The influence of institutions on organizational performance via network structure

Regarding the institutional context of the Open Source community

Authors: Elmira van den Broek (3969088) & Irene Westra (3973530)
Conclusion and Discussion by Elmira van den Broek (3969088)
Department of Sociology, Faculty of Social Sciences,
University of Utrecht, the Netherlands

Abstract

This research focusses on the influence of institutions on organizational performance via network structure. Two meta-hypotheses are tested: (1) institutions influence network structures and (2) network structures influence organizational performance. By applying the idea of institutions and organizations to the Open Source (OS) community, it is expected that restrictiveness of licenses influence the network structure of OS projects, and network structure influences process- and outcome-based performance of OS projects. Empirical analyses are performed using the *SourceForge* database, a compilation of more than 30,000 active OS projects. The theories of Burt (2000, 2005) and Coleman (1990) are used in explaining the influence of levels of brokerage and closure as indicators of network structure. Findings show that restrictiveness of licenses is a negative predictor of both levels of brokerage and closure. Furthermore, levels of brokerage predict higher levels of both process-based and outcome-based performance. Levels of closure show no distinct pattern in explaining performance of OS projects. Higher levels of process-based performance are an indicator of higher overall performance outcomes. The findings are broadly consistent with our theoretical framework.

Preface

We, Elmira van den Broek and Irene Westra, started both our bachelor Sociology at Utrecht University in September 2012. Three years later, after positive experiences of teamwork, we were more than ready to write our bachelor thesis together. The dataset on Open Source projects of David Macro draw our attention, since both of us are interested in applying sociological knowledge to new, evolving social contexts. Writing our bachelor thesis was an interesting and challenging experience, were we sought to apply as much of our skills as possible. With our diverse backgrounds, Irene working at Utrecht Data School and Elmira also studying Economic and Business Economic, we were able to practice a variety of knowledge. Though the easiest way was not always chosen and some struggles did exist, we are very proud on what we achieved with this thesis. We would like to thank David Macro in particular, for all of his guidance and challenging input during the meetings. Thank you for all of the valuable advice and encouraging ideas.

Table of contents

1. Introduction

Organizations can be viewed as open systems, strongly influenced by their environments. Institutions shape these environments, by not only providing competitive and efficiency-based forces, but also socially constructed belief and rule systems (Scott, 2003). The rule systems consist of formal and informal constraints, that influence both internal structure and economic performance (North 1990; Scott 2003). Gaining insight in how institutions structure incentives in human exchange can increase understanding on how organizations evolve through time and how they carry out their work. In this study the nature of institutions and the consequences of institutions for organizational structure and economic performance will be examined from a social network perspective. We expect that institutions influence economic outcomes by shaping the social structure of organizations (Granovetter, 2005).

The Open Source (OS) community provides a new, interesting opportunity for investigating the influence of institutions on economic performance. The OS community is a rapidly evolving movement, where information, knowledge, and resources flow more freely across boundaries (Singh, 2010). In recent years, this movement has permitted users to take advantage of the freely accessible software and developers to modify source code (Kapitsaki, Tselikas & Foukarakis, 2015). By participating concurrently across several OS projects, OS developers create a network structures of projects of the OS community. They create channels between different projects, facilitating the flow of information, knowledge, and resources (Singh, 2010). However, OS software of projects have to be provided under a license, with each license granting certain freedoms and boundaries to the developers. Licenses can be viewed as the "most important institution in the governance structure of OS projects" (Bonaccorsi & Rossi, 2003: 1248). This may have important implications for the ties between OS projects, established by developers. By applying the idea of institutions and organizations to the OS community, we will investigate whether the restrictiveness of licenses influence the network structure of OS projects, and the consequences of this structure for performance of OS projects.

Among others, Granovetter (2005) and Uzzi (1996) gave an important role to social structures in explaining the influence of institutions on economic performance. However, examining these relations for the OS community has been less common. Previous research regarding the OS community has focussed in particular on the determinants of OS license choice, for example by looking at the motives of individual licensors or the community of developers

(Lerner & Tirole, 2005; Henley & Kemp, 2008; Sen, Subramaniam & Nelson, 2011). These license choices of individual licensors or developers shape OS projects, since OS licenses provides certain rules imposed from the original creator by copyright law (Everts, 2000).

Less research has been carried out on the influence of licenses on the success or failure of OS projects. Past studies on success of OS projects include Lerner and Tirole (2005), who showed that projects with less restrictive licenses tend to attract more contributors. Also, Subramaniam, Sen and Nelson (2009) and Chen (2010) argued that restrictiveness of OS licenses negatively predicts project success. Additionally, Stewart, Ammeter and Maruping (2006) looked at the impact of restrictiveness of licenses and organizational sponsorship on two indicators of project success: user interest and development activity. They showed that users are most attracted by unrestrictive licenses and that the influence of licensing on development activity depends on what kind of organizational sponsor a project has. However, former research scarcely studied the possible mechanism between the restrictiveness of licenses and performance. We expect that network structure might play an important role in explaining this relation.

A small amount of studies has applied a social network approach in explaining OS performance. For example, Cowan and Jonard (2003) showed that system performance of OS groups is strongly predicted by the communication network structure. Further, Grewal, Lilien and Mallapragada (2006) found that network embeddedness is a strong predictor of both technical and commercial success, but these effects are complex and need further investigation. In line with Grewal, Lilien and Mallapragada (2006), Singh, Tan and Mookerjee (2008) showed that both direct and indirect ties between team members positively influence the productivity of OS projects, and that the greater the cohesive ties between team members, the more productive they are. Additionally, Singh (2010) found that small-world properties of a community play an important role in explaining both technical and commercial success of the software produced by its members. These studies show that network structures play an important role in explaining performance of OS projects. By including the influence of social networks on performance, better insights can be provided in explaining the relation between institutions and organizational performance.

Our study contributes to the existing literature on institutions by offering a mediating role for network structure in explaining the influence of institutions on organizational performance. Thus, the main research question of our study is: “How do institutions affect organizational performance, via network structure?”

The institutional context of the OS community will be used to provide insights in the role of institutions on organizational performance in a new, rapidly evolving environment. The relation between institutions, network structure, and organizational performance is unique in existing studies regarding the institutional context of the OS community. Therefore, our study can make important contributions to existing scientific research related to this topic. Further, by performing Structural Equation Modelling (SEM) analysis additional to Ordinary Least Squares (OLS) regression analyses, insights will be provided in the direct, indirect and total effects of restrictiveness of licenses on performance. This results in a broader view on the influence of institutions on organizational performance.

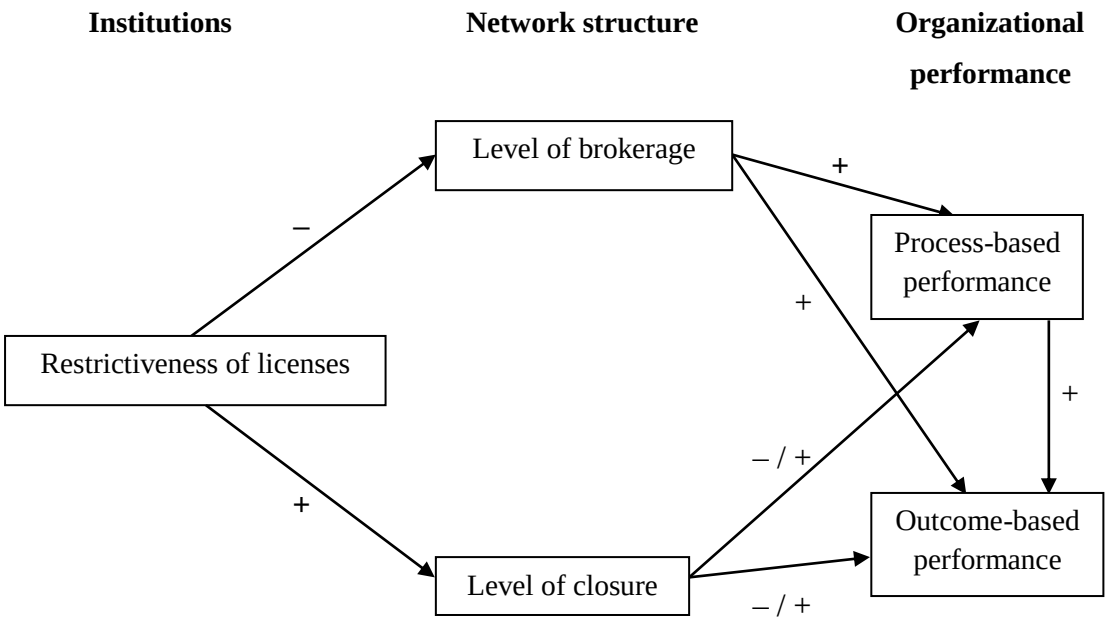
Furthermore, this study can have practical implications for governments, since governments try to influence network structures with institutions (Granovetter, 1985). Information on how institutions affect economic performance can offer input for policy proposals on the role of the government in forming institutions. For example, benefits of restiveness of licenses can provide input on how to shape property rights for stimulating organizational performance. Practitioners wishing to improve organizational performance can take advantage of this extended knowledge and apply this to organizations.

Our paper is structured as followed. First, we provide a brief discussion on the definition of performance and the relation between institutions and licenses. Second, we develop hypotheses by drawing on the views of Burt (2000, 2005) and Coleman (1990) on sources of social capital. Third the empirical methodology used will be described. OLS regression analyses and SEM analysis will be performed. After this, the obtained results will be reviewed and discussed. Finally, we review the limitations of this study and offer input for future research.

2. Theory and Hypotheses

This section introduces two meta-hypotheses that form the general focus in this research: institutions influence network structures and network structures influence organizational performance. To establish these relations, we will start with summing up different indicators of performance. Second, we will argue why licenses are a form of institutions. Thereafter predictions on the relations between types of licenses and performance will be formulated. To ground these expectations, we will draw on influential theories in research on social network effects: Burt's (2000, 2005) theory of structural holes and Coleman's theory (1990) on network closeness will be used.

Figure 1. Predicted relations between institutions, network structure and performance for the OS community.



2.1. Indicators of performance

To study the influence on performance, it is important to have a clear definition of this concept. Previous studies have investigated possible performance indicators for information systems (DeLone & McLean, 2002; Seddon, 1997). Crowston, Annabi and Howison (2003) built on this by complementing previous research on information systems with information on OS systems. This led to a final overview from possible performance indicators for OS systems. In our research will we focus on two groups of performance indicators: indicators on the process level and on the outcome level. The foundation of these dimensions will now be discussed.

In their study, Crowston et al. (2003) combined DeLone and McLean's (1992, 2002, 2003) model and Seddon's (1997) model on information system (IS) performance. They presented four possible success indicators: system and information quality (code and documentation quality), user satisfaction (among others user ratings and opinions in mailing lists), usage (among others the number of downloads and views of information page) and finally individual and organizational impacts (Crowston et al., 2003). Crowston et al. (2003) mentioned two important shortcomings of these success indicators. First of all, it is argued that these indicators do not take into account the *process* of a project, but only focus on the *use* and *use environment* of the software. This may be appropriate for information systems,

but not for OS systems. Second, it is stated that the IS performance model reasons from the perspective of a publicly visible use environment, instead of a development environment. The opposite is mostly true for the context of OS systems, where the use environment is difficult to study (Crowston et al., 2003).

Therefore, Crowston et al. (2003) came up with an additional re-examined literature model that including indicators of the process of developing OS systems. These indicators can be distinguished in three groups: (1) indicators of the output of system development, (2) indicators of the process of system development and (3) indicators of outcomes for project members. First, the output of system development can be indicated by the progress of a project from alpha to beta to stable status as self-reported in *SourceForge* (Crowston & Scozzi, 2002). An alternative measure is developer satisfaction with help of survey data, as used by the research of Scacchi (2002). Second, the process of system development can be reflected by the level of activity of a project, thus the contribution of developers to a project. Examples, are submitting code and bug reports of developers. Alternatively, the number of developers can be an indicator of success, as many OS system projects depend on volunteers with no monetary incentives (Crowston et al., 2003). Finally, outcomes for project members can be indicated by among others employment opportunities or the ability of providing salaries for project members (Lerner &Tirole, 2002; Roberts, Hann & Slaughter, 2006).

Based on the IS performance model and the re-examined model of Crowston et al. (2003) three groups of performance indicators can be distinguished: indicators concerning the *outcome* of a project (among others the number of users and downloads), indicators concerning the *development process* of a project (among others the level of activity of a project) and indicators of *developer outcomes* (expressed by developer satisfaction and job opportunities). According to Crowston et al. (2003) it is important to use not a single indicator of success, but a portfolio of indicators. This is because many of the indicators seem likely to show measurement problems, and the use of only one indicator of performance could therefore lead to unreliable results. Because our research will focus on the project level instead of developer level, we choose to focus on two of the three distinguished groups of performance: *process-based* indicators and *outcome-based* indicators.

2.2. Institutions and licenses

After having defined OS-performance, we will explain why licenses can be regarded as a form of institutions. Institutions in general can be defined as the rules of the game in society or, more formally, “the humanly devised constraints that shape human interaction” (North,

1990: 3). Institutions shape organizations by providing political, social and economic frameworks. This shows the important distinction between institutions and organizations: while institutions define the rules of the game, organizations are the players (Mantzavinos, North & Shariq, 2004).

North (1990) distinguishes formal and informal institutions. Formal constraints are rules that human beings devise, such as constitutions, laws and property rights. These rules have legal consequences for offenders. Informal constraints are conventions and codes of behaviour, for example, common norms, sanctions, taboos, traditions and codes of conduct. They are not centrally designed or enforced by legal institutions (North, 1991 in Williamson, 2000). Formal and informal constraints can be compared with the rules of the game in a competitive team sport: both formal written rules as unwritten codes of conduct underlie and supplement formal rules that play a role in the game. These constraints play as major role in reducing uncertainty in society, by providing a stable structure to human interaction. In other words, institutions provide predictability of behaviour. Uncertainty in society arises because actors frequently must act on incomplete information and process the information that they do receive through limited mental constructs (North, 1990). When uncertainty arises, costs arise because of actions actors take in examining the multiple valuable dimensions involved in exchange – broadly, information costs – and because of enforcing agreements. These costs are referred to as transaction costs (North, 1992). Coase (1960) pointed out that markets can only be efficient in the absence of those transaction costs. When transaction costs are significant and positive, institutions are needed to provide low-cost transacting and provide efficient economic growth. By reducing uncertainty in society, institutions can increase attractiveness of relation-specific investments and therefore stimulate performance (Zaheer & Venkatraman, 1995).

