7-4-2007

The Determinants of Open Source Quality: An Empirical Investigation

Ioana Popovici

The Determinants of Open Source Quality: An Empirical Investigation

Ioana Popovici

Florida International University

April 2007

Open source (OS) licenses differ in the conditions under which licensors and OS contributors are allowed to modify and redistribute the source code. While recent research has explored the determinants of license choice, we know little about the impact of license choice on project success. In this paper, we measure success by the speed with which programming bugs are fixed. Using data obtained from SourceForge.net, a free service that hosts OS projects, we test whether the license chosen by project leaders influences bug resolution rates. In initial regressions, we find a strong correlation between the hazard of bug resolution and the use of highly restrictive licenses. However, license choices are likely to be endogenous. We instrument license choice using (i) the human language in which contributors operate and (ii) the license choice of the project leaders for a previous project. We then find weak evidence that restrictive licenses adversely affect project success.

JEL Classifications: L86, K39, O30

Keywords: open source software, property rights, copy-left

Department of Economics, Florida International University, Miami, FL 33199. email: Ioana.Popovici@fiu.edu. Helpful comments from Peter Thompson are gratefully acknowledged.

1. Introduction

The source code of open source (OS) software programs is made available for everybody to use, modify, or redistribute free of charge, but under certain conditions. Usually, the contributors to OS projects are not paid for their work and are free to make contributions to the area that might interest them. Nevertheless, certain OS products have become predominant in their category (e.g. the Apache web server or the Linux Operating System) and large corporations (like IBM, Sun, or HP) have launched major projects to develop OS products. The OS phenomenon seems to contradict basic economic principles, creating puzzles for economic analysis: Why do developers around the world make voluntary and unpaid contribution to the provision of a public good? What makes OS programs thrive in a world dominated by proprietary standards?

A recent surge in interest in the economics of OS has attempted to shed light on a number of issues. Some authors explore the contributors' incentives to write OS code [Ghosh et al (2002), Lerner and Tirole (2002, 2004)] or the identity of the contributors [Mockus et al (2002), Von Krogh et al (2003), Lakhani and von Hippel (2000)]. Others analyze the need of restrictive licenses [Dahlander and Magnusson (2005), Lerner and Tirole (2002b), Bonaccorsi and Rossi (2003)] or the quality of OS relative to proprietary software [Kuan (2001), Bessen (2002)]. One important question is what makes some OS projects thrive, while others are abandoned from the start. What affects the chances of a project becoming successful? Most OS projects are initiated by one or more individuals, the leaders of the project, who write a piece of code and release it under some OS license hoping it will attract other contributions that will build on the initial code. This initial mass of code is critical for the future success of the project. Its quality, the promise of exciting challenging programming problems, and the demand for the application are all factors that affect its appeal to potential contributors. The reputation of the leader is important in attracting new contributions as well. A credible leader will make the initial code more appealing to the programming community. An unknown leader can compensate and attract contributors by writing quality code and identifying challenging tasks that will need to be solved in the future [Lerner and Tirole (2002)].

Besides all of the above, the OS license choice could be one of the factors that affect the success of the project, as programmers seem to care critically about it. Indeed, when licenses terms are altered, there is considerable controversy. In 2002, Wine (a program loader

capable of running on Linux and other POSIX compatible operating systems applications originally written for Windows) switched from an unrestricted license according to which modified source code does not have to be made freely available as long as the original source is acknowledged, to one that requires that. Wine's alteration to the licensing terms created considerable furor among contributors.

In this paper, we explore whether certain types of licenses attract new contributions and foster the innovation process more readily than others. We analyze whether employing a specific OS license affects the success of the project, if certain types of licenses make projects more likely to thrive, receive ongoing contributions, and have a greater life span than others.

Although the source code of an OS program is publicly available, the OS innovation method rests on property rights. Projects are initiated under OS licenses that differ in the conditions under which licensors and contributors are allowed to modify and redistribute the source code. On one hand, restrictive "copy-left" licenses like the Netscape's Mozilla Public License require that the modified code be made available for free use, modification, and redistribution. Unrestrictive licenses, like the Berkeley Software Distribution (BSD) license do not impose this restriction. On the other hand, project leaders have the choice to employ highly restrictive licenses, like the "viral" GNU General Public License (GPL), that requires any piece of modified code be released under the same license. Less restrictive licenses like the Lesser GPL (LGPL) require that the altered code be made generally available, but not necessarily under the same license. It is not obvious whether the OS method of innovation needs licenses and what restrictions are necessary when licenses are needed [Maurer, Scotchmer (2006)]. The existing empirical literature focuses on the determinants of OS license choice at firm or project level [Lerner and Tirole (2005), Koski (2005), Bonaccorsi and Rossi (2003)]. Nevertheless, none of the previous studies analyzes whether the license adopted affects the outcome, the success and quality of the OS project. Our paper explores whether the initial license choice decision made by the leaders influences project success.

The question of what motivates software programmers to contribute to OS programs has been a major one. Theory suggests a number of incentives that drive developers to participate in OS projects. They might work to fix a bug or customize a program for their own benefit; they might want to signal talent to their peers or potential future employers when solving a challenging problem; or they might want to improve their

programming skills or simply enjoy working on a challenging programming task [Lerner and Tirole (2002), Lerner and Tirole (2004)].

Because of these various motivations, the relationship between license type and developer contributions is *a priori* ambiguous. On the one hand, licenses that are more restrictive might encourage OS developers to contribute. Highly restrictive licenses like the GPL do not allow the bundling of the open source code with proprietary code. In contrast, less restrictive licenses do not prevent commercial software firms from "hijacking" OS code, mixing it with proprietary code and selling the resulting product. When hijacking occurs, OS contributors are inevitably deprived of some of the benefits they had expected from writing code, benefits that had motivated them to make their contribution in the first place (Lerner and Tirole, 2005). OS contributors might have to pay for the final product, they might be unable to modify it for their own use, and their contributions might not be visible in the final product sold by the commercial firm. The career prospects and talent-signaling incentive is stronger the more visible and informative about talent is the performance of the programmer [Holmstrom (1999)]. Moreover, the OS community might lose interest in the project when a commercial variant is sold, making the initial OS contributions less visible.

However, there are also reasons why less restrictive licenses might attract greater involvement. First, some OS developers may have their own use of the code in mind and less restrictive licenses give them greater freedom to develop as they see fit their own projects that incorporate the OS code. In particular, they may be attracted to working on problems with unrestrictive licenses by the prospect of commercializing their own adaptation of the code. Second, developers may believe that many OS projects will be of little value unless they are supplemented with complementary proprietary code, and restrictive licenses may therefore diminish their chance of success [Lerner and Tirole (2005)].

Measuring the success of a software project is a difficult task. One aspect of success is the eventual quality of the software, but there are numerous dimension of quality including flexibility, robustness, ease-of-use, and speed. Kuan (2002) proposes measuring the bug resolution rate (the period of time in which bugs are fixed) as a proxy for the rate of quality improvement. She argues that the list of bugs itself would not be a good quality proxy, as a long list might only suggest bugs are discovered more frequently, and discovery might be driven in large part by the perceived quality of an application. We follow Kuan (2002) and employ the bug resolution time as a proxy for quality improvement.

We use the SourceForge.net, a free service that hosts about 100,000 open source projects initiated since 1999, to construct a dataset consisting of over 300 projects and more than 1,700 bugs. We estimate non-parametric Kaplan Meier survival curves and Cox proportional hazard models to show, as hypothesized, that projects with highly restrictive licenses attract enough effort to have their bugs fixed more rapidly: the hazard of bug resolution for projects that employ highly restrictive licenses is 30 percent to 50 percent greater than for projects that employ less restrictive licenses. This positive correlation between the bug resolution rate and the restrictiveness of the license is robust to the inclusion of numerous controls.

Despite the long list of controls we employ in the Cox hazard regressions, we ought nevertheless be concerned that unobserved technical characteristics of the OS project might indicate to the project leaders that bugs will be unusually hard and time-consuming to solve and induce them to choose a particular type of OS license. For example, if a project leader knows that a programming problem is particularly challenging, he or she may prefer a restrictive license that prevents commercial competitors from hijacking the code; for simple problems, the project leader knows that commercial competitors could easily solve the problem on their own and therefore will be less concerned with potential hijacking.

To address the potential endogeneity problem, we use two distinct instruments for license type. First, we use the human language in which programmers operate. Second, we use (on a reduced sample size), the project leader's previous project license type. The first instrument is relatively weak, but we have a high degree of confidence in its validity. The second instrument is much stronger, but we are less confident of its validity: if a project leader repeatedly initiates projects of a similar type and these projects involve unobserved characteristics, then the instrument will be correlated with the disturbance term in the same direction as the endogenous regressor. Nonetheless, in both cases, there is a fundamental change in our results. The positive correlation between the bug resolution rate and the restrictiveness of the license is eliminated. Using human language, we find that restrictive licenses reduce the bug resolution rate. Using the previous license type chosen by the project leader, we find no effect of license type on the resolution rate. If the second instrument is invalid, we can anticipate that the coefficient on license type is biased toward the inconsistent coefficient obtained without instrumenting. Taking these results as a whole, we

conclude there is weak, but hardly convincing, evidence that restrictive licenses reduce bug resolution rates.

The rest of the paper is organized as follows. Section 2 presents a brief history of the OS licenses. Section 3 describes our dataset. Section 4 presents our estimation results. Section 5 discusses instrumental variable estimation. Section 6 concludes.

2. A Brief History of Open Source Licensing

In the 1960s and 1970s, mainframe software was freely shared among users, it came with the source code and it could be modified and improved. At the time, the selling of the hardware was seen as the revenue generator and higher quality software encouraged people to buy more hardware. In 1969, the U.S. Department of Justice filed an antitrust suit against IBM. IBM was accused of taking advantage of its strong market position in hardware. The company responded by unbundling its software. This was the beginning of the modern commercial software industry. In the 1980s, AT&T started to enforce intellectual property rights related to the UNIX operating system, to which numerous academic and corporate researchers contributed. In response to the litigation treats over UNIX, in 1984, MIT researcher Richard Stallman founded the Free Software Foundation that developed and distributed software under a formal open source license called the General Public License (GPL). It is also called a "copy-left" license because it seeks to prevent software developed cooperatively from becoming proprietary. Software under the GPL is free to use, modify, or redistribute, but it requires that any modified versions or parts of the copy-left code be made freely available under the GPL. Moreover, even GPL source code mixed with code developed under other licenses has to be licensed under the same GPL conditions. This is called the "viral" provision of the GPL.

