DISCUSSION PAPER SERIES

No. 4329

THE DETERMINANTS OF OUTPUT PER CONTRIBUTOR IN OPEN SOURCE PROJECTS: AN EMPIRICAL EXAMINATION

Chaim Fershtman and Neil Gandal

INDUSTRIAL ORGANIZATION

Centre for Economic Policy Research

www.cepr.org

Available online at: **www.cepr.org/pubs/dps/DP4329.asp**

ISSN 0265-8003

THE DETERMINANTS OF OUTPUT PER CONTRIBUTOR IN OPEN SOURCE PROJECTS: AN EMPIRICAL EXAMINATION

Chaim Fershtman, Tel Aviv University
Neil Gandal, Tel Aviv University, Michigan State University and CEPR

Discussion Paper No. 4329
March 2004

Centre for Economic Policy Research
90–98 Goswell Rd, London EC1V 7RR, UK
Tel: (44 20) 7878 2900, Fax: (44 20) 7878 2999
Email: cepr@cepr.org, Website: www.cepr.org

CEPR Discussion Paper No. 4329

March 2004

ABSTRACT

The Determinants of Output Per Contributor in Open Source Projects: An Empirical Examination*

Using a unique dataset we examine empirically which factors explain output per contributor in open source projects. We find that the output per contributor of open source programmes is much higher when licenses are less restrictive. Further examination suggests that the difference in output per contributor is in large part due to many more contributors for projects that employ restrictive licenses. The results suggest a status/signaling or ideological motivation for participation in open source projects with restrictive licenses.

JEL Classification: D20 and L86
Keywords: empirical study, open source, restrictive licenses and software

Chaim Fershtman
Berglass School of Economics
Tel Aviv University
Tel Aviv 69978
ISRAEL
Tel: (972 3) 640 7167
Fax: (972 3) 640 9908
Email: fersht@ccsg.tau.ac.il

For further Discussion Papers by this author see:
www.cepr.org/pubs/new-dps/dplist.asp?authorid=104930

Neil Gandal
Department of Public Policy
Tel Aviv University
Tel Aviv 69978
ISRAEL
Tel: (972 3) 640 6742
Fax: (972 3) 640 7382
Email: gandal@post.tau.ac.il

For further Discussion Papers by this author see:
www.cepr.org/pubs/new-dps/dplist.asp?authorid=144974

*We are grateful to Judith Chevalier, David Evans, Bernard Reddy, David Steinberg, Manuel Trajtenberg, and seminar participants at the 2004 AEA meetings and Tel Aviv University for helpful comments. Financial support from NERA is gratefully acknowledged. Any opinions expressed are those of the authors.

Submitted 01 March 2004

1. Introduction

The open source model is a form of software development with source code that is made available, free of charge, to all interested parties and all users have the right to modify and extend the program.[1] The open source model has become quite popular and the open source process is sometimes referred to as a movement with an ideology and enthusiastic supporters. At the core of this process are two interesting aspects. The first is that unpaid volunteers do a significant portion of the development of open source programs[2] and the second is that many of the open source programs employ restrictive licenses.

Having unpaid volunteers is always a puzzling issue for economists. What are the incentives of these programmers to invest time and effort in developing open source programs? One possible explanation, provided by Lerner and Tirole (2002a), is that developers of open source programs acquire a reputation, which is eventually rewarded in the job market.[3] But the general view is that in order to understand this phenomenon one needs to move to the relatively unfamiliar territory of non-monetary incentives. That is, personal satisfaction, ideology, and professional status may be important enough to provide incentives to software developers. This argument suggests that the open source process has incentives that are similar to those in academia, although monetary incentives play an important role in academic research.

The analogy between academia and the open source movement may help shed light on the development of open source software. It is clear that commercial R&D and academic R&D are not perfect substitutes. The focus is often on different types of research, different degrees of applicability, etc. Thus it will not be surprising if similar

[1] Open source is different than "freeware" or "shareware." Such software products are often available free of charge, but the source code is not distributed with the program and the user has no right to modify the program. Like other products based on intellectual property, the intellectual property in software is typically "licensed" for use, not sold outright. Someone who purchases a music CD buys the physical CD and the right to play the music under specific circumstances (which do not include the right to play it on the radio, etc). Software, whether proprietary or open-source, is similarly "licensed for use."

[2] Some of the work is now performed by employees of companies. Unpaid volunteers did most or all of the work until 2000/2001, but the model has changed somewhat.

[3] Johnston (2002) develops a formal model of open source software as private provision of a public good.

differences will be found between open source programs developed for commercial purposes and open source programs developed for non-commercial purposes.