Licenses can be viewed as a form of institutions, since they define the rules of the game for the OS community. They can be referred to as the most important institution in the governance structure of OS projects (Bonaccorsi & Rossi, 2003). First of all, licenses propose formal constraints indicated by levels of restrictiveness of licenses. For example, highly restrictive licenses prohibit the mixing of open- and closed-source software in any distributed networks. This results in restrictions on behaviour of developers. Second, informal constraints are presented in the ideological views of the licenses. The programmer Richard Stallman played a major part, by emphasizing the importance of the rights of software users to freely learn and create. His movement was based on philosophical grounds, such as the moral rightness and importance of offering free and open source software for users (von Hippel &

von Krogh, 2003). Stallman's moral view is reflected in the highly restrictive General Public License (GPL), and helps creating a common code of behaviour which motivates and shapes the behaviour of participants of the OS community. Licenses therefore provide a useful institutional framework for developer communities and help to structure behaviour. In further sections, we will regard licenses as a form of institutions.

2.3. Restrictiveness of licenses

OS software is defined by its attached license which commits essential rights imposed from the original creator by copyright law (Everts, 2000). This can be viewed as ironic, given that OS was founded as a collectivist and anti-intellectual property movement (Fitzgerald, 2006). In this study, the focus will be on the three most used licenses: General Public License (GPL), Lesser/Library General License (LGPL) and Berkely Software Definition (BSD) (Lerner & Tirole, 2005). In explaining these licenses we distinct between highly restrictive (GPL), modestly restrictive (LGPL) and unrestrictive (BSD) licenses (Lerner and Tirole, 2005).

2.3.1. Highly restrictive license

The GPL is the first OS license and became widespread in the 80s. The license is restrictive in two ways: the license (1) insists that "any derivative work remained subject to the same license" and (2) prohibits "the mixing of open and closed source software in any distributed works" (Lerner & Tirole, 2005: 23). Coming up with a license that prevented misuse was of importance for the founder of GPL, Richard Stallman. Misuse of programs was frequent, and involved modifying codes a little and turning them into proprietary software (Evers, 2000). The restrictions imposed by the GPL-license lower this risk of commercial exploitation by prescribing that all enhancements to GPL licensed code – even when code intermingled the cooperatively developed software with that developed separately – have to be licensed under the same terms (Lerner & Tirole, 2002). Furthermore, the rules for redistribution create a situation where people cannot use OS software without paying back to the community, because they have to make their modifications available to the OS community for free use. This method is called copyleft (Evers, 2000). The strict characteristics of GPL lead to naming this license 'highly restrictive'.

2.3.2. Modestly restrictive license

Closely related to the GPL is the LGPL. The LGPL was developed since many viewed the restrictiveness of GPL to be impractical. LGPL was first intended to use for software libraries,

hence it is also known as Library GPL (Fitzgerald, 2006). LGPL offers an practical alternative to GPL, because it allows mixing of programs between open source and non-open source licenses (Lerner & Tirole, 2005). This means that when a developer modifies a LGPL library, like C library, the modified version must be LGPL–licensed. However, when a developer creates an application that relies on a LGPL library the application can remain closed, it only has to be dynamically linked to the LGPL library (CNXSoft, 2015). Dynamic linking means that actual linking with library routines does not occur until the program is run, when both the executable and the library are placed in memory (Indiana University, 2013). Benefits are for instance that dynamically linked shared libraries are easier to create and easier to update in comparison to static linked shared libraries (Shared Libraries, 1999). Thus dynamically linking makes it possible, in contrary to GPL, to use parts of LGPL code without having to redistribute the code of LGPL. This leads to view of LGPL as an 'modestly restrictive' license.

2.3.3. Unrestrictive license

The most popular alternative license for GPL/LGPL is the BSD (Lerner & Tirole, 2005). The first freely-redistributable source code from Berkeley was released in 1989. The licensing terms were liberal: "A licensee could release the code modified or unmodified in source or binary form with no accounting or royalties to Berkeley" (McKusick, 1999 : 12). Users only had to keep the copyright notices in the source file intact and indicate in their documentation that the product contained code from the University of California and its contributors. Thus, the main requirement is the preservation and acknowledgment of previous contributors' work (Fitzgerald, 2006), but in contradiction to GPL/LGPL it allows appropriation of software modifications (Bonaccorsi & Rossi, 2003). The largest difference between BSD and GPL/LPGL is thus that code created under BSD, or derivations from that code, can get "closed" and be commercialized by anyone (Onetti & Capobianco, 2005). This is extremely contradicting with GPL, where strict rules are set to narrow down the risk of commercial exploitation. Because of the flexibility, BSD can be considered as an 'unrestrictive' license.

In general, there can be made a distinction between two dimensions of OS licenses. First, the degree to which licenses permit closed derivatives. Second, the degree to which licenses provide credits to the developers for their contribution to the OS code. In this research, these two dimensions are viewed together as indicators for restrictiveness.

2.4. Network structure

Institutions can affect network structures, since they provide formal and informal constraints that shape behaviour (North, 1990). We conjecture that network structure can play an important role in explaining the effects of restrictiveness of licenses on project performance. Multiple OS software developers can contribute to an OS project and OS software developers can work concurrently across several projects. Therefore two projects are related to each other when they share at least one developer (Grewal et al., 2006). Social ties in general may facilitate the flow of information and resources in a network, and are thus of importance for the spread and creation of knowledge (Coleman, 1990; Granovetter 1973). This may affect the performance of projects (Burt, 2000; Singh, 2010). To draw conclusions on performance of OS projects, it is therefore important to pay attention to the influence of restrictiveness of licenses on network structure, and network structure on performance.

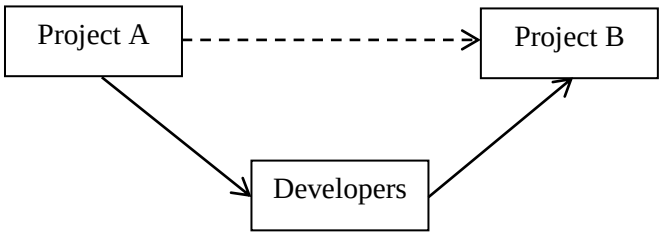
In this section, we will explore the effect of restrictiveness of licenses on performance, via network position. The structural holes theory of Burt (2000, 2005) and the theory of network closure of Coleman (1990) will be used. Both theories pay attention to the concept of social capital, which can be defined as "the advantages that individuals or groups have because of their location in social structure" (Burt, 2000: 347). First, the theory of Burt (1992) states that competitive advantages in social capital can be obtained from access to information and control: networks that include structural holes result in broad and early access to, and control over, information. Second, the theory of Coleman (1990) states that competitive advantage can be obtained from managing risk: closed networks improve communication and facilitate enforcement of sanctions. Both theories will be applied to explain the effect of network structure on the relation between restrictiveness of licenses and performance of OS projects. Before applying both theories to this idea, attention will be paid the mechanism behind the influence of restrictiveness licenses on network measures.

2.4.1. Self-selecting mechanism

According to the principle of *methodological individualism* economic and social change can only arise from the action and interaction from individuals (Elster, 1982). This implies that to draw conclusions on the performance of OS projects it is important to look at the mechanism that operates on developer level. Figure 2 shows that ties between OS projects will rise as follows: developers choose to which kind of project they want to contribute. Since it is possible for developers to participate concurrently among several projects – for example,

project A and project B – projects establish a tie when they share developers. Thus, in this study ties involve relations between projects, not individual relations of developers *within* a project. The studied network in our analyses is therefore a unipartite projection of a biparitite network. An unipartite projection is a transformation of a bipartite (a network with two types of actors) "affiliation" graph into a one-mode network (Kossinets, 2008). The subject of research in this study is a project-project network.

Figure 2. Ties between projects.



It can be expected that clustering of developers of highly restrictive-licensed projects occur, because of the strict rules the these license prescribe. This idea is explained best by the example of 'SCO vs. IBM'. This example is a well-known lawsuit about OS in the United States. The SCO Group claimed that IBM illegally used copyrighted Unix software into the OS Linux, which would lead to the allegation that every Linux distributor violated copyright and therefore can be sued (Goettsch, 2003). This led to much revolt in the OS community, because the basis of Linux is offering reliable software at little or no cost. Eric Raymond (2003), president of the OS Initiative, wrote an paper with Rob Landley on the case and they stated that

> *"We wrote our Unix and Linux code as a gift and an expression of art, to be enjoyed by our peers and used by others for all licit purposes both non-profit and for-profit. We did* not *write it to have it appropriated by men so dishonorable that after making profit from our gift for eight years they could turn around and insult our competence"* (Raymond, 2003: 54)

The anger of the OS community spoke through his words. On the 2nd of February in 2004, Jim Kerstetter wrote an article in Business Week on SCO with the title 'The most hated company in tech'. SCO accused more than 1.000 Linux customers of illegally using SCO's property (Kerstetter, 2004). The company created anxiety and confusion among Linux users, something developers wished to avoid. Developers even suggested to rewrite the code, to

prevent getting sued: “Should there are really tainted portions of code, why not to rewrite them?” (Marti, 2003). This example shows that it is safer to contribute to projects written under the same license. It minimizes the risk of illegally using code from one project in another, as it minimized the risk of a repetition of SCO vs. IBM. Therefore, it is expected that some kind of ‘self-censorship’ among developers exists in choosing projects to which they want to contribute, since this is a ‘safer’ option. In particular, self-censorship is expected to be higher for developers working under more restrictive licenses, since these licenses imply stricter rules.

Besides the safer option, OS developers might want to distinct themselves from commercial partners by ideological ideas, where the first focuses more on the ‘software freedom’ of code than the latter (Hertel, Niedner & Hermann, 2003). Highly restrictive licenses protect this software freedom more than less restrictive licenses. Colazo & Fang (2009) studied the ‘selective benefits’ and ‘ideological identification’ motives and found that copyleft licensing, which is included in highly restrictive licenses, is preferred by developers. Thus, ideological ideas can be of importance for self-selection of developers.

Finally, tendencies towards clustering of highly restrictive licenses can also be expected according to “dynamic strategic complementarities” or “dynamic network externalities” (Lerner & Tirole, 2005). This concept explains that if a licensor has agreed upon a restrictive license, he or she is more likely to choose a similar-licensed project in anticipation of future user benefits from combining end results (Lerner & Tirole, 2005). Building on the former concept, it can be expected that highly restrictive licensors will most likely cooperate with similar-licensed projects, because of perceived benefits.

Thus, it is expected that self-censorship of developers especially towards highly restrictive licenses exists because of three reasons: (1) choosing similar-licensed projects is safer for developers, (2) ideological views are included in highly restrictive licenses and (3) cooperating with similar-licensed projects is beneficial for combining end results. We predict that the self-selecting mechanism has important implications for the influence of restrictiveness of licenses on network structures.

2.4.2. Network structure: level of brokerage

Network structure is expected to influence the social capital in the network. First, Burt (1992) suggests that structural holes in social networks are the source of social capital. Structural holes are defined as “holes in the structure of information flow” (Burt, 2005: 16). These holes contain a lot of potential since they separate non-redundant sources of information (Burt,

2005). Bridging these structural holes gives actors a favourable position, since they combine different non-redundant sources of information. An actor forms a bridge when he or she fulfils the only possible paths between two points (Harary, Norman & Cartwright, 1965). Specifically, the position of a bridge relation comes with three advantages: (1) actors have superior access to a wider diversity of information, (2) actors have early access to that information, and (3) actors can control over information diffusion and therefore exploit information in their advantage (Burt, 2000). The actors that build bridges between structural holes are also called brokers, and maintain the most favourable position in a network (Burt, 2000).

As explained before, ties between projects exist when they share developers, and the network structure of these developers – and therefore indirect of these projects – can be affected by the restrictiveness of licenses through the self-selecting mechanism. A clustering of highly restrictive licensed projects is expected, since the self-selecting mechanism implies that developers of highly restrictive licensed projects tend to contribute to projects written under the same license.

According to Burt's theory (2000, 2005) the emerge of different communities, with more restrictive licensed projects clustered together, gives rise to structural holes in the OS network. Developers of less restrictive licensed projects are expected to profit from these structural holes, since they may apply less self-censorship and are therefore able to work concurrently across different licensed projects. Since clustering of highly restrictive licensed projects is expected, chances to profit from structural holes are for this group smaller. Therefore it is hypothesizes:

H1: Restrictive licensing has a negative effect on brokerage. It is therefore expected that OS projects with more restrictive licenses have lower levels of brokerage.

Previous research has shown that actors that function as bridge for structural holes have been frequently shown to perform better than actors that do not have this position (McEvily & Zaheer, 1999; Zaheer & Bell, 2005). This is because of their access to unique resources and capabilities. Therefore, it is expected that OS projects with higher levels of brokerage – established by participating developers who function as a broker – perform better than OS projects with lower levels of brokerage. In the previous section, a distinction has been made

between two types of performance: process-based performance and outcome-based performance.

First, process-based performance provides information on the activity levels of a project, indicated by, among others, commits of developers (Gutwin, Penner & Schneider, 2004). Higher levels of brokerage may increase activity levels of projects, since projects have more access to different and unique communities. Access to these communities might increase the reach and awareness of a project, and can attract more unique developers. We assume that the more unique, different developers contributing to a project, the higher the activity levels of a project will be. This leads to our second hypothesis regarding to brokerage:

H2: Brokerage positively affects process-based performance. It is therefore expected that OS projects with lower levels of brokerage have lower levels of process-based performance.

Second, outcome-based performance says something about the use of a project and can, among others, be indicated by the number of downloads and project pages views. In line with Burt's theory (1992) we expect that the higher level of brokerage, the higher the outcome-based performance is. The argument of access to unique developer user communities can also be applied to the outcome-based performance of OS projects. Access to unique resources and capabilities can increase differentiation of the project and help to create a competitive advantage (Besanko, Dranove, Shanley & Schaefer, 2013). Crowston et al. (2003) showed that developers of projects are often also users, thus access to different, unique developer communities implies also access to user communities. This may have positive implications for the use of a project. Thus, higher levels of brokerage lead to more unique resources, capabilities and project awareness which is beneficial for the outcome-based performance. This leads to our second hypothesis:

H3: Brokerage positively affects outcome-based performance. It is therefore expected that OS projects with lower levels of brokerage have lower levels of outcome-based performance.

2.4.3. Network structure: level of closure

Whereas brokerage says something about the probability that an actor serves as a bridge between different groups of actors, closure measures the extent to which alters are connected

(Gargiulo, Ertug & Galunic, 2009). The higher level of closure, the higher the amount of alters are connected. This implies that OS projects with high levels of closure have a high amount of alters that are connected via sharing developers.

As explained by the self-selecting mechanism, we expect self-censorship of developers especially towards highly restrictive licenses because of three reasons: (1) choosing similar-licensed projects is safer for the developer, (2) ideological views are included in highly restrictive licenses and (3) cooperating with similar-licensed projects is beneficial for combining end results. This is expected to result in higher levels of closure for more restrictive licensed projects, since they may be more clustered:

H4: Restrictive licensing has a positive effect on closure. It is therefore expected that OS projects with more restrictive licenses have higher levels of closure.

