Evaluating Longitudinal Success of Open Source Software Projects: A Social Network Perspective

Jing Wu
National University of Singapore
wujing@comp.nus.edu.sg

Khim Yong Goh
National University of Singapore
gohky@comp.nus.edu.sg

Abstract

To date, numerous open source projects are hosted on many online repositories. While some of these projects are active and thriving, some projects are either languishing or showing no development activities at all. This phenomenon thus begs the important question of what are the influential factors that affect the success of open source projects. In a quest to deepen our understanding of the evolution of open source projects, this research aims to analyze the success of open source projects by using the theoretical lens of social network analysis. Based on extensive analyses of data collected from online repositories, we study the impact of the communication patterns of software development teams on the demand and supply outcomes of these projects, while accounting for project-specific characteristics. Using panel data analysis of data over 13 months, we find significant impacts of communication patterns on project outcomes over the long term.

1. Introduction

Recent years have seen a rapid growth of open source software (OSS). Ever since the first OSS was developed by Richard Stallman (GNU) in the 1970's, a multitude of open source applications have been developed, ranging from office productivity software such as StarOffice, to database and thousands of specialized scientific applications. It is reported that more than 65 percent of public websites are now backed by the open-source Apache web server; 80 percent of the world's e-mail traffic is managed by Sendmail and 40% of large U.S. corporations make use of the open-source GNU Linux operating system [40].

The growing popularity of OSS has garnered increasing attention not only from practitioners in the industry, but also from academic scholars who are interested in examining this phenomenon in a rigorous in-depth manner. Various case studies have contributed to a better understanding of the OSS phenomenon. [22] considered the nature and the functioning of the community of developers of the Apache software. [16] focused on factors determining the level of engagement in the Linux project. [20] analyzed the strategic process by which new individuals joined the community of developers of FreeNet, a peer-to-peer network of information distribution. These studies shed new light on how large communities of developers arise, work and coordinate to achieve the success of an open source project. However, previous case studies are limited to large and popular projects only. While in-depth examinations on such large and popular projects are crucial to better understand how communities work effectively, findings from such studies may not be representative of the OSS community in general.

Several large open source projects have achieved extraordinary success and are among the most prominent software used in the technology industry. However, many open source projects have been lackluster with few or no development activities at all. Many flounder at the beginning, while others survive, but with little momentum behind them [38]. The failure of a large number of open source projects begs the following key question: What contributes to the long-term sustainability of the OSS movement? In the other words, what are the factors, besides project-specific characteristics, that could influence the success or failure of open source projects? To deepen our understanding of OSS, it is essential for Information Systems (IS) researchers to study these questions theoretically and provide insights to practitioners.

The open source community is characterized by the voluntary participation of software developers collaborating over the Internet with the aim to produce license-free software. The developers have been creating value through developing and spreading new knowledge and capabilities, fostering innovations, and building and testing trust in working relations, relying heavily on information and communication technologies to accomplish their tasks [30]. For the development teams, to achieve their objectives and successfully complete their tasks, information must be effectively exchanged. Thus, communication and coordination have been found to be two major aspects that significantly affect the performance of such teams [18]; [28]. OSS development is a complex socio-technical activity, requiring people to interact with each other. Thus, it is interesting to study the communication patterns of open source development

978-0-7695-3450-3/09 $25.00 © 2009 IEEE

teams to investigate the relation between coordination and communication characteristics (i.e., the social network attributes) of OSS project teams and the evolving outcomes of open source projects.

While others have studied the determinants of open source success (e.g., [9]; [2]; [35]; [1]; [36]; [14]), this study is among the first to explore open source project success through the lens of social network perspective. Through social network analysis of empirical data collected from open source projects, we study the impact of the communication patterns of open source projects on the outcomes of these projects, while accounting for project-specific characteristics. Such a novel approach thus incorporates both the supply side (developers) and the demand side (end users) factors. As we know, communication patterns may change with time and thus success or failure of OSS projects is transient. It is therefore important to examine the dynamic impacts of communication patterns on project success such that we can assess the long term sustainability of OSS projects. Thus, in this study, we observe the changes of communication pattern of each project across an extended period of 13 months, and investigate the evolving success of open source projects by looking at the dynamic impacts of communication patterns.

2. Theoretical Background

2.1. Communication Pattern of Project Teams

Open source developers collaborate mainly over the Internet. The advent of information and communication technologies provides instantaneous global accessibility for the open source community. Software development is a complex socio-technical activity. The developers of an open source project collaborate via interactions or communications in the form of email exchange, message boards, etc. [32]. The communication and interaction among individuals and groups form the network of relationships inside the project team. To better understand the impact of such communications on the success of open source projects, we employ the social network analysis (SNA) method, which helps to identify the prominent patterns in such networks, trace the flow of information (and other resources), and discover potential relationships between the social structure and the final product, i.e. the software system [19].

SNA aims to understand the relationships between people, groups, organizations, and other types of social entities [12]; [39]; [41] by description, visualization, and statistical modeling. It models social relationships in terms of nodes and ties. Nodes represent the individual actors or groups within the network, and ties or links show interactions or exchange of information flows between the nodes. In the context of open source projects, nodes are the developers, and ties are the interactions or communications between developers. In the MIS field, previous literature which focused on OSS research has shown that social networks operate on many levels and play a critical role in determining the way of solving problems, running organizations, and the degree to which individuals achieve their goals [17]; [26]; [42].