Assuming that participation in an open source project is (in part) motivated by the quest for professional recognition or status implies the need to examine status concerns as an incentive.[4] Status is a relative ranking of individuals and as such it cannot be a perfect substitute for monetary gains. Giving status to one individual implies, by definition, lower status to others. Such a characterization may imply tournament-like incentives, but it also implies that the use of status as professional incentives will be limited. If, for example, a society gives too many medals then the value of a medal diminishes.[5] Thus, in the case of open source software, it is not clear if programmers will continue to have "status" incentives when the value of status diminishes. This may have an important effect on the development and continuation of the open source movement. For example the status of the first person to climb Mount Everest is much higher than the status of the 100th person who climbed the mountain. The motivation for the 100th person is likely associated with self-esteem rather than status.

Obtaining status by engaging in open source software development requires that contributions be assigned to individuals. That is, it requires observable effort or observable contributions. In order to earn recognition, it must be clear who made the major contributions to the project. Teamwork and "large teams" do not typically generate professional status. This is similar to the way academia treats papers with multiple coauthors and papers with a single author.

Another question is if all types of software programs generate status. For example programs that help consumers fill out their tax returns will likely provide less status than mathematical programs. Consequently, the different incentives may affect the type of open source projects developed for commercial purposes and those developed for non-commercial purposes, as well as the level of participation in the project.

[4] See for example Frank (1985), Fershtman and Weiss (1993,1996), and Auriol and Renault (2003).
[5] For more details, see Fershtman and Weiss (1993).

What kinds of open source projects will we see in the future? In order to answer this question, we need to have a better understanding of what factors determine success and participation in open source software projects.

One possibly important factor is the type of license under which open source programs are licensed. Open source software is often licensed under the General Public License (GPL). Any program that incorporates software from an open source program under a GPL license must distribute the changes as open source software as well, also under GPL. However, some open source software programs are distributed with BSD[6]-type licenses. Commercial products can be developed using software licensed under a BSD-type license as long as credit is given to the organization responsible for the underlying code.[7] We want to examine how the type of license affects the output (per contributor) of open source projects.

Clearly the potential for non-monetary incentives creates difficulties for both theoretical and empirical work. An additional difficulty for conducting empirical work is that there is no readily observable quantitative measure of success in open source projects. In the case of proprietary software, we can measure sales, licenses, and revenues. These measures are not available for most open source software programs, since the product is distributed free of charge.

We empirically examine which factors explain output per contributor (programmers, developers, etc.) in open source projects. Our main result is that the output per contributor of open source programs is much higher when licenses are less restrictive. Further examination suggests that the difference in output per contributor is in large part due to many more contributors for projects that adopt restrictive licenses.

This result is consistent with the hypothesis that status or self-esteem is a key source of motivation for participation in open source projects with restrictive licenses. Potential contributors in such projects have strong incentives to contribute up to the minimum threshold level in order to be included in the "list of contributors." Once

[6] BSD = Berkeley Software Distribution.

[7] Nevertheless, the BSD license is not like a commercial license, since commercial firms do not typically provide the source code to users, nor do they let users redistribute the code.

someone is included in the list of contributors, the incentive to contribute beyond that level is diminished considerably.

Previous research suggests that projects employing restrictive licenses are less commercial.[8] Hence our results support an ideological or status/signaling motivation in the programmer's decision to participate in a project with a restrictive license. This is because status and signaling may be obtained simply by being on the list of contributors and less by the size or significance of the contribution.

2. Literature Review of Case Studies and Empirical Work

Several papers examine high profile open source projects using case study methodology. Hann, Roberts, and Slaughter (2002) examine whether increased open source participation leads to higher wages using contributors to the Apache HTTP Server Project. Their results find mixed evidence: contributions are not correlated with higher wages, but a higher ranking within the Apache Project is indeed correlated with higher wages. This can be interpreted as signaling high productive capabilities of programmers.

Kuan (2002) argues that since programmers write open source software for their own use,[9] open-source programs will be of greater quality than similar closed-source programs. To test her hypothesis, she examines bug resolution data from three open source programs: The Apache web server, the Linux operating system, and the Gnome user interface. She then compares bug resolution rates in these programs with three similar closed source programs. She finds some support for the hypothesis that open-source bug reports are resolved faster than closed-source service requests. In reviewing Kuan (2002), Shah (2002) notes that "the three open-source projects that Kuan chose are relatively large and well-known, and these results may not be representative; after all, open-source projects differ a great deal with respect to their size, age, formal and informal rules, and so forth."[10] This is a key point; there may be

[8] See Lerner and Tirole (2002b) and Bonaccorsi Rossi (2002).
[9] See Von Hippel (2002).
[10] http://www.researchoninnovation.org/tiip/archive/kuan.htm

potential differences between a relatively small group of well-known open source projects (Linux, Apache, Sendmail, Bind, and Perl) and the general population of open source projects. We discuss this issue further in the conclusion.

While there are several case studies of well-known open source projects, there is little empirical (econometric) work on open source. Krishnamurthy (2002) examined 100 mature products available on the SourceForge web site. He finds that most of these products have very few developers and that most open source software programs do not generate a great deal of discussion.