First, according to Burt (1992), high levels of closure are expected to constrain the access to different, unique developer-user communities. This results in limited reach and awareness of OS projects. As illustrated above, limited reach and awareness is expected to lead to lower activity levels of projects and thus in lower levels of process-based performance. This leads to our fifth hypothesis:

H5A: Closure negatively affects process-based performance. It is therefore expected that OS projects with higher levels of closure have lower levels of process-based performance.

An alternative idea to the theory of structural holes of Burt (2005) is Coleman's view (1990) of social capital. Contrary to Burt (2005), he argues that networks with closure – networks in which everyone is connected such that no one can escape the notice of others – are the source of social capital. Closure is of importance because of two things: (1) network with closure lead to more reliable communication channels, and (2) network with closure facilitate norms and sanctions that make it less risky for people in the network to trust each other (Coleman, 1990). These factors are of importance for reducing uncertainty between actors, by providing stable structures for human interaction. Lower levels of uncertainty lower transaction costs for OS users and developers and can increase attractiveness of a project (North, 1990). This can have a positive influence on the activity levels of a project, since developers may be more willing to contribute when the project takes place in a reliable network. Furthermore, it can be expected that OS projects with higher levels of closure share a distinct set of ideological principles, with "software freedom" taking a prominent place. This may lead to higher levels

of identification of developers with these OS projects and stimulate frequently participation in group's collective activities (Gamson, 1991; Simon et al., 1998). Frequent participation is expected to result in higher levels of project activity. This leads to the following hypothesis:

H5B: Closure positively affects process-based performance. It is therefore expected OS projects with higher levels of closure have higher levels of process-based performance.

The views of Burt and Coleman also provide different expectations for the effect of closure on outcome-based performance. Burt (2000) argues that closure eliminates structural holes, since networks with closure are networks in which everyone is connected and new sources of information and resources are limited. Research has shown that actors that function as bridge for structural holes have been frequently shown to perform better than actors that do not have this position (McEvily & Zaheer, 1999; Zaheer & Bell, 2005). Since networks with closure eliminate structural holes, a negative relation between closure and outcome-based performance is expected:

H6A: Closure negatively affects outcome-based performance. It is therefore expected that OS projects with higher levels of closure have lower levels of outcome-based performance.

However, according to Coleman (1990), it can be expected that higher levels of closure in network lead to higher outcome-based performance. Because of the higher levels of trust in networks with closure, relation-specific investments are more attractive, thus leading to higher performance (Zaheer & Venkatraman, 1995). The relation-specific investments increase the unique resources of a project and can help create a competitive advantage. This competitive advantage can lead to higher levels of outcome-based performance. Furthermore, common norms can improve mutual understanding, leading to less misinterpretations of a firm's actions by its network partners, and in turn leading to better performance (Dyer & Nobeoka, 2000). This leads to the following hypothesis:

H6B: Closure positively affects outcome-based performance. It is therefore expected that OS projects with higher levels of closure have higher levels of outcome-based performance.

Finally, it can be argued that process-based performance has a positive effect on outcome-based performance (Jemison & Sitkin, 1986). In this research the main focus will be on the influence of institutions on economic performance, but the possible relation between the two types of performance cannot be left behind and therefore we state:

H7: Process-based performance positively affects outcome-based performance. It is therefore expected that OS projects with higher levels of process-based performance also have higher levels of outcome-based performance.

3. Data and Methods

In this section the data collection and sampling method will be discussed. Further, attention will be paid to the explanatory and predictive variables of our analyses. As figure 1 showed, a relation is expected between restrictiveness of licenses and network measures, and network measures and performance. Restrictiveness of licenses can be considered as our explanatory variable and performance measures as our predictive variables. Network structure is expected to have a mediating effect on restrictiveness of licenses and performance. Descriptive statistics of all variables used in analyses are shown in table 4.

3.1. Data collection and sampling method

Our research made use of a unique dataset on network structure and performance of OS projects. The dataset has been built from data provided by the website of *SourceForge* (http://sourceforge.net). *SourceForge* is an OS community resource which thrives on community collaboration to help open source projects to be as successful as possible (SourceForge, n.d.). It is the most important hosting place for OS projects, with 90% of all OS projects listed on this website. Therefore, *SourceForge* can be regarded as the most representative of the OS movement (Singh, 2010). At time of the data collection in 2012, the portal contained of 30,031 active projects. Only active projects were taken into consideration to keep the data manageable and suitable, since inactive projects perform few activities and make no code contributions (Lerner & Tirole, 2005).

The level of aggregation is the project or team level instead of the single user-developer. This has a certain advantage with regard to our research: projects can function as an equivalent for organizations, and information on the influence of institutions on the performance of projects can be translated to the effects of institutions on organizations. Therefore, the project level is

a useful level of aggregation for our analyses. A "project" here can be defined as a group of users-developers working on the same software. For all projects characteristics are documented, among others the age of the project, the number of developers of the project and the type of license of the project. Further, indicators on performance are available: for example, the number of downloads and forum messages. Finally, the dataset contains information on the relation between projects on *SourceForge*, which provides information on several network-measures.

3.2. Restrictiveness of licenses

The explanatory variable used in our analyses is the restrictiveness of licenses. In the previous section, a distinction has been made between three types of restrictiveness of licenses: highly restrictive (GPL), modestly restrictive (LGPL), and unrestrictive (BSD) licenses. These three types of licenses count for the largest share of projects of the *SourceForge* data (86.2%). Besides these licenses, the dataset contains 11 other types of licenses, among others the Apache License V2.0, Mozilla Public License (MPL) 1.1 and the Academic Free License (AFL). These licenses have shares less than 4% and are less relevant for our analyses. Therefore, a selection has been made for only highly restrictive (GPL), modestly restrictive (LGPL), and unrestrictive (BSD) licenses. The three types of license are coded as dummy-items, ranging from 0 to 1, with 1 representing projects that contain the type of license mentioned. In some particular cases, projects are written under more than one license (hybrid-licensed projects).These projects account for only for a small share in the dataset (3.8%), and were included in the analyses.[1] This explains the observation that the sum of frequencies of the three types of licenses exceeds the 100%.

3.3. Network measures

Network measures are considered as mediating variables in our analyses. Two types of network measures have been distinguished: the level of brokerage and the level of closure.

The level of brokerage is operationalized by the level of betweenness centrality. Betweenness centrality can be defined as "an index that measures the extent to which a person brokers indirect connections between everyone else in a network" (Freeman, 1992 in Burt, 2005: 27). The dataset contains six items on betweenness, ranging from a cut-off path of three to a cut-

[1] We have repeated all analyses with exclusion of hybrid-licensed projects. No significant differences in results were found. Therefore, we continued interpretation of our first analyses without exclusion of hybrid-licensed projects.

off path of eight. Betweenness centrality thus only considers paths of length cut-off or smaller. The items show high correlations (see Appendix 3 table 1), therefore it is suitable to select one item of betweenness centrality for our analyses. The medium of six items has been chosen. However, the variable is highly skewed: 84% of the OS projects are never the shortest path of length five cut-off or smaller. Because of the highly skewed distribution of the variable, the logarithm has been computed. (Log) betweenness will be used as indicator for the level of brokerage of OS projects.

The level of closure of OS projects expresses the average social distance from each project to other projects in the network. Closure is operationalized by the transitivity index of a project. This index is used in previous social network research, measuring the average fraction of an actor's friends who are also friends with one another (Tam Cho & Fowler, 2007). This implies that the higher the proportion of transitive relations in a network - thus the more alters a project has that share developers - the higher the level of closure in a network. Almost half of all projects have a transitivity of zero (49.1%). This implies that almost half of all OS projects have no alters that share developers.

3.4. Performance indicators

Performance measures are considered as predictive variables of our analyses. In our research, a distinction has been made between two groups of performance measures: process- and outcomes-based performance measures. Both performance measures are indicated by two items. Table 3 shows the average performance levels of highly restrictive, restrictive, and unrestrictive licenses. Lower levels of performance are observed for highly restrictive licenses, whereas levels of performance are higher for modestly restrictive and unrestrictive licenses. Overall differences between modestly restrictive and unrestrictive licenses are smaller than differences between highly restrictive and restrictive licenses.

Process-based performance is measured by the number of reads and the number of commits. A commit occurs when a programmer decides to submit a file with changes in the source code, whereas a read implies that a programmer makes a request for the source code (Comino, Manenti, & Parisi, 2007). Both measures can be viewed as indicators of developer activity levels, and provide information on the process of a project (Gutwin, Penner & Schneider, 2004). An important note is that the number of commits include a large amount of missings (17,285 cases). However, since better process measures are not available, this variable will still be used for our analyses.

Table 3. Restrictiveness of licenses and performance indicators.

	Highly restrictive licenses	Modestlyly restrictive licenses	Unrestrictive licences
Process-based performance			
Reads	0.084	0.152	0.147
Commits	0.041	0.056	0.049
Outcome-based performance			
Downloads	2.057	2.148	2.088
Page views	5.134	5.327	5.320

Note: licenses are coded as dummy items. Reference groups of a license are in this case the two other indicated levels of restrictiveness.

Both process measures are ratio variables, measured from the start of the project. Since it can be expected that older projects have higher levels of commits and reads in absolute terms, averages per day are computed. Furthermore, it can be argued that small increases in process measures at the start weigh more for projects than small increases when higher levels of process-based performance are already obtained. In other words, an increase from 0 to 1 weighs more than an increase from 100 to 101. To capture these differences and to prevent negative values, the logarithm + 1 has been taken of both variables.

Outcome-based performance is measured by the number of downloads and the number of views of project pages. Both items are ratio variables measured over a 60 day window. As with process-based performance, it can be expected that increases in outcome measures weigh more at the start than when higher levels of outcome-based performance are already obtained. Therefore, also for the outcome-based measures the logarithms + 1 have been taken. For (log) number of downloads, more than 40% of all projects has not been downloaded in the 60 day window, whereas for (log) number of project page views only 7% of all OS projects has not been viewed in the 60 day window.

3.5. Control variables

The number of developers, the age of the project, the quadratic term of age of the project, the development status of the project, the program language and the type of users of the project are included as control variables.

The number of developers can be viewed as the most important control variable in our analyses. Previous research showed that the team size of a project can be considered as an

important control variable, since network measures are generally related to the number of participants, and this can influence performance (Reagans & Zuckerman, 2001; Singh, 2010). Since we expect that the number of developers affect the network location and performance of OS projects, this variable is included in the first model of our analyses. Furthermore, there is controlled for the age of a project, since the software life cycle may also influence network and performance measures (Singh, 2010). Since this variable is measured in days, this results in large numbers. To keep the data manageable, age of the project is divided by a thousand. To control for a potential nonlinear relationship between the age of the project, network formation, and project performance, the square of age of the project is computed. The centralized measures of age and age squared are included to avoid multicollinearity problems. Also, the development status of the project may influence the location in the network and the performance.

Table 4. Descriptive statistics variables included in analyses

Variable	*N*	*Mean*	*Std. Dev.*	*Min.*	*Max.*
Licenses					
Highly restrictive license	24951	0.77	0.42	0	1
Modestly restrictive license	24951	0.18	0.38	0	1
Unrestrictive license	24951	0.10	0.29	0	1
Network measures					
(Log) level of betweenness	24951	0.77	2.12	0	14.11
Level of closure	24951	0.08	0.09	0	0.25
Performance measures					
(Log) average commits	7666	0.04	0.12	0	1.91
(Log) average reads	24951	0.10	0.38	0	5.80
(Log) downloads	24951	2.06	2.35	0	17.25
(Log) total pages	24951	5.17	2.39	0	17.84
Control variables					
Number of developers	24951	2.61	4.72	1	247
Age	24951	0.00	0.93	-1.52	1.98
Age^2	24951	0.87	0.87	0.00	3.91
Status of the project	24951	3.27	1.60	1	6
Language : Java	24951	0.29	0.45	0	1
Language : C + +	24951	0.23	0.42	0	1
Language : PHP	24951	0.13	0.33	0	1
Language : C	24951	0.22	0.42	0	1
Users : Developers	24951	0.50	0.50	0	1
Users : End Users/Desktop	24951	0.43	0.49	0	1
Users : System Administrators	24951	0.17	0.37	0	1
Users : Advanced End Users	24951	0.12	0.33	0	1

Previous research showed that the development stage of projects plays an important moderating role in determining performance outcomes (Stewart & Gosain, 2006). The development status consists of five phases: Developing, Alpha, Beta, Stable and Mature. Only 2% of all projects are in the mature phase of their development. At last, two groups of variables are included to account for the different characteristics of projects. There has been controlled for the program languages and the type of users of the project by constructing dummies variables. Only the four most named program languages and types of users are included in our analyses.

4. Results

To test our meta-hypotheses on the influence of institutions on network structure, and network structure on performance, we performed linear ordinary least squares (OLS) regression analyses and structural equation modelling (SEM). Three OLS-regression analyses were performed to gain insights in the effects of the restrictiveness of licenses on network measures, network measures on process-based performance, and network measures on outcome-based performance. To determine the overall direct, indirect and total effects of the variables, a SEM analysis was performed additionally. SEM has several advantages compared to OLS. First, in SEM, a variable can be both predictive and explanatory. This means that multiple equations can be modelled simultaneously, whereas in OLS regression, a variable can only either be response or explanatory (Xiao, 2013). Our theory involves variables off both roles, hence SEM has a big advantage over OLS. Further, SEM includes more flexible assumptions, in particular allowing interpretation even when multicollinearity exists (Xiao, 2013). Finally, it possible for SEM to estimate the model with maximum likelihood with missing values (mlmv). This method aims to retrieve as much information as possible from observations containing missing values (Acock, 2013). This is beneficial since it increases the sample of our SEM analysis. Therefore, SEM is a useful complement to our research.

In this section, the results regarding our three OLS regression analyses will be presented and discussed, as well as the results of our SEM analysis. Attention will be paid to the direct, indirect and total effects of the explanatory variables.

4.1. OLS regression analyses

OLS regression analyses were performed to estimate the effects of the restrictiveness of licenses on network measures, network measures on process-based performance, and network measures on outcome-based performance. For each hypothesized relation, two or more models are presented. The first model contains the explanatory variables and the most important control variable – the number of developers – used in explaining variation in the dependent variable. Consecutive models add relevant other explanatory variables. The final model includes all relevant explanatory variables and control variables. This model will be fully interpreted.