Social structure, a term frequently used in social theory, refers to entities or groups in definite relation to each other, to relatively enduring patterns of behavior and relationship within social systems [34]. The social structure of an open source development team describes how people interact, behave and organize in the community. Investigating social structure is a useful way to understand team practice such as coordination, control, socialization, continuity and learning [10]; [33]. [6] interviewed a member of the Apache Foundation's incubator team at ApacheCon 2003. The study of social structure helps to identify the reasons for such concerns since it provides an assessment measure of finding the crucial members as well as their importance with regard to the project.

The communication pattern describes the structure of interactions during communication. It can be characterized by several attributes. According to social network theory, the centrality and density of a group are related to its efficiency of problem solving, perception of leadership and the personal satisfaction of participants [34]. The concepts of density and centrality refer to different aspects of the overall "compactness" of the network [34]. Density describes the general level of cohesion in the network while centrality describes the extent to which this cohesion is organized around particular focal points. Centrality and density, therefore, are important complementary measures of the communication pattern.

Density measures how closely a network is connected, which in turn determines the readiness of a group in response to changes in processes and outcomes. It is defined as the percentage of ties that exist in a network out of all possible ties.

Centrality can be defined on an individual or overall level for a network. The centrality of an individual node refers to the number of direct links to other nodes in a network. If we define the link between nodes as communications, a person with a high centrality represents a major channel of information exchange. In some sense he is a focal point of communication, at least with respect to others who has contact with him. At the opposite extreme is a point of low centrality degree. The occupant of such a position is likely to be seen as peripheral. His position isolates

him from direct involvement with most of the others in the network and cuts him off from active participation in major communication processes. Thus, the centrality measure indicates whether a group member is "in the thick of things" [10]; [29]. In order to track the influence of the project leader(s), we examine the individual centrality measure of project leader(s) since the centrality of the leader(s) indicates the stature and influence of the leader(s) in the project team [15].

One can also define the centrality of a network as a whole. Project centrality, centrality of an entire project team, captures the inequality of the developers' contributions to the project: high score of project centrality implies that the power of individual developers varies rather substantially, and overall, positional advantages are rather unequally distributed in this network. Social network theory [23] suggests that the speed and efficiency of a network in solving problems are related to the inequality of the developers' contributions to the project.

2.2. Success of Open Source Projects

Unlike traditional firm-driven endeavors, open source projects are not always driven by direct profit motives [8]; [22]. The success indicators of commercial software such as market share, on time and on budget delivery cannot be readily applied in the OSS setting. In the OSS environment, there is usually no pre-determined deadline, a priori budget, or a set of specifications [33], and market share of OSS is difficult to assess. Thus, a set of different indicators are necessary to define the success of open source projects.

Success is a subjective concept and therefore it is not always clear on how to define success. [31] defined successful OSS projects as those characterized by a continuing process of volunteer developers fixing bugs, adding features and releasing software "often and early". Since a large number of OSS projects are abandoned by their developers, it is critical to attract contributors on an on-going basis to keep the project sustainable [27]. [4] explored success measures in the IS literature and suggested a portfolio of success measures, including measures of the development process. Subsequently, [5] analyzed four success measurements by using data from SourceForge.net and suggested that a project that attracts developers, maintains a high level of activity, fixes bugs and has many users downloads can described as successful. There are some other scholars advocating different success measurements: number of developers, development stages, output per contributor, project popularity, user interest, number of CVS commits[1], number of downloads [1]; [2]; [9]; [35]; [36]; [14].

In our study, we consider success from both the supply side (developers) and the demand side (end users). Since open source development relies on voluntary input, attracting and motivating contributors are key factors for its success. In other words, development activity is a key indicator of project success: high development activity shows that the developers in the project continuously contribute to the project; the project will evolve until it has no development activity at all. On the demand side, project popularity is a key measure of the project's success: high popularity shows that there are many users using or are interested in using the open source software. On the other hand, an OSS project will cease to exist or progress if there is no demand or if no one makes use of the end product for an extended period.

3. Research Model

We propose hypotheses with regard to how communication patterns may affect the success of open source projects. We define constructs that capture the communication pattern of an open source project: 1) project centrality, which measures the inequality of the developers' contributions in the project, and 2) project density, which measures the closeness of a network and its readiness to respond to changes, and 3) leadership centrality, which measures the influence of the project leader(s). We use the level of development activity and project popularity to measure the degree of success from the supply and demand side respectively.