Lerner and Tirole (2002b) examined the choice of licenses using a very large database of open source projects from the SourceForge web site. They find that projects that run on commercial operating systems and projects that are designed for developers tend to use less restrictive licenses, while projects that are targeted for end users tend to use more restrictive licenses.

Bonaccorsi and Rossi (2002) conducted surveys using Italian firms. They find that (i) firms that employ restrictive licenses supply (on average) fewer proprietary products than firms that do not employ restrictive type licenses and (ii) firms that use restrictive licenses attach greater importance to "social motivation."[11]

3. Data

3.1 Methodology and Data

A major barrier to empirical work on open-source software is that sales/license data are not available for open source projects, in part because the product is typically not sold or licensed for a fee. Hence, we will not be able to measure value or use. But we can measure the size of the open source projects in the form of source lines of code (SLOC). We know that SLOC is not an ideal proxy for success and the largest products (by category) are not necessarily the ones that succeed commercially. Nevertheless, according to "Cost Estimation and Project Management" the most

[11] See Bonaccorsi and Rossi (2002) for discussion.

common measure of productivity is lines of code.[12] We thus believe that it's reasonable to think of SLOC as a proxy for output.

We use the unique number of contributors (denoted CONTRIBUTORS) as a measure of input in a project. All open source projects in our sample list their contributors' email addresses, and the algorithm we employ for counting contributors is designed to control for contributors who list multiple email addresses.

It would be helpful to know whether a small number of individuals made most of the contributions, but such information is not available. Many projects list different roles for individuals (programmer, developer, project manager, etc.), but it is still not possible to quantify the individual contribution. Since the projects are quite heterogeneous, it makes sense to measure output per contributor, which is defined as the ratio of SLOC/CONTRIBUTORS.

The initiators of an open source project likely first choose the license to employ, the operating system, and the intended audience, as well as other characteristics of the project. Once these attributes have been chosen, they typically do not change.[13] Indeed, these variables do not change over time in our data set. Hence, although these variables (characteristics) are clearly not exogenous, they are chosen at an earlier (initial) stage.

Once the characteristics of the project have been chosen, contributors join, code is written, more contributors join, mode code is written, etc. Hence it's likely that both the number of contributors and the size of the program are endogenously determined. Our dependent variable is the ratio of these two endogenous variables, while the other variables are "predetermined."[14]

We employ a unique data set consisting of 71 open source projects hosted at the SourceForge web site. This sample was observed over an eighteen-month period

[12] available at http:///www.comp.glam.ac.uk/pages/staff/bfjones/ils/cocomo.htm (accessed 8/25/03).
[13] There are a few well-known cases in which the license type was changed, but no changes in license type were made in the projects in our sample.
[14] Hence, there does not appear to be any endogeneity issues.

from January 2002 through the middle of 2003, with data collected at two-month intervals.[15] Hence, there are nine observations on each project. The 71 projects in the sample were chosen (in January 2000) from more than 31,000 projects that were listed at the SourceForge web site at the time by selecting the most active projects in the top-level list of "topics" at SourceForge.

It's important to point out that the projects in our sample do not include the well-known successful open source projects, such as Apache, Bind, Linux, Perl and Sendmail since these projects are not hosted at SourceForge.

It is difficult to compare lines of code per contributor between "high" and "low" level programming languages. This is because "lower level" programming languages have more lines of code and take longer to develop than higher level programming languages. Fortunately, every product in our sample uses high level programming languages. Despite the homogeneity in the level of the programming languages, there are likely differences in lines of code across different programming languages. Hence we will also examine the large subset of the data where the programs were written in either C or C++. Of the 71 SourceForge projects in our sample, 56 are written in either C or C++.[16]

Type of license: An important question we wish to examine is whether different types of open source licenses lead higher output per contributor. We therefore distinguish between three levels of license restrictiveness (Very restrictive, Moderately restrictive and Non-restrictive).[17]

- The most popular open source license is the GNU General Public License or GPL. This license requires that the programming commands (or source code) be made available and that other programs that incorporate code from a GPL licensed program must also make the source code fully available under the GPL. Hence, programs that use code from a GPL licensed program cannot

[15] We are grateful to NERA for providing us with the data.
[16] Some of the programs in our sample are available in more than one (high level) programming language.
[17] The classification of licenses into the three categories employs the definitions of Lerner and Tirole (2002b).

become proprietary software. The GPL will be classified as a very restrictive license.

- Another popular license is the GNU Lesser GPL or LGPL. It is also quite restrictive, but less restrictive than the GPL restrictive license. We will refer to this as a moderately restrictive license. Most of the moderately restrictive licenses in our data set use the LGPL. Other moderately restrictive licenses in our data set include the Mozilla, MPL, and NPL licenses.
- The main alternative to the licenses described above is the Berkeley Software Distribution (BSD) type license which has fewer restrictions than the GPL and LGPL licenses. For example, commercial products can be developed using software licensed under a BSD license as long as credit for the underlying code is given to the University of California. Hence, we refer to the BSD-type licenses as "nonrestrictive" licenses. Other licenses in this category that appear in our data set include an MIT license, a Sun Industry Standards Source License, an Intel OSL and an Apache Software License.

