use Github (a popular OSS development platform) as a collaboration tool within the company and is similar to the process of traditional firms. However, public repositories with opportunities to collaborate with external developers are similar in terms of innovation activities, such as mergers and acquisitions.

From the perspective of OSS development organizations, the composition of developers is a critical issue when these organizations manage various projects involving external and internal developers. In project management, the number of people involved or efficient external developers in innovation performance is an issue that managers face. Ferriani et al., (2009) ragards as team project composition. To establish new firms or new projects, considering the existing employees and newcomers for the team composition could affect team performance (Ensley & Hmieleski, 2005; Forbes, Borchert, Zellmer-Bruhn, & Sapienza, 2006). In addition, previous studies on traditional firms report the strong tendency to prefer colleagues who already have interpersonal relationships, as well as reluctance to work with newcomers (Ruef, Aldrich, & Carter, 2003).

The present study adopts organizational learning theory in explaining the role of internal and external developers on project performance. The innovation process is divided into two stages, namely, search and implementation stages (Zaltman, Duncan, & Holbek, 1973). Previous research categorized the innovation process as exploration focusing on searching for and exploiting a new external idea, as well as integrating such idea with the existing knowledge. Perretti & Negro, (2007) indicated that integrating ideas with existing knowledge is an exploitation activity, whereas hiring new employees is an exploration activity. Moreover, this research divided the exploration and exploitation activities whether developers belong to an organization. Collaborating with developers who have worked together could easily establish work reliabilities or routines for efficient work. Therefore, routines originate from previous collaboration experiences, that is, the exploitation of existing knowledge diminishes the value of a new idea (Chatman, 1989; Smith-Doerr & Powell, 2005). In this case, a firm requires exploration by involving the external members. Therefore, managing OSS projects entails providing opportunities for the participation or encouraging the motivations of external developers, which are important governance strategies.

Accordingly, inconclusive arguments toward that opening OSS projects to external developers are provided. Crowston and Howison, (2005) explained that in practice, an OSS project endeavors to maximize the results through attempts at "going open" for OSS development. Raymond, (1999) cited Linus Torvalds, a famous OSS developer and Linux founder, who said that "the person who understands and fixes the problems is not necessarily or even usually the person who characterizes it." (Crowston & Howison, 2005). Therefore, in Raymond (1999), they formulated this assertion as "Linus' law." However, contrary opinions are noted on the number of external developers. Previous research indicates that substantial participation of external developers decreases performance innovation by the loss of external resources control (Dahlander & Magnusson, 2008; Laursen & Salter, 2006). In addition, Raymond and Young (2001) asserted that if a project leader focuses on innovation by excessively accepting outside ideas, then the OSS performance will deteriorate.

First, we verify Linus' law with data from Github. Second, we explain the role of the internal members of an organization and external developers based on organizational learning theory. In addition, we analyze the efficient team composition of an OSS project using empirical data from Github. Accordingly, we divide the main objective of this research into two. In Study 1, we analyze the relationship between project size and team composition related to project performance. Study 2 shows the team composition by considering the fit of internal and external developers using the response surface model. This study provides theoretical contributions from the perspective of organizational learning theory. Previous research regards OSS development as an exploration activity. However, we divide the OSS development activities into two, namely, exploration and exploitation, according to the position of developers. In particular, the results of Study 2 explain the composition of external and internal developers with project performance.

The rest of this paper is structured as follows. The second chapter reviews previous research with theoretical background, including organizational learning and team composition. The third chapter proposes a research model and the relevant hypotheses to verify such model. The fourth chapter presents an explanation of the Github data. The fifth chapter presents the analytical result.

Finally, the sixth chapter provides the conclusion and the limitations of this study.

5. 2. Theoretical Background

5. 2. 1 Organizational Learning

Organizational learning is defined as "the experiential production and reproduction or organizational rules, leading to behavioral stability or behavioral changes." (Kieser, Beck, & Tainio, 2001; Levitt & March, 1988). Organizational learning itself involves unique behaviors, such as those of humans, by considering different outcomes as performances (Holmqvist, 2004). Normally, learning activities can be divided into two: exploitation and exploration. Scholars define exploration and exploitation from the technological innovation perspective by considering knowledge-creating activities (Argyres, 1996; Ahuja and Lampert, 2001; Rosenkopf and Nerkar, 2001; Benner and Tushman, 2002; Katila and Ahuja, 2002; Nerkar, 2003; Dowell and Swaminathan, 2006). Exploration emphasizes developing and adopting new technologies, whereas exploitation emphasizes developing technologies using existing technologies. Benner and Tushman (2003) also defined exploitation and exploration in the knowledge and innovation areas. They defined exploitative innovation as improving existing components and building on existing technological trajectories. By contrast, exploratory innovation is defined as a shift to a different technological trajectory.

Researchers have consistently argued that exploration and exploitation draw on different structures, processes, and resources (He & Wong, 2004), that generate significantly different performances over time. March, (1991) explained that exploitation is considered a refinement and extension of existing capabilities and technologies. In particular, exploitation is a process by which organization creates reliability in experience through refinement, production, and focused attention (Levinthal & March, 1993). Uotila, Maula, Keil, and Zahra (2009), explained that exploitation can lead to short-term positive effects. Exploitation often reduces variety, adapts external knowledge, and enhances efficiency in improving performance from a short-term perspective. Exploitation is also a

process of reutilization to add value to existing knowledge. Therefore, exploitation could increase the innovation performance of firms by enhancing their competence in a certain knowledge area as a result of specialization (Gilsing et al., 2008). Typically, exploitation focuses on the refinement and incremental extension of existing capabilities (Voss & Voss, 2013). Considering long-term performance, exploitation hinders a firm from adapting a new or rapid-change environment. Therefore, firms that emphasize exploitation activities may lack the capability to adapt to significant environmental changes; thus, the formula that makes these firms successful in the short term may endanger their long-term success.

On the other hand, exploration provides different strategic advantages to firms by exploring external knowledge. Such exploration-oriented activities assist a firm in developing new knowledge and creating capabilities necessary for survival and long-term success. However, exploration activities are uncertain in their payoffs, and performance effects often occur in the long run. Therefore, focusing solely on exploration can be detrimental for a firm, thereby locking it into a cycle in which "failure leads to search and change which leads to failure which leads to more search and so on." (Levinthal & March, 1993). The discovery and experimentation of new technology are core activities of a firm (March, 1991; Nooteboom, 2000). Exploration focuses on creating a variety of experiences and thrives on experimentation and free association (Holmqvist, 2004). Accordingly, companies are able to obtain new opportunities, insights, and knowledge (Nooteboom, 2000; Rowley, Behrens, & Krackhardt, 2000). Gilsing, Nooteboom, Vanhaverbeke, Duysters, and van den Oord (2008) considered exploration as an activity that creates new knowledge that is not novel to industry or to the world but differs from the existing knowledge of a firm. Moreover, exploration focuses on radical new knowledge by transforming existing knowledge or combining new and existing ones (Ahuja & Morris Lampert, 2001; Henderson & Clark, 1990; Nelson & Winter, 2009; Tushman & Rosenkopf, 1992).

Previous research emphasized that exploration and exploitation contain different learning models (Argyris and Schön, 1978; Benner and Tushman, 2003; Eisenhardt and Martin, 2000). Although exploration and exploitation produce tangible output, such as patents and products, the process and objective of these two activities are quite different from each other (Li et al., 2008). Moreover, each

activity settles down within firms as a problem-solving mechanism; thus, the activity turns out routine because of path (Dosi, 1988; Nelson & Winter, 1982). Gilsing et al. (2008) asserted that firms are familiar with existing technologies and previous experiences.

Scholars describe these routinization using path dependence as a lock-in situation with explanation of the tractable approach toward the problems of technologies (Arthur, 1989; Levitt and March, 1988). Numerous researchers describe these types of lock-in behavior as routines, such as habits, rules, standard operating procedures, or heuristics (Cyert and March, 1963; March and Simon, 1958; Nelson and Winter, 1982; Simon, 1947). Weick, (1979) attempted to explain the types of routine within an organization: "organization can and do act like closed systems… organizational attentiveness to one's own past experience can continue unpunished for surprisingly long periods of time." Considering organizational rules, if one firm successfully and repeatedly implements exploration or exploitation as a knowledge creation process, then such firm could have competence on the process (Holmqvist, 2004). Thereafter, companies attempt to focus on current success and reinforce the process to retain stable status. For a long time, a trade-off between exploration and exploitation is complicated in strategic fields because two strategic investments require spending enormous costs (Lavie et al., 2010). With realistic limitations, a considerable choice between exploration and exploitation is required for a radically changed environment.

Despite the focus of scholars on compatible concepts, inconsistent interpretations of exploration and exploitation are still noted (Li et al., 2008). Accordingly, Li et al., (2008) reviewed the conceptual definitions and measurements of exploration and exploitation (See Table 10), as well as explained that several scholars adopt various concepts of these two activities. In addition, they also provided new definitions of exploration and exploitation in a different manner. In addition, researchers vary the notions of exploration and exploitation as a process or outcome of innovation (Li et al., 2008). A few researchers regard exploration and exploitation as innovation processes, such as organizational learning activities, behaviors, investments, or strategies (Jayanthi and Sinha, 1998; Nerkar, 2003; He and Wong, 2004; Nerkar and Roberts, 2004; Van Looy, Martens and Debackere, 2005; Phene, Fladmoe-Lindquist and Marsh, 2006; Sidhu,

Commandeur and Volberda, 2007). Other researchers treat these concepts as outcomes of innovation performance (Argyres, 1996; Katila and Ahuja, 2002; Nerkar and Roberts, 2004).

