Network Dynamics and Knowledge Transfer

Network Dynamics and Knowledge Transfer in Virtual Organizations[1]

Neil Gandal
Berglas School of Economics
Tel Aviv University
Ramat Aviv, Tel Aviv 69978, Israel
Email: gandal@post.tau.ac.il

Uriel Stettner
Recanati Business School
Tel Aviv University
Ramat Aviv, Tel Aviv 69978, Israel
Email: urielste@tau.ac.il

January 21, 2015

[1] We appreciate the invaluable research assistance of Yaniv Friedensohn and gratefully acknowledge the financial support of the Israel Science Foundation (grant No. 1287/12) and the support of a grant by the Research Program for the Economics of Knowledge Contribution and Distribution. We are grateful to seminar participants at UT-Austin and the Economics of Knowledge Contribution and Distribution conference at Georgia Institute of Technology for helpful comments. Any opinions expressed are those of the authors.

Network Dynamics and Knowledge Transfer

Abstract:

In this paper, we examine whether particular network structures foster knowledge transfer among distinct open-source projects. Deploying panel data, we compare organizations that were in a giant component of a network for relatively long periods of time with organizations that joined the giant component during the sample period. Our findings show that joining a large pool of knowledge (i.e., a giant component) allows projects to gain access to novel knowledge and ideas. Moreover, established projects within the giant component benefit differently from changes in network structures than projects that only recently entered such giant component of a network.

Keywords: Network Dynamics, Knowledge Spillovers, Social Network, Open Source

Network Dynamics and Knowledge Transfer

1. Introduction

Product development in community-based organizational settings is becoming an increasingly important mechanism through which individual knowledge holders create and disseminate knowledge in joint efforts to generate products. OSS generally implies that a particular computer software source code is available to the broad public under an OSS license (Laurent, 2004.). Such licenses grant the rights to use an entire work, to create a derivative work, or to share or market such work subject to the license governing the specific open-source project (Bonaccorsi, Rossi, & Giannangeli, 2006; Von Hippel & Von Krogh, 2003; Lerner & Tirole, 2002). Accordingly, one of the central aspects of OSS development is the project's ability to share and absorb knowledge that has been created within or outside of a distinct OSS project. Such spillovers facilitate the transfer of knowledge and ideas within and across researchers and development teams. External knowledge may provide a particular project with highly specialized competencies and technical flexibility through the formation of informal "learning alliances" that may provide accelerated learning processes, a contraction of the product development life cycle, and ultimately a sustainable competitive advantage.

In its traditional form, open-source software (OSS) development is a collaborative effort of loosely coordinated and geographically dispersed developers who contribute their time and knowledge to establishing and improving software and whose underlying knowledge is made accessible to the general population. OSS projects, like virtual teams, are semi-structured groups of skilled developers working on interdependent tasks using informal, non-hierarchical, and decentralized communication with the common goal of creating a valuable product (Lipnack & Stamps, 1997). In contrast to traditional work teams, which enjoy the benefits of face-to-face communication, OSS projects face exceptional challenges in forming personal relationships

(Beyerlein, Johnson, & Beyerlein, 2001), team communication (Pinto & Pinto, 1990) and, ultimately, performance (Jehn & Shah, 1997). Accordingly, due to the nature of its organizational design and structure, members of dispersed development teams are restricted in their exposure to knowledge and know-how.

Clearly, the evolving social structure that underlies distinct OSS development efforts is a critical point of distinction from traditional proprietary, closed-innovation development mechanisms. The open-source structure emphasizes the significance of social capital in defining organizational traits such as the accessibility of diverse knowledge, the aptitude to recruit qualified human capital (Lacetera, Cockburn, & Henderson, 2004), and/or the capacity to increase product visibility and increase adoption rates (Burt, 1992; Granovetter, 1985, 2005; Uzzi & Gillespie, 2002). This architecture of network ties offers a glimpse into the extent to which an entity (i) is rooted in a network, (ii) connects with other entities, and (iii) connects with other structurally embedded entities. Accordingly, an entity that is characterized by higher levels of embeddedness is expected to possess higher levels of social capital, which should, in turn, exert a positive impact on both the technical and commercial successes of the open-source project with which the entity is associated (Grewal, Lilien, & Mallapragada, 2006).