In the empirical work, we control for the operating system that the project employs, as well as whether the project was written for single or multiple operating systems. We also control for the age of the project and program type by including information about whether the program is intended for developers, system administrators, or end users.

3.2 Descriptive Statistics

Data were collected on 71 SourceForge hosted open source projects over an eighteen-month period at two-month intervals. Hence there are nine observations on each project. The following variables are available for the study.

- SLOC - the source lines of code at each point in time for each project.
- CONTRIBUTORS – the number of contributors to the project at each point in time for each project.
- OUTPUT PER CONTRIBUTOR – the ratio of SLOC/CONTRIBUTORS.
- NEW VERSION – This variable takes on the value one if a new version of the program was developed in the two month period between observations.

- ADVANCED STAGE – The variable takes on the value one if the project was either in stage 5 or stage 6, where stage 5 is defined to be "Production/Stable" and stage 6 is defined to be "Mature," and zero otherwise.[18]
- Operating systems:
    1. LINUX - This variable takes on the value one if the project is written for the Linux operating system and takes on the value zero otherwise.
    2. MICROSOFT – This variable takes on the value one if the project is written for one of the Microsoft operating systems and takes on the value zero otherwise.
    3. POSIX – This variable takes on the value one if the project is written for the POSIX standards and takes on the value zero otherwise.
    4. MAC - This variable takes on the value one if the project is written for the MAC operating system and takes on the value zero otherwise.
    5. BSD OS - This variable takes on the value one if the project is written for one of the BSD Unix-like operating systems (i.e., FreeBSD, NetBSD, or OpenBSD) and takes on the value zero otherwise.
    6. SUN OS - This variable takes on the value one if the project is written for the SUN operating system (Solaris, a variant of Unix) and takes on the value zero otherwise.
    7. OSI - This variable takes on the value one if the project is independent of any operating system and takes on the value zero otherwise.
- SINGLE OS – This variable takes on the value one if the project was written for a single operating system and zero otherwise.

- We have several variables that identify how restrictive is the license:[19]
    1. RESTRICTIVEV – This variable takes on the value one if the project is licensed under the General Public License (GPL). Otherwise, this variable takes on the value zero.

[18] Each project was rated by stage of production, ranging from 1 to 6. Nearly all of the projects in our sample were in stages 3-6. We employ the variable in this manner, because some projects listed multiple stages at a single point in time, while other projects listed a single stage. Only a few projects that listed a single stage at each point in time changed stages during our time period.

[19] The empirical results suggest that there is little difference between very restrictive and moderately restrictive licenses. Hence we generally use the variable RESTRICTIVE. Nevertheless, we also discuss how the results change when we delineate restrictive licenses into very restrictive and moderately restrictive licenses.

2. RESTRICTIVEM –This variable takes on the value one if the project uses any one of the following licenses: LGPL, Mozilla, MPL, or NPL. Otherwise, the variable takes on the value 0.
3. RESTRICTIVE – This variable takes on the value one if the license is either very restrictive (RESTRICTIVEV=1) or moderately restrictive (RESTRICTIVEM=1).
4. NONRESTRICTIVE - This variable takes on the value one if the license employed is a BSD-type license.

- AGE – This variable is the age of the project in years.
- DESKTOP – This variable takes on the value one if the intended audience is end users, i.e., users with desktops or laptops, and zero if the intended audience does not include end users.
- SYSTEM – This variable takes on the value one if the intended audience is system administrators and zero if the intended audience does not include system administrators.
- DEVELOPER - This variable takes on the value one if the intended audience is developers and zero if the intended audience does not include developers.
- LANGUAGE – This variable takes on the value one if the program is written in C or C++ and 0 otherwise.

Descriptive statistics are shown in Table 1. Table 1 shows that 62% of all projects in the sample employed the GPL license. The table also shows that only a few projects in the sample employ multiple types of licenses. The categories RESTRICTIVE and NONRESTRICTIVE are nearly exclusive. Indeed, only two projects in the sample offered both restrictive and nonrestrictive licenses.[20]

Table 1 also shows that the most popular "operating systems" are the LINUX, MICROSOFT, and POSIX operating systems. Indeed one of these operating systems is employed in every project in our sample except for those projects that are exclusively operating system independent (OSI). Hence we focus on these operating

[20] The two projects were available under both licenses throughout the period for which we have data.

systems in the analysis. Table 2 shows the breakdown of the projects according to these operating systems.[21]

Table 3a shows the differences on key variables between projects with restrictive licenses and projects with non restrictive licenses. The most striking difference in Table 3a is that the output per contributor is approximately 6642 for the programs employing non-restrictive licenses and approximately 2319 for the programs employing restrictive licenses.