In the 1980s, the GPL was the most widely used open source license. In the 1990s this changed as Debian, an organization founded to disseminate an early Linux operating system, faced the problem of defining the licenses of different building parts of the Debian GNU/Linux operating system. To address this issue, the leader of the Debian developed in 1995 the "Debian Social Contract", which allowed the bundling of open source and proprietary software.

In 1998, a number of open source developers adopted this licensing arrangement, which became known as the "Open Source Definition". The licenses that fall under the Open Source Definition include copy-left licenses (like Netscape's Mozilla Public License) that are not "viral" (like the GPL), meaning they do not require that the modified source code fall under the same license as the original code. Nevertheless, they require that the modified code be made available for free use, modification, and redistribution.

Larry Wall, the founder of Perl, a UNIX based programming language, initially released it under the GPL, later decided the license is too restrictive and developed the

Artistic License, a non-copyleft license that allows users to mix proprietary and open source code and does not place restrictions on the modified source code. Non-copy-left licenses such as the Berkeley Software Distribution or the Apache licenses allow a lot of flexibility to contributors, as well. They allow the free use, redistribution, and modification of the software and the modified source code does not have to be made freely available as long as the original source is acknowledged. In the recent years, the less restrictive licenses have become more and more popular, as numerous developers believe that the bundling of open source with proprietary code in areas poorly served by the open source community will improve the success of the open source movement.

We follow Lerner, Tirole (2005) and define three types of OS licenses: unrestrictive, restrictive, and highly restrictive. Highly restrictive licenses (like the GPL) are those that contain the "viral" provision that requires that modified versions of the code be made available under the same license as the original code. Restrictive licenses (e.g. Mozilla Public License), also called "copy-left" licenses, require that altered code be made generally available, but not necessarily under the same license. Table 1 summarizes the OS licenses under which the projects in our sample operate. Some projects operate under two licenses as parts of the source code fall under different licenses, or the contributor may be able to decide what license he prefers for his contribution. Our sample contains 121 bugs from 15 such projects.

Table 1. Types of Open Source Licenses in Our Sample

	Restrictive	Highly Restrictive	Number of observations
Apache Software License	No	No	23
Artistic License	No	No	11
BSD License	No	No	79
Eiffel Forum License	Yes	No	2
General Public License	Yes	Yes	1002
Lesser GPL	Yes	No	220
MIT License	No	No	40
Mozilla Public License 1.1	Yes	No	172
Nethack Public License	Yes	No	10
Qt Public License	Yes	No	26
Zlib/libpng License	No	No	10
Proprietary License	Yes	Yes	39
Public Domain	No	No	8
Other	?	?	69

3. Dataset Construction

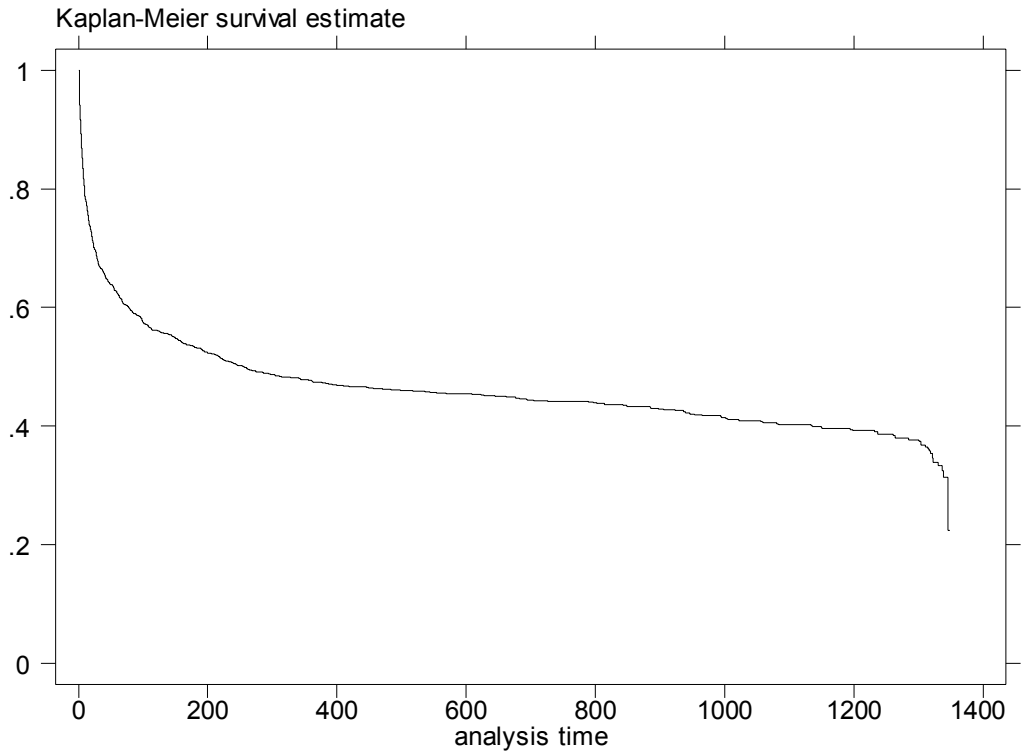
To construct our dataset, we use the SourceForge.net, a free service that hosts about 100,000 open source projects since 1999. First, we chose all the projects registered on SourceForge during the month of January 2003. There were 932 such projects. For each project, we sorted the list of bugs in alphabetical order by the summary of the bug written by the contributor who posted it, in descending order. We then took the first 10 bugs, in case there were more than 10, and all the bugs listed otherwise. A great number of the projects had no activity at all. For each bug, we extracted from SourceForge the following details: bug ID, summary, date submitted, bug status (open, closed, pending, or deleted), date closed (whenever the status was closed), and priority. To each bug, we attached the following information about the project: project ID, project name, registration date, development status, license under which the project operates, operating system, audience, programming language, number of contributors, topic, human languages in which the programmers operate, and database environment. We had no pending bugs (the pending status is used when additional information is needed in order to resolve the issue), but we deleted the observations that consisted of deleted bugs because we are analyzing the bug resolution rates and the deleted bugs are typically bugs that were cancelled by the project team before getting solved. The final sample includes 317 projects with 1711 bugs. Table 2 summarizes the percentage of observations along characteristics of the projects. Most of the bugs in our sample come from projects oriented towards end-users and developers, geared to the POSIX family of operating systems or Microsoft Windows. Also, the sample is dominated by bugs coming from software development and dynamic content projects, in the Beta or Production development stage, with less than 5 contributors. In addition, 78% of the bugs have a 5 (normal) level of priority, where priority ranges from 1 to 9, with 9 being the most important and highest priority.

Table 2. Percentage of bugs classified by the characteristics of the projects

License		**Intended Audience**		**Operating System**		**Number of developers (N)**	
All highly restrictive	56%	End Users/Desktop	44%	POSIX	39%	N=1	34%
All restrictive	79%	Developers	53%	MS Windows	38%	1<N≤5	37%
Some highly restrictive	60%	System Administrators	22%	Independent	29%	5<N≤10	13%
Some restrictive	80%					N>10	16%
		Project Topic				**Project Status**	
Communications	7%	Education	2%	Printing	1%	1 (Planning)	2%
Security	3%	Internet	8%			2 (Pre-Alpha)	4%
Software Development	16%	Site Management	6%			3 (Alpha)	16%
Desktop	3%	Human Machine Interfaces	1%			4 (Beta)	33%
Text Editors	1%	Office/Business	5%			5 (Production/Stable)	48%
Database	5%	Dynamic Content	14%			6 (Mature)	3%
Terminals	2%	Games	3%			7 (Inactive)	3%

633 bugs are still open at the time of the study. Figure 1 shows the Kaplan-Meier estimate of the survivorship function. The initial steep decline shows that numerous bugs were solved shortly after the day they were posted on SourceForge. After that, the slow descent of the curve shows many bugs were not solved for long periods. Approximately 50% of the bugs are estimated to be still unsolved at any time during the study. Also, the minimum value of the survivorship function is not zero, as the largest observed time was a censored observation.

Figure 1. Kaplan-Meier Estimate of the Survivorship Function



4. Results

Our sample contains 1711 bugs out of which 633 were still open at the time of our study. The presence of right censoring in our sample lends itself to the use of survival analysis that uses information from all the observations available, both censored and uncensored.

Our independent variable of interest is the type of license under which the project operates. We define four dummy variables. ALL HIGHLY RESTRICTIVE LICENSES is equal to 1 when all code falls under highly restrictive licenses and 0 otherwise. SOME HIGHLY RESTRICTIVE LICENSES is equal to one when at least part of the code falls under highly restrictive licenses. ALL RESTRICTIVE LICENSES is equal to one when all code is govern by licenses that are at least restrictive, and SOME RESTRICTIVE LICENSES is equal to one when at least some of the code is govern by licenses that are at least restrictive.

Figure 2 depicts the Kaplan-Meier estimator of the survivorship function for the group of bugs coming from projects with ALL HIGHLY RESTRICTIVE LICENSES

compared to the rest of the bugs in the sample. The estimated survivorship function for bugs of projects with ALL HIGHLY RESTRICTIVE LICENSES lies completely bellow that for the rest of the bugs. This means bugs of projects with ALL HIGHLY RESTRICTIVE LICENSES are solved quicker than the rest of the bugs in the dataset at any point during the study. In other words, at any point in time the proportion of bugs estimated to be fixed is greater for projects that operate under one or two highly restrictive licenses. The two curves do not cross at any point and show a similar pattern of survival. Therefore, to decide whether the difference observed in Figure 1 is significant, we conduct a log-rank test for equality of the survivor functions. The test value is 42.88; hence, we reject the null hypothesis that the two survivorship functions are the same (the statistic is significant at beyond the 1% level).