We contribute to research on organizational learning by studying how changes in network structures can foster knowledge transfers in the case of OSS projects hosted at sourceforge.net. Sourceforge.net facilitates software developer collaboration by providing a free online platform for managing projects, communications, and software code. It is, by far, the largest repository of registered OSS development projects during the period of our study. In addition to providing information about the project (date established, project stage, etc.), each Sourceforge.net project contains a list of registered team members who contribute their time and knowledge to the advancement of one or more OSS projects. Following Grewal, Lilien, & Mallapragada (2006) and Fershtman and Gandal (2011), we construct the project network by defining two projects to be connected if they have a developer in common. We can also construct the related contributor network by defining two contributors as connected if they work together on the same project.

Several other recent studies have examined the relationship between network structure and performance (Ahuja, 2000; Calvó-Armengol, Patacchini, & Zenou, 2009).2 Our paper is closest to that of Fershtman and Gandal (2011), who focus on spillovers that occur by means of the interactions of different researchers or developers in OSS projects. The theoretical model developed by Fershtman and Gandal (2011) shows that project spillovers imply (i) positive association between degree and project success and (ii) positive associations between closeness and project success. Using cross-sectional data, they demonstrate that the structure of the product network is associated with the project's success, which provides support for knowledge spillovers.

However, none of these papers discussed above focuses on the relationship between changes in the network architecture and changes in success over time, which is a key focus of our paper. Further, as network structures evolve over time, projects connect to (and disconnect from) one another. Typically, mature network structures reflect one giant and many small components. The dynamics of network formation will generate the following two different sets of organizations in a giant component: (i) organizations that were in the giant component

---

[2] Some recent studies have examined the relationship between network structure and behavior (e.g., Ballester, Calvó-Armengol, & Zenou, 2006; Calvo-Armengol & Jackson, 2004; Jackson & Yariv, 2007; Karlan, Mobius, Rosenblat, & Szeidl, 2009). Goyal, van der Leij and Moraga-Gonzalez (2006) constructed a co-authorship network using data on published papers that were included in EconLit between 1970 and 2000 to study network properties over time.

throughout a specified period of time and (ii) organizations that joined the giant component sometime during such specified period of time.

What differences are there between these two sets of organizations? Because we have yearly panel data from 2006-2009, the second key focus of the paper is to compare organizations that were in the giant component of a network for relatively long periods of time with organizations that joined such giant components during the sample period. To the best of our knowledge, this aspect of network formation has not been explored in the literature.

We find that in general, changes in the network architecture are positively associated with changes in project success. However, we also find that established projects within the giant component benefit differently from changes in network structures than do projects that only recently entered such a giant component of a network. In particular, we find the following:

1. For both sets of organizations, there are positive associations between the change in degree and the change in an organization's performance. Yet, these associations are stronger for organizations moving into the giant component than for organizations that have always been in the giant component.
2. For both sets of organizations, there are positive associations between the change in closeness and the change in an organization's performance. Further, these associations are stronger for organizations moving into the giant component than for organizations that have always been in the giant component.
3. The addition of 'Stars' (developers who work on five or more projects) is positively associated with changes in project success for projects already in the giant component in January 2006, but this association does not exist for projects that joined the giant component during the 2007-2009 period.

The first two results suggest that joining a large pool of knowledge (i.e., the giant component) allows projects to gain access to high-impact, novel knowledge and ideas. We believe that the third result can be explained as follows: the more prolonged the exposure of projects to external projects and developers, the greater is the positive impact on project success from the addition of a Star.

2. Research Setting and Data

This paper uses a replica of publicly available data from Sourceforge.net that is hosted at Notre Dame University. Sourceforge.net facilitates software developer collaboration by providing a free online platform for managing projects, communications, and software code. Sourceforge.net is the largest repository of registered OSS development projects during the period of our study.

Each SourceForge.net project contains a list of registered team members who contribute their time and knowledge to the advancement of an OSS project. Each project links to a "developer page" that contains meta-information on a particular contributor, including the date the developer joined the project, the developer's functional description (e.g., administrator, developer) and his or her geographic location. These projects are managed by project administrators. Because accessibility to OSS projects is unrestricted and because the contributors can be identified by their unique user names, we utilize this information to construct a two-mode network that relates projects via registered contributors. Accordingly, we define two OSS projects as being connected when there are common contributors who participate in both projects.[3]

[3] We assume that project members are added to the list because they make a contribution to the project that involves an investment of time and effort. A project is thus understood as a collaborative effort by its contributors.