Table 3b provides summary data for projects written for developers. The summary data in Table 3b are consistent with the Lerner and Tirole (2002) result that programs written for developers tend to use less restrictive licenses and employ commercial operating systems. The table shows that 80 percent of the programs not written for developers employ very restrictive licenses, while only 52 percent of the programs written for developers employ very restrictive licenses. Additionally 43 percent of programs written for developers are written for a Microsoft operating system, while 26 percent of programs not written for developers employ a Microsoft operating system. Similarly 44 percent of programs written for developers employ a Linux operating system, while 64 percent of programs not written for developers are written for the Linux operating system.

4. Model

We examine factors that explain output per contributor. We have a panel data set, that is, nine observations for nearly all projects over an eighteen month time period. Hence we employ a random effects model of the form

(1) $y_{it}=X\beta + \mu_i + \varepsilon_{it}$,

where the index "i" represents the project and index "t" represents the time period. y_{it} is the output per contributor in project i at time t, X is a matrix that contains the characteristics of each project (license type, operating system, etc.), β measures the

[21] It is our understanding that POSIX isn't technically an operating system, but rather a set of standards. Nevertheless, we'll refer to POSIX as an operating system. No projects were written for both LINUX and POSIX operating systems.

effect of these characteristics, μ_i is the random effect of project i and ε_{it} is a white noise error term.

Alternatively, we could have employed a fixed effects model, but the random effects model is appropriate since the 71 projects are a random sample from the larger population of projects hosted at Sourceforge. (The main results are robust to employing a fixed effects model.)

Perhaps more importantly, (i) the Breusch and Pagan (BP) Lagrange multiplier test and (ii) the Hausman (HM) specification test indicate that the random effects model is appropriate. The results are robust to using maximum likelihood estimation or generalized least squares (GLS) estimation. In order to be concise, we report only the GLS results.

5. Empirical Results

5.1 The Effect of Restrictive Licenses

Table 4 presents three regressions using the full data set. The main result in Table 4 is that after controlling for operating system, audience, etc., projects that employ restrictive licenses result in significantly less output per contributor than projects that do not employ restrictive licenses. This is a striking result since, unlike many of the other characteristics, the type of license does not "technically" affect the writing of code. The result is "economically" as well as statistically significant: other things being equal, output per contributor is approximately 5000 lines less when a restrictive license is employed.

The result is very robust to different specifications as shown in Table 4. The difference between regressions 1 and 2 in Table 4 is that in table 2, RESTRICTIVEV and RESTRICTIVEM are employed rather than restrictive. Similarly, regression 3 in Table 4 shows that result is robust to employing LINUXONLY, MICROSOFTONLY, and POSIXONLY, rather than LINUX, MICROSOFT and POSIX.[22] The result is also robust to adding LANGUAGE to the set of explanatory variables as well.

[22] Of course in this case, SINGLEOS has to be excluded due to multicollinearity.

In the first regression in Table 5, we present regression results for projects written for either C or C++. Table 5 shows that the estimated coefficient on RESTRICTIVE is of the same order of magnitude as the estimated coefficients on RESTRICTIVE in Table 4 (-6207, t=-2.81).

As noted by Lerner and Tirole (2002b) and Bonaccorsi and Rossi (2002), projects with restricted licenses likely have limited commercial potential. Our empirical result that the ratio of output per contributor is larger in our data for projects that employ nonrestrictive licenses suggests an ideological or status/signaling motive in the programmer's decision to participate in a project with a restrictive license. This is because status and signaling may be obtained simply by being on the list of contributors and less by the size or significance of the contribution.

An interesting question is whether this result is due to the fact that there are more contributors or less code per project, or some combination of both. Some clues can be obtained by examining medians rather means. The median output per contributor is approximately 2125 for the programs employing non-restrictive licenses and approximately 1367 for the programs employing restrictive licenses. Hence, even in the case of medians, there is a large difference.

There is a relatively small (approximately 13 percent) difference in the median lines of code per project: the median is approximately 52,978 source lines of code for programs employing restrictive licenses and 60,309 source lines of code for projects that do not employ such licenses.

On the other hand, there is a very large difference in the median number of contributors: 35 for projects that employ restrictive licenses and 13 for projects that do not employ restrictive licenses. This provides support for the ideological or status/signaling incentive to be a contributor to a project with a restrictive license.

5.2 The Effect of Program Type

We included the variables SYSTEM, DEVELOPER, and DESKTOP in the regressions in Table 4 in order to control for the heterogeneity among projects. The

results indicate that output per contributor in projects oriented towards end users (DESKTOP) and system operators (SYSTEM) is significantly lower than that in projects for developers. This may be because these projects are less commercial than projects for developers or it may be due to technical issues or some combination of both.

In order to explore this issue further, the second regression in Table 5 includes only projects written for developers. Since many of the projects are for multiple audiences, we still control for end users and system operators. The overall R-squared of this regression is much higher than the other regressions in Tables 4 and 5, suggesting that it is important to control for program type.