Table 9. Concepts of Exploration and Exploitation

Reference	Exploration	Exploitation
Audia and Goncalo (2007)	New subclasses and number of new citations	
Ahuja and Lampert (2001)	Distant Search	Local Search
Cantwell and Mudambi (2005)	Competence-creating subsidiary	Competence-exploiting subsidiary
Nerkar (2003)	Temporal spread	Temporal recency
Nerkar and Roberts (2004)	Distal experience in technology and market	Proximate experience in technology and market
Phene et al. (2006)	Distant search in technical knowledge dimension and geographic/spatial dimension.	Local search in technical knowledge dimension and geographic/spatial dimension.
Geiger and Makri (2006)	Science search	Technology search
Argyres (1996)	Technological capability broadening	Technological capability deepening
Bierly and Daly (2007)	Experiment with radical new ideas or ways of doing things.	Refining and leveraging existing knowledge and focus on efficiency of current practices
Dittrich and Duysters (2007)	Non-equity alliances with new partners, who have different technologies	Equity alliances with existing partners
Gilsing and Nooteboom (2006)	Searching and recombining technology and science;	Search market knowledge
Danneels (2002)	Develop new technology to serve new customers,	Strengthen existing technology to serve existing customers
Perretti and Negro (2007)	Recombining old and new (hiring new employees)	Recombining old, reuse existing, leverage prior knowledge

Phene et al., (2012) divided an acquirer's knowledge creation capacity within one company into exploitation and exploration activities. They analyzed the effect of technological uniqueness of a target firm on an acquirer's exploitation and exploration activities. In addition, common technological knowledge, which is considered "technological relatedness" in other studies, play moderating roles between the technological uniqueness of a target firm and knowledge creation capability of an acquirer. Research also verifies the effect of the extent of common geographic bases and equity interest on an acquirer's exploration and exploitation activities. They only divided the acquirer's knowledge creation strategy that failed

to obtain the strategy mode of a target. However, they did not consider the limitation of the acquirer's resources in an actual business environment. Accordingly, only a limited number of firms could invest in exploitation and exploration activities. Therefore, assuming exploitation and exploration by measuring data from the patents of certain firms is difficult. Laursen and Salter (2006), explained that exploitation and exploration activities are influenced by previous experiences and future expectations. Therefore, determining the optimal balance between the two activities is difficult (Levinthal & March, 1993). Considerable historical research streams identify two main activities in the innovation process, namely, search and implementation (Zaltman et al., 1973).

The current research regards exploitation as coordination work with prior collaborators and exploration with newcomers. Prior collaborators share the same language or similar idea (Jones, Hesterly, & Borgatti, 1997), and the benefits of prior experience with coworkers is based on a matter of trust (Hansen, 1999). By establishing relationships and experiences, employees could enhance trust among them, thereby enhancing the coordination and knowledge creation activities. Therefore, routines for knowledge are also easy to develop. However, previous research warned that trust make firms blind toward new external opportunities (Ferriani et al., 2009; Smelser & Swedberg, 2010). In the same context, Perretti and Negro, (2007) indicated that one role of newcomers is to search for new ideas that do not exist in firms and provide and integrate them with existing knowledge in these firms. Following the limitations of existing employees (Chatman, 1989), newcomers could expand the boundaries of firms to new external resources. March, (1991) also suggested that experienced members already contribute their knowledge in organizations and they have low possibilities of contributing new knowledge. By contrast, newcomers have limited knowledge in the organizational domain but they are likely to contribute new knowledge from the perspective of the organization. Therefore, March's simulation model states that old-timers would elicit exploitation (i.e., production, efficiency, and implementation), whereas newcomers will increase exploration (i.e., search, discovery, and innovation).

5. 2. 2. Organizational Learning and OSS

The current research adopts organizational learning theory as a theoretical lens in interpreting the role of external and internal developers on the OSS project performance. In the case of OSS projects, outside developers are regarded as developers who search new idea as self-activity. However, the OSS projects are conducted by volunteers who actively enjoy confusing stages. Moreover, the OSS projects are easily supported by domain-specific experts through the Internet, thereby reducing efforts in searching for new knowledge for innovation. The succeeding primary activity is the implementation of a new idea. In the OSS project, implementation processes are organized by internal developers who have the authority to accept the code submitted by external developers.

Osterloh and Rota (2007) regarded OSS as a low-cost collective intention and compared other technological inventions with OSS based on organizational learning theory. However, the current study considers OSS with balanced activities of the exploration and exploitation activities regardless of specific knowledge creation process within the OSS development process. Rullani and Frederiksen, (2010) analyzed the interactions among the developers by employing organizational learning theory. They characterized the communication between developers regardless of projects. Communication with developers who are involved in multiple projects could describe the characteristics of communities. They regarded project launching as exploration, which is a new code based on a new idea of OSS. Even specific codes come from existing knowledge and joining projects as exploitation because they contribute their knowledge to existing projects. David and Rullani (2008) also asserted the improvements in the exploitation capabilities of the OSS project when additional resources are allocated to the existing software.

5. 2. 3. Team Composition

Traditional companies frequently face decision-making toward team composition (Ferriani et al., 2009). When they organize project teams or establish new

companies, managers need to assign existing employees and hire newcomers. In this case, the ratio of existing employees and newcomers are important for an efficient team (Ensley & Hmieleski, 2005; Forbes, Borchert, Zellmer-Bruhn, & Sapienza, 2006). In particular, people prefer to work with colleagues who have collaborated with them from the perspective of employees (Ruef, Aldrich, & Carter, 2003). This phenomenon could be described as path dependency in the field of management. Path dependency explains the tendency to prefer to work or communicate with someone who already has experience.

Edmondson, Bohmer, and Pisano (2001) elucidated that project teams with complex tasks require sophisticated team coordination. Considering the circumstances of the OSS project team, software development also requires specialized and complex knowledge creation process. Therefore, the coordinating team of the OSS development is also crucial to their performance. Tan, Mookerjee, and Singh (2007) also emphasized the importance of the optimal team composition for efficient team performance in the OSS project as implications. They investigated the collaboration networks and flow of information and know-how within OSS projects. Crowston, Annabi, Howison, & Masango, (2004) also proposed the importance of team positions in the OSS project for efficient team performance. However, previous research interprets the roles of developers as positions of network within the OSS projects.

5. 3. Hypotheses

Study 1

Brook's law states that "Too many cooks spoil the broth"(Brooks Jr, 1995), thus, assuming that the number of employees does not constantly and positively influence the innovation performance is easy. By contrast, Torvalds asserted that the number of developers will reduce the problems of OSS projects. Torvalds stated that "Given enough eyeballs, all bugs are shallow." In research streams, inconclusive arguments are presented toward exploring external knowledge as components of openness strategies. Even various studies consistently argued the importance of searching for external knowledge for new innovative opportunities,

whereas others identify the ineffectiveness of considerable openness on firm performance.

From the positive perspectives of organizational size on innovation performance, we could assume that various experiences of organizational members will enhance innovational performance. Watson, Kumar, and Michaelsen, (1993) advocated the positive roles of members with multiple experiences and perspectives in solving problems and improving the ability of teams. Torvalds shared the positive effects of numerous developers on OSS projects. Raymond (1999) formulated the assertions of Tovalds as "Linus' law." Crowston and Howison (2005) also indicated that OSS projects attempt to "going open" to succeed in OSS projects.

By contrast, Laursen and Salter (2006) asserted that substantially open innovation negatively affects on firm performance. Studies determined that returns of openness are decreasing if they conduct additional search for external knowledge. They use 2707 manufacturing firms in verifying meaningful contributions to organizational learning theory. Even if many considerations of open innovation measurement scales are provided, such as costs or input of open innovation, open innovation is evidently an inconsistently effective method in improving firm performance. Their hypothesis emphasized that the search for new knowledge will have negative effects on innovation performance (Laursen & Salter, 2006).

Therefore, to clarify the relationship between openness and innovation performance, researchers attempted to consider moderating or mediating the effects on dependent variables. Katila and Ahuja (2002) also determined a curvilinear relationship between search activities and innovation performance, which have a tendency to "over search." Katila (2002) and Katila and Ahuja (2002) argued that the relationship between search strategy and innovation performance focuses on the age of knowledge, as well as the depth and scope of the search. The current study regards the participation of developers as knowledge creation activities. Therefore, even if positive influence of active involvements of developers in OSS projects is present, numerous developers result in difficulty to control the resource on the part of organizations or managers. From these assumptions, our hypotheses are presented as follows.

Hypothesis 1: The relationship between the number of developers in an OSS project and project performance is curvilinear (inverse U-shape)

We consider the knowledge creation of internal members of organizations as an exploitation activity based on organizational learning theory, whereas developers from outside are involved in exploration activity. The definition of exploitation implies that developers create code based on existing knowledge and experiences. Therefore, we assume the number of internal developers involved in OSS organizations as sources of potential exploitation. Furthermore, exploration is interpreted as activities for searching a new idea outside the firm. From this perspective, we regard the role of external developers as exploration activities.

The capabilities of exploration and exploitation are affected by the size of companies. The resources and information of small companies are limited compared with large ones (Lee, Park, Yoon, & Park, 2010). Therefore, focusing on external resources in implementing internal knowledge is difficult. The negative effects of over search could also be explained by the attention-based view (Ocasio, 1997). Koput (1997) provided three reasons for the difficulties of over search. First, substantial knowledge from outside firms leads to easily losing control during the implementation stages involving internal knowledge. Second, even useful knowledge is inconsistently useful in every situation. Third, the human resource is limited to attend to and implement external knowledge to the firm; hence, only a few of them focus attention to this matter. Therefore, attention-based theory explains the limited attention of owners, particularly decision-makers, who need to concentrate their efforts to several issues. Accordingly, with a limited number of developers, substantial external knowledge results in negative effects on innovation performance.

