Determinants of Open Source Software License Choice: A Social Influence Perspective

Param Vir Singh  
Carnegie Mellon University, psidhu@andrew.cmu.edu

Corey Phelps  
HEC Paris

Follow this and additional works at: http://repository.cmu.edu/tepper
Part of the Management Information Systems Commons
Determinants of Open Source Software License Choice: A Social Influence Perspective

Param Vir Singh
psidhu@cmu.edu
David A. Tepper School of Business
Carnegie Mellon University,

Corey Phelps
phelps@hec.fr
HEC Paris

We thank Sandra Slaughter, Ray Reagans, David Krackhardt, Yong Tan, and participants at the Workshop of Information Systems Economics and seminar participants at the university of Washington. This work was partially supported by the Berman Faculty Development fund.
Determinants of Open Source Software License Choice: A Social Influence Perspective

Abstract

This study presents a social influence model of open source license choice. For our theoretical foundation, we build on the heterogeneous diffusion model of social influence from the sociology literature. We characterize specific open source licenses as discrete practices that are at risk of being adopted by new open source projects. Specific hypotheses are proposed and tested on a sample of 5,307 open source projects hosted at SourceForge. Our findings suggest the most important factor that determines which license will be adopted by a new project is the type of license chosen by licensors more socially proximate in the social network of a new project’s licensor. Moreover, the likelihood that a new open source project adopts a particular license increases when additional similar open source projects have previously adopted such a license and when these projects are large and successful. We also find that project managers with longer tenure in open source environments or who have been members of successful open source projects are less susceptible to social influence. This suggests that project managers who are inexperienced with open source software development or who have contributed to unsuccessful open source projects are most influenced by their social context.

Key words: open source software license, social networks, adoption and diffusion, IT governance, social influence, hierarchical Bayes.
1. Introduction

In the last decade, the open source model has revolutionized traditional software development. The number of open source software (OSS) projects has grown dramatically and established firms such as IBM and Microsoft have begun to use elements of OSS in their own development efforts. The open source model has even begun to be applied and adapted to products beyond software (O’Mahony 2003). As a result, academic research on the OSS movement has exploded (von Krogh and von Hippel 2006). A primary area of inquiry into OSS focuses on how such projects are governed (von Krogh and von Hippel 2006). Indeed, governance is central to defining and understanding the phenomenon of OSS.

The contractual agreement (i.e., license) that governs how a software product is distributed determines its status as “open source,” rather than the way it was developed (Lerner and Tirole 2005a). For software to be considered OS, its source code must be made generally available to its users, ensuring no individual actor can appropriate it (McGowan 2000). The advent of the OSS licensing framework was an “institutional innovation” (Hargrave and Van de Ven 2006) in that it was a novel and unprecedented departure from previous legal mechanisms to promote cooperation and beneficial exchange among actors with diverse incentives (Demil and Lecocq 2006). It does so by allocating rights on intellectual property (software code) to a commons, which is freely accessible by the community of users and developers, rather than to individual contributors (Kogut and Metiu 2001).

Within the broad OSS framework, there is a variety of specific licenses. This diversity stems from differences in the number of restrictions a particular license places on how users may use and modify the software (Lerner and Tirole 2005a). For example, the inclusion of a “copyleft” provision requires that, when modified versions of the program are distributed, the source code must be made generally available. In contrast, a “reciprocal” or “viral” clause prohibits co-mingling of the code with other software that does not use the same license. These contractual provisions have been found to affect an OSS product’s market potential and the number and quality of developers interested in contributing to it (Comino et al. 2007, Stewart, Ammeter, Likoebe 2006, Lerner and Tirole 2005b).

Given the diversity of possible licenses and their potential influence on project performance, what determines which type of license will be adopted to govern a new OSS project? In contrast to the growing body of research that examines why individual developers contribute to OSS projects (von Krogh and von
Hippel 2006), only a few studies have examined the determinants of OSS license choice (Edwards 2005, Gambardella and Hall 2006, Lerner and Tirole 2005b, Sauer 2007, Sen et al. 2009). This is surprising given the important role contractual governance plays in the viability and success of OSS projects. In one example, Lerner and Tirole (2005b) developed a model in which a project manager considers how her choice of a license influences the incentives of developers to contribute to the project and thus its success. The manager then chooses the license that maximizes her expected benefits from the project. Lerner and Tirole’s (2005b) empirical results were largely consistent with the predictions of their model. The remaining studies on OSS license choice also model a licensor’s problem as an optimizing act between attracting more contributors, thus increasing the value of the software and project performance, and preserving the licensor’s ability to appropriate the benefits of commercializing the software.

Despite the insights offered by these studies, substantial opportunity exists to improve our understanding of how and why particular licenses are chosen to govern particular OSS projects. A potential limitation of these studies is that they assume OSS project managers possess sufficient knowledge about the variety of OSS licenses available and how these licenses influence a given project’s attractiveness to developers and therefore its potential success. However, a project manager’s licensing decision is likely to be characterized by substantial uncertainty. The OSS licensing framework is novel and many specific licenses exist, all of which are complex and untested in the courts (Lerner and Tirole 2005b). Rosenberg (1998) observed that OSS licenses are surrounded by substantial confusion. Moreover, the choice of a license is a one-shot, largely irreversible decision and licensors are typically software developers with little or no legal expertise (McGowan 2001). Under such conditions of uncertainty, a substantial body of research indicates that managers will turn to their social networks for help in determining the appropriate choice (e.g., Buskens 2002, DiMaggio and Powell 1983, Rogers 1995). Prior research ignores the potential influence of social context on individual licensing choice. Accordingly, we seek to complement this research by investigating the role that a project manager’s social network plays in influencing her adoption of a particular license to govern a new OSS project.

To examine this issue, we draw on the heterogeneous diffusion model of social influence from the sociology literature as our theoretical foundation (Greve, Strang, Tuma 1995, Strang and Tuma 1993). This model was developed to parsimoniously explain how social context influences an actor’s adoption of an
innovation and its diffusion within a population of actors (Strang and Tuma 1993). We characterize specific OSS licenses as discrete, innovative governance practices that are at risk of being adopted by new OSS projects. Drawing on this model, we argue the likelihood of the adoption of a specific license type will depend upon three classes of factors: the susceptibility of the project manager to information about the practice, the infectiousness of information about the practice from prior adopters of the practice, and the social proximity between the project manager and prior adopters of the practice. We expect the infectiousness of a particular license will increase as the number, size, and performance of other OSS projects that have adopted the license in the past increases. We also expect a new OSS project will be more likely to adopt a specific license when the project manager is more socially proximate to individuals contributing to other projects that have adopted the same license, either through direct contact with them or by occupying a similar structural role in the network. Additionally, we expect the likelihood of adoption of a particular license by a new project to increase with the adoption of the same license by projects that are in the same technological domain. Finally, we expect project managers will be less susceptible to adopting a specific license when they have worked on large and successful OSS projects in the past.

We test these hypotheses on 5,307 OSS projects hosted at SourceForge. Following Lerner and Tirole (2005b), we categorize OSS licenses as permissive, restrictive and highly restrictive, based on the key contractual clauses they contain. After controlling for the economic incentive explanations put forth by Lerner and Tirole (2005b), our results suggest the likelihood that a new OSS project adopts a particular license increases with the number of OSS projects in the managers’ social network that previously adopted such a license and when these projects are large and successful. We also find that project managers who have been members of successful prior OSS projects and with longer tenure in OSS environments are less susceptible to social influence. Finally, we find that a project manager is more likely to adopt a particular license when technically similar projects have adopted that license and when her project shares more indirect social ties to the same developers as other projects that have adopted the same license.

The results of this study contribute to the literature on licensing choice, particularly in the OSS setting. Our results complement Lerner and Tirole’s (2005b) study and expand our understanding of licensing choice, both in the OSS setting and beyond. While Lerner and Tirole (2005b) showed that OSS license choice is shaped by the economic incentives of both the licensor and the community of developers,
we show how the social context in which licensors are embedded influence their choice of license and how their experience moderates this influence. Given the growing popularity and importance of OSS projects and the influence a project’s license has on its performance, our results suggest the viability of the OSS movement may be constrained to the extent social context influences licensors to adopt licenses based on a logic of social influence rather than economic maximization.

This study also contributes to the innovation diffusion literature. While research into the adoption and diffusion of technical innovations, such as new products or services, is vast (Rogers 1995), research into the adoption of innovative administrative practices, such as novel governance practices, is rare (Wejnert 2002). Our study contributes to the innovation diffusion literature by showing how the social structure in which a licensor is embedded influences the adoption of a novel governance practice. Our use of the heterogeneous diffusion model is one of the first applications of this model in the Information Systems literature.

In the next section, we discuss the OSS phenomenon and the different OSS licenses. In section 3, we provide a brief review of relevant research and the motivation for our theoretical approach. We develop hypotheses in section 4, followed by a description of data and methods in section 5. Results are provided in section 6 and the discussion and conclusion in section 7.