The results in Table 5 show that projects that employ restrictive licenses have much smaller output per contributor than projects that do not employ restrictive licenses (-7906, -3.37). Since programs written for developers tend to be commercial, these results reinforce the notion that contributors in non-commercial programs do so for a signaling, status or ideological motive.

5.3 The Effect of Operating Systems

The regressions in Table 4 provide some evidence that projects written for the Linux operating system have lower OUTPUT PER CONTRIBUTOR than projects written for other operating systems. The effect is statistically significant in the first two regressions in Table 4 at the 10% level and the difference is approximately 800 lines per contributor. This difference is much smaller than the difference in output per contributor for different license types. In the third regression in Table 4, when we include the LINUXONLY, MICROSOFTONLY, and POSIXONLY, rather than LINUX, MICROSOFT and POSIX, the difference increases to more than 2000 lines per contributor and is more statistically significant.

In the case of programs written in C/C++ (first regression in Table 5), the magnitude of the effect is similar to the third regression in Table 4. In the case of projects written for developers (second regression in Table 5), the LINUX effect (-1556, -1.04) is statistically insignificant, even though the estimated magnitude is fairly large.

In this regression, the estimated coefficient on Microsoft is positive and statistically significant (4024, t=1.82).

It's possible that these differences are due to technical issues, that is, programs written for the LINUX operating system require less code overall. On the other hand, the difference is possibly due to an ideological or status motivation in the programmer's decision to participate in a project.

Tables 4 and 5 show that the magnitude of the POSIX effect is similar in magnitude to the LINUX effect. Nevertheless, the effect is not statistically significant in regressions 1 and 2 in Table 4.

5.4 Other factors that affect the size of open source programs

We now discuss the other factors that affect the output per contributor of open source programs. The coefficient on NEWVERSION is negative as expected, although not statistically significant in either Table 4 or Table 5. Hence there is some (weak) evidence that per contributor output is slightly lower around the time when a new version is introduced. The other factors generally do not explain the variance in output per contributor, but we include them as controls. There is a potential sample selection issue associated with AGE, since age is the initial date at which the project was first hosted at SourceForge. If the project began elsewhere, it is indeed older than the age we employ. This likely does not have any effect on our main results because actual age is likely quite highly correlated with the age from when the project first appeared at SourceForge. Additionally, the estimated coefficient on AGE is virtually zero, suggesting that age is not a factor in explaining the variance in output per contributor.

6. Conclusions and Further Discussion

The main result is that other things being equal, output per contributor of open source programs is much higher when licenses are less restrictive. This result suggests an ideological or status/signaling motivation in the programmer's decision to participate in a project with a restrictive license. This is because status and signaling may be obtained simply by being on the list of contributors and less by the size or significance of the contribution.

This research has answered some questions, but also has raised several interesting issues for future work. SourceForge is the largest open source development site and it hosts a very large number of open source projects. Nevertheless, the well known open source projects such as LINUX (operating system), APACHE (web server), BIND (A domain name software), Perl (a scripting language), and Sendmail (an E-mail server) are not hosted at SourceForge. It is possible that the well-known open source projects differ from the projects hosted at SourceForge in important ways. For example, of the above open source programs, only LINUX was exclusively released under the GPL. The other four projects use BSD-type licenses.[23] It would be interesting to empirically compare well-known (successful) open source projects (which are not typically hosted at SourceForge) with projects hosted at SourceForge.

Additional research questions include whether proprietary and open source programs can co-exist in the same markets and if so which markets (i.e., office suites, servers, etc.). Finally, can we expect the same level of updating, maintenance, and compatibility between open source and proprietary software programs? We leave these questions for future work.

[23] It is our understanding that Perl uses both a BSD-type license and a GPL license.

References:

Auriol, E. and R. Renault (2003), "Status and Incentives" Working paper, available at http://www.idei.asso.fr/Commun/Articles/Auriol/status_incentives.pdf

Bonaccorsi, A., and C. Rossi (2002), "Why Open Source Software can Succeed," mimeo.

Fershtman, C. K. Murphy and Y. Weiss (1996) "Education, Status and Growth" *Journal of Political Economy*, 104, 108-132.

Fershtman C. and Y. Weiss (1993), "Social Status, Culture and Economic Performance" *Economic Journal*, 103, 946-950.

Frank, R. (1985), *Choosing the right pond*, Oxford: Oxford University Press.

Hann, I., Roberts, J., and S. Slaughter, "Delayed Returns to Open Source Participation: An Empirical Analysis of the Apache HTTP Server Project, Carnegie Mellon University mimeo, 2002.

Johnson, J. (2002). "Open Source Software: Private Provision of a Public Good," *Journal of Economics & Management Strategy*, 11: 637-662.