Assumptions of no perfect multicollinearity between explanatory variables and homoskedasticity were tested for all OLS regression analyses. Perfect multicollinearity occurs when two or more predictors show a perfect linear relationship, and poses a problem since it becomes impossible to obtain unique estimates (Field, 2009). Perfect multicollinearity is rare. However, less than perfect multicollinearity, a strong correlation between two or more predictors, is virtually unavoidable (Field, 2009). This can result in larger estimates of the variance of the coefficients, which in turn causes larger standard errors and higher p-values. With help of the Variance Inflation Factor (VIF), problems of multicollinearity can be identified. As a rule of thumb, VIF's greater than 10 indicate that variables could be considered as a linear combination of other independent variables (Myers, 1990). Table 2 in Appendix 3 shows that mean VIF's are between 1.28 and 1.46. Thus, it can be concluded there are no problems of multicollinearity identified.

To meet the assumption of no heteroskedasticity, the error term should have a constant variance ($Var(\varepsilon_i) = \sigma^2$). If this assumption is violated, coefficients remain the same, but variance of the coefficients is biased. This means that hypotheses tests cannot be performed (Breusch & Pagan, 1979). Heteroskedasticity can be diagnosed with the Breusch-Pagan test. This involves regressing the squared residuals on all explanatory variables. Heteroskedasticity is observed when the explanatory variables have a jointly significant impact on ε_i^2.Results of the Breusch-Pagan test show that the assumption of no heteroskedasticity is violated for all analyses. Therefore, equations have to be estimated with heteroskedasticity-robust standard errors. OLS regression analyses with heteroskedasticity-robust standard errors have been performed and interpreted. Results of the robust OLS regression analyses are presented in table 5, 6 and 7, and will be reviewed.

4.1.1. Restrictiveness of licenses on network measures

The effects of the restrictiveness of licenses on two network measures were studied: effects on the level of brokerage and on the level of closure. The results of these regression analyses are shown in table 5.

For the level of brokerage, the restrictiveness of licenses and the number of developers were significant predictors (R^2 = .261, F(3, 24947) = 68.01, p < .001). Including other control variables resulted in an explained variance of 30.0% (R^2 = .300, ΔF(11, 24936) = 46.08, p <.001*).*

Table 5. The restrictiveness of licenses predicted by network measures.

	Level of brokerage			Level of closure		
	B	*SE(B)*	*Hyp.*	*B*	*SE(B)*	*Hyp.*
Licenses						
Highly restrictive license	-0.118**	0.040	H1: -	-0.005**	0.002	H4: -
Modestly restrictive license	0.061	0.046		-0.002	0.002	
Control variables						
Number of developers	0.213***	0.023		-0.002***	0.000	
Age	0.299***	0.019		0.003***	0.001	
Age^2	0.129***	0.016		0.000	0.001	
Status of the project	0.073***	0.007		0.004***	0.000	
Language: Java	0.001	0.033		0.007***	0.001	
Language: C++	0.026	0.032		-0.003*	0.001	
Language: PHP	-0.116***	0.032		-0.010***	0.002	
Language: C	0.315***	0.035		0.004**	0.002	
Users: Developers	0.082**	0.025		0.006***	0.001	
Users: End Users/Desktop	-0.001	0.025		-0.001	0.001	
Users: System Administrators	-0.109**	0.033		0.000	0.002	
Users: Advanced End Users	0.035	0.031		0.000	0.002	
Constant	-0.149	0.063		0.077	0.003	
R^2	.300			.017		
N	7,666			7,666		

**p <.05. **p <.01. ***p<.001 ; unrestrictive licenses are the reference category of restrictiveness of licenses; heteroskedasticity-robust standard errors used.*

Table 6. The effect of network measures on process-based performance.

	Number of reads			Number of commits		
	B	*SE(B)*	*Hyp.*	*B*	*SE(B)*	*Hyp.*
Network measures						
Level of brokerage	0.035***	0.003	H2: +	0.003*	0.001	H2: +
Level of closure	-0.040*	0.017	H5a: +	-0.006	0.010	H5a: -
			H5b: -			H5b: -
Licenses						
Highly restrictive license	-0.023**	0.008		-0.004	0.004	
Modestly restrictive license	0.020*	0.010		0.004	0.005	
Control variables						
Number of developers	0.030***	0.003		0.010***	0.001	
Age	0.006**	0.002		-0.004	0.002	
Age^2	-0.009**	0.003		-0.013***	0.002	
Status of the project	0.024***	0.001		0.008***	0.001	
Language: Java	0.011	0.005		0.006*	0.003	
Language: C++	0.017**	0.006		0.008**	0.003	
Language: PHP	-0.012	0.006		0.000	0.004	
Language: C	0.022***	0.006		-0.002	0.003	
Users: Developers	0.005	0.004		0.001	0.003	
Users: End users/Desktop	0.005	0.004		0.005	0.003	
Users: System Administrators	-0.008	0.006		-0.005	0.004	
Users: Advanced End Users	0.021**	0.006**		0.005	0.003	
Constant	-0.076			-0.009		
R^2	.267			.306		
N	7,666			7,666		

**p <.05. **p <.01. ***p<.001; unrestrictive licenses are the reference category of restrictiveness of licenses; heteroskedasticity-robust standard errors used.*

Table 7. The effect of network measures on outcome-based performance.

	Number of downloads			Number of page views		
	B	*SE(B)*	*Hyp.*	*B*	*SE(B)*	*Hyp.*
Network measures						
Level of brokerage	0.114***	0.016	H3: +	0.146***	0.017	H3: +
Level of closure	-0.353	0.250	H6a: -	0.487	0.275	H6a: -
			H6b: -			H6b: -
Process measures						
Number of commits	1.638**	0.500	H7: +	1.421**	0.517	H7: +
Number of reads	0.825***	0.121	H7: +	0.943***	0.107	H7: +
Licenses						
Highly restrictive license	0.222**	0.072		0.108	0.081	
Modestly restrictive license	0.068	0.080		0.047	0.090	
Control variables						
Number of developers	-0.005	0.009		0.004	0.007	
Age	0.726***	0.040		0.564***	0.043	
Age^2	-0.008	0.035		0.024	0.039	
Status of projects	0.613***	0.014		0.432***	0.016	
Language: Java	-0.024	0.057		0.034	0.061	
Language: C++	0.132*	0.061		0.006	0.067	
Language: PHP	-0.257***	0.073		0.019	0.082	
Language: C	0.131	0.068		0.204**	0.074	
Users: Developers	-0.045	0.052		-0.077	0.057	
Users: End users/Desktop	0.402***	0.053		0.202**	0.059	
Users: System Administrators	0.131	0.077		0.158	0.088	
Users: Advanced End Users	0.014	0.065		-0.037	0.072	
Constant	-0.025			3.666		
R^2	.409			.303		
N	7,666			7,666		

**p <.05. **p <.01. ***p<.001 ; unrestrictive licenses are the reference category of restrictiveness of licenses; heteroskedasticity-robust standard errors used.*

As expected, in the final model highly restrictive licenses significantly predicted levels of brokerage (B = -0.118, p =.003). Since the level of brokerage is indicated by (log) level of betweenness, this can be interpreted as follows: using a highly restrictive license compared to an unrestrictive license, predicts a (-0.118*100) 11.8% decrease in the level of brokerage. This lends support to our first hypothesis.

For the level of closure, the restrictiveness of licenses and the number of developers were also significant predictors, but only explained 0.55% of the variance in the level of closure of projects (R^2 = .006, F(3, 24947) = 40.37, p < .001). Including other control variables resulted in an explained variance of 1.71% (R^2 = .017, ΔF(11, 24936) = 27.39, p <.001*).* Contrary to our expectations, it was found that in this final model highly restrictive licenses significantly predicted lower levels of closure. Using a highly restrictive license compared to an unrestrictive license, predicts a 0.00534 decrease in the level of closure. This does not support our fourth hypothesis.

4.1.2. Network measures on process-based performance

Two process-based performance measures were distinguished: the number of reads and the number of commits. Results of these regression analyses are shown in table 6.

For the number of reads, the network measures and the number of developers are significant predictors (R^2 = .254, F(3, 24947) = 234.87, p <.001). Adding the effect of licenses results in a small, but significant increase in the explained variance in the number of reads (ΔR^2 = .002, ΔF(2, 24945) = 26.24, p < .001). The final model, which also captures the effect of all control variables, results in an explained variance of 26.7% in the number of reads (R^2 = .267, ΔF(11, 24934) = 47.40, p < .001). As predicted, the level of brokerage is a positive, significant predictor of the number of reads. Since both the level of brokerage and the number of reads are indicated by logarithm-variables, the coefficients can be interpreted as an elasticity: a one-percentage increase in the level of brokerage predicts a 0.035% increase in the number of reads. Also in line with our expectations, the level of closure is a significant predictor of the number of reads: a one-unit increase in the level of closure predicts a decrease of 0.040 in the number of reads. Highly restrictive licenses compared to unrestrictive licenses have a negative, significant effect on the number of reads, while modestly restrictive licenses compared to unrestrictive licenses have a positive effect on the number of reads.

For the number of commits, the network measures and the number of developers are significant predictors (R^2 = .288, F(3, 7662) = 108.56, p < .001). Including restrictiveness of licenses predicts a small, but significant increase in explained variance in the number of

commits (ΔR^2 = .001, ΔF(2, 7660) = 4.41, p = .0122). The final model, which also captures the effect of all control variables, predicts 30.6% of the variance in the number of commits (R^2 = .306, ΔF(11, 7649) = 14.57, p = <.001). As expected, in the final model the level of brokerage is a positive, significant predictor of the number of commits. As with the number of reads, the coefficient can be interpreted as an elasticity: a one-percentage increase in the level of brokerage predicts a 0.003% increase in the number of reads. Thus hypothesis 2 is confirmed. Inconsistent with our expectations, the level of closure is not a significant predictor of the number of commits. Therefore, hypothesis 5a is partly confirmed, whereas hypothesis 5b is rejected. Also the restrictiveness of licenses has no significant effect on the numbers of commits.

4.1.3. Network measures on outcome-based performance

As with process-based performance, two measures have been chosen for outcome-based performance: the number of downloads and the number of project page views. Results are shown in table 7.

For the number of downloads, the network measures and the number of developers are significant predictors (R^2 = .129, F(3, 7662) = 217.62, p <.001). Including process-based performance predicts a significant increase in explained variance (ΔR^2 = .068, ΔF(2, 7660) = 108.97, p <.001), as for the consecutive model that adds the effect of licenses (ΔR^2 = .002, ΔF(2, 7658) = 10.55, p <.001). In the final model all other control variables are added. This model predicts an explained variance of 40.1% of the number of downloads (R^2 = .4085, ΔF(11, 7647) = 254.25, p <.001). In line with our expectations, the final model shows that process-based performance significantly predicts higher levels of downloads. For both process-variables, coefficients can be interpreted as an elasticity: a one-percentage increase in the number of commits predicts a 1.64% increase in the number of downloads, whereas a one-percentage increase in the number of reads predicts a 0.82% increase in the number of downloads. Consistent with our expectations, the level of brokerage is also a significant predictor for the number of downloads. A one-percentage increase in the level of brokerage predicts a 0.11% increase in the number of downloads. Contrary to our expectations, the level of closure is not a significant predictor of the number of downloads. Highly restrictive licenses predict significant, higher numbers of downloads.

For the number of page views, 13.6% of the variance is explained by network measures and the number of developers (R^2 = .136, F(3, 7662) = 236.28, p <.001). Including process measures predicts a significant increase in the explained variance in the number of

page views (ΔR^2 = .0606, $\Delta F(2, 7660) = 112.64$, $p < .001$) Adding licenses adds small, but significant value compared to the model before ($\Delta F(2, 7658) = 2.78$, p =.0622). The final model, with all other control variables included, predicts an explained variance of 30.3% of the number of page views (R^2 = .303, $\Delta F(11, 7647) = 100.50$, p <.001). As expected, this final model shows that process-based performance significantly predicts higher levels of page views. Thus, hypothesis 7 can be confirmed. As for the number of downloads, coefficients can be interpreted as an elasticity: a one-percentage increase in the number of commits predicts a 1.42% increase in the number of page views, whereas a one-percentage increase in the number of reads predicts a 0.94% increase in the number of page views. As expected, the level of brokerage predicts significantly higher levels in the number of total pages. Thus, support has been found for hypothesis 3. In line with hypothesis 6b, higher levels of closure predict higher levels of page views. However, this effect is small and only significant at the 10% α-level (p = 0.077). Significance is therefore not reported in table 7. Thus, hypothesis 6b is partly confirmed and hypothesis 6a rejected. Restrictiveness of licenses is no significant predictor for the number of page views.

4.2. SEM analysis

We used SEM analysis to estimate the direct, indirect and total effects of our explanatory variables on network structure and performance indicators. As proposed by our theoretical model, we expect a mediating effect of network structure on the relation between restrictiveness of licenses and performance. Therefore restrictiveness of licenses may have direct and indirect effects on performance. To account for these effects, SEM differentiates between these relations and calculates total effects of our explanatory variables. In summary, total, direct and indirect effects are shown in table 9.

Table 8. Goodness of fit tests and indexes (non-robust model).

Fit statistic	Value Indicating Good Fit[1]	Value
Model versus saturated model		$\chi^2(3) = 22{,}001.786$, $p<.001$.
Baseline versus saturated model		$\chi^2(99) = 60{,}657.289$, $p<.001$.
RMSEA (CI 90%)	<.08	0.542
AIC	Smallest value	853,901.141
CFI	>.9-.95	0.637
TLI	>.9-.95	-10.988
CD	Close to 1	0.686

Note. RMSEA = root mean squared error of approximation; CI = confidence interval; AIC = Akaike's Information Criterion; BIC = Bayesian information criterion; CFI = Bentler's Comparative Fit Index; TLI = Tucker–Lewis Index, CD = Coefficient of determintation.

[1] Indexes and values indicating good fit based on Byrne (2001), Hu and Bentler (1999), and Kline (1998) in Reniers et al. (2011).

Table 8 presents the results of the goodness of fit tests and indexes. The first two tests present the likelihood ratio of the model. First, the statured model is the model that fits the co-variances perfectly (Kline, 2013). It can be rejected at the 1% level that our model fits as well as the saturated model, χ^2 (9) = 22,001.79, p <.001. The second test shows the baseline versus the saturated comparison. The baseline model includes the mean and variances of all observed variables plus the co-variances of all observed exogenous variables. We can reject at the 1% level that the baseline model fits as well as the saturated model, χ^2 (99) = 60,657.29, p <.001. This is generally a sign of a poor fit (Kline, 2013). Further, results of the goodness of fit indexes show that values do not correspond with the indices of a good fit. Since our main goal of using SEM is to gain insights in the total effects of our model, we will continue using this model to make comparison to the OLS regression results possible. Furthermore, "closer to fit in SEM does not mean closer to truth" (Kline, 2013: 201).

As shown by the Breusch-Pagan test of the OLS regressions the assumption of homoskedasticity has not been met. Therefore, SEM analysis has been performed with heteroskedasticity-robust standard errors. This will be our final interpreted model. However, correction for heteroskedasticity makes it impossible to perform among others goodness of fit tests and indexes. Therefore, table 8 includes the goodness of fit tests and indexes of our non-robust model. Table 9 presents the total, direct and indirect effects of our SEM analysis predicted with heteroskedasticity-robust standard errors.