2. Open Source Software

2.1 Open Source Software Development Process

All OSS projects follow a similar process. An “initiating developer” begins a project by working on an idea and partially implementing it. The initiating developer then hosts the source code and invites other developers to contribute to the project. Developers volunteer to perform specific tasks and collaborate as a team, incorporating their individual creations into a single, seamless body of source code. Once an executable version of the software is developed, it is released to users for testing and feedback. The software evolves as new features are added, existing features are modified, and bugs get fixed. The whole process involves the sharing of ideas and joint problem-solving that fosters social bonds among collaborators. Except for a few well-known projects such as Linux, Apache, and PHP, the majority of OSS projects involve a small number of developers, typically between one and ten, besides the project initiator (Krishnamurthy 2002). Given the small number of developers typically involved in a project and the
frequency and intensity of their interactions over several months to several years, the strength of social ties among them can be considerable (Hahn et al. 2008, Singh 2009). Because OSS projects bring together and stimulate the formation of social ties among teams of developers and because developers often work on multiple OSS projects, either concurrently or sequentially over time, a large scale social network is produced that directly and indirectly connects developers participating in the broader OSS community. While projects create social ties among developers, this affiliation network also socially connects projects when they share common developers (Grewal, Lilien, Mallapragada 2006). It is the influence of this latter inter-project social network that we examine in this study.

2.2 Open Source Software Licenses

For software to be characterized as “open source,” it must be offered under a license that satisfies several conditions. Among other things, these conditions include free redistribution; access to the source code; freedom to experiment with and redistribute modifications; prohibition of discrimination against people, groups, or fields of endeavor; integrity of the author’s source code; and non-contamination of other software distributed with software subject to the license\(^1\) (Opensource.org 2009). The Free Software Foundation (FSF) and the Open Source Initiative (OSI) are two bodies that approve OSS licenses.

While the OSI and FSF terms are the basic tenets of the OSS community, OSS licenses differ in the extent to which they restrict how users may use and modify the software under license. At one extreme are the highly restrictive licenses such as GNU general public license (GPL), and at the other extreme are the permissive licenses, such as Berkeley Software Distribution (BSD) licenses. As the name suggests, permissive licenses have fewer restrictions than highly restrictive ones. Highly restrictive licenses differ from permissive licenses in two key ways (de Laat 2005):

1. They require that, when modified versions of the program are distributed, the source code must be made generally available. This provision is called the “copyleft” clause.
2. They prohibit the software to be mingled with other software that does not use the same license. This provision is called the “viral” or the “reciprocal” clause.

\(^1\) The lists of conditions and the rationale behind them are available at http://www.opensource.org/docs/definition.php and http://www.gnu.org/philosophy/free-sw.html
The highly restrictive licenses were the first free software licenses and their proponents argue that developer communities that support OSS production can only be sustained if software, once developed free, stays free (Stallman et al. 2002). The most famous of these licenses, GPL, was authored by Richard Stallman, an early proponent of the OSS movement. Frustrated by the proprietary restrictions associated with UNIX software development, Stallman decided to build a free operating system, GNU, in 1984. Stallman recognized that, without proper protection, someone could appropriate the source code, modify it, and make it proprietary, contradicting his beliefs that software must be “free.” He also feared that, if he did not protect the software from appropriation, others might be reluctant to share their code.

The “copyleft” and “viral” clauses were specifically designed to protect the software from being hijacked by proprietary software developers. The implication of these clauses is that any modification or derivation of the software has to be offered under GPL, making GPLed software less attractive to commercial actors. Use of the “viral” clause restricts the software from taking advantage of complementarities with other software, reducing its appeal to both contributors and users.

Proponents of Berkeley Software Distribution (BSD) argue that it is better for the community to be able to keep the modified or derivative works private and to release code back to the community if and when they please. While the BSD license makes software attractive for commercial software such as embedded systems, it does not protect the software from being hijacked. Recognizing the provisions in GPL make software less attractive to commercial actors and that its alternative, the BSD license, does not protect the software from being hijacked, Stallman authored the Lesser General Public License (LGPL) in 1991 as a compromise between the two (Stallman et al. 2002). The LGPL includes the copyleft provision but does not include the viral provision. Thus, if someone modifies the software and distributes it, the source code has to be made generally available. However, it allows the software to be combined with proprietary software without affecting the proprietary software’s license. Lerner and Tirole (2005b) refer to LGPL-type licenses as restrictive licenses. Any license that is approved by the OSI and the FSF can be categorized as permissive, restrictive, or highly restrictive. Please refer to the online supplement for a table of individual licenses and their classification.

The creation of the OSS licensing framework was an “institutional innovation” (Hargrave and Van de Ven 2006) because it was a novel and unprecedented departure from previous legal mechanisms to
promote cooperation and beneficial exchange among actors with diverse incentives (Demil and Lecocq 2006). OSS licenses promote cooperation and exchange by allocating rights on intellectual property (i.e., software code) to a commons, which is freely accessible by the community of users and developers, rather than to individual contributors (Kogut and Metiu 2001, Shah 2006).

3. Prior Research on the Choice of Open Source License

Before an initiating developer makes his or her source code available to others for further development, he or she must decide the scope of the license by choosing a particular OSS license. This choice is crucial as it signals to potential contributors their ability to benefit from their participation, which directly affects the type and number of developers attracted to the software’s development, and thus its pace of development and viability (Lerner and Tirole 2005b, Sauer 2007, Stewart et al. 2006). Licensing choice is also significant because a license, once adopted, is seldom changed in the future. The few attempts that have been made in the past have faced fierce opposition from contributors, have required substantial effort on the part of licensors to acquire permission from individual contributors, and have been found wanting in their ability to force existing licensees to honor alterations to the terms of license (McGowan 2001, Shankland 2004, Wheeler 2008).

Research on the choice of the scope of an OSS license is dominated by an economic incentive perspective: the conflicts between the licensor and subsequent contributors’ incentives are the defining characteristics of this approach. The licensor and potential contributors are considered economic agents who seek to maximize their own incentives. Both the licensor and contributors can derive commercial and non-commercial benefits from the project. Non-commercial benefits include peer recognition, opportunities for signaling to potential employers, the pleasure of participating in code development, and the ability to modify the software for their own use (Lerner and Tirole 2005b). The commercial benefits include the money one can make by providing modified or complementary products and services. While the non-commercial benefits received are likely to be proportional to one’s effort in the project, the commercial benefits flow disproportionately to the licensor (Lerner and Tirole 2005b). This creates conflict between the licensor’s and the contributors’ incentives. The licensor would prefer a permissive license as it would provide him or her with greater opportunities to commercialize the software. However, contributors would prefer a more restrictive license to protect their contributions from being hijacked. The overall benefits
depend on how many contributors actually participate in the project. With signaling, peer recognition, and the value of the software increasing with the number of contributors, the licensor’s problem is typically modeled as an optimizing balancing act between attracting more contributors and preserving his or her ability to commercialize. Several studies have followed this approach (e.g., Edwards 2005, Gauduel 2004, Gambardella and Hall 2006, Lerner and Tirole 2005b, Sauer 2007, Sen et al. 2009).

Projects that are more attractive to contributors, such as applications designed for commercial settings, proprietary operating systems and the Internet, are the ones that have greater appeal to the community (Lerner and Tirole 2005b). A licensor is more likely to use a permissive license for such projects because contributors are more likely to participate in such projects. Software in this category provides greater benefits to contributors based on their ability to tailor the code for specific applications, greater signaling incentives, ego gratification, or peer recognition opportunities. Lerner and Tirole’s (2005b) empirical results were generally consistent with this perspective.

Despite the insights this research provides into understanding how and why particular licenses are chosen to govern particular OSS projects, it is limited in at least one important respect. These studies assume OSS project managers possess sufficient knowledge about the variety of OSS licenses available and how these licenses influence a given project’s attractiveness to developers and therefore its potential success. However, a project manager’s licensing decision is likely to be characterized by substantial uncertainty. During the time of this study, nearly 40 licenses were recognized as OSS by the Open Source Initiative. Moreover, OSS licenses are complex legal documents that have not yet been tested in court (Lerner and Tirole 2005b). This complexity is compounded by the fact that a licensor is typically a software developer with little or no legal expertise. The American Bar Association recently observed, “The terms of various OSS licenses may pose inherent problems that may not be apparent to those not skilled in the legal nuances of licenses” (ABA 2008). Because of their newness, innovative practices such as OSS licenses, typically lack reliable and rich information about their strengths and weaknesses (Rogers 1995). Consequently, Rosenberg (1998) argued that OSS licenses are surrounded by a lot of confusion. This confusion is likely to be exacerbated by the inherent riskiness in choosing a particular license. While the ability to use a practice on a trial basis before committing to it reduces the risk of its adoption (Rogers
the choice of an OSS license is a one-shot, largely irreversible decision (McGowan 2001, Shankland 2004, Wheeler 2008).