Figure 3 presents the estimated survivorship functions for the group of bugs of projects with ALL RESTRICTIVE LICENSES and the rest of the bugs in the dataset. In this case, the bug resolution experience for these two groups appears similar. A log-rank test supports this impression; it fails to reject the hypothesis of equality of the two estimates. In other words, there is no observed difference between the bug resolution rates of projects with one or two restrictive licenses and the rest of the projects. According to Figure 4, bugs coming from projects that operate under at least one highly restrictive license are solved faster than the rest of the bugs. This observation is confirmed by a highly significant log-rank test. The situation is not as clear-cut in Figure 5, where projects are divided in those that are govern by at least one restrictive license and the rest of the projects. Also, tests for the equality of the survivor functions are not conclusive. The log-rank test and the Tarone-Ware test reject the null of equality at the 5% and 10% level, respectively. On the other hand, the Wilcoxon and the Peto-Peto-Prentice tests fail to reject the null.

Figure 2. Kaplan-Meier Survival Estimates,
Comparison by All Highly Restrictive Licenses

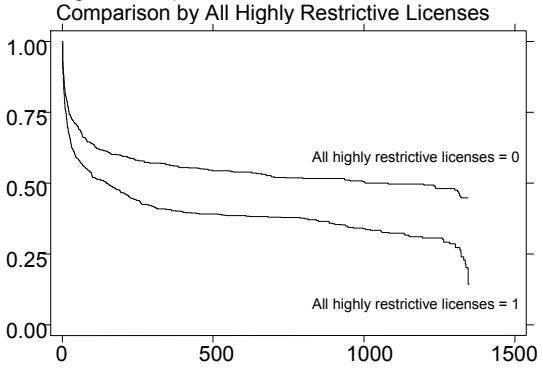


Figure 3. Kaplan-Meier Survival Estimates,
Comparison by All Restrictive Licenses

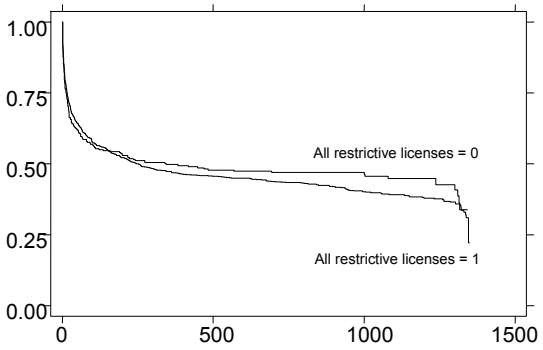


Figure 4. Kaplan-Meier Survival Estimates,
Comparison by Some Highly Restrictive Licenses

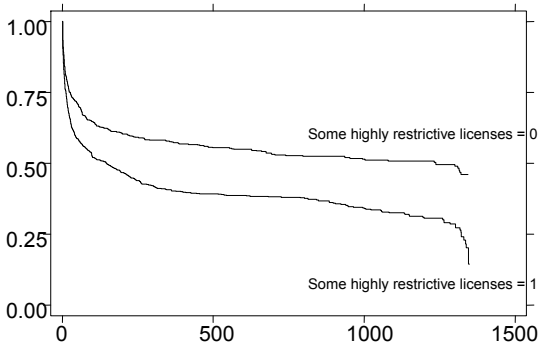
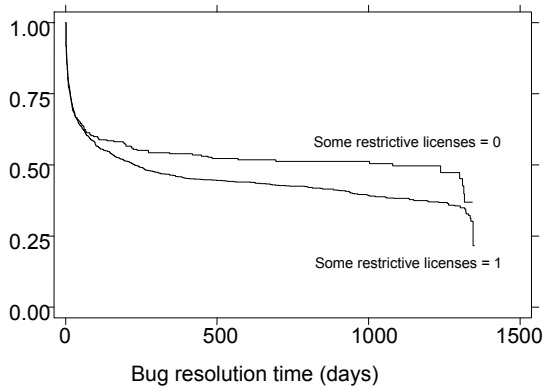


Figure 5. Kaplan-Meier Survival Estimates,
Comparison by Some Restrictive Licenses



These results suggest that bugs are solved at a faster rate when project leaders choose at least one highly restrictive license (that includes the viral provision) to govern their project. On the other hand, choosing at least one restrictive license (copy-left license) does not appear to affect the bug resolution rate. In other words, if the bug resolution rate is a proxy for project quality improvement, it appears that projects improve at a faster rate when contributors and users are not allowed to mix the source code with code that falls under different licenses (for example, proprietary licenses). The less restrictive copy-left requirement does not seem to have the same effect on quality improvement.

Of course, it is necessary to control for other factors influencing the bug resolution rate. We now turn to the results of our estimation. We use a Cox proportional hazards model to estimate the effect of license type on bug fixing rates. Table 3 reports the hazard ratios estimates when we divide projects in two groups: those that operate only under highly restrictive licenses (ALL HIGHLY RESTRICTIVE LICENSES) and the rest of the projects in our dataset. A hazard ratio greater than 1 in column (1) shows that, at any time during the study, bugs of projects operating under all highly restrictive licenses are fixed 56% faster than bugs of the rest of the projects. The hazard ratio estimate is significant at the 1% confidence level. Kuan (2002) finds that higher priority bugs are solved faster than lower priority bugs for FreeBSD (an open source Unix operating system). Moreover, we expect that bugs are solved sooner when there are many developers working on the project, interested in improving it to use it for their own use or to make their contributions more visible. We control for these confounding effects by including two independent variables in the specification: the bug priority level chosen by the individual who reported the bug and

the number of developers working on the project. Contrary to our expectations, column (2) reports estimated hazard ratios close to 1 and insignificant. This indicates that the bug priority level and the number of developers working on the project do not affect the bug resolution time. The hazard ratio estimate on ALL HIGHLY RESTRICTIVE LICENSES does not change.

Table 3. Cox Proportional Hazards Model Results for All Highly Restrictive Licenses (Hazard Ratios reported)

	(1)	(2)	(3)	(4)	(5)	(6)
All Highly Restrictive Licenses	1.56	1.57	1.45	1.34	1.30	1.30
	(6.45)	(6.45)	(4.70)	(3.68)	(3.24)	(3.25)
Bug Priority		1.00	1.01	1.02	1.01	1.01
		(-0.04)	(0.19)	(0.59)	(0.20)	(0.34)
Number of Developers		1.00	1.00	1.01	1.01	1.01
		(0.62)	(0.49)	(2.18)	(2.08)	(1.28)
Project Audience Dummy Variables	No	No	Yes	Yes	Yes	Yes
Project Topic Dummy Variables	No	No	No	Yes	Yes	Yes
Project Status Dummy Variables	No	No	No	No	Yes	Yes
Project Operating System Dummy Variables	No	No	No	No	No	Yes
Number of observations	1529	1529	1529	1529	1529	1529

Z scores in parentheses

The project audience dummies are: End-users, Developers, and System Administrators. The project topic dummies are: Communications, Security, Software Development, Desktop, Text Editors, Database, Terminals, Education, Internet, Site Management, Human Machine Interfaces, Office/Business, Dynamic Content, Game, and Printing. The project status dummies are: 1 (Planning), 2 (Pre-Alpha), 3 (Alpha), 4 (Beta), 5 (Production/Stable), and 6 (Mature). The project operating system dummies are: POSIX, MS Windows, and Independent.

We expect projects targeted towards developers or system administrators to attract contributors that are more skilled than those that work on applications for other audiences, like end-users. Contributors' skill is likely to affect the bug resolution rates. Leaders of projects report on SourceForge.net the following four categories of intended audience: end-users/desktop, developers, system administrators, and others. We construct three dummy variables that capture the type of audience targeted by the project: End-users, Developers, and System Administrators. We add these dummies and results are reported in column (3). The hazard ratio estimate for ALL HIGHLY RESTRICTIVE LICENSES is slightly smaller. At any time during the study, when projects operate only under highly restrictive licenses, bugs are fixed 45% faster. The hazard ratio estimate on the Developers dummy is significant at the 5% level. Contrary to our expectations, it is less than 1, implying that bugs from

projects targeted towards software developers are solved at a slower pace than bugs from projects targeted towards other audiences. The estimate on the End-users and System administrators dummies are larger than 1, suggesting bugs of projects targeted towards these types of audiences are solved faster, but the estimates are not statistically significant. A possible explanation could be that projects targeted towards developers might have bugs that are more difficult.

We hypothesize that the difficulty of the bugs, that affects their resolution rates, is related to different characteristics of the software application. System complexity increases debugging time. In low-level applications (such as communication applications, like real time digital telephony), bugs tend to be very hard to spot. Also, in systems where performance is an issue (like databases), identifying the source of the bug takes time. In security applications, bugs are critical. If left unresolved, they may compromise the user's entire operation. There are 16 types of project topics in our dataset: communications, security, software development, desktop, text editors, database, terminals, education, internet, site management, human machine interfaces, office/business, dynamic content, games, printing, and others. We construct dummies that capture every topic, except for the one denoting other topics. Column (4) specification includes these dummies. The hazard ratio estimate for ALL HIGHLY RESTRICTIVE LICENSES decreases to 1.34 when we control for the project topic. It is still highly significant. When compared to bugs of projects with other topics, bugs of security and dynamic content projects are solved faster and bugs of software development, terminals, and site management projects are solved more slowly. The rest of the dummies hazard ratio estimates are insignificant.

We also expect to find that the bug resolution rate is correlated to the project development stage. For example, when projects are in the planning stages, developers could be more interested in developing new features for the application than by fixing existing bugs. At the same time, in more mature projects, there could be less room for improvement in terms of new features; fixing bugs could become of primary interest. There are seven project development stages and we construct dummies to denote the first six. Column (5) reports the results of the model that includes these dummies. The hazard ratio estimate for ALL HIGHLY RESTRICTIVE LICENSES is 1.30 and still significant at the 1% level. The four most advanced development stages have highly significant hazard ratio estimates (at the 1% level).