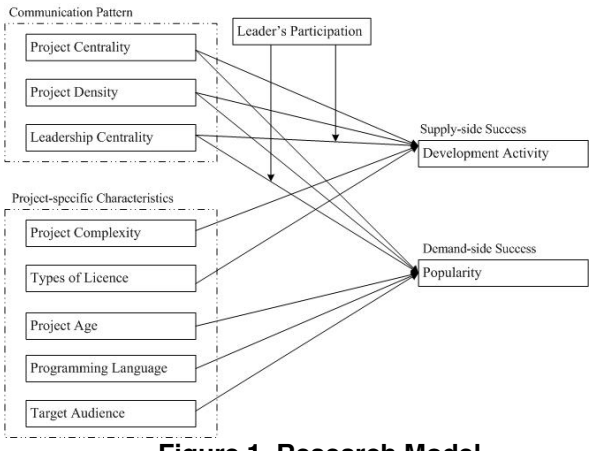

Figure 1. Research Model

[1] Concurrent Versions System (CVS) is a program that lets a code developer save and retrieve different development versions of source code. It also lets a team of developers share control of different versions of files in a common repository of files. This kind of program is sometimes known as a version control system.

3.1. Communication Pattern & Project Success

Project centrality measures the difference or inequality of contributions among developers, i.e., it examines whether there is an outstanding group of contributors in the project. Past research in social networks has shown that centrality is an important measure of group performance [11]. The investigation of project centrality can shed light on whether the inequality of the developers' contributions affects the success of the project. When the project centrality is high, the power of individual developers varies rather substantially. Social network theory suggests that networks with high centrality have the advantage of speedy and flexible information diffusion within the network [7]. In a network with high centrality score, there are certain developers who have access to more resources of the network than others in the network. These "core" developers are responsible for exchanging information and allocating resources among the team members. In most cases, these core developers are the most capable developers in the team. They filter out less meaningful messages while distributing the most useful information and allocate resources to its best usage. This increases the efficiency and quality of the communication among the developers and enhances the team members' access to resources and information. Therefore, the development process, which involves collaborative tasks such as debugging, document writing, upgrading, patching, and consulting, can be better handled with better resource allocation and higher quality delivery. In the strategic management literature, good coordination within development teams (which helps to attract users, resources, and collaborators) is considered as a key determinant of software development success [19]. Since higher project centrality is associated with better organization and more efficient information exchange, we propose that a more centralized project is likely to achieve higher development activities:

Hypothesis 1: Project centrality will positively affect the level of project development activities.

In the strategic management literature, well-balanced development teams, who are able to attract users, resources and collaborators, are considered as a key determinant of the software project development success. In the high centralization project, the power of individual developers varies rather substantially. The higher the project centralization measure, it is more likely that there are more linkages between the core developers and the other contributors. Thus, in the high centralization projects, the communications are largely dependent on the core developers. Moreover, as mentioned previously, core developers take charge of exchanging most information or resource among the project team members. In this way, the core developers can disseminate useful information and helpful resources, while discarding the less meaningful or useful messages. This thus increases the efficiency and quality of the communication among the developers, in turn enhancing the quality of software development, maintenance and support. Open source users, especially new users, usually put more emphasis on the continual support and maintenance provided by the project teams. Consistent, periodic maintenance and support of open source projects by developers typically generate positive word of mouth which enhances the reputation of such projects. Given the increased likelihood of positive word of mouth and reputation in the OSS community, it is thus likely that the demand popularity of such projects increase over time. Therefore, we propose that highly centralized OSS projects will be more likely to attract end users, i.e., more likely to increase the popularity of such projects:

Hypothesis 2: Project centrality will positively affect the level of project popularity.

In a network of software developers, a higher density indicates a greater degree of interaction among the members and thus closer collaboration among members. However, a higher project density makes the dissemination of knowledge more time-consuming, as information and knowledge needs to travel through the extended hierarchies of the project team. An example of a three-developer project illustrates this problem: suppose developer A has communications with B, B with C, and C with A, then developer C may obtain the same information from both A and B. In projects with many developers, there may be more occurrences of repeated information exchange. The efficiency of the communication is therefore substantially reduced in projects with high density. Previous literature indicates that effective communications among the team members is a key factor to project success [37]. For open source development teams, to achieve the objectives and to successfully complete their development tasks, information must be effectively exchanged [30]. Thus, we propose that the density of a project will negatively affect the level of development activities of a project:

Hypothesis 3: Project density will negatively affect the level of project development activities.

As mentioned in the above, a higher project density makes the dissemination of knowledge more time-consuming, as information and knowledge needs to travel through the extended hierarchies of the project team. In projects with many developers, there may be more occurrences of repeated information exchange. The efficiency and quality of communication is therefore substantially reduced in projects with high

density. Consequently, from the perspective of OSS end users, software technical support and maintenance may be negatively impacted such that the frequency of technical updates or bug fixes is delayed, or that the outcome quality of maintenance and support is less than optimal. These negative consequences of high network density in project teams in turn would have detrimental impacts on the reputation and eventual popularity of projects among end users. Thus, we propose that the density of a project will negatively affect the level of project popularity:

Hypothesis 4: Project density will negatively affect the level of project popularity.

A project manager plays the key role of coordinating overall project development activity. As [24] pointed out, the project manager should carry out some critical tasks such as attracting new programmers, ensuring an efficient division of the project into modules, allowing contributors to perform their tasks independently from the rest of the contributors, and managing conflicting views and approaches among participants. In a project with higher leadership centrality, the manager has higher stature and more influence on the project team. In high manager centrality projects, the managers can attract and communicate with many developers, and those developers may actively exchange information with their managers because of the manager's stature and influence. Under such communication structures, developers derive the majority of necessary information and resources from the managers, and can perform their tasks independently from the other developers. The project manager thus serves as a pivotal conduit for information communication among the project team. Therefore, we propose that project leadership centrality will positively influence the level of development activity of the project:

Hypothesis 5: Project leadership centrality will positively affect the level of project development activities.