With the mass media attention on OSS licensing and the online environment in which the OSS community interacts, obtaining information about a license is not difficult. Rather, a licensor’s problem is in knowing how to filter the volume of information that reaches him or her for relevancy and how to evaluate this information using incomplete knowledge about the decision problem (Coleman 1966). Under conditions of uncertainty, when an actor needs to, but cannot independently or via market mechanisms, efficiently ascertain the true impact of adopting a practice, his or her social network will appreciably affect the adoption decision (Rangan 2000). A substantial body of research in sociology and social psychology shows that individuals turn to their social networks for help in determining the appropriate choice when faced with significant decision uncertainty (e.g., Buskens 2002, DiMaggio and Powell 1983). Although prior research ignores the potential influence of social context on a project developer’s license choice, given the substantial uncertainty licensors face when making such a decision, we expect their social networks significantly influence their choice of license. To examine this issue, we draw on the heterogeneous diffusion model of social influence from sociology as our theoretical foundation (Strang and Tuma 1993).

4. Heterogeneous Diffusion Model

The heterogeneous diffusion model was developed to summarize how social context influences an actor’s adoption of an innovation and its diffusion within a population of actors (Greve, Strang, Tuma 1995, Strang and Tuma 1993). In contrast to diffusion models that assume spatial and temporal homogeneity, the heterogeneous diffusion model allows for spatial heterogeneity in the influence of social proximity on individual adoption behavior and accommodates temporal heterogeneity by allowing the influence of prior adopters and social proximity to vary over time (Strang and Tuma 1993). In this model, the extent to which an actor’s social context influences its adoption behavior depends on three factors: infectiousness, social proximity and susceptibility (Greve 2005). The infectiousness of a prior adopter of an innovation (i.e., source) describes how strongly information about its actions affects a potential adopter. The social proximity of the source to a potential adopter describes how easily information is transmitted between them, based on their social distance from each other (Greve 2005). The susceptibility of a potential adopter
describes how much the actor is influenced by the information available about the innovation and depends on factors that influence its motivation and ability to adopt the particular innovation (Greve 2005). In sum, information about an innovation adopted by a source project becomes available to potential adopters at a rate determined by the source’s infectiousness. The information reaches a potential adopter at a rate determined by the social proximity of the source to the potential adopter and increases the potential adopter’s likelihood of adoption to the extent determined by its susceptibility (Greve 2005). The model also specifies that an actor has intrinsic propensities to adopt a particular innovation based on its own inherent characteristics, independent of social context. Because the focus of this study is on social influence processes, we consider intrinsic characteristics affecting adoption as control variables that provide a baseline model for comparison to our social influence variables.

Prior research has used this model to explain an organization’s adoption of innovative governance practices (Davis and Greve 1997). Because the OSS licensing framework represents a novel mode of contractual governance (Demil and Lecocq 2006), and thus an institutional innovation (Hargrave and Van de Ven 2006), we characterize specific OSS licenses as discrete, innovative governance practices that are at risk of being adopted by new OSS projects. In the next section we develop hypotheses linking variables associated with each of the three broad factors in the heterogeneous diffusion model to the likelihood a new OSS project will adopt a particular license.

4.1 Social Proximity

Social proximity refers to the social distance between two actors in a social network and determines how easily information is transmitted between them and the relevance of this information (Coleman 1966). Because of the richness, volume and relevance of information they provide, socially proximate actors exert greater social influence by more effectively reducing uncertainty about appropriate attitudes and behavior (Davis and Greve 1997). A socially proximate actor provides an influential frame of reference by which a focal actor evaluates and interprets information (Leenders 2002). Two different approaches to conceptualizing social proximity in a social network exist, each with its own respective causal mechanism linking proximity with social influence. The first approach, known as social cohesion, defines social proximity in terms of the number, length and strength of the paths that connect actors in a network
The second approach defines social proximity in terms of the similarity of two actors’ profiles of network relations, or “structural equivalence” (Marsden and Friedkin 1993).

### 4.1.1 Social Cohesion

Social cohesion defines social proximity in terms of the number, length and strength of the paths that connect actors in a network (Marsden and Friedkin 1993). The simplest form of cohesion is when two actors, such as a potential and prior adopter of an innovation, share a direct social tie. Directly connected actors communicate and share information with each other more frequently and with greater fidelity than indirectly connected actors (Burt 1982). Direct ties are conduits for the communication of rich, personalized information, which tends to be more influential than impersonal information sources (Rogers and Kincaid 1981). The volume and fidelity of information decays as the number of nodes indirectly connecting actors increases (Shannon 1949), making indirectly connected actors less socially influential on a potential adopter than direct ties (Burt 1982). Research in social psychology suggests involvement in shared activities provides opportunities for social cohesion to develop and that shared attitudes develop from social cohesion (Homans 1961). Faced with an uncertain situation, such as the adoption of an innovation, individuals discuss it with their proximate peers and develop a consensual normative understanding of the associated costs and benefits (Rogers 1995). Social ties provide detailed, personalized and more persuasive information on costs and benefits of adoption than general information sources (Rogers 1995). Discussions with prior adopters of an innovation build social pressures on the potential adopter to adopt the innovation when faced with an opportunity to do so (Rogers and Kincaid 1981). Social pressure increases with social cohesion and hence a potential adopter is more likely to adopt an innovation that has been adopted by his or her most proximate peers. In support of these arguments, research has found that shared attitudes and behavior develop among people or organizations that are connected through strong and direct communication channels. For example, firms are more likely to adopt corporate governance or merger-and-acquisition practices that are adopted by their direct partners (Davis and Greve 1997, Haunschild 1994), and physicians are more likely to adopt a new drug if it is adopted by their peers (Coleman 1966).

In the OSS context, projects on which a licensor has worked in the past provide greater opportunities for communication and thus for social cohesion. Through his or her discussions with
developers on prior projects, a focal project’s licensor develops a shared understanding of the costs and benefits associated with the license chosen for those projects. While direct involvement in a project provides opportunities to observe the consequences of adopting a license choice, a focal project’s licensor may also receive useful and persuasive information from projects with which he or she is not involved but has social ties with developers who are. These ties can provide the licensor with detailed, personalized and persuasive information on the costs and benefits of the particular licenses adopted by these projects. Prior adopters have experience with a particular license and thus understand it better than a potential adopter and may communicate their preferences persuasively via social ties with the focal project administrator. Hence, socially proximate prior adopters exert social pressure on a licensor to adopt the same license for a new project that they adopted.

HYPOTHESIS 1: A licensor is more likely to choose a license type that is adopted by his or her contacts.

4.1.2 Structural Equivalence

An alternative conceptualization defines social proximity in terms of the similarity of two actors’ profiles of network relations, or “structural equivalence” (Marsden and Friedkin 1993). Structurally equivalent actors occupy qualitatively similar roles and have similar patterns of relationships with the same types of other actors (Marsden and Friedkin 1993). Because equivalent actors occupy similar social roles, they are substitutes and competitors (Burt 1987). Competition among two equivalent actors increases their incentives to monitor and compare behaviors to ensure neither has an advantage or falls behind. As a result, equivalent actors view information about each other’s behavior as more relevant and are more motivated to imitate each other’s adoption of an innovation (Burt 1987). The structural equivalence model of social proximity highlights symbolic communication among social substitutes rather than direct communication among contacts. Several studies suggest that structural equivalence exerts significant effects on imitative adoption (Bothner 2003 Galaskiewicz and Burt 1991). In a classic study on the diffusion of medical innovations, Burt (1987) found that physicians were more likely to adopt a new drug when it was adopted by other structurally equivalent physicians.

The intensity of competition between two OSS projects increases with their potential for substitutability in each other’s relations. OSS projects compete with each other in attracting developers (Hahn et al. 2008). Thus, a licensor competes for contributors with its structurally equivalent projects.
because they both tap into the same pool of developers for contributors. Therefore, a licensor will be more likely to adopt a particular OSS license when projects that are structurally equivalent, in terms of the pattern of their relationships with developers, have previously adopted the same license.

HYPOTHESIS 2: *A licensor is more likely to choose a license type that has been chosen by projects that are structurally equivalent to his or her project.*

Defining structural equivalence between two actors in terms of the similarity of the pattern of their relationships with the same alters is highly restrictive (Marsden and Friedkin 1993). Broadly, equivalence refers to the degree to which two actors occupy similar roles and therefore serve as common referents (Marsden and Friedkin 1993). In this vein, equivalence among organizations is often defined by the extent to which they produce or provide similar products or produce products using similar technologies and therefore compete for similar customers and suppliers (Bothner 2003, Davis and Greve 1997, Fligstein 1985). Prior research has found that the extent to which organizations produce similar, competitive products or employ the same technologies are more likely to imitate each other’s adoptions of innovations (Bothner 2003, Davis and Greve 1997, Fligstein 1985).

In the OSS context, projects are equivalent to the extent they employ the same technology platform in their development. Indeed, the primary data source we use in this study, SourceForge, organizes all OSS projects it hosts into common domains or “foundries” based on their common usage of a technology platform such as projects that use the Perl programming language. Projects in the same foundry typically target and compete for similar users and compete for developers with similar skills (Hahn et al. 2008). Because of the complementarities that often exist among products in the same foundry, licensors may also have strong incentives to adopt the license of other complementary projects in the same technical domain to ensure the licenses of the products are compatible and therefore the products can be mingled and mixed by developers and users (Lerner and Tirole 2005b). Hence, a licensor will be more likely to adopt a specific OSS license that has been adopted by other projects in its foundry.