Column (6) includes dummies for another characteristic of the project, the operating system under which the application runs: POSIX, MS Windows, and an Independent operating system. We omit the dummy for other operating systems. The ALL HIGHLY RESTRICTIVE LICENSES hazard ratio estimate does not change. The POSIX and Independent operating system dummies have high hazard ratio estimates significant at less than the 1% level.

After controlling for other observable confounding effects, for projects that operate under highly restrictive licenses, the bug resolution process seems to be significantly faster than for other projects. In other words, our results suggest that, when the license of the project requires that any subsequent contributions be made available under the original license, the quality of the application seems to improve at a faster rate.

We then classify the projects in the dataset in another two groups. We set to compare bug resolution rates for projects that are govern by at least one highly restrictive license (SOME HIGHLY RESTRICTIVE LICENSES) with the rest of the projects. Table 4 reports the results of our estimation.

Table 4. Cox Proportional Hazards Model Results for Some Highly Restrictive Licenses (Hazard Ratios reported)

	(1)	(2)	(3)	(4)	(5)	(6)
Some Highly Restrictive Licenses	1.62 (6.78)	1.64 (6.80)	1.52 (5.19)	1.40 (4.10)	1.40 (3.99)	1.41 (4.02)
Bug Priority		1.00 (0.01)	1.01 (0.20)	1.02 (0.56)	1.00 (0.12)	1.01 (0.25)
Number of Developers		1.00 (1.84)	1.00 (0.72)	1.01 (2.23)	1.01 (2.16)	1.01 (1.40)
Project Audience Dummy Variables	No	No	Yes	Yes	Yes	Yes
Project Topic Dummy Variables	No	No	No	Yes	Yes	Yes
Project Status Dummy Variables	No	No	No	No	Yes	Yes
Project Operating System Dummy Variables	No	No	No	No	No	Yes
Number of observations	1529	1529	1529	1529	1529	1529

Z scores in parentheses.

The project audience dummies are: End-users, Developers, and System Administrators. The project topic dummies are: Communications, Security, Software Development, Desktop, Text Editors, Database, Terminals, Education, Internet, Site Management, Human Machine Interfaces, Office/Business, Dynamic Content, Game, and Printing. The project status dummies are: 1 (Planning), 2 (Pre-Alpha), 3 (Alpha), 4 (Beta), 5 (Production/Stable), and 6 (Mature). The project operating system dummies are: POSIX, MS Windows, and Independent.

We add the same controls as above. We find that the hazard ratio estimate is greater than 1 and highly significant. The results are again robust to the introduction of different controls. They suggest that bugs of projects operating under at least one highly restrictive license are fixed 40 to 64% faster than bugs of the rest of the projects at any time during the study. The coefficients on the priority level of the bug and the number of developers working on the project are again very close to 1 and insignificant. Together with our earlier results, our findings suggest that the viral provision of highly restrictive licenses seems to affect the speed of the bug resolution process. In other words, projects protected against the risk of "hijacking" by commercial vendors attract enough effort to have their bugs fixed more rapidly than the rest of the projects.

We next turn to explore the effect of the copy-left provision of less restrictive licenses. These require that the modified code be made freely available for the public. However, the contributor can decide on the type of OS license to govern the modified or extended piece of code. The results of estimation of specifications with the ALL RESTRICTIVE LICENSES dummy variable are reported in Table 5.

Table 5. Cox Proportional Hazards Model Results for All Restrictive Licenses
(Hazard Ratios reported)

	(1)	(2)	(3)	(4)	(5)	(6)
All Restrictive Licenses	1.06	1.07	1.01	1.03	1.00	1.03
	(0.65)	(0.74)	(0.08)	(0.28)	(-0.03)	(0.35)
Bug Priority		0.99	1.01	1.02	1.01	1.01
		(-0.32)	(0.31)	(0.77)	(0.39)	(0.47)
Number of Developers		1.00	1.00	1.01	1.01	1.00
		(-0.65)	(-0.20)	(1.89)	(1.86)	(1.03)
Project Audience Dummy Variables	No	No	Yes	Yes	Yes	Yes
Project Topic Dummy Variables	No	No	No	Yes	Yes	Yes
Project Status Dummy Variables	No	No	No	No	Yes	Yes
Project Operating System Dummy Variables	No	No	No	No	No	Yes
Number of observations	1529	1529	1529	1529	1529	1529

Z scores in parentheses.

The project audience dummies are: End-users, Developers, and System Administrators. The project topic dummies are: Communications, Security, Software Development, Desktop, Text Editors, Database, Terminals, Education, Internet, Site Management, Human Machine Interfaces, Office/Business, Dynamic Content, Game, and Printing. The project status dummies are: 1 (Planning), 2 (Pre-Alpha), 3 (Alpha), 4 (Beta), 5 (Production/Stable), and 6 (Mature). The project operating system dummies are: POSIX, MS Windows, and Independent.

We compare projects that operate under one or two restrictive or highly restrictive licenses and the rest of the projects. We cannot find a significant difference in the bug resolution speed between the two groups of projects. The estimates are close to 1 and insignificant.

We repeat the analysis from Table 5 for the SOME RESTRICTIVE LICENSES dummy. The estimation results are presented in Table 6. When projects are governed by at least one restrictive or highly restrictive license, bugs are fixed around 20% faster. Estimates are significant at either the 5 or 10% level.

Table 6. Cox Proportional Hazards Model Results for Some Restrictive Licenses
(Hazard Ratios reported)

	(1)	(2)	(3)	(4)	(5)	(6)
Some Restrictive Licenses	1.22	1.23	1.18	1.21	1.17	1.20
	(2.16)	(2.26)	(1.72)	(1.98)	(1.68)	(1.85)
Bug Priority		0.99	1.00	1.02	1.00	1.01
		(-0.46)	(0.12)	(0.60)	(0.23)	(0.33)
Number of Developers		1.00	1.00	1.01	1.01	1.01
		(-0.77)	(-0.36)	(1.87)	(1.90)	(1.15)
Project Audience Dummy Variables	No	No	Yes	Yes	Yes	Yes
Project Topic Dummy Variables	No	No	No	Yes	Yes	Yes
Project Status Dummy Variables	No	No	No	No	Yes	Yes
Project Operating System Dummy Variables	No	No	No	No	No	Yes
Number of observations	1529	1529	1529	1529	1529	1529

Z scores in parentheses.

The project audience dummies are: End-users, Developers, and System Administrators. The project topic dummies are: Communications, Security, Software Development, Desktop, Text Editors, Database, Terminals, Education, Internet, Site Management, Human Machine Interfaces, Office/Business, Dynamic Content, Game, and Printing. The project status dummies are: 1 (Planning), 2 (Pre-Alpha), 3 (Alpha), 4 (Beta), 5 (Production/Stable), and 6 (Mature). The project operating system dummies are: POSIX, MS Windows, and Independent.

To conclude, our results seem to support the hypothesis that the highly restrictive viral provision motivates individuals to contribute to projects and to respond faster to service requests. Nevertheless, our results are mixed when we analyze the less restrictive copy-left licenses. They are not consistent with the hypothesis that bug resolution rates are shorter when projects employ these licenses as opposed to unrestrictive licenses. Therefore, in the next section, we continue to explore the correlation between highly restrictive licenses

and the bug resolution speed and try to address the potential endogeneity problem stated earlier.

5. Instrumental Variables Estimation Results

Although we found strong evidence that highly restrictive licenses are correlated with the rate of bug resolution, we are concerned that the license choice might be endogenous. It is quite possible that certain technical characteristics of the projects (apart from those included in the specification) indicate to project leaders that bugs are unusually hard or time-consuming to solve. This knowledge might induce them to choose a particular type of OS license. We try to address this issue by using two different instruments: (i) the human language in which the contributors operate and (ii) the project leader's previous project license type as instrumental variables.

5.1. Human Language as Instrumental Variable

Lerner and Tirole (2005) find that projects in which contributors operate in Japanese are far less likely to have highly restrictive licenses than projects in English. On the other hand, projects in German and Spanish are much more likely to have highly restrictive licenses. Although the authors do not provide an explanation for the correlation between license type and language, we suspect that patent litigation and cultural differences are to blame. In European countries like Germany and France, the patent litigation decisions are reached reasonably quickly and the proceedings are not too expensive when compared to the U.S. This could explain the tendency of German and Spanish nationals to choose highly restrictive licenses, for example, licenses that do not allow the bundling of open source with proprietary code and protect the open code from potential infringement actions. On the other hand, in Japan, proceedings involve a series of meetings with the judges and the costs tend to be high. Thus, projects in Japanese are more likely to operate under less restrictive licenses (for example, licenses that allow contributors to use commercial code as part of modified versions). Moreover, cultural differences might provide a potential explanation for the correlation between language and license. For example, in Europe, where Roman law was adopted, law was regarded as the regulatory system for human relations and individuals could form secure relationships by entering a contract. On the other hand, in

Japanese history, the legal system did not play a role in human relationships; they were perceived as moral relationships. That could be the reason why the licenses adopted by projects in Japanese are less likely than projects in English to be highly restrictive.

Six languages in which open source contributors operate dominate our sample: English, French, German, Spanish, Russian, and Japanese. We construct dummies for all, except for English. Because our dependent variable, the license type, is a dummy, we employ a logit specification for the first stage of the instrumental variable estimation. We control for all observed characteristics of the project, as these might affect the license choice. Odds ratios are reported in Table 7.

Table 7. First Stage Logit Estimates Using Natural Language as an Instrument for License Type

(Odds ratios reported)

	(1) Dependent Variable: All Highly Restrictive Licenses	(2) Dependent Variable: Some Highly Restrictive Licenses
Natural Language:		
- French	0.64 (-0.66)	0.68 (-0.55)
- Spanish	7.68 (1.51)	-
- German	3.01 (1.93)	2.85 (1.75)
- Russian	0.13 (-1.71)	0.26 (-0.98)
- Japanese	-	-
Number of Developers	0.95 (-1.85)	0.94 (-2.09)
Project Audience Dummy Variables	Yes	Yes
Project Topic Dummy Variables	Yes	Yes
Project Status Dummy Variables	Yes	Yes
Project Operating System Dummy Variables	Yes	Yes
Number of observations	313	301

Z scores in parentheses.