However, Gregorian (2014) asserted the OSS philosophy as open collaboration. From the different definitions of open collaboration, Neus and Scherf (2005) defined open collaboration as “a meritocratic philosophy that invites feedback from everyone, regardless of official status or formal training and frequent releases of interim versions to encourage testing, feedback and quick evolution of solutions.” Therefore, increasing size could offer additional

opportunities in solving the problems (Blau, 1970) from the OSS project, thereby requiring specialized knowledge. Increased specialization itself could result in a structural difference that could enlarge firms; hence, the positive network effects could be expected. Coordinating employees is difficult because the complexity of organizations that results from their sizes. However, with the OSS project as a self-organizing community, developers themselves contribute the proper knowledge to the proper place. In addition, previous research also explains the capabilities of large firms for innovation. Various studies report that open innovation adopters are mostly large firms (Bianchi, Cavaliere, Chiaroni, Frattini, & Chiesa, 2011; Keupp & Gassmann, 2009; Lichtenthaler & Ernst, 2009). In particular, in the OSS development environments, additional structured projects are efficient for collaborating with the modularity of software. Through modularity, developers could work for parallel development with large-scale process (Rossi, 2004).

Hypothesis 2: Project size will moderate the relationship between the ratio of external developers and project performance. The influence of exploration measured by the ratio of external developers is negative in small projects. By contrast, the influence of exploration measured by the ratio of external developers is positive in large projects.

5. 4. Data Collection and Analysis

5. 4. 1. Variables

In this study, the total number of project commit was used as a measurement tool in evaluating the dependent variable, that is, the OSS project performance. Many existing studies use the commit number of a project or an individual as a performance measure. Various research consider the numbers of commit as dependent variables. Crowston et al. (2003) regarded that the commit number measured the success of an OSS project as performance during the development process. Existing studies used the commit numbers of contributors as individual performance when measuring the level of contribution of each individual developer.

Moreover, Adams et al. (2009) and Grewal et al. (2006) used commit numbers to measure technical success. In particular, this research use natural log for normalization of development variable.

Our research use numbers of developers as indicators of project size. Public project are open to developers who want to contribute their knowledge to an OSS project. In this case, number of developers could be important factors in several reasons. Previous research also indicate the size of project or size of firms with number of employees (Glancey, 1998; Reid & Adams, 2001; Zenger & Lawrence, 1989). For team composition, previous research use numbers of new comers with previous experienced employees as indicators (Perretti & Negro, 2007). In this research we regard external developers as newcomers in team composition. Especially, newcomers could contribute the new knowledge to organization, we interpret newcomer as source of exploration. Our main focus of team composition is external developers who can carry the value of new idea, we calculated team composition centered with ratio of external developers among total numbers of contributors in OSS development project. Table 10. summarize the descriptions of variables

Table 10. Description of Variables

Variables		Measurement
Dependent Variable	*(Ln)Project_Performance*$_i$	Number of commits of repository$_i$
Independent Variables	*Project_Size*	Total numbers of developers on repository$_i$
	Team Composition	Ratio of external developers among total numbers of developers on repository$_i$

5. 4. 2. Data Collection

The current study gathered 116644 repositories that were created from January 2014 to December 2015 in Github with over 5 stars as indicators of active repositories. We excluded the repositories managed by individual organizations in investigating the innovation performance of the OSS development organization.

Figure 9 show the page of organizations in Github. Therefore, we can obtain 34163 repositories managed by organizations. After eliminating the duplicate organizations, we retained 20696 repositories that have over 2 contributors according to the definition of cooperation. To analyze the organization of each project, we eliminated organizations with only one member based on the definition of an organization. Finally, 18844 repositories comprised our final data set.

In addition, we delete numbers of external members and internal members exceeds three standard deviation to detect outliers. Since in normal distribution for 99.87% of the data within the range, researchers usually adopt standard deviation to eliminate outliers (Howell, Rogier, Yzerbyt, & Bestgen, 1998). As a results, internal members who involved in OSS development organization are 14, and external developers are below 83. Therefore, total number of our data is 18557 repositories.

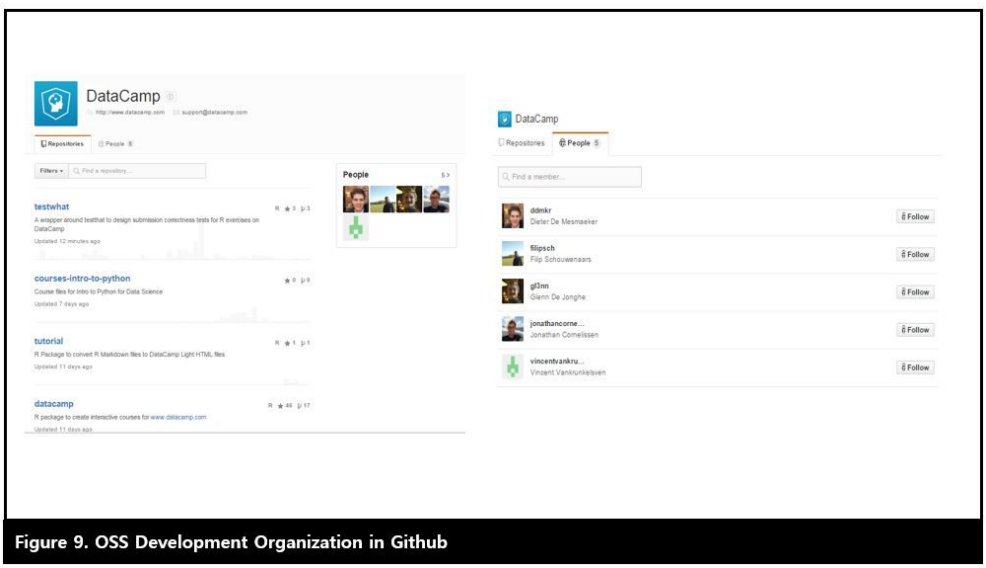


Figure 9. OSS Development Organization in Github

5. 5. Results

5. 5. 1 Descriptive Statistics

Table 11. presents the descriptive statistical value of each variable. As a result of conducting the analysis focusing on 18557 projects (i.e., the analysis target), the commit of each project was 4.533 and the average project size was 7.470. The average ratio of external developers per project is approximately 0.612.

Table 11. Descriptive Statistics

Variables		N	Mean	SD	Min	Max
Dependent Variable	*(Ln)Project_Performance*$_i$	18557	4.533	1.523	.693	12.062
Independent Variables	*Project_size*$_i$	18557	7.470	9.188	2	94
	Team_Composition$_i$	18557	.612	.288	0	1

5. 5. 2 Results

We use the hierarchical regression analysis model to verify Hypothesis 1 (see Table 12). The hierarchical regression analysis is a method in evaluating the effect of the independent variable by the explanation quantity, with the exception of the explanation quantity of covariance in the entire account quantity (R^2) by injecting the variables the researcher would see in order. As a result of injecting in stages in consideration of the covariance of main variables, the number of developers as project size on a performance of project was 30.2% (model 1). Subsequently, the square of project size ($B = -0.002$, $p < 0.01$) had a static effect on the project performance increased to 38.3% (model 2). From the preceding results, we concluded the inverse-U shape relationship between project size and project performance. Figure 10 shows the clear curvilinear shape of the project commit count following the project size.

Table 12. Results of Hierarchical Regression

	Model 1		Model 2	
	B	SE	B	SE
Project_size	*.091***	*.001*	*.161***	*.002*
Project_size^2			-.002***	.000
Cons	4.533***	.009	4.700***	.0094
R^2	.302		.383	
Adjusted R^2	.302		.383	
N	18557		18557	
VIF	1.00		3.21	

※ Dependent variable: (Ln)Project_Performance$_i$
Note: * p < .1, ** p < .05, *** p < .01

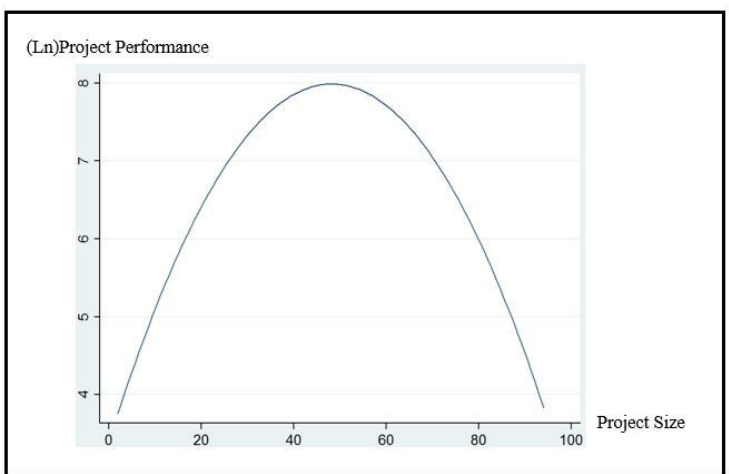


Figure 10. Relationship between Project Size and Project Performance

For Hypothesis 2, we conducted two group comparisons using project size. Following the average of project size (7.470), we divided the project size into two groups, namely, small group with below 8 developers and a large group with over 8 developers. The results indicate that when project size is small, developers from outside negatively influence project performance ($B = -0.135$, $p < 0.01$). The explanation power of model 1 is approximately 17.3%. By contrast, when project size is large, developers from outside positively influence project performance ($B = 0.343$, $p < 0.01$) (See Table 13).