Network Dynamics and Knowledge Transfer

Each project links to a standardized “Project page” that lists descriptive information on a particular project, including a statement of purpose, the intended audience, the license, and the operating system for which the application is designed. Moreover, a “Statistics page” shows various project activity measures, including the number of project page views and downloads registered for the focal project. Although some data are available for other periods, statistics on downloads are available only for the 2006–2009 period.[4] Therefore, we deploy panel data from 2006–2009 to construct two distinct two-mode networks: (i) the project network and (ii) the contributor network. In the former, the nodes are the OSS projects, and two projects are linked when there are common contributors who work on both. In the latter, the nodes of the contributor network are the contributors, and two contributors are linked if they participated in at least one OSS project together.

Regarding the project network in 2009, we find that 84.3% percent of the projects have either one or two contributors, 9.2% have three to four contributors and 6.5% have five or more contributors (see Table 1). With regard to the contributor network in January 2009, 91.3% of the contributors worked on one or two projects, 6.5% of the contributors worked on three to four projects, and 2.1% of the contributors worked on five or more projects.[5] While we focus on the project network, our analysis also includes a key feature of the contributor network: contributors who work on five or more projects. We define such contributors as ‘Stars.’

[4] Page view data are not available over time, but page views are highly correlated with downloads.
[5] These percentages were virtually identical in 2006 as well.

Table 1: Distribution of components in project networks—January 2009

Project Network		**Contributor Network**	
Contributors per project	*Percent of total projects*	*Projects per contributor*	*Percent of Contributors*
1	69.9	1	77.2
2	14.4	2	14.1
3-4	9.2	3-4	6.5
5-9	4.8	5-9	1.9
10 or more	1.7	10 or more	0.2

Having panel data from 2006 to 2009 allows us to focus on differences over time. This approach is helpful because it is difficult to determine causality from cross-sectional data, and, therefore, unobserved fixed project effects might be driving success. Because we do not have data on these fixed project effects, they are included in the error term when running cross-sectional analyses. If these unobserved effects are correlated with the right-hand-side variables, the estimates from the cross-sectional analysis will be biased; however, this problem is eliminated when using data on differences over time.

2.1 Dependent Variable

We wish to examine whether knowledge spillovers play a significant role in the development of OSS projects and evaluate the importance of Stars. Consistent with prior research, we measure project performance by examining the number of times a project has been downloaded (Fershtman & Gandal, 2011; Grewal et al., 2006). We focus on downloads of the executable, compiled product because users will not typically download the source code. We define ΔDownloads as the difference between the total numbers of downloads in January 2009 and January 2006. We further define lΔDownloads ≡ ln(1+ΔDownloads), where “ln” means the natural logarithm, and Δ is the difference operator.

2.2 Independent Variables

Knowledge spillovers from project to project occur via individuals. In the case of OSS projects, contributors frequently port code that is embedded in one project into another project to which they contribute. Direct spillovers occur when projects have a common developer who transfers information and knowledge (primarily code) from one project to another. Project spillovers may also be indirect, i.e., when knowledge is transferred from one project to another when the two projects are not directly linked (there is no common contributor). Because we do not directly observe spillovers, we will examine the relationship between the network structure and project success to identify the relative importance of knowledge spillovers.

We define two network centrality measures: (i) a project's *degree* is defined as the number of projects with which the focal project has a direct link or common developers and (ii) a project's *closeness centrality*, which is defined as the inverse of the sum of all distances between a focal project and all other projects multiplied by the number of other projects.[6] Intuitively, closeness centrality measures how far each project is from all the other projects in a network.[7]

Accordingly, we define ***ΔDegree*** as the difference in the degree centrality of the project between January 2006 and January 2009. Similarly, we define **ΔClose** as the difference in the closeness centrality of the project between January 2006 and January 2009. Next, we define **ΔC*pp*** as the change in the number of contributors that participated in the project during the three-year period from January 2006 to January 2009. Because the number of contributors might fall or rise over time, ΔDegree, ΔClose, and ΔCpp can be either positive or negative.

---

[6] See Freeman (1979), pp. 225-226 and Wasserman & Faust (1994), pp. 184-185 for details on how closeness centrality is calculated.
[7] Closeness centrality lies in the range [0,1]. In the case of a Star network with a single project in the middle that is connected to all other projects, the closeness centrality of the project in the center is one.

In addition to project downloads and the network variables described above, we have data for a group of control variables. In Sourceforge.net, projects evolve through six stages, beginning with planning (1) and continuing to pre-alpha (2), alpha (3), beta testing (4), production (5), and finally maturity (6). We define a dummy variable, Δstage, that assumes the value one if there was stage progression (e.g., from alpha (3) to production (4)) and zero if there was no change in stage.