Kuan, J., "Open Source Software as Lead User's Make or Buy Decision: A Study of Open and Closed Source Quality," Stanford Institute for Economic Policy Research Mimeo, 2002.

Lerner, J., and J. Tirole, 2002a, "Some Simple Economics of Open Source" *Journal of Industrial Economics*, 52: 197-234.

Lerner, J., and J. Tirole, 2002b, "The Scope of Open Source Licensing, mimeo, available at http://opensource.mit.edu/papers/lernertirole2.pdf (accessed August 28, 2003).

Shah, S., "A Summary of 'Open Source Software as Lead User's Make or Buy Decision: A Study of Open and Closed Source Quality," TIIP Newsletter, available at http://www.researchoninnovation.org/tiip/archive/kuan.htm (accessed August 28, 2003).

Von Hippel, E., "Open Source Software Projects as User Innovation Networks, MIT mimeo 2002.

Table 1: Descriptive Statistics (N=636)[24]

VARIABLE	MEAN	STD. DEV.	MIN	MAX
SLOC	120,355.10	266,341.30	275	2,337,136
CONTRIBUTORS	83.32	157.50	1	1091
OUTPUT PER CONTRIBUTOR	3358.91	6485.26	68.75	55024.50
AGE	2.06	0.73	0.027	3.50
NEW VERSION	0.50	0.50	0	1
ADVANCED STAGE	0.58	0.49	0	1
LINUX	0.50	0.49	0	1
MICROSOFT	0.37	0.48	0	1
OS INDEPENDENT	0.30	0.46	0	1
POSIX	0.26	0.44	0	1
BSD	0.13	0.33	0	1
MAC	0.08	0.28	0	1
SUN	0.08	0.27	0	1
LINUXONLY	0.13	0.34	0	1
POSIXONLY	0.063	0.24	0	1
MICROSOFTONLY	0.13	0.33	0	1
SINGLE OS	0.30	0.46	0	1
RESTRICTIVEV	0.62	0.49	0	1
RESTRICTIVEM	0.18	0.39	0	1
RESTRICTIVE	0.76	0.42	0	1
NONRESTRICTIVE	0.27	0.44	0	1
DESKTOP	0.64	0.48	0	1
SYSTEM	0.31	0.46	0	1
DEVELOPER	0.65	0.48	0	1
LANGUAGE	0.79	0.41	0	1

[24] There are potentially 639 observations. We lack some data on three of the observations.

Table 2: Number of projects per operating system

Operating System	Number of Projects
LINUXONLY	9
POSIXONLY	9
MICROSOFTONLY	4
LINUX + MICROSOFT	16
POSIX + MICROSOFT	7
LINUX + OTHER OS	12
POSIX + OTHER OS	2
EXCLUSIVELY OSI	12

Table 3a: Descriptive statistics according to license use

Projects that employ restrictive licenses (N=483)				
VARIABLE	**MEAN**	**STD. DEV.**	**MIN**	**MAX**
SLOC	81,531.07	99,870.05	3619	530,314
CONTRIBUTORS	64.20	78.38	1	469
OUTPUT PER CONTRIBUTOR	2,318.63	2,462.53	250.95	10,904.8
DESKTOP	0.61	0.49	0	1
SYSTEM	0.24	0.45	0	1
DEVELOPERS	0.63	0.48	0	1
LINUX	0.52	0.50	0	1
MICROSOFT	0.42	0.49	0	1
POSIX	0.28	0.49	0	1
OSI	0.27	0.44	0	1
SINGLE OS	0.35	0.48	0	1
LANGUAGE	0.85	0.85	0	1
Projects that do not employ restrictive licenses (N=153)				
SLOC	242,917.40	494,786.20	275	2,337,136
CONTRIBUTORS	143.70	281.64	1	1091
OUTPUT PER CONTRIBUTOR	6,641.61	11,924.35	68.75	55,024.50
DESKTOP	0.70	0.46	0	1
SYSTEM	0.28	0.49	0	1
DEVELOPERS	0.71	0.46	0	1
LINUX	0.46	0.50	0	1
MICROSOFT	0.24	0.43	0	1
POSIX	0.18	0.38	0	1
OSI	0.48	0.50	0	1
SINGLE OS	0.24	0.43	0	1
LANGUAGE	0.59	0.49	0	1