Table 13. Results of Hypothesis 2

	Model 1			
	Small Project (Contributors<8)		Large Project Contributors≥8)	
	B	SE	B	SE
Project_size	*.340***	*.007*	*.044***	*.067*
Ratio_External	-.135***	.034	.343***	.091
Cons	2.846	.029	4.830	.067
R^2	.173		.202	
Adjusted R^2	.173		.201	
N	13377		5180	

※ Dependent variable: (Ln)Project Performance$_i$

※ Contributor (Mean) = ~8 (7.470)

Note: * p < .1, ** p < .05, *** p < .01

Study 2

In study 2, we verify the specific combination of internal developers and external developers on project performance. Since previous research assert limitation of difference between two components with difference score, we adopt alternative methodologies called response surface model.

Polynomial Model

Polynomial modeling enables the analysis of the complex relationship between independent variables and an outcome variable. Based on a basic theoretical model, that is, Z = f (X, Y), this analysis enables the assessment of a substantially accurate relationship between independent and dependent variables (Edwards, 1994), (Edwards & Van Harrison, 1993). Polynomial modeling involves a hierarchical analysis of polynomial equations.

$$Z = b_0 + b_1 X + b_2 Y + b_3 X_2 + b_4 XY + b_5 Y_2 + e$$

To verify the relationship among X, Y, and Z, we need to conduct a hierarchal analysis. The first step in this analysis is testing the linear relationship between X

and Y, and Z. Second, higher order (X^2 and Y^2) with product tem (XY) is tested for the presence of a quadratic relationship (Edwards 2002; Edwards and Harrison 1993). Specific description of variables in polynomial modeling is shown in Table 15.

Table 14. Description of Variables

Variables		Measurement
Dependent Variable	*Project_Performance*	Total number of commits in repository i
Independent Variables	*AM*	Number of organizational members who work for OSS repository i
	NM	Number of contributors those who are not group member but work for OSS repository i

To test our independent variables, that is, the number of internal members and external developers, we tested the quadratic polynomial equation as follows:

$$Project\_Performance = b_0 + b_1\,AM + b_2\,NM + b_3\,AM^2 + b_4\,AMNM + b_5\,NM^2 + e$$

Table 15. shows the results of the higher-order analysis. From the results of the analysis, we determine that relationships among internal members and external developers influence project performance. However, the first-order analysis does not describe the precise relationship toward project performance. Therefore, as the first stage of hierarchical analysis, higher-order term with interaction terms are analyzed.

Table 15. Results of First and Second Order Analysis

Dependent Variable	Independent Variables	First-Order Linear Equation		Second-Order Quadratic Equation	
		R^2	β	R^2	β
Project Performance	AM	0.3127	0.178216***	0.3872	.230236***
	NM		0.082210***		.15526***
	AM^2				-.011556***
	AMNM				-.00182***

	NM^2				-.004917***

※ Dependent variable: (Ln)Project_Performance$_i$

※ AM, NM: Mean-centered Measurement

Note: *p<.05, **p<.01, ***p<.001.

Response Surface Model

The response surface methodology is a visual test that provides a profound understanding of polynomial models. Accordingly, polynomial models are difficult to interpret, and the response surface model enables the easy understanding of the complex relationship between independent and dependent variables. The coefficients that describe the surface (Edwards 2002; Edwards & Parry, 1993) require a detailed test on the plotted surface. Therefore, for statistical test, the response surface model normally tests three main features of the surface generated from the polynomial models.

- A stationary point is defined as a point at which the slope of a surface is zero in all directions (Edwards & Parry, 1993)
- Principal axes run perpendicular to each other and intersect at the stationary point (Edwards & Parry, 1993)
- For a convex surface, the upward curvature is greatest along the first principal axis and least along the second principal axis (Edwards & Parry, 1993)
- For a concave surface, the downward curvature is least along the first principal axis and greatest along the second principal axis (Edwards & Parry, 1993)

The other features of the response surface model are the confirmation axis (X = Y) and disconfirmation axis (X = −Y) (Edwards & Parry, 1993). In RSM, the X =Y and X = −Y axes are of limited interest to this study because innovation performance does not necessarily match the number of external and internal developers. Moreover, the difference between external and internal developers is peripheral.

Edwards and Parry (1993) show the equation of stationary point

$$X_0 = (b_2b_4) – (2\ b_1b_5) / (4\ b_3b_5) – b_4$$

$$Y_0 = (b_1 b_4) - (2\, b_2 b_3) / (4\, b_3 b_5) - b_4$$

The stationary point can be used in the equations for the first and second principal axes. A provided the equation of the first principal axis, which is the line along which the slope of a convex surface is maximum and is given by

$$Y = p_{10} + p_{11} X_0$$

Where

$$p_{11} = (b_5 - b_3 + \text{sqrt}\,((b_3 - b_5)^2 + b4^2)) / b_4$$
$$p_{10} = Y_0 - p_{11} X_0$$

Furthermore, (Edwards & Parry, 1993) explained that the second principal axis (i.e., the line along which the slope of a convex surface is minimum) is given by

$$Y = p_{20} + p_{21} X_0$$

Where

$$p_{21} = (b_5 - b_3 - \text{sqrt}\,((b_3 - b_5)^2 + b_4^2)) / b_4$$
$$p_{20} = Y_0 - p_{21} X_0$$

Through the three features of response surface model (stationary point, first principal axis, and second principal axis) we observe the relationship between external and internal members with project performance. Through jackknife method, we test the statistical significance of three features (See Table 16).

Table 16. Three Features of Response Surface Model

		Estimates (t-value)
Stationary point	X_0	*1.2428****
	Y_0	*40.9750****
First principal axis	P_{10}	*46.19313****

	P_{11}	*-4.19863****
Second principal axis	P_{20}	*40.67903****
	P_{21}	*0.23817****

Note: *p<.05, **p<.01, ***p<.001
jackknife method

From this formula, we can draw the relationships among the internal members, external members and project performance. Figure 11 show the response surface with three features. From this surface, we can observe that numbers of external developers have inverse-U shape relationship with project performance.

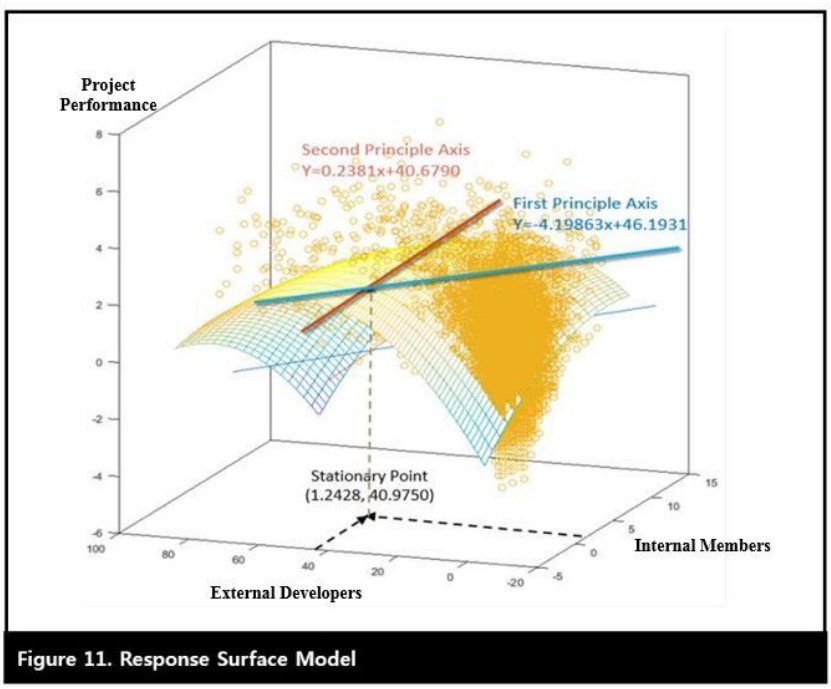


Figure 11. Response Surface Model

From this surface model, we can observe three different point of external developers. First, minimum numbers of external developers with internal member, stationary point of external developers with internal members, and maximum number of external developers and internal members (See Figure 12).

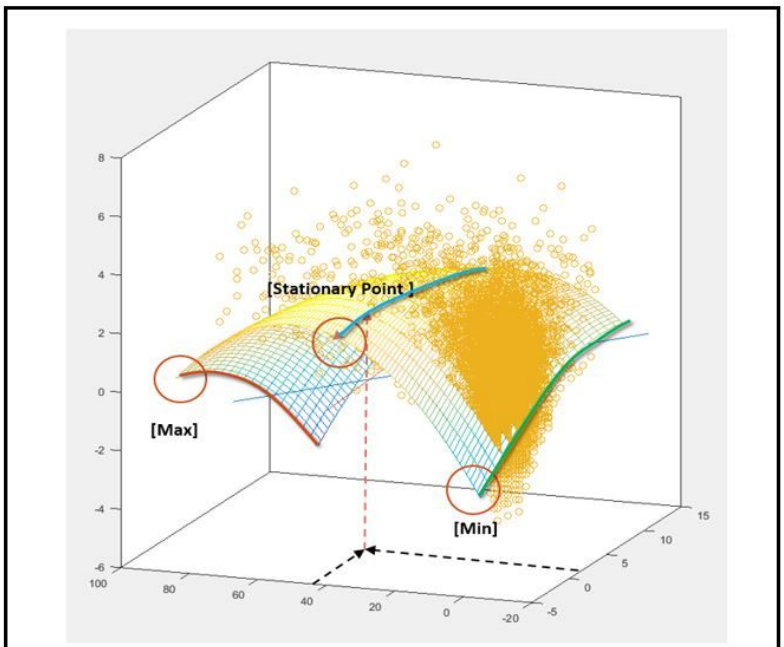


Figure 12. Three Points of Response Surface Model

Table 17 show the exact point of these three points with equation. Min, Stationary, and Max point can be draw through the equation of polynomial model. Since numbers of external developers are changed by mean-centering, minimum number of external members are – 5.