To control for the amount of time that the project has been in existence, we define the variable years_since as the number of years that have elapsed since the project first appeared at Sourceforge.net: lyears_since = ln(years_since).

Finally, we define a Star as a contributor who worked on five or more projects. This variable comes from the contributor network, not the project network. Clearly, having a "Star" contributor join a project gives that project more connections to other projects. An interesting question is whether adding a "Star" to the team of developers has an effect on the success of a project. To examine this effect, we include a variable, denoted as ΔStar5, which can take on positive or negative values and is defined as the change in the number of Stars on a project from 2006 to 2009.

2.3 Discussion of the Data

In our panel data set, we have 42,796 projects with complete information.[8] Complete information indicates that the projects existed in both 2006 and 2009 and that we have data for all the relevant variables discussed above. We exclude observations for Δdegree, Δcloseness, and

[8] Importantly, because we have data on the participants in every project, our networks are constructed using all projects, including projects without complete information.

ΔCpp that are (approximately) in the lowest 5% of these distributions. Specifically, we exclude 961 observations of Δdegree that are less than or equal to -4, an additional 394 observations of ΔCpp that are less than or equal to -1, and an additional 557 observations of Δcloseness that are less than -0.0037. We exclude these observations because large negative changes in Cpp, degree, and closeness might simply be explained by those particular projects being more likely to remove any inactive programmers from their projects' websites in comparison with other projects. Our results are also robust to including all 42,796 observations. We report these results in the appendix.

After excluding the 1,912 projects discussed above, we are left with 40,884 observations for the analysis. Approximately one-third of the projects in the main part of the paper (13,474) are in the giant component, and the second-largest component is small (64 projects.). This distribution (one giant component and many small components) is typical of many networks.

Particularly interesting are the 2,656 projects that were not in the giant component in 2006 but were included in the giant component in 2009. These projects comprise 20% of the giant component. Not surprisingly, these observations exhibit relatively large changes in degree, Cpp, closeness, stage and Stars. An interesting question is whether these projects have different properties than other projects in the giant component.

Descriptive statistics are shown in Table 6 in the appendix. The mean and median download changes for projects in the giant component (mean = 66,819 and median = 930) is much greater for projects in the giant component than for projects outside of the giant component (mean = 20,734 and median = 373).

When we compare the two subgroups within the giant component—namely the projects in the giant component throughout the 2006–2009 period and the projects that moved into the giant component during the 2006–2009 period—we find no difference in the mean or median changes in downloads among the groups. Projects that moved into the giant component have much higher changes in degree, closeness, and the number of Stars than projects in the giant component throughout the 2006–2009 period (see Table 6).

Correlations between changes in degree, closeness, Stars, and Cpp are all relatively low, as shown in Table 7 of the appendix. The highest correlation is between ΔCpp and Δdegree, but that correlation is only 0.53. No other correlation exceeds a magnitude of 0.34.

3. Empirical Analysis:

The relationship between the number of contributors and *downloads* is likely non-linear: additional contributors are likely associated with a larger number of downloads, but the marginal effect of each additional contributor declines as the number of contributors increases. The same is likely true for the relationship between network variables and downloads as well, which suggests that a "log/log" model is appropriate.[9] Thus, we use the following estimating equation:

$$[1] \quad l(\Delta \text{Downloads}) = \beta_0 + \beta_1 (l\Delta \text{Cpp}) + \beta_2 (l\Delta \text{Degree}) + \beta_3 (l\Delta \text{Close}) + \beta_4 (\Delta \text{Star5}) + \beta_5 (\Delta \text{Stage}) + \beta_6 (\text{lyears\_since}) + \varepsilon,$$

[9] We estimate a log/log specification. As with the case of downloads, all independent variables (except changes in the number of Stars and changes in stage) are in logarithmic form. We denote this by including an 'l' before the variable name—e.g., lΔCpp is the logarithm of the change in the number of contributors. We add a constant to lΔCloseness, lΔCpp, and lΔDegree such that the logarithm is defined.

where Δ is the difference operator and ε is a white-noise error term. [10] We estimate [1] for the following four cases:

Case I:	Projects outside of the giant component
Case II:	Projects in the giant component in January 2009
Case IIA:	Projects in the giant component throughout the 2006–2009 period
Case IIB:	Projects that moved into giant component during the 2006–2009 period

3.1 Knowledge Spillovers via Contributors

Table 2 shows that a change in the degree centrality is positively associated with a change in the number of downloads for projects outside the giant component (Case I: $\beta = 0.42$, $p<0.0001$) and projects in the giant component (Case II: $\beta = 0.34$, $p < 0.0001$.) In fact, Table 2 also shows that the effect is approximately twice as large for the projects that moved into the giant component (Case IIB: $\beta = 0.63$, $p < 0.0001$) than for the projects that were always in the giant component (Case IIA: $\beta = 0.30$, $p < 0.0001$).