However, conventional wisdom suggests that heavy reliance on a single developer-manager threatens the survival of the open source projects. This is because a developer-manager usually simultaneously participates in more than one open source projects or is affiliated to other commercial projects. Once he (or she) loses interest in the current focal project or becomes engaged in other projects, the focal project will lose its central emphasis or focus, leading to delayed responses to end users or other developers' requests. This situation would largely decrease the information quality and level of team coordination, which in turn reduces the efficiency and productivity of the whole development team. Thus, there are the potential risks associated with a project with high leadership centrality. Therefore, we propose that a leader's heightened participation in other projects will yield a negative effect on project development activities for project teams with high leadership centrality:

Hypothesis 6: Leader's heightened participations in other projects will yield a negative effect on project development activities for projects with high leadership centrality.

In a project with higher leadership centrality, the manager has higher stature and more influence in the project team. Therefore, these managers are able to attract high capability developers and communicate with many developers effectively and efficiently, leading to developers actively exchanging information with their managers. Consequently, with the project manager serving as a pivotal conduit for effective information communication among the project team, it is more likely that outcome of OSS development turns out to be positive in terms of software release quality and maintenance support frequencies. These outcomes would in turn positively impact on the demand popularity of the project among OSS end users. Therefore, we propose that project leadership centrality will positively influence the level of project popularity:

Hypothesis 7: Project leadership centrality will positively affect the level of project popularity.

Since the open source developers (including managers) usually simultaneously participate in more than one projects or are affiliated to other commercial projects, there are effort and time constraints on part of the individual developers's fixed amount of resources and effort. Thus, as many anecdotal evidence have indicated, heavily reliance on a single developer-manager threatens the survival of the open source projects. This may largely deter the users who want to seek continual consistent support and maintenance from OSS project teams. Therefore, we propose that a leader's heightened participations in other projects will yield a negative effect on project popularity for projects with high leadership centrality:

Hypothesis 8: Leader's heightened participations in other projects will yield a negative effect on project popularity for projects with high leadership centrality.

3.2. Project Characteristics and Success

Apart from communication pattern, project-specific factors may also influence the success of OSS projects. Project characteristics considered in this study include type of licenses, project complexity, programming language, project age, and target audience.

[25] suggested that the restrictiveness of license protects the developers from being exploited by the commercial software firms by limiting the privatization of their intellectual products. Commercialization of

open source projects is undesirable because it can reduce the visibility of the developer's contribution and reputation. Therefore, commercialization may drive away developers. Thus, we propose the restrictiveness of licenses may play a positive role on the level of OSS development activities. In addition, more complex projects with more features usually require more development activities such as feature development and patching. Thus, we expect that a more complex project is likely to achieve a higher level of development activities.

Hypothesis 9A: A project with a restrictive license will be likely to achieve a higher level of development activities.

Hypothesis 9B: A more complex project will be likely to achieve a higher level of development activities.

From the end users' perspective, software written in more popular programming languages may be more popular among end users, since more people can modify (or customize) the open source software. In addition, the popularity of an OSS project is likely to increase with a longer history of existence, because the longer the project has been developed and distributed in the open source community, the more users can find and adopt the OSS software. Finally, some open source projects are specifically developed for particular user groups. Apparently, software targeted at the general end users may appeal to more users, and thus leading to increased project popularity.

Hypothesis 10A: A project written in a popular programming language will be likely to achieve a higher level of popularity.

Hypothesis 10B: A project with a longer time history will be likely to achieve a higher level of popularity.

Hypothesis 10C: A project targeting at end users will be likely to achieve a higher level of popularity.

4. Research Method

The objective of this study is to investigate the longitudinal impacts of communication patterns of open source teams on project success. We collect data of various OSS projects over an extended period of 13 months and utilize cross-sectional time-series panel data analysis methods [13].

4.1. Project Selection

As with most empirical studies on open source projects, the data is collected from SourceForge.net, which is the world's largest online repository of open source applications. At the start of our data collection (in November 2006), SourceForge.net hosted 133,029 open source projects on a wide variety of areas, and had 1,425,354 registered developers. SourceForge.net also provides useful tools to control and manage open source development. It offers a variety of services including hosting, mailing lists, bug tracking, patch tracking, support request tracking, feature request tracking, message boards, file archiving, and other project management tools. SourceForge.net provides a large sampling population of open source projects with extensive details, and is the best site to collect data on OSS projects' development activities and attributes.

In order to investigate the communication patterns of the project teams, we observe and analyze the developers' interactions through bug, patch, support request, and feature request (BPSF) tracking systems hosted on SourceForge.net. These tracking systems enable users and developers to report and discuss BPSF. Each report includes basic information about the BPSF as well as their correspondences that deal with bug fixing, patches updating, support and feature requests responding in time sequence.