Table 17. Three Points of External Developers

	Point	Equations
[1] Min	-5	Project_Performance=-1.015+0.2547AM-0.01155AM^2
[2] Stationary Point	40.9750	Project_Performance=3.1128+0.02872AM-0.01155AM^2
[3] Max	84	Project_Performance=0.0067-0.1822AM-0.01155AM^2

In each three point, we could identify the different relationship with numbers of internal members and project performance. Figure 13 identify the different shape of slop of three points.

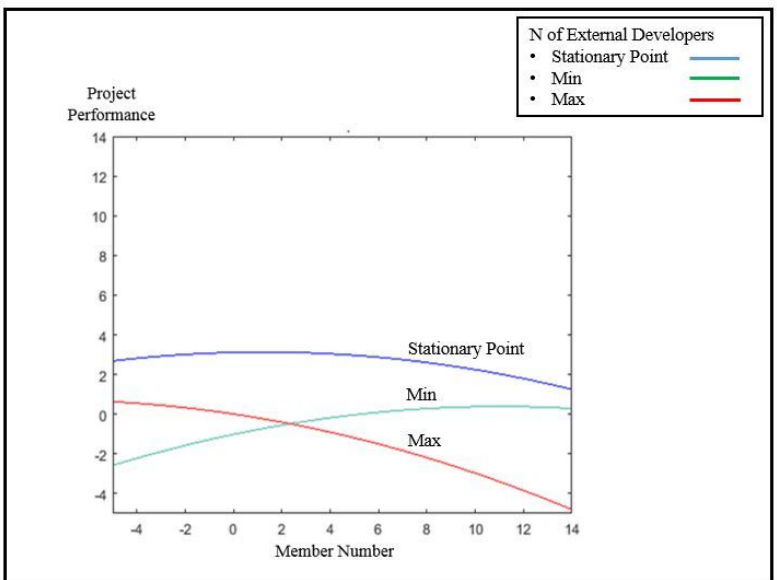


Figure 13. Relationship between Number of Member and Project Performance

First, if the number of external developers are minimum, numbers of internal members are positively associated with project performance. Of course, it is obvious that if the external members are minimum, there are lack of developers who can contribute the project. Therefore, the large number of internal members are positive impacts on performance. Interesting points of study 2 lay in stationary point and maximum numbers of external developers. In stationary point, numbers of internal member could no impacts on project performance. The degree of slops are minor. However, with maximum numbers of external developers, influence of internal developers are negatively related to project performance. From this results, we can assume that the phenomenon of negative effects of internal members with maximum numbers of external developers interpret with organizational learning theory. Since the OSS project which facilitate the activities of exploration, the pressure or control of internal members who regard as exploitation activities could reduce the activation of external developers.

7

5. 6. Conclusion

In this research, we assert the role of project size on project performance. If the project size is large, the performance of project have inverse U shape relationships. The results of hypothesis 1 conclude the inconclusive discussion among the previous research. Since the OSS project based on open collaboration, the philosophies of projects indicate 'going open' with considering the collaborators could contribute through new idea that could solve the problems of OSS project. However, others are worried about the large numbers of developers could reduce the efficiency of performance. Our research draw the hypothesis from the organizational learning theory which assert that too much exploration activities could diminish the innovation performance.

Another contributions of this research focus on the compositions of internal and external developers. Internal developers who already share the purpose of software and have experience with other collaborators could internalize the purpose of software or development norms of organizations. However, with the limitation of routinization, internal developers are easy to fail to recognize the value of new knowledge. Therefore, roles of external developers are also very important. In this study, we verify the specific compositions of internal and external developers in OSS project using response surface model. The results indicate that with low numbers of external developers, numbers of internal developers are positively influence on project performance. However, if the numbers of external members are large, internal developers are decrease the project performance.

Therefore, we could draw the contributions in practical governance of developments. If OSS project with large openness to external developers, it is better not to interrupt the process of knowledge creations even the internal developers feel far from the developments norms and initial purpose of projects. The results of our research precisely suggest how to govern the project considering the team compositions.

REFERENCES

Adler, P. S. 1999. Building better bureaucracies. ***The Academy of Management Executive***, 13(4): 36-47.

Adams, P. J., Capiluppi, A., & Boldyreff, C. 2009. ***Coordination and productivity issues in free software: The role of brooks' law***. in Proceedings of the Software Maintenance, 2009. ICSM 2009. IEEE International Conference on.

Ahn, H. K., S. 2003. A Study on the Victim Prevention through Community Network : focusing on effect of social capital and network. ***Correction Review***, 18: 121-143.

Ahuja, G., & Morris Lampert, C. 2001. Entrepreneurship in the large corporation: A longitudinal study of how established firms create breakthrough inventions. ***Strategic Management Journal***, 22(6-7): 521-543.

Ahuja, V., Yang, J., & Shankar, R. 2009. Benefits of collaborative ICT adoption for building project management. ***Construction Innovation***, 9(3): 323-340.

Aksulu, A., & Wade, M. 2010. A comprehensive review and synthesis of open source research. ***Journal of the Association for Information Systems***, 11(11): 576-656.

Antheunis, M. L., Valkenburg, P. M., & Peter, J. 2010. Getting acquainted through social network sites: Testing a model of online uncertainty reduction and social attraction. ***Computers in Human Behavior***, 26(1): 100-109.

Baek, H., Ahn, J., & Ha, S. 2011. Identifying Factors Affecting Helpfulness of Online Reviews: The Moderating Role of Product Price. ***The Journal of Society for e-Business Studies***, 16(3): 93-112.

Baek, H., & Oh, S. 2015. Identifying the Network Characteristics of Contributors That Affect Performance in Open Collaboration : Focusing on the GitHub Open Source. ***The Journal of Society for e-Business Studies***, 20(1): 23-43.

Baker, W. E. 1990. Market networks and corporate behavior. ***American Journal of Sociology***: 589-625.

Baldwin, C. Y., & Clark, K. B. 2006. The architecture of participation: Does code architecture mitigate free riding in the open source development model? ***Management Science***, 52(7): 1116-1127.

Balkundi, P., & Harrison, D. A. 2006. Ties, leaders, and time in teams: Strong inference about network structure's effects on team viability and performance. ***Academy of Management Journal***, 49(1): 49-68.

Barcellini, F., Détienne, F., & Burkhardt, J.-M. 2008. User and developer mediation in an Open Source Software community: Boundary spanning through cross participation in online discussions. ***International Journal of Human-Computer Studies***, 66(7): 558-570.

Barreto, A., Barros, M. d. O., & Werner, C. M. 2008. Staffing a software project: A constraint satisfaction and optimization-based approach. ***Computers & Operations Research***, 35(10): 3073-3089.

Bianchi, M., Cavaliere, A., Chiaroni, D., Frattini, F., & Chiesa, V. 2011. Organisational modes for Open Innovation in the bio-pharmaceutical industry: An exploratory analysis. ***Technovation***, 31(1): 22-33.

Bird, C., Gourley, A., Devanbu, P., Gertz, M., & Swaminathan, A. 2006. ***Mining email social networks***. in Proceedings of the 2006 international workshop on Mining software repositories.

Blau, P. M. 1970. A formal theory of differentiation in organizations. ***American Sociological Review***: 201-218.

Blincoe, K., Sheoran, J., Goggins, S., Petakovic, E., & Damian, D. 2016. Understanding the popular users: Following, affiliation influence and leadership on GitHub. ***Information and Software Technology***, 70: 30-39.

Bonaccorsi, A., & Rossi, C. 2003. Why open source software can succeed. ***Research Policy***, 32(7): 1243-1258.

Bosu, A., Carver, J., Guadagno, R., Bassett, B., McCallum, D., & Hochstein, L. 2014. Peer impressions in open source organizations: A survey. ***Journal of Systems and Software***, 94: 4-15.

Bourdieu, P. 1986. The Form of capital. Handbook of theory and research for the sociology of education. JG Richardson: New York, Greenwood Press.

Briggs, P., Burford, B., De Angeli, A., & Lynch, P. 2002. Trust in online advice. ***Social Science Computer Review***, 20(3): 321-332.

Brooks Jr, F. P. 1995. ***The Mythical Man-Month: Essays on Software Engineering, Anniversary Edition, 2/E***: Pearson Education India.

Brown, S. L., & Eisenhardt, K. M. 1997. The art of continuous change: Linking complexity theory and time-paced evolution in relentlessly shifting organizations. ***Administrative Science Quarterly***: 1-34.

Burt, R. S. 2009. ***Structural holes: The social structure of competition***: Harvard university press.

Certa, A., Enea, M., Galante, G., & Manuela La Fata, C. 2009. Multi-objective human resources allocation in R&D projects planning. ***International Journal of Production Research***, 47(13): 3503-3523.

Chatman, J. A. 1989. ***Matching people and organizations: Selection and socialization in public accounting firms***. in Proceedings of Academy of Management Proceedings.

Chengalur-Smith, S., & Sidorova, A. 2003. ***Survival of open-source projects: A population ecology perspective.*** in Proceedings of *ICIS 2003*: 66.