Case II in Table 2 also shows that changes in closeness centrality are positively associated with changes in project performance.[11] Table 2 also shows that changes in closeness are positively and significantly associated with changes in the number of downloads for both the projects that moved into the giant component (Case IIB: $\beta = 0.89$, $p < 0.01$) and the projects that were always in the giant component (Case IIA: $\beta = 0.15$, $p < 0.0001$). However, the effect is much stronger for the projects that moved into the giant component.

[10] We examine alternative functional forms as well. Not surprisingly, we find that the log/log specification has a much higher adjusted R-squared than the log/linear specification and a linear/linear specification performs even more poorly.

[11] Recall that when we employ closeness in the analysis, we must restrict attention to connected projects.

Network Dynamics and Knowledge Transfer

We believe that the results regarding the associations between (i) degree and project success and (ii) closeness and project success mean that spillovers are particularly pronounced for those projects that joined the giant component during the 2007-2009 period. This result suggests that joining a large pool of knowledge (i.e., the giant component) allows projects to gain access to high-impact, novel knowledge and ideas.

Table 2 also shows that changes in the number of contributors are positively associated with changes in the number of downloads. This association is true for projects outside the giant component (Case I) and projects in the giant component (Case II). When we split the giant component into two groups, we see that this result holds as well for projects always in the giant component (Case IIA) and projects that moved into the giant component between 2006 and 2009 (Case IIB).

3.2 Knowledge Spillovers via Star Contributors

Table 2 shows that a change in the number of Stars does not significantly influence downloads for projects outside the giant component (Case I: $\beta = -0.016$, $p = 0.81$). However, changes in the number of Stars are significantly positively associated with changes in the number of downloads for projects that are in the giant component (Case II: $\beta = 0.14$, $p = 0.01$). Thus, changes in the number of Stars are positively associated with changes in the number of downloads in the giant component even after controlling for the network structure. This effect does not exist for projects outside the giant component, which suggests that the spillovers via Stars are due in part to being in the giant component.

We then compare the impact of Star developers who were in the giant component consisting of 10,818 projects throughout the entire period of the study (Case IIA) to those associated with the 2,656 projects who later joined the giant component sometime between

January 2006 and 2009 (Case IIB). Table 2 shows that whereas changes in the number of Stars on a project are not significantly associated with changes in downloads for projects that moved into the giant component (Case IIB: β = 0.064 p = 0.60), changes in the number of Stars are significantly positively associated with changes in the number of downloads for projects that were always in the giant component (Case IIA: β = 0.14, p = 0.04). We interpret these results to mean that the more prolonged the exposure of projects to external projects and developers, the greater is the positive impact on project success from the addition of a Star.

Table 2: Main results

DV: lΔdownloads	**Case I** Outside the Giant Component	**Case II** In the Giant Component	**Case IIA** Always in the Giant Component	**Case IIB** Moved into the Giant Component
Constant	6.30 (32.42)	5.00 (14.95)	4.71 (12.92)	7.65 (5.53)
lΔCpp	1.32 (18.83)	1.73 (33.33)	1.80 (30.19)	1.42 (13.25)
lΔdegree	0.42 (7.07)	0.34 (6.53)	0.30 (5.39)	0.63 (3.76)
lΔcloseness		0.15 (4.41)	0.15 (4.22)	0.89 (2.73)
ΔStars5	-0.016 (-0.25)	0.14 (2.48)	0.14 (2.08)	0.064 (0.53)
Δstage	1.04 (17.07)	0.92 (10.56)	0.99 (8.97)	0.76 (5.53)
lyears_since	-0.87 (-12.41)	0.41 (3.69)	0.54 (4.18)	-0.04 (-0.18)
Moved into Giant Component		-0.59 (-6.41)		
# of Observations	27,410	13,474	10,818	2,656
Adjusted R-squared	0.04	0.14	0.14	0.16

4. Projects with more than one contributor

We repeat the analysis for projects with more than one contributor. Table 3 shows that all of the main results discussed above continue to hold; thus, our results are robust to excluding projects with just a single contributor. The result for Stars has borderline significance in Case IIA; again, however, Stars seem to matter more for projects that have benefitted from being in the giant component for a relatively long period of time than for projects that moved into the giant component more recently (Case IIB).