We elaborate the procedures for collecting data on the pattern of communication. First, on project's summary page, the links of bugs, patches, feature requests or support requests systems are listed. Second, by clicking the link (e.g. bugs), we see the list of bugs. Third, checking the details of each bug, we can find who submitted the bug and who responded to this bug. We define that the submitter and respondents have the interaction, which will be recorded in the sociomatrix. Each sociomatrix contains interaction information of each project per month. A sociomatrix is a standard data representation for a network analysis [39]. The sociomatrix has a row and a column for each individual, and the cells of the matrix count the number of interactions from one individual to another. It can be constructed by using the popular social network analysis tool of Ucinet 6.0.

Not all projects on SourceForge.net are suitable for our study. Three criteria are adopted to select useful projects: projects are selected from top 7000 ranked projects; projects have at least three developers; and there are enough interactions to ensure that each sociomatrix is equal to or larger than 3×3 matrix. We choose projects with at least three developers because we are interested in team communications, instead of individual basis or dyadic interactions, which are not suitable for analyzing communication patterns. Although many projects have more than three developers, yet some projects do not make use of the SourceForge's BPSF tracking system, which leaves us impossible to track the communications; some projects do not have enough bugs, or patches, or support requests, or feature requests to establish a non-null socio-matrix. We select only the top-ranked 7000

projects because in top 1000 projects, ninety projects are useful; from top 1000 to top 2000, twenty-one projects are useful; from top 2000 to top 4000, ten projects are useful; from top 4000 to top 7000, only five projects are useful. Thus, we believe that there are few useful projects after the top-ranked 7000 projects.

In top ranked 7000 projects, 6815 projects were examined because the remaining 185 projects were not ranked by the website. There were 3069 projects with only one developer; there were 948 projects with two developers; 378 projects did not have (bug, patch, support or feature request) tracking systems on SourceForge.net; 2294 projects did not have enough interactions. By excluding the unsuitable projects, the final sample was 126 projects. (The number of developers of these 126 projects ranges from 3 to 149.) In order to analyze the dynamic impacts of communication patterns on project success, we monitor each project over an extended period. Starting from 01 November 2006, we captured communication records (BPSF reports) of each project every month until 30 November 2007. We thus obtained data on 126 open source projects in 13 months. The total number of observations in our data set is thus 1638.

4.2. Project-Level Measures

Projects success is measured from both the supply side and demand side. For the supply side, development activity is calculated as the average number of total tracks and file releases. Number of total tracks includes sum of bugs, patches, support and feature requests. For the demand side, popularity is measured by the formula: "number of web hits * (1+ subscription)", in which "subscription" is measured by level of accepted donations of each project, and "hits" are measured by the number of web traffic visitations.

Project centrality can be measured by project degree centrality and betweenness centrality. Project degree centrality describes the inequality or variance of developers' contribution in the network [10]. A high score on project degree centrality implies that the power of individual developers varies rather substantially, and this means that, overall, positional advantages are rather unequally distributed in this project. Project betweenness centrality measures the inequality of developers' ability of information control. It can also be interpreted as measuring the degree of independence from others in the project. Project density measures the readiness of the group to respond to changes. It is defined as the percentage of ties that exist in a network out of all possible ties. A density of 1 implies that every actor is connected to every other actor. A density of 0 implies that no actor knows any other actor. Leadership centrality indicates the stature and influence of the project leader, which can be measured by centrality of the project administrator, or average centrality of the project administrators if there is more than one administrator. In order to moderate the effect of the leadership centrality, we include the moderating variable of the extent of leader's participations in other projects. It is defined as the average number of projects that project administrators participate simultaneously. We import interaction data from SourceForge's BPSF tracking systems to create a sociomatrix for each project per month. By analyzing each socio-matrix through the popular social network software Ucinet 6.0, we can obtain the data of project centrality, density and leadership centrality.

There are several project-specific factors that may also influence the success of the project. We measure the types of licenses according to license categories classified by [24]. A value of 1 indicates a restrictive license; 0 indicates a nonrestrictive license. Project complexity is measured by the number of software packages. Project age is defined as the months between the data collection date and the project registration date. In addition, a dummy variable is included to control for whether the project uses C/C++, which is one of the largest and most popular programming language categories on SourceForge.net. Finally, a dummy variable is employed to indicate whether the project is targeted at general end users.

5. Data Analysis and Results

The average project centrality of our sample projects is 49.43% and ranges from 0 to 100%. The average density is 22.63% and ranges from 0 to 100%. The average leadership centrality is 50.15% and ranges from 0 to 100%. The descriptive statistics of all variables is shown in Table 1.