Chesbrough, H. 2006a. Open innovation: a new paradigm for understanding industrial innovation. ***Open innovation: Researching a New Paradigm***: 1-12.

Chesbrough, H. W. 2006b. ***Open innovation: The new imperative for creating and profiting from technology***: Harvard Business Press.

Choi, J., Kim, J., Ferwerda, B., Moon, J. Y., & Hahn, J. 2013. ***Impact of social features implemented in open collaboration platforms on volunteer self-organization: case study of open source software development***. in Proceedings of the 9th International Symposium on Open Collaboration.

Colazo, J. A. 2010. Collaboration structure and performance in new software development: findings from the study of open source projects. ***International Journal of Innovation Management***, 14(05): 735-758.

Coleman, J. S. 1988. Social capital in the creation of human capital. ***American Journal of Sociology***: S95-S120.

Comino, S., Manenti, F. M., & Parisi, M. L. 2005. From planning to mature: on the determinants of open source take off, Working Paper.

Cooper, R., Edgett, S., & Kleinschmidt, E. 2001. Portfolio management for new product development: results of an industry practices study. ***R&D Management***, 31(4): 361-380.

Cosentino, V., Izquierdo, J. L. C., & Cabot, J. 2014. Three Metrics to Explore the Openness of GitHub projects. *arXiv preprint arXiv:1409.4253*.

Crowston, K., Annabi, H., & Howison, J. 2003. ***Defining open source software project success***. in Proceedings of the ICIS 2003: 28.

Crowston, K., Annabi, H., Howison, J., & Masango, C. 2004. ***Effective work practices for software engineering: free/libre open source software development***. in Proceedings of the 2004 ACM workshop on Interdisciplinary software engineering research.

Crowston, K., & Howison, J. 2005. The social structure of free and open source software development. ***First Monday***, 10(2).

Crowston, K., & Howison, J. 2006. Hierarchy and centralization in free and open source software team communications. ***Knowledge, Technology & Policy***, 18(4): 65-85.

Dabbish, L., Stuart, C., Tsay, J., & Herbsleb, J. 2012. ***Social coding in github: transparency and collaboration in an open software repository***. in Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work.

Dabbish, L., Stuart, C., Tsay, J., & Herbsleb, J. 2013. Leveraging transparency. ***Software, IEEE***, 30(1): 37-43.

Dahlander, L., & Magnusson, M. 2008. How do firms make use of open source communities? ***Long Range Planning***, 41(6): 629-649.

Dahlander, L., & O'Mahony, S. 2011. Progressing to the center: Coordinating project work. ***Organization science***, 22(4): 961-979.

Darley, J. M., & Latane, B. 1968. Bystander intervention in emergencies: diffusion of responsibility. ***Journal of personality and social psychology***, 8(4p1): 377.

David, P. A., & Rullani, F. 2008. Dynamics of innovation in an “open source” collaboration environment: lurking, laboring, and launching FLOSS projects on SourceForge. ***Industrial and Corporate Change***, 17(4): 647-710.

Davis, R. C. 2015. Git and GitHub for Librarians. ***Behavioral & Social Sciences Librarian***, 34(3): 158-164.

Dosi, G. 1988. Sources, procedures, and microeconomic effects of innovation. ***Journal of economic literature***: 1120-1171.

Dourish, P., & Bellotti, V. 1992. ***Awareness and coordination in shared workspaces***. in Proceedings of the 1992 ACM conference on Computer-supported cooperative work.

Edmondson, A. C., Bohmer, R. M., & Pisano, G. P. 2001. Disrupted routines: Team learning and new technology implementation in hospitals. ***Administrative Science Quarterly***, 46(4): 685-716.

Edwards, J. R. 1994. Regression analysis as an alternative to difference scores. ***Journal of Management***, 20(3): 683-689.

Edwards, J. R., & Parry, M. E. 1993. On the use of polynomial regression equations as an alternative to difference scores in organizational research. ***Academy of Management Journal***, 36(6): 1577-1613.

Edwards, J. R., & Van Harrison, R. 1993. Job demands and worker health: Three-dimensional reexamination of the relationship between person€ nvironment fit and strain. ***Journal of Applied Psychology***, 78(4): 628.

Engwall, M., & Jerbrant, A. 2003. The resource allocation syndrome: the prime challenge of multi-project management? ***International Journal of Project Management***, 21(6): 403-409.

Ensley, M. D., & Hmieleski, K. M. 2005. A comparative study of new venture top management team composition, dynamics and performance between university-based and independent start-ups. ***Research Policy***, 34(7): 1091-1105.

Eskerod, P. 1998. ***The human resource allocation process when organising by projects,*** Projects as arenas for renewal and learning processes: 125-131: Springer.

Fang, Y., & Neufeld, D. 2009. Understanding sustained participation in open source software projects. ***Journal of Management Information Systems***, 25(4): 9-50.

Faraj, S., Kudaravalli, S., & Wasko, M. 2015. Leading collaboration in online communities. ***MIS Quarterly***, 39(2): 393-412.

Feliciano, J. 2015. ***Towards a Collaborative Learning Platform: The Use of GitHub in Computer Science and Software Engineering Courses***. University of Victoria.

Ferriani, S., Cattani, G., & Baden-Fuller, C. 2009. The relational antecedents of project-entrepreneurship: Network centrality, team composition and project performance. ***Research Policy***, 38(10): 1545-1558.

Forbes, D. P., Borchert, P. S., Zellmer-Bruhn, M. E., & Sapienza, H. J. 2006. Entrepreneurial team formation: An exploration of new member addition. ***Entrepreneurship Theory and Practice***, 30(2): 225-248.

Fortune. 2015.09.29. Github CEO: What I Learned from Our Harassment Scandal, Daniel Roberts.

Franck, E., & Jungwirth, C. 2003. Reconciling rent-seekers and donators–The governance structure of open source. ***Journal of Management and Governance***, 7(4): 401-421.

Franke, N., & Von Hippel, E. 2003. Satisfying heterogeneous user needs via innovation toolkits: the case of Apache security software. ***Research Policy***, 32(7): 1199-1215.

Freeman, L. C. 1978. Centrality in social networks conceptual clarification. ***Social Networks***, 1(3): 215-239.

Fukuyama, F. 1995. ***Trust: The social virtues and the creation of prosperity.***

Ghosh, R. A., & Prakash, V. V. 2000. The orbiten free software survey. ***First Monday***, 5(7).

Gibbs, J. L., Ellison, N. B., & Lai, C.-H. 2010. First comes love, then comes Google: An investigation of uncertainty reduction strategies and self-disclosure in online dating. ***Communication Research***: 0093650210377091.

Gilsing, V., Nooteboom, B., Vanhaverbeke, W., Duysters, G., & van den Oord, A. 2008. Network embeddedness and the exploration of novel technologies: Technological distance, betweenness centrality and density. ***Research Policy***, 37(10): 1717-1731.

Giuri, P., Ploner, M., Rullani, F., & Torrisi, S. 2010. Skills, division of labor and performance in collective inventions: Evidence from open source software. ***International Journal of Industrial Organization***, 28(1): 54-68.

Giuri, P., Rullani, F., & Torrisi, S. 2008. Explaining leadership in virtual teams: The case of open source software. ***Information Economics and Policy***, 20(4): 305-315.

Glancey, K. 1998. Determinants of growth and profitability in small entrepreneurial firms. ***International Journal of Entrepreneurial Behavior & Research***, 4(1): 18-27.

Gregorian, A. 2014. A Tie Strength Model For Reconstructing Collaboration Networks on GitHub A study of the Ruby on Rails project network.

Grewal, R., Lilien, G. L., & Mallapragada, G. 2006. Location, location, location: How network embeddedness affects project success in open source systems. ***Management Science***, 52(7): 1043-1056.

Gulati, R., & Gargiulo, M. 1999. Where Do Interorganizational Networks Come From? ***American Journal of Sociology***, 104: 177-231.

Gutwin, C., Schneider, K., Paquette, D., & Penner, R. 2004. Supporting group awareness in distributed software development, ***Engineering Human Computer Interaction and Interactive Systems***: 383-397: Springer.

Hahn, J., Moon, J. Y., & Zhang, C. 2008. Emergence of new project teams from open source software developer networks: Impact of prior collaboration ties. ***Information Systems Research***, 19(3): 369-391.

Hansen, M. T. 1999. The search-transfer problem: The role of weak ties in sharing knowledge across organization subunits. ***Administrative Science Quarterly***, 44(1): 82-111.

Hars, A., & Ou, S. 2001. ***Working for free? Motivations of participating in open source projects***. in Proceedings of the 34th Annual Hawaii International Conference on System Sciences, 2001

He, Z.-L., & Wong, P.-K. 2004. Exploration vs. exploitation: An empirical test of the ambidexterity hypothesis. ***Organization Science***, 15(4): 481-494.

Henderson, R. M., & Clark, K. B. 1990. Architectural innovation: The reconfiguration of existing product technologies and the failure of established firms. ***Administrative science quarterly***: 9-30.

Hendriks, M., Voeten, B., & Kroep, L. 1999. Human resource allocation in a multi-project R&D environment: resource capacity allocation and project portfolio planning in practice. ***International Journal of Project Management***, 17(3): 181-188.

Heo, C. Y. 2016. Sharing economy and prospects in tourism research. ***Annals of Tourism Research***, 58(C): 166-170.

Hertel, G., Niedner, S., & Herrmann, S. 2003. Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel. ***Research Policy***, 32(7): 1159-1177.

Hippel, E. v., & Krogh, G. v. 2003. Open source software and the "private-collective" innovation model: Issues for organization science. ***Organization Science***, 14(2): 209-223.