Table 3: Projects with more than one contributor

Dept Variable: lΔdownloads	**Case I** Outside the Giant Component	**Case II** In the Giant Component	**Case IIA** Always in the Giant Component	**Case IIB** Moved to the Giant Component
Constant	6.00 (16.46)	4.67 (10.78)	4.35 (9.24)	8.25 (4.51)
lΔCpp	1.51 (17.54)	1.54 (26.88)	1.61 (24.94)	1.18 (9.31)
lΔdegree	0.41 (4.28)	0.37 (6.25)	0.33 (5.31)	0.65 (3.03)
lΔcloseness		0.15 (3.42)	0.14 (3.24)	1.08 (2.45)
ΔStars5	-0.036 (-0.32)	0.14 (1.89)	0.13 (1.62)	0.062 (0.39)
Δstage	0.89 (8.50)	0.75 (7.11)	0.80 (6.15)	0.63 (3.61)
lyears_since	-0.61 (-4.38)	0.77 (5.21)	0.90 (5.37)	0.23 (0.74)
Moved into Giant Component		-.060 (-5.10)		
# of Observations	8,094	8,632	7,061	1,571
Adjusted R-squared	0.07	0.15	0.15	0.15

In Table 8 in the appendix, we include all observations. Although the R-squared coefficients are much smaller in the regressions in Table 8 than in Table 2, the results are qualitatively unchanged, which greatly strengthens the main results of the paper.

5. Testing For Endogeneity

Although our discussion focuses on how the network structure affects success, the reverse may be true as well: contributors may want to join popular projects. Developers may want to be associated with more successful projects, thereby making the number of contributors (and thus the degree) endogenous.[12] In fact, the Sourceforge.net website states that, "as a project's activity rises, SourceForge.net's internal ranking system makes it more visible to other developers who may join and contribute to it. Given that many open-source projects fail due to a lack of developer support, exposure to such a large community of developers can continually breathe new life into a project."

[12] Closeness can also be endogenous, but only under a very unlikely scenario.

Here, we discuss the tests that we employ to investigate potential endogeneity. Because we are using panel data with network variables, several approaches to test for the endogeneity of *degree* and *closeness* are possible. We believe that the most convincing test for endogeneity is to restrict ourselves to those projects that had no changes in the number of contributors over the 2006–2009 period. In such a case, reverse causality (i.e., the effect that describes the tendency to join popular projects) is absent.[13] Note that the degree can change for projects that have no changes in the number of their contributors. The mechanism by which this change can occur is that the degree centrality of the original project also increases when a contributor on a particular project joins another project.[14]

Our results describing what occurs when we restrict the analysis to projects that had no change in the number of contributors are reported in Table 4 for Cases II, IIA, and IIB. As expected, we find that the effect of changes in closeness on changes in downloads is completely robust to all these 'tests' for endogeneity, which is not surprising because closeness can only be endogenous under an unlikely scenario. Similarly, the results regarding Stars are virtually unchanged from the results provided in Table 2.

In the case of degree, a comparison between Tables 2 and 4 shows that the results for degree are slightly smaller in Table 4 because of the 'joining popular projects effect.' Nevertheless, in all three cases (II, IIA, and IIB,) the estimated coefficients for degree are statistically significant. This analysis suggests that reverse causality is not driving the results.

---

[13] Of course, it is possible that some contributors joined and some left with a net change of zero, but the overwhelming majority of such projects had no changes in personnel.

[14] Similar to degree, the number of Stars on a project can change even when the number of contributors does not; this occurs when a contributor on one project joins other projects and transitions from working on fewer than five projects to working on five or more projects.