Results show *direct* effects of highly restrictive licenses compared to unrestrictive licenses. Highly restrictive licenses predict significant, lower levels of brokerage. This is as expected. However, highly restrictive licenses predict significant, lower levels of closure. This is not in line with our expectations. For both process-based performance variables, the level of brokerage is a positive, significant predictor. Levels of closure only show a negative, significant direct effect for the number of reads. Restrictiveness of licenses are only a significant predictor of the number of reads and not for commits. For both outcome-based performance measures, the level of brokerage again is a significant, positive predictor, whereas the level of closure only is a significant, positive predictor of total page views. Process measures significantly predict outcome measures, where higher levels of process-based performance lead to higher levels of outcome-based performance. This is as expected. Highly restrictive licenses only have a positive, significant direct effect on the number of downloads and not on total pages.

Table 9. SEM Analysis – total, direct and indirect effects.

	Hyp.	*Total effects*		*Direct effects*		*Indirect effects*	
		B	*SE(B)*	*B*	*SE(B)*	*B*	*SE(B)*
(Log) Level of betweenness							
Highly restrictive license	H1: +	-0.118**	0.040	-0.118**	0.040	0	
Modestly restrictive license		0.061	0.046	0.061	0.046	0	
Number of developers		0.213***	0.023	0.213***	0.023	0	
Level of closure							
Highly restrictive license	H4: -	-0.005**	0.002	-0.005**	0.002	0	
Modestly restrictive license		-0.002	0.002	-0.002	0.002	0	
Number of developers		-0.002***	0.000	-0.002***	0.000	0	
(Log) average commits							
(Log) Level of betweenness	H2: +	0.004**	0.001	0.004**	0.001	0	
Level of closure	H5: -	0.005	0.013	0.005	0.013	0	
Highly restrictive license		-0.008	0.005	-0.007	0.005	-0.001*	0.000
Modestly restrictive license		0.006	0.006	0.006	0.006	0.000	0.000
Number of developers		0.011***	0.001	0.010***	0.001	0.001**	0.000
(Log) average reads							
(Log) Level of betweenness	H2: +	0.035***	0.003	0.035***	0.003	0	
Level of closure	H5a: +	-0.040*	0.017	-0.040*	0.017	0	
Highly restrictive license		-0.027**	0.008	-0.023**	0.008	-0.004**	0.001
Modestly restrictive license		0.023*	0.010	0.020*	0.010	0.002	0.002
Number of developers		0.037***	0.003	0.030***	0.003	0.007***	0.001
(Log) downloads							
(Log) Level of betweenness	H3: +	0.169***	0.012	0.119***	0.013	0.050***	0.013
Level of closure	H6: -	-0.059	0.133	-0.077	0.166	0.017	0.108
(Log) average commits	H7: +	8.098***	0.415	8.098***	0.415	0	
(Log) average reads	H7: +	0.493***	0.075	0.493***	0.075	0	
Highly restrictive license		0.086*	0.042	0.178**	0.056	-0.092*	0.042
Modestly restrictive license		0.133**	0.047	0.064	0.064	0.069	0.049
Number of developers		0.091***	0.011	-0.044***	0.009	0.135***	0.013
(Log) total pages							
(Log) Level of betweenness	H3: +	0.196***	0.013	0.143***	0.015	0.053***	0.014
Level of closure	H6b: +	0.500**	0.144	0.482**	0.179	0.019	0.114
(Log) average commits	H7: +	8.600***	0.451	8.600***	0.451	0	
(Log) average reads	H7: +	0.513***	0.074	0.513***	0.074	0	
Highly restrictive license		-0.039	0.045	0.063	0.061	-0.102*	0.045
Modestly restrictive license		0.112*	0.051	0.039	0.069	0.073	0.052
Number of developers		0.102***	0.013	-0.043***	0.011	0.145***	0.014
Log Likelihood	-426842.57						
N	24,951						

**p <.05. **p <.01. ***p<.001; unrestrictive licenses are the reference category for restrictiveness of licenses; heteroskedasticity-robust standard errors used.*

Concerning the *indirect* effects, restrictiveness of licenses and levels of brokerage play an important role. Highly restrictive licenses are for all process- and outcome-based performance indicators significant, negative predictors. Furthermore, the level of brokerage shows a strong, positive, significant indirect effect on both outcome-based performance measures. Levels of closure have no indirect effect on outcome-based performance.

With regard to the *total* effects, most results are in line with our expectations. Results show that highly restrictive licenses compared to unrestrictive licenses have a negative, significant total effect on the level of brokerage. This is in line with our first hypothesis. Highly restrictive licenses compared to unrestrictive licenses also predict significant, lower levels of closure. This is not in line with our fourth hypothesis. For both process-based performance variables, the level of brokerage is a significant, positive predictor, as set by hypothesis 2. Levels of closure are only a significant, negative predictor for the number of reads, but not for the number of commits. Thus, hypothesis 5a is partly confirmed, whereas hypothesis 5b is rejected. Restrictiveness of licenses show no significant total effects for commits, but they do for reads. Highly restrictive licenses compared to unrestrictive licenses have a total negative effect on reads, whereas modestly restrictive licenses compared to unrestrictive licenses have a positive effect on reads. For both outcome-based performance measures, the level of brokerage is a significant, positive predictor, as in line with hypothesis 3. The level of closure only shows positive, significant total effects for page views. Thus, hypothesis 6b is partly confirmed, whereas hypothesis 6a is rejected. Process-based performance measures significantly predict outcome-based performance measures, where higher levels of process leads to higher levels of outcome. This is as expected by hypothesis 7. Highly restrictive licenses are only significant, positive predictors of the number of number of downloads, whereas modestly restrictive licenses are significant, positive predictors of both outcome-based performance measures.

The findings of the SEM analysis are in line with the results of the OLS regression analyses. In particular cases, significance of explanatory variables on predictive variables increased. This did not result in major changes in findings, with exception of the influence of levels of closure on our second outcome-based performance measure (total page views). Findings of the SEM analysis provide more insights in the relations between institutions, network structure and performance.

4.3. Control variables

For both analyses, a wide arrange of control variables is used. The number of developers is viewed as our most important control variable, and shows strong effects for both network measures. The number of developers predicts higher levels of brokerage, whereas it predicts lower levels of closure. For all process-based performance measures the number of developers is a positive predictor. Finally, as showed by the OLS regressions, the number of

developers is no significant predictor of outcome-based performance, whereas the SEM analysis shows a that the number of developers is a significant, positive predictor of both outcome-based performance measures.

Regarding age and the status of the project, OLS regression analyses and SEM analysis report the same findings. Age is a significant, positive predictor for both network measures and outcome-based performance measures. Age shows no distinct pattern for process-based performance. Furthermore, the status of the project is a positive, significant predictor for both network measures, both process-based performance measures and both outcome-based performance measures.

There is evidence for a potential positive, non-linear relationship of age on brokerage. This implies that the shape of the relation between age and brokerage is convex: with higher levels of brokerage at the start and at the end of a project. Also a potential negative, non-linear relationship is observed for process-based performance. Only SEM shows a negative, non-linear relationship for age and outcome-based performance. The type of program language shows no distinct pattern for the OLS regressions analyses. For SEM analysis, languages PHP and C are significant predictors for both network measures, one indicator of process-based performance and outcome-based performance. However, language PHP is a negative predictor, whereas language C is a positive predictor. Finally, the type of users show no distinct pattern in the analyses.

5. Conclusion

Our study demonstrates that overall, institutions affect performance via influencing network structures. This confirms our two meta-hypotheses. For the OS community, restrictiveness of licenses predict network characteristics of OS projects, which affect the process- and outcome-based performance. Through the network approach, our study provides an encompassing view of one of the underlying mechanisms on how institutions affect performance.

Specifically, we found that highly restrictive licenses compared to unrestrictive licenses predict lower levels of brokerage. This supports the argument that developers partake in active self-censorship which decreases opportunities for projects written under more restrictive licenses to take a broker position, while it increases opportunities for projects written under less restrictive licenses. This is an important result, since it points out that

institutions, in the form of licenses, impact network structure. This supports our first meta-hypothesis on the influence of institutions on network structure.

Further, our study showed that higher levels of brokerage predict both higher levels of process- and outcome-based performance. This supports the argument of Burt (2000), who stated that a brokerage position is the source of a wide diversity of unique information and resources, which helps establish a competitive advantage. This is in line with our second meta-hypothesis on the influence of network structure on performance. Furthermore, we found that process-based performance predicts higher levels of outcome-based performance, showing that activity levels of projects are important indicators of actual performance outcomes.

Additionally, this present study tried to present a more complete image of the influence of institutions on performance by providing insights in the total, direct and indirect effects. Findings showed that total effects of highly restrictive licenses on performance were not as clear-cut as predicted. Though highly restrictive licenses indirectly predicted lower levels of performance, these effects were in some cases off-set by opposing direct effects. A possible explanation for the observed differences in direct and indirect effects could be that other mechanisms are here at work. This matter will be discussed in the following section. Noteworthy are also the positive total effects of modestlyly restrictive licenses compared to unrestrictive licenses on three of the four performance measures. These findings suggest that overall, modestlyly restrictive licenses predict higher levels of performance, but that differences in performance are smaller compared to unrestrictive licenses than to highly restrictive licenses. Thus, even small changes in regulations (from highly restrictive to modestlyly restrictive licenses) can cause large differences in performance.

Contrary to our expectations, we found that highly restrictive licenses compared to unrestrictive licenses predict lower levels of closure. This finding questions the self-censorship mechanism, as confirmed earlier with the level of brokerage. Furthermore, levels of closure show no clear pattern in predicting performance. No effects of closure were observed for the first process- and outcome-based performance indicator, while levels of closure do predict lower numbers of reads and higher numbers of page views of OS projects. This provides mixed support for the theories of Burt (2000, 2005) and Coleman (1990). It can be concluded that our second meta-hypothesis on the influence of network structure on performance is partly confirmed, since levels of closure do not affect all indicators of performance. The mixed support regarding levels of closure will be discussed in the following section.

The control variables also yielded some interesting findings. The number of developers contributing to a project predict positive levels of brokerage and negative levels of closure. This finding suggests that an increase in developers improve chances to establish a broker position for the project, whereas it worsens chances to become a closed community. Also, the number of developers positively predicts overall performance of OS projects. Furthermore, the development status of the project is a strong, positive predictor of both network measures and performance. This is consistent with previous research (Stewart & Gosain, 2006).

6. Discussion

Our findings make important contributions to the theoretical understanding of the relationships between institutions, network structures and performance in three aspects. First, our study contributed to the literature on institutions and OS literature, by enhancing the understanding of the influence of institutions on network structure (meta-hypothesis 1). By establishing the relevance of institutions to network structure in a new, rapidly evolving institutional context as the OS community, a new theoretical lens is offered in understanding the role of institutions.

Second, this study complements existing literature by emphasizing the importance of network structure on performance (meta-hypothesis 2). Taking a brokerage position is of great importance in explaining the success of a project. The indistinct pattern of levels of closure gives rise to new questions, as discussed later on.

Third, by combining the two meta-hypotheses, our study presents a new, overall framework on how institutions affect organizational performance. Although prior work has implied the influence of license choice on project performance, the workings of these relations have not been explored in depth. We showed that licenses can have opposing direct and indirect effects on project performance. This finding suggests that different underlying mechanisms play a role: though restrictiveness of licenses indirectly predicts lower levels of project performance because of the negative influence on brokerage, restrictiveness of licenses can also result in positive project performance because of other factors. A possible explanation is that ideological views included in more restrictive licenses provide benefits (Colazo & Fang, 2009). By revealing differences in direct and indirect effects of institutions, this study offers important input for different underlying mechanisms that could be at work in explaining organizational performance.

Methodologically, we have extended previous related empirical studies by making use of a unique, representative dataset on network structure and performance of OS projects of *SourceForge*. To make our outcomes as reliable as possible, we have tested for multicollinearity and heteroskedasticity. Because there was serious evidence for heteroskedasticity, it was decided to work with heteroskedasticity-robust standard errors. Furthermore, and most importantly, our study made use of SEM analysis in addition to OLS regression analyses to differentiate between the total, direct and indirect effects of licenses on network structure and performance. This provided some important advantages compared to OLS, since (1) multiple equations could be modelled simultaneously, (2) more flexible assumptions were allowed for, and (3) the model could be estimated with maximum likelihood with missing values (mlmv). This latter benefit also provided a solution for the large amount of missings with regard to the process measure commits.

In addition to the theoretical implications of this study, our findings on social structures in organizational relations with third parties can have important implications for efficient governance via contractual and non-contractual means (Granovetter, 1985). Our study suggests that to increase organizational performance, institutions should impose regulations that stimulate cooperation, by on the one hand lowering the risk (Coleman, 1990) and on the other hand increasing the value (Burt, 2000).

First, lowering the risk of cooperation can be established by reducing trust problems of organizations, which can increase attractiveness of relation-specific investments and therefore stimulate performance. For example, by facilitating property rights or patents institutions can provide safe structures for interaction. Specifically for the OS community, code should be authorised to avoid risks of third party claims and to stimulate incentives to innovate. Since highly restrictive licenses are only minimally excludable, some critics have argued that these type of licenses are socially detrimental, because they reduce the incentives to innovate (Mundie, 2001). In this view, governments and organizations should stimulate less restrictive licensing.

Second, increasing the value of cooperation can be established by facilitating networks that bring together alters with different types of resources and capabilities. Organizations that can exploit these different resources in their benefit, can create a competitive advantage which results in higher organizational performance. Property rights and patents are also of importance, by helping to maintain a competitive advantage. However, governments and

organizations should take the institutional context in mind, as property rights and patents may conflict with the ideological views of the participants (Hertel, Niedner & Hermann, 2003).

Overall, the findings illustrate a clear picture of the effects of institutions on the four key indicators chosen for performance of OS projects. However, the findings of the present study need to be taken in proximity with an awareness of several limitations.

First, in this study we were unable to assess sample selection bias. Sample selection bias may arise for two reasons: (1) because of self-selection by individuals being investigated; and (2) because of selection decisions made by analysts (Heckman, 1979). Sample selection bias according to this first reason is plausible since the commit-data included a large amount of missings. This could indicate development activities outside the platform of submitting commits and have consequences for self-selection by individuals. However, by estimating SEM analysis with maximum likelihood with missing values (mlmv), the problem of the missings is sought to be solved. Furthermore, since we have decided to only include active projects, possibilities of sample selection bias with regard to the second reason can also arise. A solution for this bias would be the Heckman selection model (Heckman, 1979). This model estimates behavioral relationships free of selection bias in the case of a censored sample. Unfortunately, this test proposes difficulties for SEM analysis, since selection effects for direct and indirect relations cannot be easily computed. Further work should try to apply the Heckman selection procedure to account for sample selection bias.