The project audience dummies are: End-users, Developers, and System Administrators. The project topic dummies are: Communications, Security, Software Development, Desktop, Text Editors, Database, Terminals, Education, Internet, Site Management, Human Machine Interfaces, Office/Business, Dynamic Content, Game, and Printing. The project status dummies are: 1 (Planning), 2 (Pre-Alpha), 3 (Alpha), 4 (Beta), 5 (Production/Stable), and 6 (Mature). The project operating system dummies are: POSIX, MS Windows, and Independent.

We find that our language dummies are correlated with the ALL HIGHLY RESTRICTIVE LICENSES dummy [column (1)]. They are jointly significant at the 10% level. The results are consistent with Lerner and Tirole (2005), as projects in Spanish or

German are much more likely to operate only under highly restrictive licenses. Projects in French and Russian are less likely to be governed only by highly restrictive licenses. Nevertheless, only two of the dummies are statistically significant: the German dummy at the 5% level and the Russian dummy at the 10% level. The coefficient on the Japanese dummy is not estimated as all projects in Japanese in the sample operate only under highly restrictive license. When the dependent variable is SOME HIGHLY RESTRICTIVE LICENSES, our instrumental variables are jointly insignificant. The coefficients on the Spanish and Japanese dummies are not estimated as all projects in our sample that operate in these two languages are governed by at least one highly restrictive license.

We also estimate a reduced form equation with the instruments and the exogenous variables as explanators for the bug resolution rate. All coefficients on human language are statistically insignificant, except for German, which is marginally significant at the 10%. Its hazard ratio of 1.20 supports the identification story. Projects in German are much more likely to have highly restrictive licenses and bugs are solved faster when projects operate under highly restrictive licenses.

A better alternative for using the human language in which the contributors operate as instruments would be to use the country where the developers are located. Unfortunately, we do not have projects' country data available, as individuals are not required to share their locations on SourceForge.net. We argue that the human language is a good proxy for the location of the leaders of the project. When leaders choose only one language in which contributors operate, they do not expect the project to cross the country's borders.

The other characteristic of a valid instrumental variable is the lack of correlation with the disturbance term. In other words, conditional on the controls included in the regression, the only effect of language on the bug resolution rate should be through the project's license choice. We have no reasons to believe that Germans, for example, are more efficient at solving bugs than other nationals. In addition, we control for different characteristics of the projects that might affect the decision to choose a certain license type.

Table 8 presents the results of our instrumental variable estimation. We report the hazard ratios. The specification in columns (1) and (2) compares projects with ALL HIGHLY RESTRICTIVE LICENSES with the rest of the projects in the dataset.

Table 8. Cox Proportional Hazards Estimation Using Natural Language as an Instrument for License Type

(Hazard ratios reported)

	(1)	(2)	(3)	(4)
ALL HIGHLY RESTRICTIVE LICENSES	0.53 (-1.67)	0.61 (-4.46)	-	-
SOME HIGHLY RESTRICTIVE LICENSES	-	-	0.35 (-2.18)	0.68 (-3.38)
Bug Priority	1.02 (0.78)	1.01 (0.39)	1.02 (0.83)	1.01 (0.36)
Number of Developers	0.99 (-0.11)	1.00 (-0.50)	0.99 (-0.80)	1.00 (-0.43)
Project Audience Dummy Variables	Yes	Yes	Yes	Yes
Project Topic Dummy Variables	Yes	Yes	Yes	Yes
Project Status Dummy Variables	Yes	Yes	Yes	Yes
Project Operating System Dummy Variables	Yes	Yes	Yes	Yes
Number of observations	1498	1529	1426	1529

Z scores in parentheses.

The project audience dummies are: End-users, Developers, and System Administrators. The project topic dummies are: Communications, Security, Software Development, Desktop, Text Editors, Database, Terminals, Education, Internet, Site Management, Human Machine Interfaces, Office/Business, Dynamic Content, Game, and Printing. The project status dummies are: 1 (Planning), 2 (Pre-Alpha), 3 (Alpha), 4 (Beta), 5 (Production/Stable), and 6 (Mature). The project operating system dummies are: POSIX, MS Windows, and Independent.

In column (1) we use the predicted value of the ALL HIGHLY RESTRICTIVE LICENSES variable from the logit model. The predicted values are therefore probability values (bounded between 0 and 1). The sign of the coefficient has changed and it is only marginally significant at the 10% level. In column (2), we construct a dummy variable for ALL HIGHLY RESTRICTIVE LICENSES equal to 1 when the predicted value is higher or equal to 0.50, and 0 otherwise. The coefficient is still negative and, although our instrument is only weakly correlated with the endogenous variable, it is highly significant at the 1% level. In column (1), the hazard ratio decreases by 7% for every 10% increase in the probability of

ALL HIGHLY RESTRICTIVE LICENSES. The hazard ratio in column (2) shows that, when projects operate only under highly restrictive licenses, bugs are fixed 39% slower than the rest of the bugs. The estimates of the coefficients on bug priority level and number of contributors do not change. They are close to 1 and insignificant, as they are without instrumenting. Columns (3) and (4) report the results of our instrumental variable estimation when projects with SOME HIGHLY RESTRICTIVE LICENSES are analyzed. Hazard ratio estimates are less than 1 and significant at the 5% level (when predicted values are used) and 1% level (when we construct a new dummy using the same method we used for the ALL HIGHLY RESTRICTIVE LICENSES dummy). Bugs of projects governed by at least one highly restrictive license are fixed at a slower rate than bugs of the rest of the projects. Column (3) shows that the hazard ratio decreases by 10% for every 10% increase in the probability of SOME HIGHLY RESTRICTIVE LICENSES. According to column (4)'s estimate, bugs are fixed 32% slower for projects operating under at least one highly restrictive license. Thus, contrary to our hypothesis, controlling for the endogeneity of the license type, we find that projects that operate under highly restrictive licenses are actually slower at addressing bug requests than other projects.

We are nevertheless concerned that the weak correlation between our instrument and the license choice could lead to bias in the instrumental variable estimator if the instrument is even moderately correlated with the disturbance. We try to address this issue by employing an alternative instrument strongly correlated with the endogenous variable, the project leader's previous project license type.

5.1. Project Leader's Previous Project License Type as Instrumental Variable

We suspected a strong correlation between the types of license chosen for different projects by the same leader. It is quite probable to observe a strong correlation between a previous license choice and the current one. We hypothesize that a leader who chose a highly restrictive license in the past is more likely to choose the same type of license for the current project.

We construct two dummy variables: one equal to 1 when a previous project operates under all highly restrictive licenses and another equal to 1 when it operates under at least one highly restrictive license. The sample is greatly reduced due to data availability. Table 9

presents the results of the first stage logit estimation. We report the odds ratios. Column (1) reports the results of the logit specification with ALL HIGHLY RESTRICTIVE LICENSES as the dependent variable. In column (2), SOME HIGHLY RESTRICTIVE LICENSES is the dependent variable. We control for the characteristics of the project in both specifications. Our results suggest that a project leader who chooses highly restrictive licenses for a project is indeed much more likely to release the code of the next project under the same type of licenses. The odds ratios are high and significant at the 1% level in both regressions. Consistent with our intuition, our alternative instrument has a very strong correlation with the endogenous variable.

Table 9. First Stage Logit Estimates Using the Project Leader's Previous Project License Type as an Instrument for License Type

(Odds ratios reported)

	(1) Dependent Variable: All Highly Restrictive Licenses	(2) Dependent Variable: Some Highly Restrictive Licenses
Previous Project – All Highly Restrictive Licenses	8.65 (3.45)	-
Previous Project – Some Highly Restrictive Licenses	-	10.25 (3.44)
Number of Developers	1.03 (0.55)	1.03 (0.50)
Project Audience Dummy Variables	Yes	Yes
Project Topic Dummy Variables	Yes	Yes
Project Status Dummy Variables	Yes	Yes
Project Operating System Dummy Variables	Yes	Yes
Number of observations	119	119

Z scores in parentheses.

The project audience dummies are: End-users, Developers, and System Administrators. The project topic dummies are: Communications, Security, Software Development, Desktop, Text Editors, Database, Terminals, Education, Internet, Site Management, Human Machine Interfaces, Office/Business, Dynamic Content, Game, and Printing. The project status dummies are: 1 (Planning), 2 (Pre-Alpha), 3 (Alpha), 4 (Beta), 5 (Production/Stable), and 6 (Mature). The project operating system dummies are: POSIX, MS Windows, and Independent.

Nevertheless, we are less confident than before about the validity of this second instrument. It is possible that a project leader initiates projects with similar unobserved characteristics that determine his license choices. In this case, the instrument would be correlated with the error term in the same direction as the endogenous regressor.

Table 10 presents the results of instrumental variable estimation using the project leader's previous project license choice as the instrument. We report the hazard ratios.

Table 10. Cox Proportional Hazards Estimation Using the Project Leader's Previous Project License Type as an Instrument for License Type

(Hazard ratios reported)

	(1)	(2)	(3)	(4)
ALL HIGHLY RESTRICTIVE LICENSES	1.09 (0.24)	0.84 (-0.91)	-	-
SOME HIGHLY RESTRICTIVE LICENSES	-	-	1.07 (0.19)	0.77 (-1.48)
Bug Priority	0.92 (-1.88)	0.92 (-1.94)	0.92 (-1.77)	0.92 (-1.81)
Number of Developers	1.00 (-0.19)	1.00 (-0.30)	1.00 (-0.30)	0.99 (-0.60)
Project Audience Dummy Variables	Yes	Yes	Yes	Yes
Project Topic Dummy Variables	Yes	Yes	Yes	Yes
Project Status Dummy Variables	Yes	Yes	Yes	Yes
Project Operating System Dummy Variables	Yes	Yes	Yes	Yes
Number of observations	669	669	657	657

Z scores in parentheses.

The project audience dummies are: End-users, Developers, and System Administrators. The project topic dummies are: Communications, Security, Software Development, Desktop, Text Editors, Database, Terminals, Education, Internet, Site Management, Human Machine Interfaces, Office/Business, Dynamic Content, Game, and Printing. The project status dummies are: 1 (Planning), 2 (Pre-Alpha), 3 (Alpha), 4 (Beta), 5 (Production/Stable), and 6 (Mature). The project operating system dummies are: POSIX, MS Windows, and Independent.