	Mean	*Std. Dev.*	*Min*	*Max*
Activity	26.627	34.408	0	358
Log-Activity	1.145	0.532	-0.301	2.554
Popularity	16.78×10^6	117×10^6	0	1480×10^6
Log-Popularity	4.97	1.66	0	9.17
Project Centrality	49.434	0.307	0	100
Log-Project Centrality	1.47	0.629	0	2
Project Density	22.628	0.18	0	100
Log-Project Density	1.16	0.51	0	2
Leadership Centrality	50.149	0.331	0	100
Log-Leadership Centrality	1.419	0.704	0	2.301
Participation	3.116	2.481	1	15
Complexity	4.09	3.7	1	22
Restrictive Licence	0.746	0.436	0	1
Project Age	55.132	25.482	-9.1	95.467
Target Audience: End user	0.611	0.488	0	1
Language: C/C++	0.61	0.49	0	1

Table 1. Descriptive Statistics of Variables

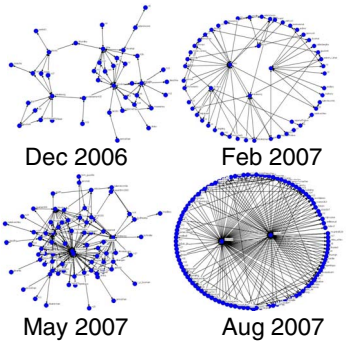

Figure 2. Communication Structure of Project "YUI_Library"

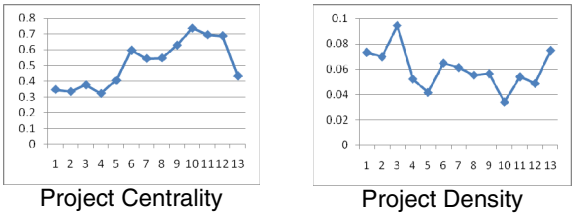

Figure 3. Centrality and Density Measures of Project "YUI_Library"

We measure the success of OSS projects from both the supply and demand side. It is widely acknowledged that the supply and demand of goods are endogenous and would influence each other simultaneously over time. To account for the endogeneity issue, we estimate the parameters of our research model by using the simultaneous equation modeling and estimation approach. We consider three model estimation methods: (1) Three-Stage Least Squares (3SLS), (2) Weighted Two-Stage Least Squares (WTSLS), and (3) Generalized Method of Moments (GMM). Such estimation techniques account for the endogeneity of the demand and supply factors and thus minimize the bias in the parameters of the proposed model.

The proposed empirical model of this study is:

$$Activity = \alpha_1 Popularity + \alpha_2 Project\_Centrality + \alpha_3 Project\_Density + \alpha_4 Leader\_Centrality + \alpha_5 Participation + \alpha_6 Leader\_Centrality*Participation + \alpha_7 Complexity + \alpha_8 Licence \quad (1)$$

$$Popularity = \beta_1 Activity + \beta_2 Project\_Centrality + \beta_3 Project\_Density + \beta_4 Leader\_Centrality + \beta_5 Participation + \beta_6 Leader\_Centrality*Participation + \beta_7 Age + \beta_8 Language + \beta_9 TargetAudience \quad (2)$$

By studying the histograms of dependent variables (development activity and popularity), centrality and density measures, it appeared that these variables were skewed to the left and not normally distributed. Accordingly, we use the logarithmic transformations of these variables in our empirical estimations.

To account for the potential issue of endogeneity between the dependent variables and independent variables, we used lagged independent variables. To obtain robust reliable results, we compare estimation results of different lags of the independent variables (lag=2, lag=1, lag=0, lag=-1 and lag=-2). As expected, we found that using a 1-month lag (i.e., lag=-1) in the independent variables gives the best set of model estimation results in terms of model fit, coefficient signs and statistical significances. Table 2 documents the model estimation results using 2 different estimation methods using a 1-month lag structure.

	3SLS		WTSLS	
	Coefficient	*Std Error*	*Coefficient*	*Std. Error*
Log-Popularity	0.033*	0.018	0.030*	0.018
Log-Project Centrality	0.500***	0.039	0.503***	0.039
Log-Project Density	-0.398***	0.044	-0.403***	0.044
Log-Leadership Centrality	0.464***	0.066	0.466***	0.066
Participation	0.163***	0.026	0.163***	0.026
Log-Leadership Centrality*Participation	-0.137***	0.021	-0.137***	0.021
Complexity	0.030***	0.004	0.030***	0.004
Restrictive License	0.033	0.031	0.055*	0.032
	R^2 = 0.021		R^2 = 0.021	
Log-Activity	0.39	0.445	0.343	0.445
Log-Project Centrality	0.688**	0.276	0.712**	0.276
Log-Project Density	-0.603**	0.256	-0.637**	0.256
Log-Leadership Centrality	1.914***	0.353	2.019***	0.353
Participation	0.777***	0.128	0.820***	0.128
Log-Leadership Centrality*Participation	-0.641***	0.105	-0.676***	0.105
Project Age	0.029***	0.002	0.029***	0.002
Language: C/C++	0.119	0.135	-0.057	0.138
Target Audience: End user	0.338**	0.115	0.425**	0.118
	R^2 = -0.305		R^2 = -0.362	

Note: significance: 10% *; 5% **; 0.01% ***

Table 2. Regression Results (3SLS, WTSLS)

Based on the estimation results, we find significant effects of communication patterns on our project success measures. H1 proposes that project centrality will positively affect the level of project development activities. When project centrality is measured by degree centrality, the model shows significant positive effect of degree centrality on development activity. When project centrality is measured by betweenness centrality, the model also shows significant positive effect of project betweenness centrality on development activity. Thus, H1 is supported.