HYPOTHESIS 3: *A licensor is more likely to choose a license type that is prevalent in his or her project’s foundry.*
4.2 Infectiousness of Prior Adopters

The infectiousness of a prior adopter of an innovation describes how strongly information about its actions affects a potential adopter. That is, prior adopters can be more or less influential as referents for subsequent adopters based on their individual characteristics such as their size, performance or status (Greve 2005). In particular, adoptions by successful OSS projects will be more infectious for several reasons. First, successful projects attract more attention, which leads to more information being available to potential adopters about them (Greve 2005). Second, a project’s level of success increases the infectiousness of information about its adoption of a license because the adopted innovative practice may be interpreted by potential adopters as a cause of its success (Greve 2005). Third, while the adoption of an OSS license is characterized by substantial uncertainty, innovations adopted by high status organizations are viewed as less uncertain and therefore more likely to be imitated by others (DiMaggio and Powell 1983). In the OSS community, the success of a project reflects its social status in the community (Stewart 2005). Thus, adoption by successful projects is likely to be imitated by others.

HYPOTHESIS 4: A licensor is more likely to choose a license type that is adopted by successful projects.

4.3 Susceptibility of Licensor

Susceptibility refers to how much a licensor is affected by the practices adopted by others (Greve 2005). Licensors may differ in their susceptibility to the influence from prior adopters based on differences in their motivations to search for new practices and learn from the actions of others (Greve 1998). Individual efforts are directed at achieving goals and a failure to achieve them increases an individual’s propensity to search for new behavior or practices (Cyert and March 1963). Consistent with this argument, poor performing organizations have been found to be more susceptible to social influence when faced with decisions to adopt new practices or innovations (Davis and Greve 1997). Poor performance in prior projects may increase a licensor’s propensity to search for appropriate practices, which will increase his or her susceptibility to outside influence. In contrast, prior success reduces incentives to change existing behavior and practices (Greve 2005). Prior successful experience also sends a positive signal about the capabilities of the licensor to the community, which increases the project’s attractiveness to potential contributors, irrespective of the license adopted. Thus, such a licensor is less likely to seek help from others to determine
the appropriate license for his or her new project. Therefore, we argue that a licensor’s susceptibility to influence from others decreases with the level of success of his or her prior OSS projects.

The need to learn from others’ behavior increases with a potential adopter’s unfamiliarity with an innovation or the setting in which it is used (Wejnert 2002). Newcomers to an organization or community have to cope with unfamiliar situations and practices. When encountering something previously unknown or out of the ordinary, an individual begins a process of inquiry (Shultz 1964) and relies upon information and interpretation from others to make sense of the situation (Louis 1980). As individuals gain experience in a particular setting, they normally know what to expect of a situation, making them less susceptible to others’ interpretations or behaviors. While the adoption of an OSS license has uncertain implications, licensors may differ in their familiarity with the OSS community and its practices, owing to differences in their levels of experience. Greate experience and knowledge increases individuals’ sense of self-efficacy and reduces their incentives to alter their behavior (Bandura 1985). A licensor with longer tenure or involvement in a number of projects is likely to be better informed about the potential implications of adopting a particular license type and hence more familiar with it, which will reduce his or her susceptibility to social influence.

HYPOTHESIS 5: The susceptibility of a licensor to social influence when choosing a particular license type decreases with his or her success on prior OSS projects.

HYPOTHESIS 6: The susceptibility of a licensor to social influence when choosing a particular license type decreases with his or her experience in the OSS community.

5. Data and Methods

5.1 Sample and Social Network Construction

To test our hypotheses we constructed a dataset that consists of OSS projects hosted at SourceForge.net (SF). SF is the largest hosting site for OSS projects and accounts for about 90% of all OSS projects. SF provides web space as well as services such as e-mailing facility, discussion forums, CVS\(^2\) repository hosting, and download servers to OSS projects to organize and coordinate their development activities. Many studies of OSS have used SF data (e.g., Comino et al. 2006, Grewal et al. 2006). Lerner and Tirole

---

\(^2\) Concurrent Versioning System (CVS) is a software tool that stores the project source code, tracks changes made to the source code and stores programmers’ comments about the changes they made.
(2005) also used SF data, which enables us to compare our results with theirs’. We consider all projects that were registered at SF between November 1999 (the start date of SF) and December 2003. Data regarding projects such as performance, size, type, intended audience, start date, operating system, user interface, translation language and developers associated with each project were obtained from the project websites and the SF database hosted by the OSTG group. To ensure the developers whose names are associated with the projects in the SF database actively participated in the projects during the period under consideration, we matched their contribution efforts and time of involvement through CVS log files, project communications, and the project documentation that typically lists all contributors.

The hypotheses focus on the social networks of the licensor. Thus, to test the hypotheses, we need to construct these networks. Constructing a social network starts with the identification of the network’s boundary—the collection of actors who, for analytical purposes, is regarded as a bounded social collective (Marsden 2005). Building on prior research that identifies criteria for establishing network boundaries for empirical research (Laumann et al. 1983), we used participation in a foundry as the network boundary. A foundry includes all projects that use a common technology platform, such as all projects that share the Perl programming language as their technology platform. Using a foundry as a network boundary criterion is valid for two reasons. First, projects within foundries are technologically similar and, as such, foundries provide a meaningful context for knowledge sharing among network members. Participation in a foundry as a network boundary for OSS projects at SF has been used in related research (Grewal et al. 2006, Singh 2009, Singh et al. 2007). Second, social relationships almost always occur among individuals within a foundry and rarely occur across foundries. We analyzed memberships for 2000 randomly selected developers who work on multiple projects and found that only about 4% worked across two or more foundries.

In constructing our sample, we considered all projects that were registered in all 22 foundries at SF between November 1999 and December 2003. We discarded projects that never showed any development activity, leaving 29,995 projects. Although we use all of these projects to construct our networks, we limit our analysis to explaining variation in license adoption to projects that were started between January 2002 and December 2003 (n = 21,220). Because relationships endure over time, constructing social networks using only relationships formed at a particular point in time would greatly understate the true connectivity
of the network. Consider the initial month when we observe license choice by a project (January 2002): using only relationships formed in January 2002 would not capture the relationships formed prior to, yet maintained through, January 2002. Data on both pre-sample relationship formation and relationship duration is needed to accurately assess network structure at a point in time. We assume that relationships formed in the pre-sample period endure through the end of the sample period (December 2003). The assumption of individual social ties decaying after four years is consistent with prior research on interpersonal affiliation networks (e.g., Cattani and Ferriani 2008, Uzzi and Spiro 2005). Because our social network measures are meaningful for only those new projects at risk of license adoption that are directly socially connected to other projects, we remove from our final analysis all projects that are socially isolated (n = 15,913), leaving us with a final sample of 5,307 projects. However, as we explain below, we account for the possibility that this self-selection into or out of our foundry-based social networks biases our results.

5.1.1 Explanation of Network Construction.

To capture social proximity, we constructed 22 affiliation networks. Each network contains projects assigned to a particular foundry. An affiliation network is a network of actors connected by their participation in common events, while events are connected by common actors (Wasserman and Faust, 1994). In our case, the actors are individual developers and the events are projects. Developers have social ties with one another as a result of working together on the same project and projects are related to one another as a result of sharing developers. Data on projects and associated developers are represented using an affiliation matrix, where A={aj}, where aj is “1” if i

otherwise. Figure 1 provides an example with five projects (P1, P2, P3, P4, and P5) and five developers (D1, D2, D3, D4, and D5).

<table>
<thead>
<tr>
<th>Developers</th>
<th>Projects</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P1</td>
</tr>
<tr>
<td>D1</td>
<td>1</td>
</tr>
<tr>
<td>D2</td>
<td>1</td>
</tr>
<tr>
<td>D3</td>
<td>0</td>
</tr>
<tr>
<td>D4</td>
<td>0</td>
</tr>
<tr>
<td>D5</td>
<td>0</td>
</tr>
</tbody>
</table>

Jan  Feb  Mar  Apr  May

Figure 1. Affiliation network
A value of one for element (D1, P1) and zero for element (D1, P2) indicates that developer D1 is associated with project P1 and D1 is not associated with project P2, respectively. Because the unit of analysis is a project, we assessed social proximity among projects by projecting each developers-by-project affiliation network into its respective unipartite inter-project network. These networks evolve over time as new projects are established and new/existing developers join/leave the projects.

Identifying the timing of the formation of new projects is critical to constructing accurate affiliation networks. A project’s administrator is responsible for choosing the project’s license. The administrator is the only developer associated with a project when it adopts a specific license. Only relationships and projects that exist prior to the adoption of a license by the focal project can influence the focal administrator’s license decision. For each focal project, the relevant affiliation network consists of all projects initiated in the respective SourceForge foundry prior to the start of the focal project. Because developers may join projects anytime after they commence, only developers who joined existing projects within the relevant foundry before the start date of the focal project were used to construct the relevant inter-project network. To ensure social ties actually existed among developers on the same project, we excluded all developers who did not actively contribute to the projects for which they volunteered.