Column (1) reports the results for the ALL HIGHLY RESTRICTIVE LICENSES dummy. We use the predicted value of the first stage logit and control for the priority level of the bug and the characteristics of the project. The hazard ratios estimates are close to 1 and insignificant. The specifications in column (2) uses a license dummy equal to 1 when the predicted value is larger or equal to 0.50, and 0 otherwise. The hazard ratio estimate is less than 1, implying projects operating only under highly restrictive licenses are slower at fixing bugs. Nevertheless, the estimate is insignificant. Columns (3) and (4) present a similar story for the SOME HIGHLY RESTRICTIVE LICENSES dummy. Again, we use the predicted value from the first stage logit in column (3) and a dummy constructed in the same manner as before in column (4). Hazard ratio estimates are insignificant in both cases. Overall, the results suggest that, after controlling for endogeneity by using the leader's previous license

choice as an instrument, there is no correlation between the license type and the bug resolution rate.

6. Conclusion

While many studies explore the OS license choice at firm or project level, few papers analyze the way different license types affect the outcome of OS projects. This paper presents an empirical analysis of the relationship between the license choice and the quality of OS programs. We use SourceForge.net to collect data on more than 1700 bugs coming from about 300 projects. There are numerous dimension of quality of a software application. We employ the bug resolution rate as a proxy for project quality improvement. We classify the OS licenses in our dataset in three groups. Highly restrictive licenses require that any modified code be released under the same license as the original code. Restrictive licenses require that the altered code be made available for free use, modification, and redistribution, but not necessarily under the original license. Finally, unrestrictive licenses do not impose any such restriction on the modified code.

The existing literature suggests different incentives for developers to contribute to OS projects: own use, talent signaling to peers and future employers, or simply enjoyment when solving a challenging programming task. Because of these various motivations, the chosen license might affect the appeal of OS projects on potential contributors in different ways. On the one hand, restrictive licenses that do not allow the bundling of the OS code with proprietary code might encourage developers to contribute. These licenses prevent commercial companies from "hijacking" OS code (mixing it with proprietary code and selling the final product). Hijacking might deprive initial contributors of some of the benefits they had expected from writing OS code. On the other hand, developers might actually be reluctant to join OS projects that operate under restrictive licenses when they believe the software could benefit from mixing its code with commercial code. In this case, unrestrictive licenses would be more appealing.

In initial regressions, we find a strong positive correlation between highly restrictive licenses and the speed of bug resolution, that is robust to controlling for several characteristics of the projects. Nevertheless, we are concerned about the potential endogeneity of our license type variable. Unobserved technical characteristics of the projects

might indicate to project leaders that bugs are hard to solve, knowledge that might induce them to choose a particular OS license. We try to address this issue by using two different instruments: (i) the human language in which the contributors operate and (ii) the project leader's previous project license type. Using human language, we find that restrictive licenses reduce the bug resolution rate. Nevertheless, this instrument is weakly correlated with our license type variable. Therefore, we use the previous license type chosen by the project leader as an alternative instrument. Although the second instrument is highly correlated with our license variable, we are less confident in his validity. This time, we find no effect of license type on the resolution rate. Taking these results as a whole, we conclude there is weak, but hardly convincing, evidence that restrictive licenses reduce bug resolution rates.

Appendix

Table 3. Cox Proportional Hazards Model Results for All Highly Restrictive Licenses (Hazard Ratios reported)

	(1)	(2)	(3)	(4)	(5)	(6)
All Highly Restrictive Licenses	1.56 (6.45)	1.57 (6.45)	1.45 (4.70)	1.34 (3.68)	1.30 (3.24)	1.30 (3.25)
Bug Priority		1.00 (-0.04)	1.01 (0.19)	1.02 (0.59)	1.01 (0.20)	1.01 (0.34)
Number of Developers		1.00 (0.62)	1.00 (0.49)	1.01 (2.18)	1.01 (2.08)	1.01 (1.28)
Project Audience:						
-End-users			1.07 (0.89)	1.07 (0.90)	1.01 (0.17)	1.01 (0.11)
-Developers			0.86 (-1.93)	1.04 (0.50)	0.93 (-0.80)	0.87 (-1.67)
-System Administrators			1.06 (0.71)	0.90 (-1.11)	0.86 (-1.58)	0.95 (-0.47)
Project Topic:						
-Communications				0.90 (-0.70)	0.83 (-1.10)	0.79 (-1.46)
-Security				2.26 (3.80)	2.21 (3.15)	2.05 (2.76)
-Software Development				0.54 (-4.47)	0.52 (-4.57)	0.60 (-3.58)
-Desktop				1.02 (0.12)	1.22 (0.96)	1.24 (0.98)
-Text Editors				0.14 (-1.95)	0.17 (-1.78)	0.16 (-1.84)
-Database				1.26 (1.48)	1.20 (1.14)	1.28 (1.55)
-Terminals				0.39 (-2.43)	0.40 (-2.39)	0.45 (-2.05)

-Education				1.00	0.96	0.78
				(0.00)	(-0.19)	(-1.05)
-Internet				1.07	1.03	1.04
				(0.50)	(0.22)	(0.27)
-Site Management				0.59	0.61	0.70
				(-2.88)	(-2.70)	(-1.95)
-Human Machine Interfaces				0.67	0.63	0.55
				(-1.01)	(-1.20)	(-1.51)
-Office/Business				1.20	1.19	1.15
				(1.22)	(1.11)	(0.88)
-Dynamic Content				1.38	1.31	0.99
				(2.77)	(2.31)	(-0.10)
-Games				0.95	1.02	1.04
				(-0.23)	(0.11)	(0.21)
-Printing				1.20	1.60	1.69
				(0.65)	(1.59)	(1.73)
Project Status:						
-1 (Planning)					1.17	1.30
					(0.59)	(0.98)
-2 (Pre-Alpha)					0.89	0.90
					(-0.61)	(-0.55)
-3 (Alpha)					1.42	1.34
					(2.91)	(2.39)
-4 (Beta)					1.30	1.28
					(2.78)	(2.45)
-5 (Production/Stable)					1.62	1.68
					(5.03)	(5.12)
-6 (Mature)					1.90	2.21
					(2.65)	(3.25)
Project Operating System:						
-POSIX						1.52
						(4.52)
-MS Windows						1.10
						(1.00)
-Independent						2.14
						(6.75)
Number of observations	1529	1529	1529	1529	1529	1529
Z scores in parentheses.						

Table 4. Cox Proportional Hazards Model Results Some Highly Restrictive Licenses

(Hazard Ratios reported)

	(1)	(2)	(3)	(4)	(5)	(6)
Some Highly Restrictive	1.62	1.64	1.52	1.40	1.40	1.41
Licenses	(6.78)	(6.80)	(5.19)	(4.10)	(3.99)	(4.02)
Bug Priority		1.00	1.01	1.02	1.00	1.01
		(0.01)	(0.20)	(0.56)	(0.12)	(0.25)
Number of Developers		1.00	1.00	1.01	1.01	1.01
		(1.84)	(0.72)	(2.23)	(2.16)	(1.40)
Project Audience:						
-End-users			1.05	1.06	0.99	0.99
			(0.59)	(0.76)	(-0.10)	(-0.16)
-Developers			0.85	1.03	0.93	0.86

			(-2.13)	(0.34)	(-0.88)	(-1.72)
-System Administrators			1.03	0.89	0.85	0.94
			(0.38)	(-1.31)	(-1.73)	(-0.59)
Project Topic:						
-Communications				0.91	0.85	0.80
				(-0.61)	(-1.01)	(-1.36)
-Security				2.33	2.39	2.18
				(3.95)	(3.42)	(2.98)
-Software Development				0.56	0.54	0.63
				(-4.12)	(-4.18)	(-3.18)
-Desktop				1.03	1.22	1.23
				(0.13)	(0.97)	(0.95)
-Text Editors				0.14	0.16	0.15
				(-1.97)	(-1.81)	(-1.87)
-Database				1.25	0.19	1.27
				(1.44)	(1.09)	(1.48)
-Terminals				0.39	0.39	0.44
				(-2.47)	(-2.42)	(-2.09)
-Education				1.02	0.97	0.80
				(0.09)	(-0.11)	(-0.96)
-Internet				1.09	1.04	1.04
				(0.63)	(0.29)	(0.29)
-Site Management				0.57	0.59	0.68
				(-3.10)	(-2.88)	(-2.11)
-Human Machine Interfaces				0.70	0.64	0.57
				(-0.97)	(-1.16)	(-1.44)
-Office/Business				1.22	1.20	1.16
				(1.34)	(1.19)	(0.99)
-Dynamic Content				1.39	1.32	1.00
				(2.85)	(2.42)	(-0.04)
-Games				0.97	1.04	1.06
				(-0.14)	(0.19)	(0.27)
-Printing				1.22	1.67	1.76
				(0.68)	(1.72)	(1.87)
Project Status:						
-1 (Planning)					1.06	1.18
					(0.22)	(0.62)
-2 (Pre-Alpha)					0.86	0.87
					(-0.82)	(-0.73)
-3 (Alpha)					1.46	1.39
					(3.14)	(2.66)
-4 (Beta)					1.31	1.29
					(2.84)	(2.57)
-5 (Production/Stable)					1.64	1.71
					(5.16)	(5.28)
-6 (Mature)					1.86	2.18
					(2.56)	(3.18)
Project Operating System:						
-POSIX						1.51
						(4.47)
-MS Windows						1.08
						(0.85)
-Independent						2.13
						(6.69)
Number of observations	1529	1529	1529	1529	1529	1529
Z scores in parentheses.						