Table 4: Testing for Endogeneity

Dept Variable: lΔdownloads	Case II: In the Giant Component: ΔCpp = 0	Case IIA: Always in the Giant Component ΔCpp = 0	Case IIB: Moved into the Giant Component ΔCpp = 0
Constant	7.28 (19.41)	7.13 (17.48)	9.06 (6.09)
lΔCpp			
lΔdegree	0.25 (3.98)	0.22 (3.35)	0.50 (2.53)
lΔcloseness	0.17 (4.54)	0.17 (4.39)	0.81 (2.21)
ΔStars5	0.17 (2.53)	0.16 (2.10)	0.11 (0.82)
Δstage	0.86 (8.71)	1.31 (8.80)	0.93 (4.81)
lyears_since	-0.077 (-0.62)	0.0043 (0.03)	-0.38 (-1.47)
Moved to Giant Component	-0.64 (-6.30)		
# of Observations	10,421	8,578	1,843
Adjusted R-squared	0.02	0.02	0.03

6. Conclusion:

Prior research studying the relationship between network structure and performance has ignored the implications of the dynamics of knowledge spillovers that occur by means of the interaction of different developers collaborating in different research projects over time. We contribute to the research on organizational learning by studying how changes in network structures can foster knowledge transfer that occurs through developers interacting across distinct development projects. Importantly, we compare organizations that were in the giant component of a network for relatively long periods of time with organizations that joined such giant components during the sample period. We find that in general, changes in the network architecture are positively associated with changes in project success. However, we find that established projects within the giant component benefit differently from changes in network structures than projects that only recently entered such a giant component of a network.

Our study advances the understanding of the link between network structures, agent network position, and organizational performance; nevertheless, it is subject to a few limitations. First, we have theorized about Stars' capacities to access, assimilate, and diffuse explicit and tacit knowledge via boundary-spanning activities. Future research should attempt to measure these latent variables that underlie the innovativeness and productivity of development teams.

References

Ahuja, G. 2000. Collaboration Networks, Structural Holes, and Innovation: A Longitudinal Study. Adm. Sci. Q. 45(3) 425–455.

Ballester, C., A. Calvó-Armengol, Y. Zenou. 2006. Who's who in networks. Wanted: the key player. *Econometrica* **74**(5) 1403–1417.

Beyerlein, M. M., D. A. Johnson, S. T. Beyerlein. 2001. Virtual teams. Jai.

Bonaccorsi, A., C. Rossi, S. Giannangeli. 2006. Adaptive entry strategies under dominant standards: Hybrid business models in the Open Source software industry. Manag. Sci. 52(7) 1085–1098.

Burt, R. S. 1992. Structural holes: the social structure of competition. Harvard University Press.

Calvo-Armengol, A., M. O. Jackson. 2004. The effects of social networks on employment and inequality. *Am. Econ. Rev.* **94**(3) 426–454.

Calvó-Armengol, A., E. Patacchini, Y. Zenou. 2009. Peer effects and social networks in education. Rev. Econ. Stud. 76(4) 1239–1267.

Cohen, W. M., D. A. Levinthal. 1990. Absorptive capacity: A new perspective on learning and innovation. Adm. Sci. Q. 35(1) 128–152.

De Jong, J. P. J., M. Freel. 2010. Absorptive capacity and the reach of collaboration in high technology small firms. Res. Policy 39(1) 47–54.

Fershtman, C., & Gandal, N. 2011. Direct and indirect knowledge spillovers: the "social network" of open-source projects. The RAND Journal of Economics, 42(1): 70–91.

Freeman, L. 1979. Centrality in social networks: Conceptual clarification. Soc. Netw. 1(3) 215–239.

Goyal, S., M. J. Van Der Leij, J. L. Moraga-González. 2006. Economics: An emerging small world. J. Polit. Econ. 114(2) 403–412.

Granovetter, M. 1985. Economic action and social structure: The problem of embeddedness. Am. J. Sociol. 91(3) 481–510.

Granovetter, M. 2005. The impact of social structure on economic outcomes. J. Econ. Perspect. 19(1) 33–50.

Grewal, R., G. L. Lilien, G. Mallapragada. 2006. Location, location, location: How network embeddedness affects project success in open source systems. Manag. Sci. 52(7) 1043–1056.

Hippel, E. von, G. von Krogh. 2003. Open source software and the "private-collective" innovation model: Issues for organization science. Organ. Sci. 14(2) 209–223.

Jackson, M. O., L. Yariv. 2007. Diffusion of behavior and equilibrium properties in network games. *Am. Econ. Rev.* **97**(2) 92–98.

Jehn, K. A., P. P. Shah. 1997. Interpersonal relationships and task performance: An examination of mediation processes in friendship and acquaintance groups. J. Pers. Soc. Psychol. 72(4) 775.

Karlan, D., M. Mobius, T. Rosenblat, A. Szeidl. 2009. Trust and social collateral. *Q. J. Econ.* **124**(3) 1307–1361.