![Diagram](image.png)

(a) Feb
Project: P1
Developers: D1, D2

(b) March
Project: P1
Developers: D1, D2

Project: P2
Developers: D2, D3

Project: P3
Developers: D3

(c) April
Project: P1
Developers: D1, D2

Project: P2
Developers: D2, D3

Project: P3
Developers: D3, D4

Project: P4
Developers: D5

(d) May
Project: P1
Developers: D1, D2

Project: P2
Developers: D2, D3

Project: P3
Developers: D3, D4

Project: P4
Developers: D5, D4

Project: P5
Developers: D1

Figure 2. An evolving sociogram corresponding to Figure 1
Licensor and focal projects for each month are represented in bold.

An evolving sociogram corresponding to the affiliation matrix of Figure 1 is shown in Figure 2. If there is a common developer between two projects, then there is a link between them as shown between projects P1 and P2 in Figure 2a. One can also see from Figure 2a that Licensor D2 for Project P2 is
drawing from his or her own experience as he or she is the one who connects P1 and P2. Because such a connection represents self-confirmatory behavior and not social influence, the social influence variable calculations do not include such pairs. However, we do account for such pairs by controlling for prior experience. Also note that, in Figure 2c, Project P4 does not have any relationship to other projects. However, in Figure 2d, Project P4 is connected to both Projects P1 and P2, owing to involvement of Developer D2 in Project P4. When the licensing decision was made, Developer D2 was not involved with Project P4 but joined afterwards.

5.2 Dependent Variables

Much of the research on OSS has distinguished OSS licenses based on whether they contain copyleft and viral clauses (e.g., de Latt 2005). These two license clauses have fundamentally different implications for legally copying, modifying and distributing computer software source code, represent fundamental ideological differences among proponents of OSS, and have important implications for the success of OSS projects (de Laat 2005, Stewart et al. 2006). Prior empirical studies of OSS licenses have also argued that the essential distinction between such licenses pertains to the presence or absence of these clauses (Lerner and Tirole 2005, Stewart et al. 2006). Accordingly, we also categorize licenses using these clauses. Because a project may offer its software under several different licenses we follow Lerner and Tirole (2005) and categorize a project under four different categories: (1) all licenses highly restrictive or not (2) some licenses highly restrictive or not, (3) all licenses restrictive or not, and (4) some licenses restrictive or not. In order to compare our results with those of Lerner and Tirole (2005), we follow them and treat each of these four categories as separate dichotomous dependent variables. We estimate four different license choice models. In Model 1, the dependent variable is coded “1” if all the licenses chosen by the focal project were highly restrictive and “0” otherwise. The dependent variables for other models were coded similarly.

5.3 Independent Variables

Social Cohesion. Social cohesion defines social proximity in terms of the number, length and strength of the paths that connect actors in a network (Marsden and Friedkin 1993). Essentially, cohesion captures the

---

3 A project may offer several software packages. The licensor may choose to offer different licenses for different packages.
frequency of communication between two actors in a network and thus the rate of information sharing between them (Burt 1982). Directly connected actors communicate and share information with each other more frequently and with greater fidelity than indirectly connected actors (Burt 1982). The volume and fidelity of information decays as the number of nodes indirectly connecting actors increases (Burt 1982, Shannon 1949). Accordingly, our measure of social cohesion is based on a measure of social distance in which distance is defined as the number of network nodes that exist on the shortest path between two connected actors. To compute our measure of social cohesion, we first calculate the social distance among each pair of projects in the appropriate inter-project affiliation matrix. The social distance matrix is then used to calculate nearness among each pair of projects using an exponential decay transformation (Burt 1982). Larger values of nearness represent stronger social cohesion between two projects. Social distance is calculated from all projects to the focal project on which the focal licensor does not work, but for which there is a path between projects. Because the social distance is only meaningful to projects that started at SF prior to the focal project, we treat the social distance between a focal project and all other projects that started after it as undefined. As we explain below, this project-specific nearness matrix values are used as a measure of social proximity in the construction of a project-specific social influence variable.

**Structural Equivalence.** Structural equivalence between two actors is defined in terms of the similarity of the pattern of their relationships with the same alters (Lorrain and White 1971). Structural equivalence is measured by using the Euclidean distance between projects $i$ and $j$ (Wasserman and Faust 1994). The Euclidian distance measures the dissimilarities in the relations of projects $i$ and $j$ to all other projects, $k$, in the same foundry. We follow Burt (1987) and transform the Euclidean distances to represent similarities in the relations of projects $i$ and $j$ by subtracting them from the maximum value of Euclidian distance involving $i$. Following Leenders (2002), we compute structural equivalence only for those projects that are at a path length of greater than one from the focal project. By doing so, we can more accurately separate the effects of structural equivalence from social cohesion. As we explain below, this project-specific matrix of equivalence values is used as a measure of social proximity in the construction of a project-specific social influence variable.

**5.3.2 Technical Equivalence.** In the OSS context, projects are equivalent to the extent that they employ the same technology platform in their development. OSS projects are assigned to foundries based on their
common usage of a technology platform. Thus, we construct four versions of the variable, Technical Equivalence, each indicating the fraction of projects in the focal project’s foundry that are governed by one of the four particular license types used as our dependent variables.

5.3.3 Susceptibility. We operationalize susceptibility to social influence by several measures. To capture the prior success of the licensor of the focal project, we use the average size ($\text{SIZE}$) of his or her projects at the inception of the focal project at SF. The size of a project is the number of developers involved in the project. An OSS project relies on voluntary contributions from developers to survive. A project that is able to attract a larger number of developers is more likely to survive and achieve a stable release (Comino et al. 2007), a measure of technical success (Grewal et al. 2006). To capture the experience of the licensor of the focal project, we use his or her tenure ($\text{TENURE}$) in the OSS environment, measured as the number of months the licensor had been active on SF, and experience ($\text{EXP}$), measured as the number of projects in which he or she had been involved by the start of the focal project. To compute susceptibility for the focal project, we multiply these measures of success with the social proximity measures.

5.3.4 Infectiousness. We measure the infectiousness of a prior adopter by its success, where success is measured as the number of developers ($\text{ASIZE}$) involved in the project at the start time of the focal project.

5.4 Control Variables

5.4.1 Self-Confirmatory Behavior. Because of behavioral inertia, a licensor who has experience with a particular license from working on other projects may choose a similar license for their focal project. To control for such self-confirmatory behavior, we construct for versions of the variable $\text{EXPL}$, each indicating the fraction of projects in the focal licensor's prior projects that are governed by one of the four particular license types used as our dependent variables.

5.4.1 Focal Project Attractiveness. Two different types of measures of attractiveness of the focal project are calculated: (1) Licensor Attractiveness and (2) Software Characteristics Attractiveness.

   **Licensor Attractiveness.** We use three variables, defined above, to account for the attractiveness of the focal project's licensor: $\text{TENURE}$, $\text{SIZE}$, and $\text{EXP}$.

   **Software Characteristics Attractiveness.** Following Lerner and Tirole (2005b), we construct an extensive list of software characteristics. Each project is characterized by the following dimensions: Intended Audience (e.g., end users, system administrators), Topic (e.g., games, Internet), Operating System
(e.g., POSIX, Windows), User Interface (e.g., GUI, text based), and Natural Language (e.g., English, Chinese). In all, we control for 46 characteristics of a focal OSS project.

5.5 Model

The basic framework for the license choice model developed here captures both economic and social influence factors that affect a licensor’s choice of a particular license type. Let, \( y_i \) be an indicator variable that equals “1” if project \( i \) has license type L and “0” otherwise. The model is specified as:

\[
p(y_i = 1) = F(\beta X + \gamma Z + \tau D) \quad \text{and} \quad p(y_i = 1) = \frac{e^{(\beta X + \gamma Z + \tau D)}}{1 + e^{(\beta X + \gamma Z + \tau D)}}.
\]

In the above, \( X \) represents the social influence variables, \( Z \) represents the software characteristics and other control variables, \( D \) represents the licensor characteristics, and \( F \) is the logistic function. The variables in \( Z \) and \( D \) correspond to the economic incentives argument put forth by Lerner and Tirole. While the construction of \( Z \) and \( D \) is straightforward, the construction of \( X \) requires some explanation.

5.5.1 Social Influence Variables. In essence, social influence variables account for three factors: social proximity of \( j \) to \( i \), infectiousness of \( j \), and the susceptibility of \( i \) (Strang and Tuma 1993). Given a concrete measure of social proximity between a focal project, \( i \), and the projects in its social network, \( j \), the focal project licensor’s subjective perception of that proximity can be represented by a power law function (Burt 1987). Let \( P \) be the set of all projects in the focal project’s foundry that started before the focal project \( i \), and \( Y_p = \{y_1, y_2, \ldots, y_p\} \) be a vector of indicator variables representing license choice of all projects in the set \( P \). Let \( R_{ij} \) capture the facility through which \( j \) influences \( i \), moderated by the infectiousness of project \( j \). \( R_{ij} \) is calculated as follows:

\[
R_{ij} = \frac{(\text{infectiousness of } j \times \text{social proximity of } i \text{ to } j)^\upsilon}{\sum_k (\text{infectiousness of } k \times \text{social proximity of } i \text{ to } k)^\upsilon} 
\]

\[
R_i = \sum_j R_{ij}.
\]

\( R_i \) varies between zero and one. Higher values of \( R_i \) correspond to higher social influence. The exponent \( \upsilon \) measures the extent to which focal project \( i \) is conservative in relying on others. Values of \( \upsilon \) much larger than one imply that a focal project’s evaluation of a license is affected by its most proximate projects, whereas fractional values indicate that it is affected almost by anyone. Hence, \( \upsilon \) determines the social frame of reference for the focal project. Social cohesion and structural equivalence are two different
measures of social proximity that enter $R_i$. The susceptibility of the licensor to social influence is accounted for by multiplying $R_{ij}$ with licensor characteristics of project $i$:

$$X_i = [\text{TENURE}_i \times R_i, \text{SIZE}_i \times R_i, \text{EXP}_i \times R_i, R_i].$$

Here the interaction terms, $\text{TENURE}_i \times R_i$, $\text{SIZE}_i \times R_i$, and $\text{EXP}_i \times R_i$ represent susceptibility variables.