Second, results of the goodness of fit indexes of SEM indicated that our model did not have a good fit. Findings of SEM could therefore be considered as disappointing or less valuable. However, SEM analysis is only used in this study to illustrate total, direct and indirect paths, and not as main tool to gain insights in the relations between institutions and performance. Therefore, SEM analysis still can provide useful insights. Furthermore, as Kline (2013: 201) stated, "closer to fit in SEM does not mean closer to truth". A possible other explanation for the bad fit of the model could be the many modifications of the original variables used for the data analysis. Further work may try to work with other constructions of the used variables on institutions, network structure and performance, which may result in a better fit of the model. However, theory should always be leading, and complexity of the model should be taken in mind.

Third, an important assumption has been made with regard to level of closure, which provides a possible explanation for the indistinct pattern of this network measure. The level of closure is indicated by transitivity, which is primary an aspect of individual alters, not of

projects. However, transitivity is defined in this study as an indicator of closure *between* projects and not *within* projects. Since this study has no information on normal levels of transitivity of an arbitrary project given network size, it has been assumed that no differentiation exists between levels of transitivity of alters of different types of projects. This is a strong assumption, since it can be expected that alters do differentiate in transitivity levels and clustering of communities may occur. Research showed that alters have preferences for similarity, which can result in clustering of alters with similar characteristics (MPherson, Smith-Lovin, & Cook, 2001). Regarding the OS community, similarity of alters can be expected in license, but also in level of skills or development phase. By assuming that transitivity is not differentiated between types of projects, complexity of underlying mechanisms can be underexposed. This can result in discrepancy. For example, it may happen that transitivity levels *within* a GPL licensed project are higher than *between* this project and his alters, explained by close relations of developers within the project, but few alters of the project sharing developers. Ideally, further work should examine the impact of network measures differentiated to type of project to avoid this kind of discrepancy. However, this will have important complications for complexity of the model, since diverse characteristics of alters can be of importance. Future work should identify relevant characteristics of alters that could explain clustering of communities, and try to reveal the different mechanisms on how this can influence organizational performance.

Fourth, our study showed that institutions affect performance in the institutional context of the OS community. Since data is obtained through *SourceForge*, generalizability to other potential new, evolving institutional contexts is unexamined. However, by highlighting the effects of institutions in this particular context, we hope to attribute to knowledge on different institutional contexts and come closer to understanding of the role of institutions.

Future research can be taken in a couple of other directions. The present findings showed that small changes in regulations may cause major changes in organizational performance, via the mechanism of network structure. Further research should extent the influence of the mechanisms of social structures, by gaining more insights in the required conditions for positive or negative effects on organizational performance, and repeating analyses in other, evolving institutional contexts. Furthermore, as showed by the opposing direct and indirect effects of institutional matters on organizational performance, other mechanisms additional to network structure can play a role in explaining the influence of institutions on organizational performance. For example, identification and ideological views of actors within the

institutional context can be beneficial for performance, even if this context negatively affects opportunities of network structure. Subsequent work should try to identify additional mechanisms that explain the influence of institutions in different institutional contexts.

To arrive at an understanding on how to effectively design institutions to maximize organizational success is an important academic and practical aspiration. Our research hopes to provide one step towards this goal.

Bibliography

Acock, A. (2013). *Discovering structural equation modeling using Stata.* College Station: Stata Press.

Besanko, D., Dranove, D., Shanley, M. & Schaefer, S. (2013). *Economics of strategy* (6th ed.) New York: Wiley.

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

Breusch, T. S., & Pagan, A. R. (1979). A simple test for heteroscedasticity and random coefficient variation. *Econometrica: Journal of the Econometric Society*, 1287-1294.

Burt, R. S. (2005). *Brokerage and closure: An introduction to social capital*. Oxford: OUP.

Burt, R. S. (2000). The network structure of social capital. *Research in organizational behavior*, *22*, 345-423.

Chen, S. (2010). Determinants of survival of open source software: An empirical study. *Academy of Information and Management Sciences Journal*, *13*(2), 119.

CNXSoft. (2011, October 10). Open Source Licenses Overview: GPL, LGPL, Apache, BSD,… Retrieved from http://www.cnx-software.com/2011/10/10/open-source-licenses-overview-gpl-lgpl-apache-bsd/.

Coase, R.H. (1960). The Problem of Social Cost. *Journal of Law and Economics, 3*, 1-44.

Colazo, J., & Fang, Y. (2009). Impact of license choice on open source software development activity. *Journal of the American Society for Information Science and Technology*, *60*(5), 997-1011.

Coleman, J.S. (1990). *Foundations of Social Theory.* Cambridge: Harvard University Press.

Comino, S., Manenti, F. M., & Parisi, M. L. (2007). From planning to mature: On the success of open source projects. *Research Policy, 36(10)*, 1575-1586.

Cowan, R., & Jonard, N. (2003). The dynamics of collective invention. *Journal of Economic Behavior & Organization*, *52*(4), 513-532.

Crowston, K., Annabi, H., & Howison, J. (2003). Defining open source software project success. *ICIS 2003 Proceedings*, 28.

Crowston, K., & Scozzi, B. (2002). Open source software projects as virtual organizations: competency rallying for software development. *IEE Proceedings-Software*, *149*(1), 3-17.

DeLone, W. H., & McLean, E. R. (1992). Information systems success: The quest for the dependent variable. *Information Systems Research*, *3*(1), 60-95.

DeLone, W. H., & McLean, E. R. (2003). The DeLone and McLean model of information systems success: a ten-year update. *Journal of Management Information Systems*, *19*(4), 9-30.

DeLone, W. H., & McLean, E. R. (2002). Information systems success revisited. In *System Sciences, 2002. HICSS. Proceedings of the 35th Annual Hawaii International Conference on* , 2966-2976.

Dyer, J., & Nobeoka, K. (2002). Creating and managing a high performance knowledge-sharing network: the Toyota case. *Strategic Management Journal, 21, 345-367.*

Duijn, M. A., van, Zeggelink, E. P., Huisman, M., Stokman, F. N., & Wasseur, F. W. (2003). Evolution of sociology freshmen into a friendship network.*Journal of Mathematical Sociology*, *27*(2-3), 153-191.

Elster, J. (1982). The case for methodological individualism. *Theory and society*, 453-482.

Evers, S. (2000). An introduction to Open Source software development. *Technische Universität Berlin, Fachbereich Informatik, Fachgebiet Formale Modelle, Logik und Programmierung (FLP).*

Field, A. (2009). *Discovering statistics using SPSS* (3rd edition). Sage Publications.

Fitzgerald, B. (2006). The transformation of open source software. *Mis Quarterly*, 587-598.

Gamson, W. A. (1991). Commitment and agency in social movements. *Sociological Forum*, *6*(1), 27-50. Kluwer Academic Publishers-Plenum Publishers.

Gargiulo, M., Ertug, G., & Galunic, C. (2009). The two faces of control: Network closure and individual performance among knowledge workers. *Administrative Science Quarterly*, *54*(2), 299-333.

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

Goettsch, K.D. (2003). SCO Group v. IBM: The Future of Open-Source Software. *Journal of Law, Technology and Policy*, 581.

Granovetter, M. (1985). Economic action and social structure: the problem of embeddedness. *American Journal of Sociology, 91*(3), 481-510.

Granovetter, M. (2005). The impact of social structure on economic outcomes. *Journal of Economic Perspectives*, *19*(1), 33-50.

Granovetter, M. S. (1973). The strength of weak ties. *American Journal of Sociology, 78*(6), 1360-1380.

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

Gutwin, C., Penner, R., & Schneider, K. (2004). Group awareness in distributed software development. In *Proceedings of the 2004 ACM conference on Computer supported Cooperative Work*, 72-81.

Harary, F., Cartwright, D. & Norman, R. (1965). Structural models: An introduction to the theory of directed graphs. New York: Wiley.

Hansen, G. S., & Wernerfelt, B. (1989). Determinants of firm performance: The relative importance of economic and organizational factors. *Strategic Management Journal*, *10*(5), 399-411.

Heckman, J. J. (1979). Sample selection bias as a specification error. *Econometrica: Journal of the Econometric Society*, 153-161.

Henley, M., & Kemp, R. (2008). Open source software: an introduction. *Computer Law & Security Review*, *24*(1), 77-85.

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

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

Indiana University. (2013, September 5). What is a linker, and what are dynamic and static linking? Retrieved from https://kb.iu.edu/d/akqn

Jemison, D. B., & Sitkin, S. B. (1986). Corporate acquisitions: A process perspective. *Academy of Management Review*, *11*(1), 145-163.

Kapitsaki, G. M., Tselikas, N. D., & Foukarakis, I. E. (2015). An insight into license tools for open source software systems. *Journal of Systems and Software*, *102*, 72-87.

Kerstetter, J. (February 1, 2004) The Most Hated Company in Tech. Retrieved from http://www.bloomberg.com/bw/stories/2004-02-01/the-most-hated-company-in-tech.

Kline, R.B. (2013). *Principles and practice of structural equation* (3rd ed.). New York, London: The Guilford Press.

Kossinets, G. (2006). Effects of missing data in social networks. *Social networks*, *28*(3), 247-268.

Lerner, J. & Tirole, J. (2002). Some Simple Economics of Open Source. *The Journal of Industrial Economics*, 50, 197-234.

Lerner, J. & Tirole, J. (2005). The Scope of Open Source Licensing. *The Journal of Law, Economics & Organization*, *21*(1).

Mantzavinos, C., North, D. C., & Shariq, S. (2004). Learning, institutions, and economic performance. *Perspectives on politics*, *2*(1), 75-84.

Marti, D. (2003, May 19). Torvalds Suggests DiBona for SCO Panel. Retrieved from http://www.linuxjournal.com/article/6879

McEvily, B. & Zaheer, A. (1999). Bridging ties: A source of firm heterogeneity in competitive capabilities. *Strategic Management Journal*, *20*(12), 1133-1156.

McKusick, M. K. (1999). Twenty years of Berkeley Unix: From AT&T-owned to freely redistributable. *Open Sources: Voices from the Open Source Revolution*, 31-46.

Mundie, Craig. 2001. ‘‘The Commercial Software Model,’’ available at http://www.microsoft.com/presspass/exec/craig/05-03sharedsource.asp (accessed September 17, 2002).

Myers, R. (1990). *Classical and modern regression with applications* (2nd ed.). Boston: Duxbury.

North, D. C. (1990). *Institutions, institutional change and economic performance*. Cambridge: Cambridge University Press.

North, D. C. (1992). *Transaction costs, institutions, and economic performance.* 13-15. San Francisco: ICS Press.

Onetti, A., & Capobianco, F. (2005). Open source and business model innovation. The

Funambol case. In *Proceedings of the first International Conference on Open source Systems*, 224-227. Raymond, E.S & Landley, R. (2003). SCO-vs.-IBM: the Open Source Initiative Position Paper on the Complaint. *Journal of AAUGN Inc, 24*(2).

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

Reniers, R. L., Corcoran, R., Drake, R., Shryane, N. M., & Völlm, B. A. (2011). The QCAE: A questionnaire of cognitive and affective empathy. *Journal of Personality Assessment, 93*(1), 84-95.

Roberts, J. A., Hann, I. H., & Slaughter, S. A. (2006). Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the Apache projects. *Management Science, 52*(7), 984-999.

Scacchi, W. (20022) Understanding the Requirements for Developing Open Source Software Systems. *IEE Proceedings Software, 149*(1), 24-39.

Scott, W.R. (2003). *Organizations: Rational, Natural, and Open System* (5th ed.) New Jersey: Pearson Education, Inc.

Sen, R., Subramaniam, C., & Nelson, M. L. (2011). Open source software licenses: Strong-copyleft, non-copyleft, or somewhere in between?. *Decision support systems*, *52*(1), 199-206.

Shared libraries. (1999, June 15). Retrieved from http://www.iecc.com/linker/linker09.html.

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

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

Simon, B., Loewy, M., Stürmer, S., Weber, U., Freytag, P., Habig, C. & Spahlinger, P. (1998). Collective identification and social movement participation. *Journal of Personality and Social Psychology*, *74*(3), 646.

Seddon, P. B. (1997). A Respecification and Extension of the DeLone and McLean Model of IS Success. *Information Systems Research*, 8(3), 240-253.

Snijders, T. A. (2001). The statistical evaluation of social network dynamics. *Sociological methodology*, *31*(1), 361-395.

SourceForge. Retrieved from http://sourceforge.net/ at 2015, April 20.

Stewart, K., Ammeter, A. and Maruping, L.M. (2006). Impacts of license choice and organizational sponsorship on user interest and development activity in open source software projects. *Information Systems Research, 17,* 126–144.

Stewart, K. J., & Gosain, S. (2006). The moderating role of development stage in free/open source software project performance. *Software Process: Improvement and Practice, 11*(2), 177-191.

Subramaniam, C., Sen, R., & Nelson, M. L. (2009). Determinants of open source software project success: A longitudinal study. *Decision Support Systems, 46*(2), 576-585.

Tam Cho, W. K., & Fowler, J. H. (2010). Legislative success in a small world: Social network analysis and the dynamics of congressional legislation. *The Journal of Politics, 72*(1), 124-135.

Uzzi, B. (1997). Social structure and competition in interfirm networks: The paradox of embeddedness. *Administrative Science Quarterly,* 35-67.

Williamson, O. E. (2000). The new institutional economics: taking stock, looking ahead. *Journal of Economic Literature,* 595-613.

Wynn, D. E. (2003). Organizational structure of open source projects: A life cycle approach. In *Abstract for 7th Annual Conference of the Southern Association for Information Systems, Georgia*.

Xiao, X. (2013). Structural equation modeling compared with ordinary least squares in simulations and life insurers' data.

Zaheer, A., & Bell, G. G. (2005). Benefiting from network position: firm capabilities, structural holes, and performance. *Strategic Management Journal, 26*(9), 809-825.

Zaheer, A., & Venkatraman, N. (1995). Relational governance as an interorganizational strategy: An empirical test of the role of trust in economic exchange. *Strategic management journal, 16*(5), 373-392.