Lacetera, N., I. M. Cockburn, R. Henderson. 2004. Do firms change capabilities by hiring new people? A study of the adoption of science-based drug discovery. Adv. Strateg. Manag. 21 133–159.

Laurent, A. M. 2004. Understanding open source and free software licensing. O'Reilly Media, Inc.

Lerner, J., J. Tirole. 2002. Some simple economics of open source. J. Ind. Econ. 50(2) 197–234.

Lipnack, J., J. Stamps. 1997. Virtual teams: Reaching across space, time, and organizations with technology. John Wiley & Sons Inc.

Pinto, M. B., J. K. Pinto. 1990. Project team communication and cross-functional cooperation in new program development. J. Prod. Innov. Manag. 7(3) 200–212.

Uzzi, B., J. J. Gillespie. 2002. Knowledge spillover in corporate financing networks: Embeddedness and the firm's debt performance. Strateg. Manag. J. 23(7) 595–618.

Wasserman, S. 1994. Social network analysis: Methods and applications. Cambridge University Press.

Appendix:

Table 6: Descriptive Statistics

	Variable	Observations	Mean	Std. Dev.	Min	Max
Projects **outside** the giant component	Δdownload	27410	20733.59	1109424	0	1.71e+08
	Δbetween	27410	-1.08e-07	1.52e-06	-.0000842	9.55e-09
	ΔCloseness	27410	-.0015933	.0067119	-.0418276	.0001101
	ΔDegree	27410	-.0322875	1.134059	-4	19
	ΔCpp	27410	.0694637	.5928583	-1	20
	ΔStage	27410	0.0444728	.2061469	0	1
	ΔStars5	27410	-.0048887	.2240058	-1	1
	years_since	27410	6.57	1.55	3.97	10.15

	Variable	Observations	Mean	Std. .Dev.	Min	Max
Projects always in the giant component	Δdownload	10818	69818.77	2052881	0	1.98e+08
	Δbetween	10818	8.90e-07	.0000388	-.0007162	.0024578
	ΔCloseness	10818	.0000653	.002364	-.0036971	.0200627
	ΔDegree	10818	.6863561	3.905711	-4	103
	ΔCpp	10818	.5878166	3.099011	-1	104
	ΔStage	10818	.0444629	0.2061308	0	1
	ΔStars5	10818	.0086892	.3565553	-1	1
	years_since	10818	7.34	1.59	3.97	10.16

	Variable	Observations	Mean	Std. Dev.	Min	Max
Projects that moved into the giant component	Δdownload	2656	54599.41	878090.8	0	3.74e+07
	Δbetween	2656	3.66e-06	.0000167	-9.03e-09	.0006224
	ΔCloseness	2656	.0297832	.004486	.0160351	.0454347
	ΔDegree	2656	1.907003	3.467798	-4	81
	ΔCpp	2656	.8524096	3.349011	-1	86
	ΔStage	2656	.1125753	0.316	0	1
	ΔStars5	2656	.1716867	.4432846	-1	1
	years_since	2656	6.34	1.60	3.97	10.11

Table 7: Correlation Among All Centrality Variables (Giant Component: N=13,474)

	ΔCpp	Δdegree	Δcloseness	Star
ΔCpp	1.00			
ΔDegree	0.53	1.00		
ΔCloseness	0.06	0.18	1.00	
Star	0.09	0.34	0.21	1.00

Table 8: Replicating Analysis in Table 2 Using All Observations

Dependent Variable: lΔdownloads	Case II: In the Giant Component	Case IIA: Always in the Giant	Case IIB: Moved into the Giant
Constant	-12.45 (-7.86)	-11.18 (-6.61)	-29.29 (-5.92)
lΔCpp	4.30 (13.25)	3.94 (11.16)	5.63 (6.71)
lΔDegree	1.10 (3.87)	0.93 (3.13)	5.26 (4.71)
lΔCloseness	0.46 (2.95)	0.34 (2.04)	1.97 (3.94)
ΔStars5	0.33 (6.03)	0.29 (4.59)	0.18 (1.52)
ΔStage	1.52 (17.55)	1.69 (15.75)	1.00 (7.13)
lyears_since	0.33 (2.87)	0.45 (3.46)	-0.10 (-0.44)
Moved into giant	-0.64 (-4.19)		
# of Observations	14,939	12,251	2,688
Adjusted R-squared	0.05	0.04	0.11