### 5.5.2 Accounting for Unobserved Heterogeneity

While we have developed an extensive list of controls besides the social influence variables, there is still scope for unobserved heterogeneity of licensors to influence license choice behavior. For example, licensors may have intrinsic preferences for particular licenses, or may have access to external source of information. Further, some individuals may be intrinsically more susceptible (infectious) than other. We allow for this unobserved heterogeneity in the license choice behavior of the licensors by allowing their coefficients for social influence and licensor characteristics to vary across them. Licensor random coefficients are introduced hierarchically. Formally, the above specified model is modified as:

$$p(y_i = 1) = F(\beta_i X_i + \gamma Z_i + \tau_i D_i),$$

and

$$\begin{bmatrix} \beta_i \\ \tau_i \end{bmatrix} = \text{MVN}(\rho, \Sigma_{\beta\tau}),$$

where $\rho$ is a vector that corresponds to the mean of $\begin{bmatrix} \beta_i \\ \tau_i \end{bmatrix}$, and $\Sigma_{\beta\tau}$ is the corresponding covariance matrix.

### 5.5.3 Accounting for Reflection Problem

A key issue in studies attempting to identify the effect of social influence is the reflection problem (Manski 1993). The reflection problem in our context would mean the licensor of a new project chooses the license chosen by projects in his or her social network not because of influence but because he or she is similar to the licensors of those projects and it is this unobserved similarity that also causes the formation of the social relationship connecting the projects. Thus, an observed relationship between social proximity and license adoption, for example, would be spurious, implying parameter estimates for social influence variables are biased by endogeneity stemming from an unobserved relationship selection process. Bramoulle et al. (2009) have suggested ways by which the true effect of social influence can be identified by accounting for the reflection problem. The solution involves introducing unobservable variables common to the projects that belong to the same network component. A component is a part of social network in which each actor can reach every other actor. The unobserved effects are allowed to be correlated with both the network structure and the license choices made by actors.
in the network. The unobservable variables are treated as component-specific fixed effects and eliminated by using appropriate differencing.

**5.5.4 Left-Censoring Bias.** In the dataset, we observe the social influence effects of prior adopters. The possibility of having social ties with prior adopters of a license is dependent on the pool of adopters. Thus, the projects that started at SourceForge close to the start of SourceForge are likely to have fewer ties compared to projects that started later. This creates concern for potential left-censoring bias. Although all projects were included for construction of a social network, for hypothesis testing, we used only those projects that were registered from January 2002 to December 2003.

**5.5.5 Sample Selection Bias.** By definition, for any project to be subjected to social influence, it has to be part of the social network. That is, it needs to have at least one tie of a social distance of 1. Of the full sample of 21,220 projects registered at SourceForge during our period of study, only 5,664 projects had at least one tie of a social distance of 1, and 5,307 projects had at least one tie of a social distance of 2. There are 15,913 projects that do not have any tie of social distance or 1 or more (i.e., these projects are isolates).

Many focal licensors who are new to SF (i.e., the focal project is the first project in which they are involved at SF) or have not worked with people who have also worked on other projects will not be subjected to the social influence arising from social cohesion or structural equivalence. Thus, the social influence variables are undefined for such projects. The way we constructed the structural equivalence measure implies that only those focal projects have a defined structural equivalence measure, which have at least one tie at a social distance of 2. Hence, the number of projects for which all social influence variables are defined is 5,307. However, not including them in the estimation may lead to sample selection bias (Heckman 1976). To control for the possibility of selection bias, we estimated a first-stage selection model for all focal projects (21,220) using an instrument and entered the inverse mills ratio into the license choice models. The ratio essentially controls for the probability that the focal project is started by a licensor who has worked with people who have worked on other projects.

The first stage used variables that would correlate with a licensor’s likelihood of working with people who have worked on other projects. The first stage estimated the selection hazard as a function of all non-social influence factors along with the technical expertise of the licensor and network density of the focal project's foundry as instruments. To measure technical expertise, we used the licensor’s self-reported
level of expertise reported to SF. The category indicating the lowest level of expertise is “want to learn,”
which we coded as “1” if the developer selected that category, and “0” otherwise. The want to learn
category developers are more likely to be new to OSS and, thus, less likely to be part of the existing social
network of projects at SF. Network density is the extent to which all the projects within a foundry are
connected with each other. Higher density implies a higher level of interconnection. Individuals who are
part of dense networks are considered more likely to experiment and produce new innovations due to the
trust and support that network density provides (Uzzi and Spiro 2005). We expect that as network density
increases, the likelihood a new project is started by a developer from within the network also increases.

5.5.6 Estimation Procedure. The estimation procedure consists of two steps. In step one, a selection
model is estimated as a Probit regression model. In step two, four different license choice models are
estimated separately and include the inverse Mills ratio obtained from step one as one of the covariates.
The license choice models are hierarchical Bayes Logit models and account for the reflection problem.
Each license choice model is estimated through a standard Markov Chain Monte Carlo (MCMC)
hierarchical Bayes estimation procedure, using a Gibbs Sampler and the Metropolis-Hastings algorithm
coded in Matlab (Rossi and Allenby 2003). To reduce the autocorrelation between draws of the Metropolis-
Hastings algorithm and to improve the mixing of the MCMC, we used an adaptive Metropolis adjusted
Langevin algorithm (Atchade 2006). In the hierarchical Bayes procedure, the first 100,000 observations
were used as burn-in, and the last 25,000 were used to calculate the conditional posterior distributions. To
assess the convergence of the MCMC, we compared the within- to between-variance for each parameter
estimated across multiple chains (Gelman and Rubin 1992). The full estimation procedure is provided in an
on-line supplement to this paper.

6. Results
Descriptive statistics for the key variables are reported in Table 1. Initial investigations revealed that SIZE,
TENURE, and EXP were heavily skewed, which may lead to spurious correlation. To address this concern,
we followed Greene (2003) and log-transformed the variables. Further investigations revealed that the
interaction terms were highly correlated with their component terms. This may raise concerns of
 multicollinearity, which, if uncorrected, may lead to inconsistent and inefficient estimates. As suggested by
Gelman and Hill (2007), we standardized the component variables (SIZE, TENURE, EXP, social cohesion,
and structural equivalence). The estimated model includes these standardized variables and their interactions. This step reduced the correlation to acceptable levels: variance inflation factors (VIFs) for all variables in all models were below the threshold value of 10. The correlations of these transformed variables are provided in the on-line supplement.

The selection model results are presented in Table 6. To conserve space, software characteristics and time period effects, while estimated, are not reported. Both technical experience \((-0.171, \ p < .001)\) and network density \((2.457, \ p < .001)\) significantly predict whether the licensor of the new project has ties to the existing network of length greater than 1.

| Table 1. Descriptive Statistics \((N = 5,307\) observations) |
|---------------------------------|-----------------|---------------|---------|---------|
| Variables                       | Mean | S.D. | Min | Max |
| 1. All licenses highly restrictive (ALLHR) | 0.55 | 0.5 | 0 | 1 |
| 2. Some licenses highly restrictive (SHR) | 0.6 | 0.49 | 0 | 1 |
| 3. All licenses restrictive (ALLR) | 0.73 | 0.44 | 0 | 1 |
| 4. Some licenses restrictive (SR) | 0.76 | 0.43 | 0 | 1 |
| 5. Social Cohesion ALLHR | 0.54 | 0.23 | 0 | 1 |
| 6. Social Cohesion SHR | 0.64 | 0.23 | 0 | 1 |
| 7. Social Cohesion ALLR | 0.54 | 0.2 | 0 | 1 |
| 8. Social Cohesion SR | 0.45 | 0.25 | 0 | 1 |
| 9. Structural Equivalence ALLHR | 0.18 | 0.26 | 0 | 1 |
| 10. Structural Equivalence SHR | 0.2 | 0.28 | 0 | 1 |
| 11. Structural Equivalence ALLR | 0.51 | 0.48 | 0 | 1 |
| 12. Structural Equivalence SR | 0.52 | 0.48 | 0 | 1 |
| 17. Technical Equivalence ALLHR | 0.58 | 0.1 | 0 | 0.76 |
| 18. Technical Equivalence SHR | 0.74 | 0.1 | 0.4 | 0.96 |
| 19. Technical Equivalence ALLR | 0.77 | 0.06 | 0.5 | 0.89 |
| 20. Technical Equivalence SR | 0.87 | 0.05 | 0.5 | 0.98 |
| 21. Licensor's experience on OSS projects \((EXP)\) | 1.94 | 1.69 | 1 | 15 |
| 22. Average size of Licensor's prior projects \((SIZE)\) | 2.03 | 1.14 | 0.69 | 6.34 |
| 23. Licensor's prior exp with this license \((EXPL-ALLHR)\) | 0.50 | 0.47 | 0 | 1 |
| 24. Licensor's prior exp with this license \((EXPL-SHR)\) | 0.65 | 0.44 | 0 | 1 |
| 25. Licensor's prior exp with this license \((EXPL-ALLR)\) | 0.71 | 0.42 | 0 | 1 |
| 26. Licensor's prior exp with this license \((EXPL-SR)\) | 0.81 | 0.36 | 0 | 1 |