Appendix 1: SPSS SYNTAX

```
*------------------------------------------------------------------------------
BACHELOR THESIS IRENE WESTRA & ELMIRA VAN DEN BROEK
DAVID MACRO
------------------------------------------------------------------------------.

GET
  FILE='C:\Users\Elmira\Dropbox\1415-BA3-ElmiraVanDenBroek    IreneWestra\data\OSS-
data-cleaned.sav'.

*------------------------------------------------------------------------------
PREPARING THE DATASET BY CREATING A FILTER FOR HIGHLY RESTRICTIVE, RESTRICTIVE AND
UNRESTRICTIVE LICENSES
------------------------------------------------------------------------------.

* FILTER USED FOR ALL ANALYSES

COMPUTE filter_$=(LIC1=1 OR LIC2=1 OR LIC3=1).
VARIABLE LABELS filter_$ '((LIC1=1 OR LIC2=1 OR LIC3=1) (FILTER)'.
VALUE LABELS filter_$ 0 'Not Selected' 1 'Selected'.
FORMATS filter_$ (f1.0).
FILTER BY filter_$.
EXECUTE.

* FILTER USED FOR CHECKS IF HYBRID LICENSED PROJECTS CHANGE OUTCOMES . NOT USED FOR
INTERPRETATION OF FINAL OUTCOMES .

COMPUTE filter_$=(((LIC1=1 OR LIC2=1 OR LIC3=1) &~(LIC1=1 & LIC2=1) &~(LIC1=1 &
LIC3=1) &~(LIC2=1 & LIC3=1))).
VARIABLE LABELS filter_$ '((LIC1=1 OR LIC2=1 OR LIC3=1) &~(LIC1=1 & LIC2=1)
&~(LIC1=1 & LIC3=1) '+'&~ (LIC2=1 & LIC3=1)) (FILTER)'.
VALUE LABELS filter_$ 0 'Not Selected' 1 'Selected'.
FORMATS filter_$ (f1.0).
FILTER BY filter_$.
EXECUTE.

*------------------------------------------------------------------------------
DESCRIPTIONS OF VARIABLES USED IN DATA ANALYSES AND CORRELATIONS BETWEENNESS
------------------------------------------------------------------------------.

FREQUENCIES LIC1 LIC2 LIC3 T60Downloads T60total_pages NDEVELOPERS COMMITS READS
AGE AGE2 STATUSCODE2 LANG1 LANG2 LANG3 LANG4 IA1 IA2 IA3 IA4 .
DESCRIPTIVES LIC1 LIC2 LIC3 T60Downloads T60total_pages NDEVELOPERS COMMITS READS
AGE AGE2 STATUSCODE2 LANG1 LANG2 LANG3 LANG4 IA1 IA2 IA3 IA4 .

FREQUENCIES BETWEENNESS3 BETWEENNESS4 BETWEENNESS5 BETWEENNESS6 BETWEENNESS7
BETWEENNESS8 .
FREQUENCIES BETWEENNESS5 TRANSITIVITY.
DESCRIPTIVES BETWEENNESS5 TRANSITIVITY.

CORRELATIONS
  /VARIABLES=BETWEENNESS3 BETWEENNESS4 BETWEENNESS5 BETWEENNESS6 BETWEENNESS7
BETWEENNESS8
  /PRINT=TWOTAIL NOSIG
  /MISSING=PAIRWISE.

GRAPH
/ SCATTERPLOT (BIVAR) = BETWEENNESS5 WITH NDEVELOPERS .

* BETWEENESS SHOWS HIGH CORRELATIONS. WE DECIDE TO USE BETWEENNESS5 FOR OUR
ANALYSES .
```

```
*------------------------------------------------------------------------------
RECODING VARIABLES
------------------------------------------------------------------------------.

COMPUTE LNBETWEEN5 = LN(BETWEENNESS5+1) .
COMPUTE LNDOWNLOAD = LN(T60Downloads+1) .
COMPUTE LNTOTPAGES = LN(T60total_pages+1) .
COMPUTE AVDEVELOP = NDEVELOPERS/AGE .
COMPUTE AVCOMMIT = COMMITS/AGE .
COMPUTE AVREADS = READS/AGE .
COMPUTE LNAVREAD = LN(AVREADS+1) .
COMPUTE LNAVCOMMIT = LN(AVCOMMIT+1) .

DESCRIPTIVES AGE .
COMPUTE CENAGE = AGE - 3305.123539 .
COMPUTE AGE2 = CENAGE*CENAGE .

*------------------------------------------------------------------------------
TESTING FIRST META-HYPOTHESIS: INSTITUTIONS ON NETWORK STRUCTURE
------------------------------------------------------------------------------.

* HYPOTHESIS 1: LICENSES ON BETWEENNESS. LIC3 IS USED AS REFERENCE CATEGORY .

REGRESSION
  /MISSING LISTWISE
  /STATISTICS COEFF OUTS R ANOVA CHANGE
  /CRITERIA=PIN(.05) POUT(.10)
  /NOORIGIN
  /DEPENDENT LNBETWEEN5
  /METHOD=ENTER LIC1 LIC2 NDEVELOPERS
  /METHOD=ENTER CENAGE AGE2 STATUSCODE2 LANG1 LANG2 LANG3 LANG4 IA1 IA2 IA3 IA4 .

* CONFIRMS HYPOTHSIS 1 .
* HYPOTHESIS 4: LICENSES ON CLOSURE .

REGRESSION
  /MISSING LISTWISE
  /STATISTICS COEFF OUTS R ANOVA CHANGE
  /CRITERIA=PIN(.05) POUT(.10)
  /NOORIGIN
  /DEPENDENT TRANSITIVITY
  /METHOD=ENTER LIC1 LIC2 NDEVELOPERS
  /METHOD=ENTER  CENAGE AGE2 STATUSCODE2 LANG1 LANG2 LANG3 LANG4 IA1 IA2 IA3 IA4 .

* DOES NOT CONFIRM HYPOTHESIS 4 .

*------------------------------------------------------------------------------
TESTING SECOND META-HYPOTHESIS: NETWORK STRUCTURE ON PERFORMANCE
------------------------------------------------------------------------------.

* HYPOTHESIS H2, H5A AND H5B .

REGRESSION
  /MISSING LISTWISE
  /STATISTICS COEFF OUTS R ANOVA CHANGE
  /CRITERIA=PIN(.05) POUT(.10)
  /NOORIGIN
  /DEPENDENT LNAVREAD
  /METHOD=ENTER LNBETWEEN5 TRANSITIVITY NDEVELOPERS
  /METHOD=ENTER  LIC1 LIC2
  /METHOD=ENTER  CENAGE AGE2 STATUSCODE2 LANG1 LANG2 LANG3 LANG4 IA1 IA2 IA3 IA4 .


REGRESSION
  /MISSING LISTWISE
  /STATISTICS COEFF OUTS R ANOVA CHANGE
```

```
  /CRITERIA=PIN(.05) POUT(.10)
  /NOORIGIN
  /DEPENDENT LNAVCOMMIT
  /METHOD=ENTER LNBETWEEN5 TRANSITIVITY NDEVELOPERS
  /METHOD=ENTER  LIC1 LIC2
  /METHOD=ENTER  CENAGE AGE2 STATUSCODE2 LANG1 LANG2 LANG3 LANG4 IA1 IA2 IA3 IA4  .

* CONFIRMS HYPOTHESIS 2 AND PARTLY H5A . DOES NOT CONFIRM HB .
* HYPOTHESIS H3, H6A, H6B AND H7 .

REGRESSION
  /MISSING LISTWISE
  /STATISTICS COEFF OUTS R ANOVA CHANGE
  /CRITERIA=PIN(.05) POUT(.10)
  /NOORIGIN
  /DEPENDENT LNDOWNLOAD
  /METHOD=ENTER LNAVCOMMIT LNAVREAD NDEVELOPERS
  /METHOD=ENTER LNBETWEEN5 TRANSITIVITY
  /METHOD=ENTER LIC1 LIC2
  /METHOD=ENTER CENAGE AGE2 STATUSCODE2 LANG1 LANG2 LANG3 LANG4 IA1 IA2 IA3 IA4  .


REGRESSION
  /MISSING LISTWISE
  /STATISTICS COEFF OUTS R ANOVA CHANGE
  /CRITERIA=PIN(.05) POUT(.10)
  /NOORIGIN
  /DEPENDENT LNTOTPAGES
  /METHOD=ENTER LNAVCOMMIT LNAVREAD NDEVELOPERS
  /METHOD=ENTER LNBETWEEN5 TRANSITIVITY
  /METHOD=ENTER LIC1 LIC2
  /METHOD=ENTER CENAGE AGE2 STATUSCODE2 LANG1 LANG2 LANG3 LANG4 IA1 IA2 IA3 IA4  .

* CONFIRMS HYPOTHESIS 3, 7 AND PARTLY H6B AT THE 10% ALPHA-LEVEL. DOES NOT CONFIRM
H6A .
```

Appendix 2: STATA DO-FILE

```
// Bachelor thesis Irene Westra and Elmira van den Broek

cd "C:\Users\Elmira\Dropbox\1415-BA3-ElmiraVanDenBroek-IreneWestra\data"
use OSS-data-cleaned.dta, replace

// SEM does not work with capital letters
rename _all, lower

// Only keep highly restrictive, restrictive and unrestrictive licenses
keep if lic1 == 1 | lic2 == 1 | lic3 == 1

// Command to check for effect of hybride licensed projects. This is only a check and since this makes no significant differences, these results will not be interpreted
gen lictotal = lic1 + lic2 + lic3
drop if lictotal > 1

// Generate variables used in analyses
qui summ age
gen agec = (age - r(mean)) / 1000
gen agec2 = agec * agec
gen lnbetween5    = log(betweenness5 + 1)
gen avcommit      = commits/age
gen lnavcommit  = log(avcommit + 1)
gen avread        = reads/age
gen lnavread    = log(avread + 1)
gen lndownload    = log(t60downloads + 1)
gen lntotpages    = log(t60total_pages + 1)

global controle  agec agec2 statuscode2 lang1 lang2 lang3 lang4 ia1 ia2 ia3 ia4

// Descriptives variables used in analyses
sum lic1 lic2 lic3 lnbetween5 transitivity lnavcommit lnavread lndownload lntotpages ndevelopers $controle

// Test classical assumptions for OLS regression analyses

// 1. Multicollineairity
nestreg: reg lnbetween5 (ndevelopers lic1 lic2) ($controle)
vif
nestreg: reg transitivity (ndevelopers lic1 lic2) ($controle)
vif
nestreg: reg lnavcommit (lnbetween5 transitivity) (ndevelopers lic1 lic2) ($controle)
vif
nestreg: reg lnavread (lnbetween5 transitivity)(ndevelopers lic1 lic2) ($controle)
vif
nestreg: reg lndownload (lnbetween5 transitivity) (lnavcommit lnavread) (ndevelopers lic1 lic2) ($controle)
vif
nestreg: reg lntotpages (lnbetween5 transitivity) (lnavcommit lnavread) (ndevelopers lic1 lic2) ($controle)
vif

// Assumption met. All VIF are between 1.28 - 1.40.

// 2. Homoskedasticity
// H1: licenses on betweenness
nestreg: reg lnbetween5 (ndevelopers lic1 lic2) ($controle)
estat hettest
// Assumption not met! Perform Breusch-Pagan test
nestreg: reg lnbetween5 (ndevelopers lic1 lic2) ($controle), robust
```

```
// H4: licenses on closure
nestreg: reg transitivity (ndevelopers lic1 lic2) ($controle)
estat hettest
// Assumption not met!
nestreg: reg transitivity (ndevelopers lic1 lic2) ($controle), robust

// H2 and H5: network measures on commits
nestreg: reg lnavcommit (ndevelopers lnbetween5 transitivity) (lic1 lic2)
($controle)
estat hettest
// Assumption not met!
nestreg: reg lnavcommit (ndevelopers lnbetween5 transitivity) (lic1 lic2)
($controle), robust

// H2 and H5: network measures on reads
nestreg: reg lnavread (ndevelopers lnbetween5 transitivity)(lic1 lic2) ($controle)
estat hettest
// Assumption not met!
nestreg: reg lnavread (ndevelopers lnbetween5 transitivity)(lic1 lic2) ($controle),
robust

// H3, H6 and H7: network measures on downloads
nestreg: reg lndownload (ndevelopers lnbetween5 transitivity) (lnavcommit lnavread)
(lic1 lic2) ($controle)
estat hettest
// Assumption not met.
nestreg: reg lndownload (ndevelopers lnbetween5 transitivity) (lnavcommit lnavread)
(lic1 lic2) ($controle), robust

// H3, H6 and H7: network measures on total pages
nestreg: reg lntotpages (ndevelopers lnbetween5 transitivity) (lnavcommit lnavread)
(lic1 lic2) ($controle)
estat hettest
// Assumption not met.
nestreg: reg lntotpages (ndevelopers lnbetween5 transitivity) (lnavcommit lnavread)
(lic1 lic2) ($controle), robust

// Conclusions regarding heteroskedasticity: All hypotheses are heteroskedastic.

// The following OLS regressions can thus be interpreted:
nestreg: reg lnbetween5 (ndevelopers lic1 lic2) ($controle), robust
nestreg: reg transitivity (ndevelopers lic1 lic2) ($controle), robust
nestreg: reg lnavcommit (ndevelopers lnbetween5 transitivity) (lic1 lic2)
($controle), robust
nestreg: reg lnavread (ndevelopers lnbetween5 transitivity) (lic1 lic2)
($controle), robust
nestreg: reg lndownload (ndevelopers lnbetween5 transitivity) (lnavcommit lnavread)
(lic1 lic2) ($controle), robust
nestreg: reg lntotpages (ndevelopers lnbetween5 transitivity) (lnavcommit lnavread)
(lic1 lic2) ($controle), robust

// SEM analysis
// 1. SEM excluding missings

sem   ( lnbetween5     <- ndevelopers lic1 lic2 $controle  )  ///
      ( transitivity   <- ndevelopers lic1 lic2  $controle ) ///
      ( lnavcommit     <- ndevelopers lic1 lic2 lnbetween5 transitivity $controle )
///
      ( lnavread       <- ndevelopers lic1 lic2 lnbetween5 transitivity $controle )
///
      ( lndownload     <- lnavcommit lnavread ndevelopers lic1 lic2 lnbetween5
      transitivity  $controle  )  ///
      ( lntotpages     <- lnavcommit lnavread ndevelopers lic1 lic2 lnbetween5
      transitivity  $controle  )
```

```
// Compute missing paths:
estat mindices

// Estimate total effects and goodness of fit statistics
estat teffects
estat gof, stats(all)

// 2. SEM including missings

sem   ( lnbetween5      <- ndevelopers lic1 lic2 $controle  )  ///
      ( transitivity    <- ndevelopers lic1 lic2  $controle ) ///
      ( lnavcommit      <- ndevelopers lic1 lic2 lnbetween5 transitivity $controle )
      ///
      ( lnavread        <- ndevelopers lic1 lic2 lnbetween5 transitivity $controle
      )///
      ( lndownload      <- lnavcommit lnavread ndevelopers lic1 lic2 lnbetween5
      transitivity   $controle  )  ///
      ( lntotpages      <- lnavcommit lnavread ndevelopers lic1 lic2 lnbetween5
      transitivity  $controle  ), method(mlmv)

// Compute missing paths:
estat mindices

// Estimate total effects and goodness of fit statistics
estat teffects
estat gof, stats(all)

// 3. SEM including missings and robust checks

sem   ( lnbetween5      <- ndevelopers lic1 lic2 $controle  )  ///
      ( transitivity    <- ndevelopers lic1 lic2  $controle ) ///
      ( lnavcommit      <- ndevelopers lic1 lic2 lnbetween5 transitivity $controle )
      ///
      ( lnavread        <- ndevelopers lic1 lic2 lnbetween5 transitivity $controle
      )///
      ( lndownload      <- lnavcommit lnavread ndevelopers lic1 lic2 lnbetween5
      transitivity   $controle  )  ///
      ( lntotpages      <- lnavcommit lnavread ndevelopers lic1 lic2 lnbetween5
      transitivity  $controle  ), method(mlmv) vce(robust)

// Estimate total effects and goodness of fit statistics
estat teffects
estat gof, stats(all)

// SEM including missings and robust checks will be interpreted as our final model
```

Appendix 3: Selected SPSS and STATA output

Table 1. Correlations betweenness measures.

betweenness3	betweeness4	betweeness5	betweeness6	betweeness7	betweeness8	
betweenness3	1					
betweenness4	0.9893	1				
betweenness5	0.9806	0.9978	1			
betweenness6	0.9744	0.9933	0.9984	1		
betweenness7	0.9685	0.9871	0.9942	0.9986	1	
betweenness8	0.9625	0.9806	0.989	0.9954	0.999	1

Table 2. VIF predictors.