Table 3b: Descriptive statistics according to audience

Projects for developers (N=414)				
VARIABLE	**MEAN**	**STD. DEV.**	**MIN**	**MAX**
SLOC	165584.00	320592.70	4044	2337136
CONTRIBUTORS	102.29	189.79	1	1091
OUTPUT PER CONTRIBUTOR	4238.83	7730.89	250.95	55024.5
DESKTOP	0.52	0.50	0	1
SYSTEM	0.27	0.44	0	1
LINUX	0.44	0.50	0	1
MICROSOFT	0.43	0.50	0	1
POSIX	0.30	0.46	0	1
OSI	0.44	0.50	0	1
RESTRICTIVE	0.74	0.44	0	1
RESTRICTIVEV	0.52	0.50	0	1
RESTRICTIVEM	0.28	0.45	0	1
SINGLE OS	0.24	0.43	0	1
LANGUAGE	0.76	0.43	0	1
Projects not for developers (N=222)				
SLOC	36009.47	27832.08	275	105345
CONTRIBUTORS	47.95	45.10	1	237
OUTPUT PER CONTRIBUTOR	1717.07	2235.70	68.75	10699.5
DESKTOP	0.85	0.36	0	1
SYSTEM	0.39	0.49	0	1
LINUX	0.64	0.48	0	1
MICROSOFT	0.26	0.44	0	1
POSIX	0.17	0.38	0	1
OSI	0.081	0.273577	0	1
RESTRICTIVE	0.80	0.402921	0	1
RESTRICTIVEV	0.80	0.402921	0	1
RESTRICTIVEM	0	0	0	0
SINGLE OS	0.48	0.50	0	1
LANGUAGE	0.84	0.37	0	1

Table 4: Dependent Variable: OUTPUT PER CONTRIBUTOR (Full Data Set)

	Regression 1		Regression 2		Regression 3	
Independent Variables	Coeff.	T stat	Coeff.	T stat	Coeff.	T stat
CONSTANT	12640.86	5.94	13157.77	6.11	10864.14	5.15
RESTRICTIVE	-4840.83	-2.90			-4926.39	-2.93
RESTRICTIVEV			-3811.57	-2.40		
RESTRICTIVEM			-6135.64	-2.79		
DESKTOP	-2943.94	-1.97	-4141.83	-2.48	-3062.4	-2.01
SYSTEM	-2433.84	-1.57	-2465.52	-1.59	-2523.48	-1.60
DEVELOPERS	168.912	0.27	258.93	0.41	134.7423	0.21
LINUX	-2009.03	-3.39	-2070.84	-3.49		
LINUXONLY					-2040.09	-3.25
MICROSOFT	323.17	0.54	265.13	0.44		
MICROSOFTONLY					813.44	1.28
POSIX	-2256.97	-2.43	-2233.54	-2.40		
POSIXONLY					-2383.97	-1.08
OSI	-2112.14	-3.42	-2067.32	-3.34	-1018.17	-1.52
AGE	-35.82	-0.37	-35.64	-0.37	-32.62	-0.34
NEWVERSION	-163.88	-1.67	-161.88	-1.65	-151.72	-1.55
ADVANCED STAGE	-282.25	-0.57	-219.23	-0.44	-222.92	-0.49
SINGLE OS	-1901.03	-3.28	-1925.04	-3.32		
Num. of observations	636		636		636	
	χ^2 stat	p val	χ^2 stat	p val	χ^2 stat	p val
BP Test[25]	2321.65	0.00	2328.33	0.00	2318.73	0.00
HM Test[26]	9.08	0.43	8.12	0.52	6.03	0.54

[25] Formally, the null hypothesis for the BR LM test is that var(u)=0. If the null hypothesis is rejected, the random effects model is appropriate. The chi-squared statistics in tables 4 and 5 clearly indicate that the null hypothesis should be rejected in all of the regressions. This result is also true for all of the supplementary regressions discussed in the text.

[26] Formally the HM statistics tests the null hypothesis that the independent variables (X_i) are not correlated with μ_i , the random effect of project i. The chi-squared statistics in tables 4 and 5 clearly indicate that the null hypothesis should not be rejected in any of the regressions. Hence the evidence indicates that X_i and μ_i are uncorrelated. The same result also holds for all of the supplementary regressions discussed in the text as well.

Table 5: Dependent Variable: OUTPUT PER CONTRIBUTOR

	Projects Written in C/C++		**Projects for Developers**	
Independent Variables	Coefficient	T-statistic	Coefficient	T-statistic
CONSTANT	13326.36	5.14	14698.89	4.01
RESTRICTIVE	-6206.95	-2.81	-7905.71	-3.37
DESKTOP	-2263.2	-1.27	-2899.94	-1.44
SYSTEM	-1296.64	-0.68	-1395.61	-0.6
DEVELOPERS	61.88	0.09		
LINUX	-2144.37	-3.27	-1555.53	-1.04
MICROSOFT	356.69	0.54	4023.50	1.82
POSIX	-2444.49	-2.34	-3804.84	-1.58
INDEPENDENT OS	-2060.47	-3.06	-1561.27	-0.95
AGE	36.611	0.31	34.51	0.26
NEWVERSION	-186.27	-1.58	-117.94	-0.88
ADVANCED STAGE	-409.64	-0.75	-2668.11	-1.21
SINGLE OS	-1915.88	-3.03	-1045.17	-0.36
Number of observations	501		414	
	χ^2 stat	p value	χ^2 stat	p value
BP Test	1806.54	0.00	1483.04	0.00
HM Test	9.27	0.41	4.06	0.40