Before we explain the license choice results, we need to explain the procedure that we followed for estimating the parameter \(\nu\), which is used to construct social influence variables. Following the methodology suggested by Burt (1987), this parameter is treated as a constant when calculating the value of social proximity to be used in the models. To identify the optimal value of \(\nu\), we compared likelihoods for
different values of $\nu$. We calculated the social proximity measures for different values of parameter $\nu$ (0.1, 0.25, 0.5, 0.75, 1, 1.5, 2, 2.5, and 5) and then chose the value that provided the best likelihood. Among these values, the value of $\nu = 1$ for both Social Cohesion and Structural Equivalence provided the best likelihood. This value indicates that the licensor gives more weight to the most proximate prior adopters. The analysis in this paper presented henceforth is for social proximity measures calculated by using $\nu = 1$.

The results of the license choice model are provided in Table 3. Coefficient posterior means and variances are reported. Model 1 tests for the probability that all the licenses chosen by the licensor for the focal project are highly restrictive. Model 2 tests the probability that some of the licenses are highly restrictive. Models 3 and 4 test the probabilities for all and some licenses being restrictive, respectively. To conserve space, software characteristics and time period effects, while estimated, are not reported. Standardized coefficients are reported for all the models for comparison purposes. As one can observe, there is a significant amount of unobserved licensor specific variance.

<table>
<thead>
<tr>
<th>Variables</th>
<th>Coefficients</th>
<th>Standard Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Want to Learn</td>
<td>-0.171***</td>
<td>0.05</td>
</tr>
<tr>
<td>Network Density</td>
<td>2.457***</td>
<td>0.17</td>
</tr>
<tr>
<td>Software Characteristics &amp; Time Controls</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Number of Projects</td>
<td>21220</td>
<td></td>
</tr>
</tbody>
</table>

*** $p < .01$, ** $p < .05$, * $p < 0.1$. Software Characteristics and Time dummies while included in the estimation are not shown to conserve space.

Consistent with hypotheses 1 and 2, the coefficients for the social cohesion and the structural equivalence terms are positive and significant in all models. Among the two social proximity effects, social cohesion effects appear to dominate the structural equivalence effects. This indicates that licensors pay more attention to direct than to symbolic communication in their social network. Another equivalence measure, fraction of projects in the foundry using the focal license type, has a much larger effect than does the social network-based structural equivalence measure. This indicates that a licensor is more influenced by the choices of technically similar projects than by the choices of structurally similar projects. This effect provides support for Hypothesis 3.
The interactions of the social proximity measures are mostly negative and significant across all models, which is consistent with hypotheses 5 and 6. This implies that focal licensors who are experienced and successful are less susceptible to social influence. The estimated coefficients for social cohesion, structural equivalence, \( EXP \), \( TENURE \), and \( SIZE \) are simple effects rather than main effects, as the interaction terms are significant (Jaccard and Turrisi 2003). To better understand the meaning of these effects, we need to understand the structure of these variables as used in the models in Table 2. To assess the net effects of any variable with interactions on the probability of choosing a certain license type, the main and the interaction effects must be combined. Using the results from Model 1, the effects for social cohesion are given as:

\[
p = \Phi(1.634XSocialCohesion - 0.451XSocialCohesionXTENURE - 0.260XSocialCohesionXSIZE - 0.241XSocialCohesionXTENURE).
\]

The coefficient estimate of 1.634 for social cohesion is conditional on \( EXP \), \( TENURE \), and \( SIZE \) taking on the value of zero (thus removing the effects of interaction with \( SIZE \), \( TENURE \), and \( EXP \)), while the effect of Social Cohesion when \( SIZE \), \( TENURE \), and \( EXP \) are not zero will depend on their values and the values of their coefficients. For example, when \( EXP \) and \( SIZE \) are at their means, the effect of a one-unit change in social cohesion decreases as the value of \( TENURE \) increases, is positive for values of \( TENURE \) less than 6.78 (i.e., 1.634/0.241), and becomes negative above that value.

Hypothesis 4 predicted that prior adopters differ in their infectiousness based on their success. To test this prediction, we needed to construct different influence structures \( (R_i) \). We constructed \( R_i \) in two different ways: (1) non-weighted social proximity (i.e., set infectiousness to 1 for all prior adopters) and (2) \( ASIZE \)-weighted social proximity (replace infectiousness by \( ASIZE \) of prior adopters). Because there are two different social proximity measures, we have four different models. As one may notice, these are non-nested models. Thus, as suggested by Newton and Raftery (1994), we compare these models on log marginal density. The Bayesian Information Criterion (BIC), commonly used for model selection in classical statistical applications, asymptotically approximates the Bayesian posterior marginal density. Based on this measure, the best fitting model is the model that minimizes -2 log marginal density. The models (each with a specific construction of \( R_i \)) were run separately and their log marginal densities were calculated. The -2 log marginal density is reported in Table 4. For all four dependent variables, the models
that account for infectiousness of prior adopters based on their \textit{ASIZE} have the lowest -2 log marginal density among their competing models. This indicates that more successful prior adopters are more infectious. This provides support for Hypothesis 4.

\textbf{Table 3. Hierarchical Bayes Parameter Estimates for Logit License Choice Model}

<table>
<thead>
<tr>
<th>Variables</th>
<th>Model 1</th>
<th>Model 2</th>
<th>Model 3</th>
<th>Model 4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>All Licenses</td>
<td>Some Licenses</td>
<td>All Licenses</td>
<td>Some Licenses</td>
</tr>
<tr>
<td></td>
<td>Highly Restrictive?</td>
<td>Highly Restrictive?</td>
<td>Restrictive?</td>
<td>Restrictive?</td>
</tr>
<tr>
<td></td>
<td>Coefficient</td>
<td>Posterior Mean</td>
<td>Coefficient</td>
<td>Posterior Mean</td>
</tr>
<tr>
<td></td>
<td>Mean</td>
<td></td>
<td>Mean</td>
<td></td>
</tr>
<tr>
<td>Social Cohesion</td>
<td>1.634***</td>
<td>1.761***</td>
<td>2.341***</td>
<td>1.737***</td>
</tr>
<tr>
<td></td>
<td>[0.311]</td>
<td>[0.355]</td>
<td>[0.469]</td>
<td>[0.314]</td>
</tr>
<tr>
<td>Structural Equivalence</td>
<td>0.116**</td>
<td>0.108**</td>
<td>0.170**</td>
<td>0.144**</td>
</tr>
<tr>
<td></td>
<td>[0.109]</td>
<td>[0.352]</td>
<td>[0.261]</td>
<td>[0.213]</td>
</tr>
<tr>
<td>Licensor's experience on OSS projects (EXP)</td>
<td>-0.008</td>
<td>-0.012</td>
<td>-0.092**</td>
<td>-0.053**</td>
</tr>
<tr>
<td></td>
<td>[0.011]</td>
<td>[0.024]</td>
<td>[0.009]</td>
<td>[0.022]</td>
</tr>
<tr>
<td>Average size of Licensor's prior projects (Size)</td>
<td>-0.426***</td>
<td>-0.537***</td>
<td>-0.372***</td>
<td>-0.353***</td>
</tr>
<tr>
<td></td>
<td>[0.215]</td>
<td>[0.195]</td>
<td>[0.204]</td>
<td>[0.179]</td>
</tr>
<tr>
<td>Licensor's tenure at SourceForge (Tenure)</td>
<td>-1.388***</td>
<td>-1.359***</td>
<td>-1.184***</td>
<td>1.175**</td>
</tr>
<tr>
<td></td>
<td>[0.352]</td>
<td>[0.206]</td>
<td>[0.105]</td>
<td>[0.045]</td>
</tr>
<tr>
<td>Social Cohesion X Exp</td>
<td>-0.451**</td>
<td>-0.515***</td>
<td>-0.591***</td>
<td>-0.356***</td>
</tr>
<tr>
<td></td>
<td>[0.211]</td>
<td>[0.301]</td>
<td>[0.314]</td>
<td>[0.231]</td>
</tr>
<tr>
<td>Social Cohesion X Size</td>
<td>-0.260***</td>
<td>-0.103*</td>
<td>-0.248***</td>
<td>-0.091</td>
</tr>
<tr>
<td></td>
<td>[0.112]</td>
<td>[0.043]</td>
<td>[0.106]</td>
<td>[0.098]</td>
</tr>
<tr>
<td>Social Cohesion X Tenure</td>
<td>-0.241*</td>
<td>-0.165*</td>
<td>-0.523**</td>
<td>-0.067</td>
</tr>
<tr>
<td></td>
<td>[0.113]</td>
<td>[0.101]</td>
<td>[0.353]</td>
<td>[0.042]</td>
</tr>
<tr>
<td>Structural Equivalence X Exp</td>
<td>-0.034*</td>
<td>-0.038*</td>
<td>-0.001</td>
<td>-0.029*</td>
</tr>
<tr>
<td></td>
<td>[0.015]</td>
<td>[0.071]</td>
<td>[0.013]</td>
<td>[0.012]</td>
</tr>
<tr>
<td>Structural Equivalence X Size</td>
<td>-0.025</td>
<td>0.003</td>
<td>-0.051**</td>
<td>-0.091**</td>
</tr>
<tr>
<td></td>
<td>[0.012]</td>
<td>[0.002]</td>
<td>[0.029]</td>
<td>[0.041]</td>
</tr>
<tr>
<td>Structural Equivalence X Tenure</td>
<td>-0.014</td>
<td>-0.034</td>
<td>0.017</td>
<td>-0.014</td>
</tr>
<tr>
<td></td>
<td>[0.012]</td>
<td>[0.002]</td>
<td>[0.013]</td>
<td>[0.003]</td>
</tr>
<tr>
<td>Fraction of projects in the foundry with this license (Technical Equivalence)</td>
<td>1.126***</td>
<td>1.192***</td>
<td>1.848***</td>
<td>1.396***</td>
</tr>
<tr>
<td></td>
<td>[0.421]</td>
<td>[0.330]</td>
<td>[0.221]</td>
<td>[0.232]</td>
</tr>
<tr>
<td>Licensor's prior experience with this license</td>
<td>0.637***</td>
<td>0.613***</td>
<td>0.774**</td>
<td>0.897*</td>
</tr>
<tr>
<td>(EXPL)</td>
<td>[0.164]</td>
<td>[0.206]</td>
<td>[0.253]</td>
<td>[0.142]</td>
</tr>
<tr>
<td>Number of Projects</td>
<td>5307</td>
<td>5307</td>
<td>5307</td>
<td>5307</td>
</tr>
</tbody>
</table>