Table 5. Cox Proportional Hazards Model Results for All Restrictive Licenses

(Hazard Ratios reported)

	(1)	(2)	(3)	(4)	(5)	(6)
All Restrictive Licenses	1.06	1.07	1.01	1.03	1.00	1.03
	(0.65)	(0.74)	(0.08)	(0.28)	(-0.03)	(0.35)
Bug Priority		0.99	1.01	1.02	1.01	1.01
		(-0.32)	(0.31)	(0.77)	(0.39)	(0.47)
Number of Developers		1.00	1.00	1.01	1.01	1.00
		(-0.65)	(-0.20)	(1.89)	(1.86)	(1.03)
Project Audience:						
-End-users			1.17	1.11	1.05	1.04
			(2.03)	(1.40)	(0.63)	(0.52)
-Developers			0.80	1.00	0.89	0.82
			(-3.04)	(-0.01)	(-1.31)	(-2.26)
-System Administrators			1.15	0.95	0.89	0.98
			(1.72)	(-0.53)	(-1.19)	(-0.16)
Project Topic:						
-Communications				0.89	0.82	0.77
				(-0.74)	(-1.19)	(-1.57)
-Security				2.07	1.98	1.88
				(3.43)	(2.73)	(2.43)
-Software Development				0.49	0.47	0.54
				(-5.35)	(-5.32)	(-4.29)
-Desktop				1.09	1.30	1.30
				(0.46)	(1.25)	(1.21)
-Text Editors				0.16	0.19	0.17
				(-1.82)	(-1.65)	(-1.73)
-Database				1.22	1.15	1.23
				(1.31)	(0.87)	(1.30)
-Terminals				0.39	0.40	0.44
				(-2.47)	(-2.39)	(-2.07)
-Education				0.99	0.94	0.77
				(-0.05)	(-0.25)	(-1.11)
-Internet				1.09	1.05	1.07
				(0.62)	(0.32)	(0.46)
-Site Management				0.58	0.61	0.70
				(-2.93)	(-2.67)	(-1.88)
-Human Machine Interfaces				0.68	0.60	0.53
				(-1.03)	(-1.31)	(-1.63)
-Office/Business				1.18	1.16	1.11
				(1.08)	(0.94)	(0.67)
-Dynamic Content				1.38	1.30	0.99
				(2.78)	(2.26)	(-0.08)
-Games				0.95	1.03	1.03
				(-0.26)	(0.12)	(0.13)
-Printing				1.15	1.53	1.61
				(0.50)	(1.44)	(1.59)
Project Status:						
-1 (Planning)					1.18	1.32
					(0.63)	(1.04)
-2 (Pre-Alpha)					0.94	0.94

					(-0.34)	(-0.31)
-3 (Alpha)					1.43	1.33
					(2.99)	(2.33)
-4 (Beta)					1.33	1.29
					(3.00)	(2.54)
-5 (Production/Stable)					1.64	1.69
					(5.25)	(5.22)
-6 (Mature)					2.11	2.43
					(3.11)	(3.61)
Project Operating System:						
-POSIX						1.54
						(4.68)
-MS Windows						1.10
						(1.01)
-Independent						2.14
						(6.76)
Number of observations	1529	1529	1529	1529	1529	1529
Z scores in parentheses.						

Table 6. Cox Proportional Hazards Model Results for Some Restrictive Licenses

(Hazard Ratios reported)

	(1)	(2)	(3)	(4)	(5)	(6)
Some Restrictive Licenses	1.22	1.23	1.18	1.21	1.17	1.20
	(2.16)	(2.26)	(1.72)	(1.98)	(1.68)	(1.85)
Bug Priority		0.99	1.00	1.02	1.00	1.01
		(-0.46)	(0.12)	(0.60)	(0.23)	(0.33)
Number of Developers		1.00	1.00	1.01	1.01	1.01
		(-0.77)	(-0.36)	(1.87)	(1.90)	(1.15)
Project Audience:						
-End-users			1.13	1.08	1.02	1.02
			(1.61)	(0.98)	(0.29)	(0.24)
-Developers			0.79	0.99	0.89	0.83
			(-3.23)	(-0.12)	(-1.36)	(-2.22)
-System Administrators			1.14	0.94	0.88	0.98
			(1.58)	(-0.64)	(-1.28)	(-0.19)
Project Topic:						
-Communications				0.89	0.81	0.76
				(-0.76)	(-1.27)	(-1.66)
-Security				2.07	1.94	1.84
				(3.42)	(2.68)	(2.36)
-Software Development				0.48	0.46	0.54
				(-5.84)	(-5.47)	(-4.37)
-Desktop				1.08	1.26	1.26
				(0.38)	(1.12)	(1.06)
-Text Editors				0.16	0.18	0.17
				(-1.84)	(-1.67)	(-1.76)
-Database				1.23	1.16	1.23
				(1.36)	(0.92)	(1.30)
-Terminals				0.38	0.38	0.44
				(-2.49)	(-2.39)	(-2.10)
-Education				1.00	0.95	0.78
				(-0.01)	(-0.21)	(-1.07)

-Internet				1.07	1.03	1.05
				(0.51)	(0.22)	(0.34)
-Site Management				0.60	0.63	0.72
				(-2.75)	(-2.49)	(-1.73)
-Human Machine Interfaces				0.70	0.61	0.54
				(-0.95)	(-1.25)	(-1.53)
-Office/Business				1.16	1.13	1.09
				(1.00)	(0.80)	(0.56)
-Dynamic Content				1.35	1.27	0.97
				(2.59)	(2.07)	(-0.28)
-Games				0.93	1.00	1.01
				(-0.36)	(0.03)	(0.07)
-Printing				1.15	1.52	1.60
				(0.48)	(1.41)	(1.57)
Project Status:						
-1 (Planning)					1.14	1.27
					(0.49)	(0.91)
-2 (Pre-Alpha)					0.93	0.94
					(-0.38)	(-0.35)
-3 (Alpha)					1.41	1.33
					(2.89)	(2.31)
-4 (Beta)					1.32	1.28
					(2.90)	(2.49)
-5 (Production/Stable)					1.62	1.67
					(5.08)	(5.08)
-6 (Mature)					2.17	2.50
					(3.23)	(3.75)
Project Operating System:						
-POSIX						1.52
						(4.55)
-MS Windows						1.08
						(0.84)
-Independent						2.13
						(6.72)
Number of observations	1529	1529	1529	1529	1529	1529
Z scores in parentheses.						

Table 7. First Stage Logit Estimates Using Natural Language as an Instrument for License Type

(Odds ratios reported)

	(1) Dependent Variable: All Highly Restrictive Licenses	(2) Dependent Variable: Some Highly Restrictive Licenses
Natural Language:		
- French	0.64	0.68
	(-0.66)	(-0.55)
- Spanish	7.68	-
	(1.51)	
- German	3.01	2.85
	(1.93)	(1.75)
- Russian	0.13	0.26
	(-1.71)	(-0.98)
- Japanese	-	-
Number of Developers	0.95	0.94

	(-1.85)	(-2.09)
Project Audience:		
-End-users	2.44 (2.90)	2.77 (3.21)
-Developers	0.73 (-0.96)	0.84 (-0.51)
-System Administrators	1.69 (1.25)	1.81 (1.35)
Project Topic:		
-Communications	1.56 (0.64)	1.25 (0.33)
-Security	0.95 (-0.04)	0.56 (-0.54)
-Software Development	0.32 (-2.16)	0.26 (-2.54)
-Desktop	2.30 (0.64)	2.13 (0.57)
-Text Editors	-	-
-Database	0.44 (-1.26)	0.56 (-0.92)
-Terminals	0.06 (-1.64)	0.06 (-1.63)
-Education	1.10 (0.11)	0.88 (-0.15)
-Internet	1.81 (0.82)	1.67 (0.70)
-Site Management	1.07 (0.08)	1.53 (0.45)
-Human Machine Interfaces	0.53 (-0.48)	0.39 (-0.69)
-Office/Business	2.33 (1.03)	2.02 (0.86)
-Dynamic Content	1.70 (0.85)	1.44 (0.57)
-Games	0.25 (-1.95)	0.23 (-2.06)
-Printing	0.23 (-1.30)	0.20 (-1.43)
Project Status:		
-1 (Planning)	0.19 (-1.56)	1.03 (0.03)
-2 (Pre-Alpha)	6.06 (2.34)	7.48 (2.47)
-3 (Alpha)	2.91 (2.06)	2.47 (1.71)
-4 (Beta)	3.05 (2.66)	2.81 (2.44)
-5 (Production/Stable)	2.11 (1.73)	1.97 (1.55)
-6 (Mature)	7.41 (1.78)	7.27 (1.76)
Project Operating System:		
-POSIX	1.57 (1.29)	1.76 (1.58)
-MS Windows	0.91 (-0.28)	1.02 (0.05)
-Independent	0.68	0.74

	(-0.89)	(-0.69)
Number of observations	313	301
Z scores in parentheses.		

Table 8. Cox Proportional Hazards Estimation Using Natural Language as an Instrument for License Type

(Hazard ratios reported)

	(1)	(2)	(3)	(4)
ALL HIGHLY RESTRICTIVE LICENSES	0.53 (-1.67)	0.61 (-4.46)	-	-
SOME HIGHLY RESTRICTIVE LICENSES	-	-	0.35 (-2.18)	0.68 (-3.38)
Bug Priority	1.02 (0.78)	1.01 (0.39)	1.02 (0.83)	1.01 (0.36)
Number of Developers	0.99 (-0.11)	1.00 (-0.50)	0.99 (-0.80)	1.00 (-0.43)
Project Audience:				
-End-users	1.15 (1.35)	1.23 (2.38)	1.26 (1.88)	1.18 (1.92)
-Developers	0.80 (-2.51)	0.78 (-2.94)	0.79 (-2.55)	0.81 (-2.43)
-System Administrators	1.09 (0.80)	1.06 (0.58)	1.19 (1.56)	1.04 (0.38)
Project Topic:				
-Communications	0.80 (-1.37)	0.80 (-1.33)	0.78 (-1.49)	0.78 (-1.52)
-Security	1.84 (2.37)	2.09 (2.88)	1.54 (1.63)	1.99 (2.71)
-Software Development	0.46 (-4.81)	0.50 (-5.04)	0.40 (-4.93)	0.49 (-5.01)
-Desktop	1.43 (1.66)	1.39 (1.56)	1.52 (1.87)	1.38 (1.53)
-Text Editors	-	0.19 (-1.63)	-	0.19 (-1.64)
-Database	1.05 (0.28)	1.11 (0.63)	1.03 (0.19)	0.10 (0.61)
-Terminals	0.38 (-2.39)	0.37 (-2.58)	0.36 (-2.48)	0.39 (-2.42)
-Education	0.76 (-1.16)	0.83 (-0.82)	0.74 (-1.32)	0.78 (-1.07)
-Internet	1.09 (0.60)	1.12 (0.77)	1.10 (0.60)	1.10 (0.63)
-Site Management	0.68 (-2.01)	0.75 (-1.59)	0.74 (-1.49)	0.78 (-1.30)
-Human Machine Interfaces	0.48 (-1.86)	0.52 (-1.63)	0.44 (-2.05)	0.50 (-1.73)
-Office/Business	1.19 (1.07)	1.24 (1.38)	1.28 (1.55)	1.15 (0.88)
-Dynamic Content	1.07 (0.51)	1.10 (0.78)	1.13 (0.93)	1.01 (0.11)
-Games	0.89	0.86	0.79	0.91