Variable	VIF H1 and H4	VIF H2 and H5	VIF H3 and H6
Level of brokerage		1.56	1.63
Level of closure		1.02	1.03
Number of commits			2.39
Number of reads			2.6
Highly restrictive license	1.85	1.84	1.84
Restrictive license	1.83	1.82	1.82
Number of developers	1.04	1.45	1.81
Age	1.39	1.38	1.42
Age^2	1.17	1.06	1.08
Status of projects	1.04	1.1	1.12
Language: Java	1.34	1.34	1.34
Language: C++	1.18	1.2	1.2
Language: PHP	1.19	1.19	1.19
Language: C	1.2	1.15	1.15
Users: Developers	1.26	1.28	1.28
Users: End users/Desktop	1.23	1.25	1.25
Users: System Administrators	1.1	1.1	1.1
Users: Advanced End Users	1.07	1.05	1.05
Mean VIF	*1.28*	*1.3*	*1.46*

Table 3. SEM Analysis total, direct and indirect effects including control variables.

	Total effects		Direct effects		Indirect effects	
	B	SE(B)	B	SE(B)	B	SE(B)
(Log) level of betweenness						
Number of developers	0.213***	0.023	0.213***	0.023	0	
Highly restrictive license	-0.118**	0.040	-0.118**	0.040	0	
Restrictive license	0.061	0.046	0.061	0.046	0	
Age	0.299***	0.019	0.299***	0.019	0	
Age^2	0.129***	0.016	0.129***	0.016	0	
Status of projects	0.073***	0.007	0.073***	0.007	0	
Language: Java	0.001	0.033	0.001	0.033	0	
Language: C++	0.026	0.032	0.026	0.032	0	
Language: PHP	-0.116***	0.032	-0.116***	0.032	0	
Language: C	0.315***	0.035	0.315***	0.035	0	
Users: Developers	0.082**	0.025	0.082**	0.025	0	
Users: End users/Desktop	-0.001	0.025	-0.001	0.025	0	
Users: System Administrators	-0.109**	0.033	-0.109**	0.033	0	
Users: Advanced End Users	0.035	0.031	0.035	0.031	0	
Level of closure						
Number of developers	-0.002***	0.000	-0.002***	0.000	0	
Highly restrictive license	-0.005**	0.002	-0.005**	0.002	0	
Restrictive license	-0.002	0.002	-0.002	0.002	0	
Age	0.003***	0.001	0.003***	0.001	0	
Age^2	0.000	0.001	0.000	0.001	0	
Status of projects	0.004***	0.000	0.004***	0.000	0	
Language: Java	0.007***	0.001	0.007***	0.001	0	
Language: C++	-0.003*	0.001	-0.003*	0.001	0	
Language: PHP	-0.010***	0.002	-0.010***	0.002	0	
Language: C	0.004**	0.002	0.004**	0.002	0	
Users: Developers	0.006***	0.001	0.006***	0.001	0	
Users: End users/Desktop	-0.001	0.001	-0.001	0.001	0	
Users: System Administrators	0.000	0.002	0.000	0.002	0	
Users: Advanced End Users	0.000	0.002	0.000	0.002	0	
(Log) average commits						
(Log) level of betweenness	0.004**	0.001	0.004**	0.001	0	
Level of closure	0.005	0.013	0.005	0.013	0	
Number of developers	0.011***	0.001	0.010***	0.001	0.001**	0.000
Highly restrictive license	-0.008	0.005	-0.007	0.005	-0.001*	0.000
Restrictive license	0.006	0.006	0.006	0.006	0.000	0.000
Age	-0.019***	0.003	-0.020***	0.003	0.001**	0.000

Age^2	-0.016***	0.002	-0.016***	0.002	0.001*	0.000
Status of projects	0.007***	0.001	0.006***	0.001	0.000*	0.000
Language: Java	0.007*	0.003	0.007*	0.003	0.000	0.000
Language: C++	0.008*	0.003	0.008*	0.003	0.000	0.000
Language: PHP	-0.003	0.004	-0.002	0.004	-0.001*	0.000
Language: C	-0.004	0.004	-0.005	0.004	0.001**	0.000
Users: Developers	0.005	0.003	0.004	0.003	0.000*	0.000
Users: End users/Desktop	0.002	0.003	0.002	0.003	0.000	0.000
Users: System Administrators	-0.011*	0.005	-0.010*	0.005	0.000*	0.000
Users: Advanced End Users	0.013**	0.004	0.013**	0.004	0.000	0.000
(Log) average reads						
(Log) level of betweenness	0.035***	0.003	0.035***	0.003	0	
Level of closure	-0.040*	0.017	-0.040*	0.017	0	
Number of developers	0.037***	0.003	0.030***	0.003	0.007***	0.001
Highly restrictive license	-0.027**	0.008	-0.023**	0.008	-0.004**	0.001
Restrictive license	0.023*	0.010	0.020*	0.010	0.002	0.002
Age	0.004	0.003	-0.006**	0.002	0.010***	0.001
Age^2	-0.004	0.003	-0.009**	0.003	0.004***	0.001
Status of projects	0.026***	0.001	0.024***	0.001	0.002***	0.000
Language: Java	0.011	0.006	0.011*	0.005	0.000	0.001
Language: C++	0.018**	0.006	0.017**	0.006	0.001	0.001
Language: PHP	-0.015*	0.006	-0.012	0.006	-0.004**	0.001
Language: C	0.032***	0.006	0.022***	0.006	0.011***	0.002
Users: Developers	0.007	0.004	0.005	0.004	0.003**	0.001
Users: End users/Desktop	0.005	0.004	0.005	0.004	0.000	0.001
Users: System Administrators	-0.012*	0.006	-0.008	0.006	-0.004**	0.001
Users: Advanced End Users	0.022**	0.007	0.021**	0.006	0.001	0.001
(Log) number of downloads						
(Log) level of betweenness	0.169***	0.012	0.119***	0.013	0.050***	0.013
Level of closure	-0.059	0.133	-0.077	0.166	0.017	0.108
(Log) average commits	8.098***	0.415	8.098***	0.415	0	(no
(Log) average reads	0.493***	0.075	0.493***	0.075	0	(no
Number of developers	0.091***	0.011	-0.044***	0.009	0.135***	0.013
Highly restrictive license	0.086*	0.042	0.178**	0.056	-0.092*	0.042
Restrictive license	0.133**	0.047	0.064	0.064	0.069	0.049
Age	0.372***	0.017	0.485***	0.025	-0.113***	0.022
Age^2	-0.122***	0.016	-0.007	0.023	-0.115***	0.018
Status of projects	0.633***	0.008	0.557***	0.009	0.076***	0.007
Language: Java	0.007	0.032	-0.054	0.040	0.061*	0.028
Language: C++	0.142***	0.034	0.063	0.043	0.078**	0.029
Language: PHP	-0.278***	0.039	-0.235***	0.051	-0.043	0.036

Language: C	0.138***	0.035	0.118*	0.046	0.021	0.033
Users: Developers	0.039	0.028	-0.013	0.037	0.052*	0.026
Users: End users/Desktop	0.318***	0.028	0.297***	0.038	0.021	0.027
Users: System Administrators	-0.037	0.036	0.070	0.052	-0.107**	0.039
Users: Advanced End Users	0.176***	0.041	0.053	0.050	0.123***	0.032
(Log) total page views						
(Log) level of betweenness	0.196***	0.013	0.143***	0.015	0.053***	0.014
Level of closure	0.500**	0.144	0.482**	0.179	0.019	0.114
(Log) average commits	8.600***	0.451	8.600***	0.451	0	
(Log) average reads	0.513**	0.074	0.513***	0.074	0	
Number of developers	0.102***	0.013	-0.043***	0.011	0.145***	0.014
Highly restrictive license	-0.039	0.045	0.063	0.061	-0.102*	0.045
Restrictive license	0.112*	0.051	0.039	0.069	0.073	0.052
Age	0.348***	0.018	0.462***	0.027	-0.114***	0.023
Age^2	-0.031	0.017	0.089***	0.025	-0.120***	0.019
Status of projects	0.463***	0.009	0.379***	0.010	0.084***	0.007
Language: Java	0.064	0.035	-0.004	0.043	0.068*	0.030
Language: C++	0.059	0.037	-0.022	0.046	0.081**	0.031
Language: PHP	-0.125**	0.044	-0.071	0.057	-0.053	0.038
Language: C	0.176***	0.038	0.147**	0.049	0.029	0.035
Users: Developers	0.046	0.030	-0.014	0.040	0.060*	0.028
Users: End users/Desktop	0.199***	0.031	0.177***	0.041	0.022	0.028
Users: System Administrators	0.001	0.039	0.117*	0.057	-0.116**	0.042
Users: Advanced End Users	0.219***	0.044	0.089	0.054	0.130***	0.034

Appendix 4: Logbook

Date	*Time*	*Activity*	*Who*
03 – 02	2h	Discuss possible topics for bachelor project	Elmira, Irene
12 – 02	2h	Looking into the data and discussing possible topics First meeting with David Macro	Elmira, Irene
13 – 02	2h	Narrowing down topic Sorting out which variables we want to use in our research and which ones are 'missing' in the dataset. Informing David Macro about the requested variable.	Elmira, Irene
16 - 02	5h	Discussing how the new variable (developer skills) can be converted into a useful and fitting variable Searching and reading literature needed for theoretical framework Handing in the concept version of the research question	Elmira, Irene
20 - 02	2h	Working on introduction	Elmira, Irene
23 - 02	4h	Fine tuning introduction Working on theory outline	Elmira, Irene
26 - 02	2h	Meeting with David Macro about our introduction and the new research proposal	Elmira, Irene
06 – 03	2.5h	Working on theory Dividing tasks for theory chapter	Elmira, Irene
16 - 03	4h	Institutions, performance part of theory Licenses, social network part of theory	Elmira Irene
20 - 03	3h	Institutions, performance part of theory Licenses, social network part of theory	Elmira Irene
24 – 03	2.5h	Rewrite theory for deadline 25/3	Elmira
27 – 03	2h 5h	Theory 'link institutions licenses' Theory 'link licenses developers'	Elmira Irene
28 – 03	0.5h	Feedback theory 'social network'	Elmira
29 – 03	3h	Rewrite part 'performance'	Elmira
30 – 03	2h	Rewrite 'performance' Theory 'social network'	Elmira, Irene
31-03	1.5h	Last checks and additions to theory chapter	Irene
31 - 03	1h	Rewrite 'performance'	Elmira
02-04	1h	Last additions to theory and hypotheses	Irene
14-04	1.5h	Start Data & Methods section	Elmira
15-04	1h	Data & Methods section	Elmira
17-04	2h	Data & Methods section	Elmira, Irene

23-04	1h	Meeting with David	Elmira, Irene
28-04	2.5h	Revising Theory chapter	Elmira, Irene
29-04	1h	Revising Theory chapter	Elmira
19-05	2h	Meeting with David on syntax	Elmira
20-05	2h	Adjusting syntax, adding more layers in the models	Irene
22-05	3h	Meeting/adjust syntax	Elmira, Irene
23-05	2h	Adjust syntax	Elmira
24-05	2h	Rewrite Introduction	Elmira
25-05	2h	Checking and adjusting Introduction	Irene
25-05	2h	Rewrite Theory	Elmira
28-05	1,5h	Write draft results section	Elmira
29-05	1h	Check draft Results	Irene
		First version of Regression tables	
30-05	1h	Result section	Elmira
1-06	2h	Meeting Introduction/Theory and planning coming weeks	Elmira, Irene
2-06	3h	Run SEM analysis + interpretation	Elmira
	3h	Completing Results	Irene
		Composing Regression Tables APA	
3-06	1h	Completing Results + SEM analysis	Elmira
4-06	3,5h	Completing results + SEM analysis	Elmira
5-06	3h	Adding ‘Control variables’ to Results	Irene
		Completing Introduction	
		Working on feedback in Theory	
	1h	Meeting David	Elmira, Irene
6-06	5h	Tables SES in STATA	Elmira
		Perform OLS regressions in STATA	
		Checking for classical assumptions OLS	
		Correct for heteroskedasticity	
		Descriptives table Data&Methods	
7-06	2h	Include assumptions in Results	Elmira
		Adjust Results with new coefficients/p-values final results OLS regression analyses	
7-06	0.5h	Finetuning Theory	Irene
8-06	1.5h	Discussing STATA Results	Elmira, Irene
	3h	Discussing what we have to do these last weeks	
		Composing new regression tables with adjusted variable	Irene
		Adjusting Results and Data&Methods chapters	
		Creating simple version of path model	
9-06	1h	Meeting	Elmira, Irene
	2h	Data & Methods, Results	Elmira
			Irene
10-06	1.5h	Data & Methods	Elmira
11-06	3h	Finish Data & Methods	Elmira

		SEM Tables Introduction	
	2h	Rewrite SEM part Individual part: Discussion & Conclusion	
12-06	7h	Finetuning of thesis	Irene
	4h		Elmira
13-06	8h	Finishing up thesis: applying feedback, Appendix, tables, literature list, etc.	Elmira, Irene
14-06	7h	Idem.	Elmira, Irene
15-15-	8h	Worked individually on Conclusion and Discussion	Elmira, Irene
18-06	8h	Applying the final feedback	Irene
	3h		Elmira
19-06	3h	Last checks thesis and handing it in	Elmira, Irene