Holmqvist, M. 2004. Experiential learning processes of exploitation and exploration within and between organizations: An empirical study of product development. ***Organization Science***, 15(1): 70-81.

Hossain, L., & Wigand, R. T. 2004. ICT enabled virtual collaboration through trust. ***Journal of Computer-Mediated Communication***, 10(1): 00-00.

Howell, D., Rogier, M., Yzerbyt, V., & Bestgen, Y. 1998. Statistical methods in human sciences. ***New York: Wadsworth***.

Howison, J., Inoue, K., & Crowston, K. 2006. Social dynamics of free and open source team communications, ***Open Source Systems***: 319-330: Springer.

Hyatt, D. E., & Ruddy, T. M. 1997. An examination of the relationship between work group characteristics and performance: Once more into the breech. ***Personnel Psychology***, 50(3): 553-585.

Jackson, D., Kirkland, J., Jackson, B., & Bimler, D. 2005. Social Network Analysis and Estimating the Size of Hard-to-Count Subpopulations1. ***Connections***, 26(2): 49-60.

Jackson, M. O. 2005. A survey of network formation models: stability and efficiency. ***Group Formation in Economics: Networks, Clubs, and Coalitions***: 11-49.

Jackson, M. O., & Wolinsky, A. 1996. A strategic model of social and economic networks. ***Journal of economic theory***, 71(1): 44-74.

Jones, C., Hesterly, W. S., & Borgatti, S. P. 1997. A general theory of network governance: Exchange conditions and social mechanisms. ***Academy of Management Review***, 22(4): 911-945.

Kalliamvakou, E., Damian, D., Blincoe, K., Singer, L., & German, D. 2015. ***Open Source-Style Collaborative Development Practices in Commercial Projects Using GitHub***. Paper presented at the Proceedings of the 37th International Conference on Software Engineering (ICSE'15).

Kalliamvakou, E., Damian, D., Singer, L., & German, D. M. 2014. The code-centric collaboration perspective: Evidence from github: Technical Report DCS-352-IR, University of Victoria.

Katila, R., & Ahuja, G. 2002. Something old, something new: A longitudinal study of search behavior and new product introduction. ***Academy of Management Journal***, 45(6): 1183-1194.

Ke, W., & Zhang, P. 2010. The effects of extrinsic motivations and satisfaction in open source software development. ***Journal of the Association for Information Systems***, 11(12): 784-808.

Keupp, M. M., & Gassmann, O. 2009. Determinants and archetype users of open innovation. ***R&D Management***, 39(4): 331-341.

Kieser, A., Beck, N., & Tainio, R. 2001. Rules and organizational learning: The behavioral theory approach. ***Handbook of organizational learning and knowledge***: 598-623.

Ko, S. H., B. Ji, Y. 2010. A Study on Social Network Service and Online Social Capital : Focusing on a Korean and Chinese Case. ***Korea Institute of Science and Technology Information***, 15(1): 103-118.

Koput, K. W. 1997. A chaotic model of innovative search: some answers, many questions. ***Organization Science***, 8(5): 528-542.

Krishnamurthy, S. 2002. Cave or community?: An empirical examination of 100 mature open source projects. ***First Monday***.

Kwak, G. 2014. Social Network Analysis ***CheongRam***.

Lakhani, K., & Wolf, R. G. 2003. Why hackers do what they do: Understanding motivation and effort in free/open source software projects.

Laursen, K., & Salter, A. 2006. Open for innovation: the role of openness in explaining innovation performance among UK manufacturing firms. ***Strategic Management Journal***, 27(2): 131-150.

Lee, S., Baek, H., & Jahng, J. 2016. Role of Project Owner in OSS Project: Based on Impression Formation and Social Capital Theory. ***The Journal of Society for e-Business Studies***, 21(02).

Lee, S., Park, G., Yoon, B., & Park, J. 2010. Open innovation in SMEs—An intermediated network model. ***Research Policy***, 39(2): 290-300.

Lerner, J., & Tirole, J. 2001. The open source movement: Key research questions. ***European Economic Review***, 45(4): 819-826.

Lerner, J., & Tirole, J. 2005. The scope of open source licensing. ***Journal of Law, Economics, and Organization***, 21(1): 20-56.

Lerner, J., & Triole, J. 2000. The simple economics of open source: National Bureau of Economic Research.

Levine, S. S., & Prietula, M. J. 2013. Open collaboration for innovation: principles and performance. ***Organization Science***, 25(5): 1414-1433.

Levinthal, D. A., & March, J. G. 1993. The myopia of learning. ***Strategic Management Journal***, 14(S2): 95-112.

Levitt, B., & March, J. G. 1988. Organizational learning. ***Annual review of sociology***: 319-340.

Liberatore, M. J. 1987. An extension of the analytic hierarchy process for industrial R&D project selection and resource allocation. ***IEEE Transactions on Engineering Management***, 34(1): 12-18.

Lichtenthaler, U., & Ernst, H. 2009. Opening up the innovation process: the role of technology aggressiveness. ***R&D Management***, 39(1): 38-54.

Lin, X., & Germain, R. 2003. Organizational structure, context, customer orientation, and performance: lessons from Chinese state-owned enterprises. ***Strategic management journal***, 24(11): 1131-1151.

Lima, A., Rossi, L., & Musolesi, M. 2014. Coding Together at Scale: GitHub as a Collaborative Social Network. ***arXiv preprint arXiv:1407.2535***.

Longo, J., & Kelley, T. M. 2015. ***Use of GitHub as a platform for open collaboration on text documents***. Paper presented at the Proceedings of the 11th International Symposium on Open Collaboration.

Ma, M., & Agarwal, R. 2007. Through a glass darkly: Information technology design, identity verification, and knowledge contribution in online communities. ***Information systems research***, 18(1): 42-67.

MacCormack, A., Rusnak, J., & Baldwin, C. Y. 2006. Exploring the structure of complex software designs: An empirical study of open source and proprietary code. ***Management Science***, 52(7): 1015-1030.

Madey, G. 2005. Modeling the Free/Open Source software community: A quantitative investigation. ***Free/Open Source Software Development***: 203-221.

March, J. G. 1991. Exploration and exploitation in organizational learning. ***Organization science***, 2(1): 71-87.

Markus, M. L., & Agres, B. M. C. E. 2000. What makes a virtual organization work? ***MIT Sloan Management Review***, 42(1): 13.

Marlow, J., Dabbish, L., & Herbsleb, J. 2013. ***Impression formation in online peer production: activity traces and personal profiles in github***. in Proceedings of the 2013 conference on Computer supported cooperative work.

McGrath, J. E. 1964. Toward a “theory of method” for research on organizations. ***New perspectives in organization research***, 533: 533-547.

McPherson, A., Proffitt, B., & Hale-Evans, R. 2008. Estimating the total development cost of a linux distribution. ***The Linux Foundation***.

Mintzberg, H. 1979. An emerging strategy of" direct" research. ***Administrative Science Quarterly***: 582-589.

Mockus, A., Fielding, R. T., & 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): 309-346.

Moon, J. Y., & Sproull, L. 2002. Essence of distributed work: The case of the Linux kernel. ***Distributed work***: 381-404.

Moore, C. 2007. Impression formation: Blackwell.

Nelson, R. R., & Winter, S. G. 1982. The Schumpeterian tradeoff revisited. ***The American Economic Review***, 72(1): 114-132.

Nelson, R. R., & Winter, S. G. 2009. ***An evolutionary theory of economic change***: Harvard University Press.

Neus, A., & Scherf, P. 2005. Opening minds: Cultural change with the introduction of open-source collaboration methods. ***IBM Systems Journal***, 44(2): 215-225.

Ngamkajornwiwat, K., Zhang, D., Koru, A. G., Zhou, L., & Nolker, R. 2008. ***An exploratory study on the evolution of oss developer communities***. Paper presented at the Hawaii International Conference on System Sciences, Proceedings of the 41st Annual.

Nooteboom, B. 2000. ***Learning and innovation in organizations and economies***: OUP Oxford.

O'Mahony, S., & Ferraro, F. 2004. Hacking alone? The effects of online and offline participation on open source community leadership.

Ocasio, W. 1997. Towards an attention-based view of the firm. ***Strategic management journal***, 1: 403-404.

Onoue, S., Hata, H., & Matsumoto, K.-i. 2013. ***A Study of the Characteristics of Developers' Activities in GitHub***. Paper presented at the Software Engineering Conference (APSEC, 2013 20th Asia-Pacific.

Osterloh, M., & Rota, S. 2007. Open source software development—Just another case of collective invention? ***Research Policy***, 36(2): 157-171.

Perens, B. 1999. The open source definition. ***Open sources: voices from the open source revolution***, 1: 171-188.

Perretti, F., & Negro, G. 2007. Mixing genres and matching people: a study in innovation and team composition in Hollywood. ***Journal of Organizational Behavior***, 28(5): 563-586.

Rau, B. L., & Hyland, M. A. M. 2002. Role conflict and flexible work arrangements: The effects on applicant attraction. ***Personnel psychology***, 55(1): 111-136.

Raymond, E. 1999. The cathedral and the bazaar. ***Knowledge, Technology & Policy***, 12(3): 23-49.

Raymond, E. S., & Young, B. 2001. The cathedral and the bazaar-musings on Linux and open source by an accidental revoltionary (rev. ed.): O'reilly.

Reagans, R., & Zuckerman, E. W. 2001. Networks, diversity, and productivity: The social capital of corporate R&D teams. ***Organization Science***, 12(4): 502-517.