	(-0.49)	(-0.68)	(-0.92)	(-0.46)
-Printing	1.33	1.36	1.13	1.43
	(0.88)	(1.03)	(0.38)	(1.19)
Project Status:				
-1 (Planning)	1.09	0.93	1.93	1.29
	(0.31)	(-0.27)	(2.18)	(0.97)
-2 (Pre-Alpha)	1.06	1.08	1.15	1.06
	(0.27)	(0.41)	(0.60)	(0.29)
-3 (Alpha)	1.47	1.50	1.35	1.42
	(2.95)	(3.24)	(2.13)	(2.81)
-4 (Beta)	1.37	1.46	1.38	1.37
	(2.83)	(3.75)	(2.62)	(3.18)
-5	1.73	1.79	1.67	1.75
(Production/Stable)	(5.28)	(5.86)	(4.30)	(5.63)
-6 (Mature)	2.80	2.73	3.09	2.62
	(3.92)	(4.14)	(3.82)	(3.99)
Project Operating System:				
-POSIX	1.59	1.65	1.66	1.66
	(4.68)	(5.43)	(4.45)	(5.39)
-MS Windows	1.11	1.07	1.20	1.08
	(1.08)	(0.72)	(1.83)	(0.83)
-Independent	2.08	2.06	2.11	2.08
	(6.45)	(6.55)	(6.48)	(6.60)
Number of observations	1498	1529	1426	1529

Z scores in parentheses.

Table 9. First Stage Logit Estimates Using the Project Leader's Previous Project License Type as an Instrument for License Type

(Odds ratios reported)

	(1) Dependent Variable: All Highly Restrictive Licenses	(2) Dependent Variable: Some Highly Restrictive Licenses
Previous Project – All Highly Restrictive Licenses	8.65	-
	(3.45)	
Previous Project – Some Highly Restrictive Licenses	-	10.25
		(3.44)
Number of Developers	1.03	1.03
	(0.55)	(0.50)
Project Audience:		
-End-users	3.80	4.97
	(1.90)	(2.27)
-Developers	0.61	0.92
	(-0.70)	(-0.11)
-System Administrators	2.91	3.68
	(1.43)	(1.72)
Project Topic:		
-Communications	1.06	0.55
	(0.05)	(-0.55)
-Security	-	-
-Software Development	0.10	0.04
	(-1.88)	(-2.24)
-Desktop	-	-

-Text Editors	-	-
-Database	1.86 (0.51)	4.20 (1.13)
-Terminals	0.30 (-0.29)	1.06 (0.01)
-Education	0.41 (-0.52)	0.20 (-0.91)
-Internet	2.92 (0.92)	0.97 (-0.03)
-Site Management	0.09 (-1.16)	0.06 (-1.30)
-Human Machine Interfaces	-	-
-Office/Business	-	-
-Dynamic Content	3.03 (0.73)	2.63 (0.64)
-Games	0.40 (-0.76)	0.44 (-0.74)
-Printing	-	-
Project Status:		
-1 (Planning)	-	1.34 (0.15)
-2 (Pre-Alpha)	0.39 (-0.36)	-
-3 (Alpha)	7.90 (1.80)	2.53 (0.77)
-4 (Beta)	5.44 (1.83)	3.93 (1.43)
-5 (Production/Stable)	8.04 (2.21)	7.93 (2.13)
-6 (Mature)	-	-
Project Operating System:		
-POSIX	1.12 (0.18)	1.48 (0.58)
-MS Windows	0.89 (-0.16)	1.20 (0.16)
-Independent	0.53 (-0.73)	0.71 (-0.40)
Number of observations	119	119
Z scores in parentheses.		

Table 10. Cox Proportional Hazards Estimation Using the Project Leader's Previous Project License Type as an Instrument for License Type

(Hazard ratios reported)

	(1)	(2)	(3)	(4)
ALL HIGHLY RESTRICTIVE LICENSES	1.09 (0.24)	0.84 (-0.91)	-	-
SOME HIGHLY RESTRICTIVE LICENSES	-	-	1.07 (0.19)	0.77 (-1.48)
Bug Priority	0.92 (-1.88)	0.92 (-1.94)	0.92 (-1.77)	0.92 (-1.81)
Number of Developers	1.00 (-0.19)	1.00 (-0.30)	1.00 (-0.30)	0.99 (-0.60)

Project Audience:				
-End-users	0.88 (-0.68)	0.92 (-0.45)	0.90 (-0.51)	0.97 (-0.16)
-Developers	0.78 (-1.29)	0.72 (-1.68)	0.77 (-1.37)	0.73 (-1.73)
-System Administrators	0.58 (-2.72)	0.63 (-2.45)	0.60 (-2.49)	0.66 (-2.22)
Project Topic:				
-Communications	0.87 (-0.63)	0.86 (-0.71)	0.86 (-0.69)	0.83 (-0.87)
-Security	-	-	-	-
-Software Development	0.46 (-3.09)	0.43 (-3.53)	0.49 (-2.66)	0.44 (-3.35)
-Desktop	-	-	-	-
-Text Editors	-	-	-	-
-Database	1.45 (1.15)	1.42 (1.08)	1.44 (1.13)	1.48 (1.20)
-Terminals	0.53 (-1.39)	0.52 (-1.40)	0.52 (-1.44)	0.52 (-1.41)
-Education	1.18 (0.47)	1.07 (0.19)	1.21 (0.51)	1.03 (0.09)
-Internet	1.17 (0.77)	1.20 (0.91)	1.21 (0.94)	1.27 (1.18)
-Site Management	0.31 (-2.19)	0.30 (-2.37)	0.32 (-2.13)	0.28 (-2.46)
-Human Machine Interfaces	-	-	-	-
-Office/Business	-	-	-	-
-Dynamic Content	1.45 (1.30)	1.49 (1.42)	1.46 (1.31)	1.48 (1.40)
-Games	0.53 (-1.62)	0.47 (-1.92)	0.54 (-1.56)	0.46 (-1.99)
-Printing	-	-	-	-
Project Status:				
-1 (Planning)	-	-	8.55 (4.49)	8.72 (4.54)
-2 (Pre-Alpha)	1.63 (1.53)	1.52 (1.32)	-	-
-3 (Alpha)	2.08 (3.34)	2.24 (3.71)	2.37 (3.14)	2.58 (3.50)
-4 (Beta)	1.52 (2.33)	1.62 (2.72)	1.63 (2.32)	1.83 (2.95)
-5 (Production/Stable)	1.87 (3.61)	1.99 (4.00)	2.03 (3.31)	2.27 (3.95)
-6 (Mature)	-	-	-	-
Project Operating System:				
-POSIX	1.59 (3.17)	1.62 (3.33)	1.54 (2.90)	1.55 (2.96)
-MS Windows	1.12 (0.69)	1.13 (0.75)	1.10 (0.60)	1.12 (0.70)
-Independent	2.12 (3.91)	2.11 (3.87)	2.09 (3.85)	2.04 (3.71)
Number of observations	669	669	657	657

Z scores in parentheses.

References:

Bessen, James (2002), "Open Source Software: Free Provision of Complex Public Goods", working paper, Research on Innovation.
Bonnacorsi, A. and C. Rossi (2003), "Licensing Schemes in the Production and Distribution of Open Source Software: An Empirical Investigation", Sant' Ana School for Advanced Studies, Institute for Informatics and Telematics Mimeo.
Dahlander and Magnusson (2005), "Relationships between Open Source Software Companies and Communities: Observations from Nordic Firms", Research Policy 34, 481.
Ghosh, Glott, Kriger, and Robles (2002), "Free/Libre and Open Source Software: Survey and Study", Universoty of Maastricht, Institute of Infonomics and Berlecon Research GmbH Mimeo.
Lakhani and von Hippel (2000), "How Open Source Software Works: "Free" User-to-User Assistance", MIT Sloan School of Management Working Paper No. 4117.
Lerner, Josh and Jean Tirole (2002), "Some Simple Economics of Open Source", Journal of Industrial Economics, 52; 197-234.
Lerner, Josh and Jean Tirole (2004), The Economics of Technology Sharing: Open Source and Beyond, NBER Working Paper 10956.
Lerner, Josh and Jean Tirole (2005), "The Scope of Open Source Licensing", Journal of Law, Economics and Organization, 21; 20-56.
Holmström, Bengt (1999), "Managerial Incentive Problems: A Dynamic Perspective," Review of Economic Studies, 66, 169-182.
Koski Heli (2005), "OSS production and Licensing Strategies of Software Firms", Review of Economic Research, vol. 2(2), pp. 111-125.
Kuan, Jennifer (2001), "Open Source Software as Consumer Integration into Production", working paper, Stanford University.
Kuan, Jennifer (2002), "Open Source Software as Lead User's Make or Buy Decision: a Study of Open and Closed Source Quality", Stanford University.
Von Krogh, Spaeth, and Lakhani (2003), "Community, Joining, and Specialization in Open Source Software Innovation: A Case Study", Research Policy 32:1217.
Mockus, Fielding, and Herbsleb (2002), "Two Case Studies of Open Source Software Development: Apache and Mozilla", ACM Transactions on Software Engineering and Methodology, 11, 309.