H2 proposes that project centrality will positively affect the level of project popularity. When project centrality is measured by degree centrality, the model shows significant positive effect of degree centrality on project popularity. When project centrality is measured by betweenness centrality, the model also shows significant positive effect of betweenness centrality on project popularity. Hence, H2 is also supported.

The hypotheses test results from H1 and H2 suggest that the centrality of OSS project teams play an important role in the evolving success of the projects. Centralized projects are good for increasing development activity and attracting end users. The managers of OSS projects need to control the communication centrality within OSS projects.

H3 proposes that project density will negatively affect the level of project development activities. The model shows significant negative effect of project density on development activity. H3 is supported.

H4 suggests that project density will negatively affect the level of project popularity. The model shows significant negative effect of project density on project popularity. Thus, H4 is supported.

H5 proposes that project leadership centrality will positively affect the level of project development activities. H6 proposes that leader's participations in other projects will yield a negative effect on project development activities for the projects with higher leadership centrality. Both H5 and H6 are supported.

H7 proposes that project leadership centrality will positively affect the level of project popularity. H8 proposes that leader's participations in other projects will yield a negative effect on project popularity for the projects with higher leadership centrality. The estimation results also support both H7 and H8.

The hypotheses test results from H5 to H8 suggest that leadership centrality is important for the OSS projects. Leaders in the project play a crucial role in improving the performance of the project teams. However, leaders may participate in several projects simultaneously, which will disperse the effort of leaders in a single project.

Hypothesis 9 (A, B) proposes the relationships between project-specific characteristics and project development activities. The results from the model estimation show the significant effect of project complexity on the level of development activities. A restrictive type of license has a significant positive effect on the level of development activities only when adopting WTSLS estimation method.

Hypothesis 10 (A, B, C) proposes the relationships between project-specific characteristics and project popularity. The results from the model estimation show significant effects of project age and end-user target audience, but an insignificant effect of popular C/C++ programming language, on project popularity.

6. Summary and Conclusion

The main purpose of this study is to investigate the long term effects of communication pattern on the success of OSS projects. Results, shown in Table 2 are generally supportive of the hypotheses posited in this paper. By observing changes in communication patterns over time, we find significant impacts of communication patterns on the outcome of the project.

The findings of our research has implications for project managers and developers in open source environments, as well as for managers of commercial software firms, such as Microsoft and IBM, which are actively participating in open source projects. The significant effects of project centrality on project development activity and popularity are examined and uncovered by our research model. In a centralized network, some core members contribute more to the project. Such core-periphery structures can potentially enhance the speed and flexibility with which information diffuses within a group [7]. Thus, centralized projects will be with higher communication efficiency and thereby related to better performance.

Project density measures the readiness of the group to respond to changes, and how close a network is to realize its potential. In high-density projects, information dissemination is impeded, which negatively affects communication efficiency. Thus, balancing project centrality and density is an important task for project managers to ensure the success of OSS projects in a competitive environment.

Leadership centrality measures the influence of managers on the projects. In high manager centrality projects, managers attract and communicate with many developers, and those developers may actively exchange information with their managers because of manager's high stature and influence. However, heavy reliance on a single developer-manager threatens the survival of the open source projects. There is a negative impact of leader's simultaneous participations in other projects on development activity and popularity of high leadership centrality projects.

From a theoretical standpoint, we apply social network theory into the information systems domain, in particular, into the study of success of OSS projects. Although previous works have been conducted on various IS phenomenon from a social network perspective, we are among the first to apply the social network theory with encouraging success to the OSS development realm. We thus motivate other IS researchers to pursue this area of research.

7. References

[1] Colazo, J. A., Fang, Y., Neufeld, D. "Development success in open source software projects: exploring the impact of copylefted licenses," in Proceedings of the 11th Americas Conference on Information Systems, USA, 2005.

[2] Comino, S., Manenti, F. M. and Parisi, M. L. "From planning to mature: on the determinants of open source take off," Working Papers 0517, Department of Economics, University of Trento, Italia, 2005.