Reid, R. S., & Adams, J. S. 2001. Human resource management-a survey of practices within family and non-family firms. ***Journal of European Industrial Training***, 25(6): 310-320.

Roberts, J. A., Hann, I.-H., & 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): 984-999.

Rossi, M. A. 2004. Decoding the" free/open Source (F/OSS) Software Puzzle", a Survey of Theoretical and Empirical Contributions.

Rothaermel, F. T., & Deeds, D. L. 2004. Exploration and exploitation alliances in biotechnology: A system of new product development. ***Strategic Management Journal***, 25(3): 201-221.

Rowley, T., Behrens, D., & Krackhardt, D. 2000. Redundant governance structures: An analysis of structural and relational embeddedness in the steel and semiconductor industries. ***Strategic Management Journal***, 21(3): 369-386.

Ruef, M., Aldrich, H. E., & Carter, N. M. 2003. The structure of founding teams: Homophily, strong ties, and isolation among US entrepreneurs. ***American sociological review***: 195-222.

Rullani, F., & Frederiksen, L. 2010. ***Individual Interaction and Innovation Capabilities: Exploration and Exploitation in Open Source Software Communities***.

Scacchi, W., & Jensen, C. 2008. Governance in open source software development projects: Towards a model for network-centric edge organizations: DTIC Document.

Sen, R., Subramaniam, C., & Nelson, M. L. 2008. Determinants of the choice of open source software license. ***Journal of Management Information Systems***, 25(3): 207-240.

Shah, S. K. 2006. Motivation, governance, and the viability of hybrid forms in open source software development. ***Management Science***, 52(7): 1000-1014.

Sharma, S., Sugumaran, V., & Rajagopalan, B. 2002. A framework for creating hybrid-open source software communities. ***Information Systems Journal***, 12(1): 7-25.

Singh, P. V. 2010. The small-world effect: The influence of macro-level properties of developer collaboration networks on open-source project success. ***ACM Transactions on Software Engineering and Methodology (TOSEM)***, 20(2): 6.

Singh, P. V., & Tan, Y. 2010. Developer heterogeneity and formation of communication networks in open source software projects. ***Journal of Management Information Systems***, 27(3): 179-210.

Singh, P. V., Tan, Y., & Mookerjee, V. 2008. Network effects: The influence of structural social capital on open source project success. ***Management Information Systems Quarterly, Forthcoming***.

Smelser, N. J., & Swedberg, R. 2010. ***The handbook of economic sociology***: Princeton university press.

Smith-Doerr, L., & Powell, W. W. 2005. Networks and economic life. ***The handbook of economic sociology***, 2: 379-402.

Sparrowe, R. T., Liden, R. C., Wayne, S. J., & Kraimer, M. L. 2001. Social networks and the performance of individuals and groups. ***Academy of Management Journal***, 44(2): 316-325.

Sproull, L., & Kiesler, S. 1986. Reducing social context cues: Electronic mail in organizational communication. ***Management Science***, 32(11): 1492-1512.

Stewart, K. J., Ammeter, A. P., & 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.

Stol, K.-J., & Fitzgerald, B. 2014. ***Two's company, three's a crowd: a case study of crowdsourcing software development***. Paper presented at the Proceedings of the 36th International Conference on Software Engineering.

Tan, Y., Mookerjee, V., & Singh, P. 2007. ***Social capital, structural holes and team composition: Collaborative networks of the open source software community.*** in Proceedings of ICIS 2007: 155.

Tsai, W. 2000. Social capital, strategic relatedness and the formation of intraorganizational linkages. ***Strategic Management Journal***, 21(9): 925-939.

Tsay, J., Dabbish, L., & Herbsleb, J. 2014. ***Let's talk about it: evaluating contributions through discussion in GitHub***. in Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering.

Tseng, C.-C., & Chen, P.-c. 2009. ***SEARCH ACTIVITIES FOR INNOVATION--AN ATTENTION-BASED VIEW***. Paper presented at the Academy of Management Proceedings.

Turner, J. R., & Speiser, A. 1992. Programme management and its information systems requirements. ***International Journal of Project Management***, 10(4): 196-206.

Tushman, M. L., & Rosenkopf, L. 1992. Organizational determinants of technological-change-toward a sociology of technological evolution. ***Research in Organizational Behavior***, 14: 311-347.

Uotila, J., Maula, M., Keil, T., & Zahra, S. A. 2009. Exploration, exploitation, and financial performance: analysis of S&P 500 corporations. ***Strategic Management Journal***, 30(2): 221-231.

Vasilescu, B., Filkov, V., & Serebrenik, A. 2015. Perceptions of diversity on GitHub: A user survey. ***CHASE. IEEE***.

Venkatraman, N., & Henderson, J. C. 1998. Real strategies for virtual organizing. ***MIT Sloan Management Review***, 40(1): 33.

Von Hippel, E. 2001. Innovation by user communities: Learning from open-source software. ***MIT Sloan management review***, 42(4): 82.

Von Krogh, G., Haefliger, S., Spaeth, S., & Wallin, M. W. 2012. Carrots and rainbows: Motivation and social practice in open source software development. ***MIS Quarterly***, 36(2): 649-676.

Von Krogh, G., Spaeth, S., & Lakhani, K. R. 2003. Community, joining, and specialization in open source software innovation: a case study. ***Research Policy***, 32(7): 1217-1241.

Voss, G. B., & Voss, Z. G. 2013. Strategic ambidexterity in small and medium-sized enterprises: Implementing exploration and exploitation in product and market domains. ***Organization Science***, 24(5): 1459-1477.

Walther, J. B. 1992. Interpersonal effects in computer-mediated interaction a relational perspective. ***Communication Research***, 19(1): 52-90.

Watson, W. E., Kumar, K., & Michaelsen, L. K. 1993. Cultural diversity's impact on interaction process and performance: Comparing homogeneous and diverse task groups. ***Academy of Management Journal***, 36(3): 590-602.

Weber, S. 2004. ***The success of open source***: Cambridge Univ Press.

Wikipedia. 2016.01.14. Github.

Wu, C.-G., Gerlach, J. H., & Young, C. E. 2007. An empirical analysis of open source software developers' motivations and continuance intentions. ***Information & Management***, 44(3): 253-262.

Xu, J., & Madey, G. 2004. Exploration of the open source software community. ***Proceedings of North American Association for Computational Social and Organizational Science (NAACSOS), Pittsburgh, PA, USA***.

Yang, H.-L., & Tang, J.-H. 2004. Team structure and team performance in IS development: a social network perspective. ***Information & Management***, 41(3): 335-349.

Ye, Y., & Kishida, K. 2003. ***Toward an understanding of the motivation of open source software developers***. in Proceedings of the 25th International Conference on.Software Engineering.

Yoo, Y., & Alavi, M. 2004. Emergent leadership in virtual teams: what do emergent leaders do? ***Information and Organization***, 14(1): 27-58.

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.

Zaltman, G., Duncan, R., & Holbek, J. 1973. ***Innovations and Organizations***: Wiley New York.

Zenger, T. R., & Lawrence, B. S. 1989. Organizational demography: The differential effects of age and tenure distributions on technical communication. ***Academy of Management Journal***, 32(2): 353-376.

Zika-Viktorsson, A., Sundström, P., & Engwall, M. 2006. Project overload: An exploratory study of work and management in multi-project settings. ***International Journal of Project Management***, 24(5): 385-394.

Abstract

혁신을 추구하는 많은 기업들이 기업 내부의 자원을 통한 혁신에서 나아가 기업 외부의 자원을 통한 혁신에 투자하고 있다. 최근 인터넷의 발전으로 자발적으로 불특정 다수가 개방적 협업을 통해 혁신을 창출하고 있으며, 이는 기업들이 투자하여 개발한 혁신보다 좋은 성과를 내기도 한다. 특별히 본 연구에서는 개방적 협업의 대표적인 예시인 오픈소스 소프트웨어 분야의 혁신 성과에 영향을 주는 요인에 대하여 검증하고자 한다. 오픈소스 소프트웨어는 불특정다수에 의하여 자발적으로 개발되나, 특정한 기능을 가진 소프트웨어를 개발하는 것이 최종 목적이기 때문에 개발을 관리하는 방법이 소프트웨어 개발 프로젝트의 성과에 영향을 미칠 수 있다. 따라서 본 연구에서는 오픈소스 소프트웨어 개발 프로젝트의 관리에 대한 세가지 차원의 분석을 시행하고자 한다. 먼저 개인의 개발자가 프로젝트를 개설하고 관리하는 경우, 프로젝트의 운영자가 되며, 운영자의 역할이 프로젝트의 성과에 영향을 미칠 수 있다. 나아가 프로젝트를 운영하는 개발자들이 조직을 이루고 개발을 할 경우, 조직마다 프로젝트를 운영하는 방식이 상이하며 이것이 프로젝트 성과에 영향을 미칠 수 있다. 마지막으로 프로젝트 별로 고유한 특징이 프로젝트의 성과에 미치는 영향을 조명하고자 한다. 본 연구에서는 세 관점에 대한 분석을 위하여 오픈소스 소프트웨어 개발 플랫폼인 깃허브에서 웹크롤링 방식으로 데이터를 수집하였다. 본 연구의 결과는 향후 개방적 협업으로 오픈소스 소프트웨어 프로젝트의 성공적인 관리에 실질적인 방향성을 제시하며, 혁신관리의 일환으로서 기존의 연구에 이론적인 기여점을 가진다.

주요어 : 혁신 성과, 개방적 협업, 오픈소스 소프트웨어, 프로젝트 관리

학 번 : 2010-20504