Posterior licensor specific variance (corresponding diagonal element of \( \Sigma_{\theta} \)) is given in square parenthesis below the coefficient mean; \(**\) implies that the 99\% confidence interval does not include zero. \(*\) implies that the 95\% confidence interval does not include zero; \(*\) implies that the 90\% confidence interval does not include zero. All models include Software characteristics and Time dummy control variables. Social Cohesion and Structural Equivalence measures account for infectiousness based on size.
Finally, a comparison of effect sizes is also instructive. By comparing the standardized coefficients in Table 3, the major determinants of license choice can be identified. The three most important characteristics that determine the type of license used for governing an OSS project are the type of licenses used by the closest projects in the social network of the licensor, the dominant license type in the project’s foundry, and the license type with which the licensor has experience.

<table>
<thead>
<tr>
<th>Table 4. Choosing Competing Models (-2 log marginal density)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Social Cohesion</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

6.1 Robustness Checks

We performed several robustness checks. We reran the analysis by constructing the structural equivalence measure including all alters that were at a distance of 1 or more from the focal project (resulting in a sample size of 5,664). We also reran the analysis by including all developers associated with a project, regardless of whether their active involvement could be established. The results from these analyses were consistent with the results in Table 3. Another concern is that the number of contributing developers (i.e., $SIZE$) may not be an appropriate measure of a project's success. Accordingly, we used an alternative measure of project success defined as a project’s SourceForge Rank. At the end of every month, SourceForge ranks hosted projects based on their activity and popularity among developers and users. This measure was highly correlated with project size. Thus, it is not surprising that the results obtained from using this alternative measure were consistent with those using size.

7. Discussion & Conclusions

Prior research that examines the choice of OSS license scope has modeled the decision problem facing a licensor as an optimizing balancing act between providing sufficient incentives to attract numerous high quality developers to the project, while preserving the licensor’s ability to commercialize the software (e.g., Lerner and Tirole 2005b). While this research has yielded fruitful insights, it ignores the substantial uncertainty facing project administrators when choosing a license and ignores the potential influence of...
their social networks on their licensing decision. Because research suggests managers turn to their social networks for help in determining the appropriate choice when faced with considerable decision uncertainty (e.g., Buskens 2002, Rangan 2005), we sought to complement extant research by investigating the role that a project manager’s social network plays in influencing her adoption of a particular license to govern a new OSS project.

We investigated this question through the theoretical lens of the heterogeneous diffusion model from sociology (Strang and Tuma 1993). This model allowed us to examine how the decision to adopt a particular type of OSS license is shaped not only by economic incentives but also by the license choices made by other projects to which a licensor is socially connected. We characterized specific OSS licenses as discrete, innovative governance practices that were at risk of being adopted by new OSS projects. After controlling for the economic incentives explanations put forth by Lerner and Tirole (2005b), our findings reveal that the most important factor that determines which license will be adopted by a new project is the type of license chosen by licensors more socially proximate in the social network of a new project’s licensor. Moreover, the likelihood that a new OSS project adopts a particular license increases when more similar OSS projects have previously adopted such a license and when these projects are large and successful. We also found that project managers with longer tenures in the OSS community and who have previously been members of successful OSS projects are less susceptible to social influence. This suggests that project managers who are inexperienced with OSS or who have contributed to unsuccessful OSS projects who are most influenced by their social context. Finally, the results indicate that social influence dominates economic incentives in the choice of license. This suggests that, on average, licensors of OSS projects are highly uncertain about the implications of different licenses. With experience and success on prior projects, some of this uncertainty is resolved and licensors become less susceptible to social influence. These results have important implications for research into the choice of contractual governance, particularly in the OSS setting, the economic viability of the so-called OSS movement, and research on innovation diffusion.

This study contributes to the literature on contractual governance of information technology projects. Efficiency-based, economic incentive explanations dominate the IT governance literature and the broader literature on contractual governance. In contrast to this research, our results suggest social context
strongly influences the scope of contractual governance chosen when licensors are uncertain about the
performance implications of key contractual terms. These socially-influenced governance choices have
implications for understanding the performance of OSS projects and their economic viability.

Given the growing popularity and importance of OSS projects and the influence a project’s license
has on its performance, our results suggest the viability of the OSS movement may be constrained to the
extent social context influences licensors to adopt licenses based on a logic of social influence rather than
economic maximization. While licensing has important long-term implications for the success of OSS
projects (Comino et al. 2007, Lerner and Tirole 2005b, Stewart et al. 2006), our results suggest the licensor
does not know the optimal license when he or she starts the project. Once the licensor starts a project and
chooses a license, the implications may start becoming clearer over time, but it is already too late to change
the license if significant work has been done. Similarly, the long-term social welfare impact of OSS and
thus its viability is contingent on the licenses that are chosen to govern the software. Uncertainty about the
implications of available OSS licenses appears to be driving licensors to be strongly influenced by the
license choices of their peers. However, this adoption logic, if sufficiently widespread, may lead to reduced
social welfare since it suggests economically inefficient licenses may be more widely adopted than more
efficient licenses to govern OSS projects. This may threaten the long-run viability of the OSS model.
Research examining the mechanisms that can reduce OSS licensor uncertainty can provide useful policy
implications for increasing OSS project performance and the long-run economic viability of the OSS
model. Such research is sorely needed. These results reinforce prior work in sociology demonstrating that
economic action is embedded in and influenced by concrete social relationships among economic agents
and that the social embeddedness of economic agents influences the efficiency of economic activity
(Granovetter 1985).

This study also contributes to the innovation diffusion literature. Although thousands of studies
have investigated factors that influence the adoption and diffusion of technical innovations (Rogers 1995),
research into the adoption of innovative administrative practices, such as novel governance practices, is
rare (Wejnert 2002). Fewer studies still examine how social structure affects the adoption and diffusion of
innovative administrative practices (Davis and Greve 1997). Our study contributes to the innovation
diffusion literature by showing how the social structure in which a licensor is embedded influences the
adoption of a novel governance practice. Moreover, most diffusion studies that examine the influence of social proximity assume proximity alone influences adoption or that social proximity has a homogenous effect across all members of the population under study. In contrast, in specifying a heterogeneous diffusion model, we allow for spatial heterogeneity in the influence of social proximity on the choice of an OSS license by constructing licensor-specific proximity measures. We accommodate temporal heterogeneity by allowing the influence of prior adopters to vary over time. Finally, by examining two different forms of social proximity –social contact (i.e., cohesion) and structural equivalence – we are able to untangle the relative strength of the influence of communication, associated with social cohesion, and social comparison, associated with structural equivalence, on adoption. Few studies of the adoption and diffusion of innovative practices and no study, of which we are aware, of licensing choice have done this.

References