[3] Crowston, K. "A coordination theory approach to organizational process design," Organization Science (8:2), 1997, pp. 157–175.
[4] Crowston, K., Annabi, H. and Howison, J. "Defining open source software project success," in Proceedings of the 11th International Conference on Information Systems, USA, 2003.
[5] Crowston, K., Annabi, H., Howison, J., and Masano, C. "Towards a portfolio of FLOSS project success measures," in Proceedings of the Open Source Workshop of the International Conference on Software Engineering, May 2004.
[6] Crowston, K. and Howison, J. "The social structure of free and open source software development," First Monday (10:2), 2005.
[7] Cummings, J. and Cross, R. "Structural properties of work groups and their consequences for performance," Social Networks (25:3), 2003, pp. 197-210.
[8] Feller, J. and Fitzgerald, B. "A framework for understanding the open source software development paradigm," in Proceedings of the 8th International Conference on Information Systems, USA, 2000.
[9] Fershtman, C. and Gandal, N. "The determinants of output per contributor in open source projects: an empirical examination," CEPR Paper, No. 4329, March 2004.
[10] Freeman, L. C. "Centrality in social networks: conceptual clarification," Social Networks (1:3), 1979, pp. 215-239.
[11] Freeman, L. C., Roeder D., and Mulholland, R. R. "Centrality in social networks II: experimental results," Social Networks (2:2), 1980, pp. 119-141.
[12] Granovetter, M. "The strength of Weak Ties," American Journal of Sociology (78:6), 1973, pp. 1360-1380.
[13] Greene, H. W. Econometric Analysis, Prentice Hall, Upper Saddle River, New Jersey, United State, 2003.
[14] Grewal, R., Lilien, G. L., and Mallapragada, G. "Location, location, location: how network embeddedness affects project success in open source systems," Management Science (52:7), 2006, pp. 1043-1056.
[15] Hanneman, R. A., and M. Riddle. Introduction to social network methods, University of California, Riverside, United State, 2005 (published in digital form at http://faculty.ucr.edu/~hanneman/).
[16] Hertel, G., Niedner, S., and Herrmann, S. "Motivation of software developers in open source projects: and internet-based survey of contributors to the Linux kernel," Research Policy (32:7), 2003, pp. 1159-1177.
[17] Hippel, E. V., and Krogh, G. V. "Open source software and the "private-collective" innovation model: special issues for organization science," Organization Science (14:2), 2003, pp. 209-225.
[18] Johansson, C, Dittrich, Y. and Juustila, A. "Software engineering across boundaries: student project in distributed collaboration," IEEE Transactions on Professional Communication (42:4), 1999, pp. 286-296.
[19] Kidane, Y. H., and Gloor, P. A. "Correlating temporal communication patterns of the Eclipse open source community with performance and creativity," Computational & Mathematical Organization Theory (13:1), 2007, pp. 17-27.
[20] Krogh, G. V., Spaeth, S., and Lakhani, K. R. "Community, joining, and specialization in open source software innovation: a case study," Research Policy (32:7), 2003, pp. 1217–1241.
[21] Lakhani, K. R. and von Hippel, E. "How open source software works: "free" user-to-user assistance," Research Policy (32:6), 2003, pp. 923–943.
[22] Lakhani, K. R. and Wolf, R. G. "Why hackers do what they do: Understanding motivation effort in free/open source software projects," Working Paper 4425-03, MIT Sloan School of Management, Cambridge, MA, 2003.
[23] Leavitt, II. J. "Some effects of communication patterns on group performance," Journal of Abnormal and Social Psychology, 1951, (46), pp. 38-50.
[24] Lerner, J. and Tirole, J. "Some simple economics of open source," Journal of Industrial Economics (50:2), 2002, pp. 197-234.
[25] Lerner, J. and Tirole, J. "The scope of open source licensing," Journal of Law, Economics and Organization (21:1), 2005, pp. 20-56.
[26] Madey, G., Freeh, V., and Tynan, R. 2002. "The open source software development phenomenon: an analysis based on social network theory," in Proceedings of the 8th Americas Conference on Information Systems, USA, 2002.
[27] Markus, M. L., Manville, B., and Agres, C. E. "What makes a virtual organization work?" Sloan Management Review (42:1), 2000, pp. 13–26.
[28] Maznevski, M. and Chudoba, K. "Bridging space over time: global virtual team dynamics and effectiveness," Organization Science (11:5), 2001, pp. 473-492.
[29] Mullen B., Johnson, C. and Salas, E. "Effects of communication network structure: Components of positional centrality," Social Networks (13:2), 1991, pp. 169-186.
[30] Powell, A., Piccoli, G. and Ives, B. "Virtual teams: a review of current literature and direction for future research," The Data base for Advances in Information Systems (35:1), 2004, pp. 6-36.
[31] Raymond, E. S. "The cathedral and the bazaar," First Monday (3:3), 1998.
[32] Sawyer, S. "Software development teams," Communications of ACM (47:12), 2004, pp. 95-99.
[33] Scacchi, W. "Understanding the requirements for developing open source software systems," IEE Proceedings Software (149:1), 2002, pp. 24–39.
[34] Scott John. (eds.). Social Networks: Critical Concepts in Sociology, London, New York, Routledge, 2002.
[35] Sen. "Open source software development projects: determinants of project popularity," Working paper 0510003, Econometrics from EconWPA, 2005.
[36] Stewart, K. J., Ammeter, A. P., and Maruping, L. M. "Impacts of license choice and organizational sponsorship on user interest and development activity in open source software projects," Information Systems Research (17:2), 2006, pp. 126-144.
[37] Suchan, J. and Hayzak, G. "The communication characteristics of virtual teams: a case study," IEEE Transactions on Professional Communication (44:3), 2001, pp. 174-186.
[38] Thomas and Hunt. "Open Source Ecosystems," IEEE Software (21:4), 2004, pp. 89-91.
[39] Wasserman, S. and Faust, K. (eds.). Social Network Analysis, Cambridge University Press, Cambridge, 1994.
[40] Weber S. (eds.). The Success of Open Source, Harvard University Press, Cambridge, 2004.
[41] Wellman, B., and Berkowitz, S. D. (eds.). Social Structures: A Network Approach, Cambridge University Press, Cambridge, 1998.
[42] Xu, J., Gao, Y., Christley, S., and Madey, G. "A topological analysis of the open source software development community," in Proceedings of 38th Hawaii International Conference on System Sciences, USA, 2005.