The Governance of Open Source Software Communities: An Exploratory Analysis

Ivan De Noni and Andrea Ganzaroli
University of Milan, Italy

Luigi Orsi
University of Padua, Italy

Abstract

In this paper, we investigate the nature of the relationships between dimensions of governance in Open Source (OS) communities. A recent review highlighted this issue as critical. Furthermore, this issue has been recognized as strategic for managing the trade-off between innovation and standardization, the capacity of firms to profit from their investment in open source, and the sustainability of OS projects. Our results are based on a comparative analysis of 40 OS projects contained in the Freshmeat dataset. For each project, we collected data on the governance solutions implemented. Governance mechanisms have been ranked for their degree of openness. Our findings show that OS governance is configurational. Those configurations are defined along two dimensions: leadership and decision-making distribution, and reciprocity of the appropriability regime. Four configurations are indeed defined: open source, sponsored, reciprocity-based, and tolerant dictatorship. Those configurations have been defined based on an exploratory factor analysis.

Keywords

OSS Governance, Open Source Software, Community Management, Design of Participation Architecture

Introduction

Scholars have become increasingly aware of the need to specify the type of governance in an Open Source Software (OSS) project (O'Mahony 2007). Governance, according to Williamson (1999), is "a means by which to infuse *order* in a relation where potential *conflict* threatens to undo or upset opportunities to realize *mutual* gains." Governance structure is defined as the implicit or explicit contractual framework within which a transaction is located. In OS communities, the licensing agreement is the contractual framework. Its main characteristic is to bound the efficiency of private appropriation as a means of incentive, coordination, and control (e.g., Damil and Lecocq 2006). OS governance

This study has received funding from the European Community's Seventh Framework Programme FP7 – 2007 – 2013 under grant agreement n° 225349.

can be defined as the means of achieving the direction, control, and coordination of wholly or partially autonomous individuals and organizations on behalf of the OSS development project to which they jointly contribute (Markus 2007). In this perspective, in a recent review of the literature, Markus (2007) argued the need for deepening our understanding of the relationships between governance dimensions. Are those dimensions independent, or is OS governance reducible to a small number of configurations? If OS governance is reducible to a limited set of configurations, do those have an impact on the efficiency and/or effectiveness of these communities? Those are the main questions to which this paper attempts to provide an answer.

Following Markus (2007), in the existing literature three main perspectives are identifiable. The first, monolithic, suggests that OS governance is an integrated and self-coherent system of coordination and control. Furthermore, this model of governance is an alternative to the market, hierarchy, and networks (Demil & Lecocq 2006). In the second, multidimensional, governance is the result of multiple equilibriums. Therefore, almost an infinite number of governance solutions can be implemented. Finally, in the third OS, governance is defined as configurational (e.g., West & O'Mahony 2008; West 2003; O'Mahony & West 2005; von Hippel & von Krogh 2003). There is a strong correlation between governance dimensions. Therefore, only a limited number of solutions can be set up. Furthermore, according to this perspective, in those configurations elements of the market and of the hierarchy are mixed together.

Among those who see governance as configurational, a distinction is made between community managed and sponsored (O'Mahony 2003; O'Mahony 2007; West & O'Mahony 2008). The first term characterizes projects whose governance is independent from the influence of a single company or a coalition of companies (West & O'Mahony 2008; West 2003; O'Mahony & West 2005; von Hippel & von Krogh 2003; Shah 2006). The second, differently, defines communities where governance is largely controlled by a single firm or a coalition of firms. Those two configurations are often defined as the opposite ends of a single continuum (O'Mahony 2003; O'Mahony 2007; O'Mahony & Ferraro 2007; West & O'Mahony 2008; Shah 2006a). Thus, governance dimensions are considered to form a single bundle, and governance is the trade-off between two contradictory objectives (West & O'Mahony 2008; O'Mahony & West 2005; Shah 2006a). The first is to stimulate participation and innovation through openness and appropriability. The second is to secure profitability and/or compatibility by retaining some degree of control over the source code development.

Our analysis is based on a sample of 40 case studies of OS projects. These cases have been extracted from the Freshmeat dataset. The cases have been sampled randomly from the dataset, taking into account the type of project, the vitality score, and the number of registered users. For each project in the sample, data about the governance have been collected according to the dimensions of governance proposed by West and O'Mahony (2008). However, this model provides us only a term of reference to conduct a comparison of those projects. Data have been collected by visiting the website of the project and, if not sufficient, by sending an e-mail to the project leader. We conducted a correlation analysis to verify whether those dimensions were correlated. This analysis confirms that OS governance is configurational. Therefore, an exploratory factor analysis with only one factor has been developed to test whether these dimensions could be reduced to a single continuum. The resulting model is structurally sound. However, a group of three variables is not explained by the model. Therefore, we conducted a second exploratory factor analysis with two factors. The second model is also robust. Therefore, OS governance is reducible to two latent factors: leadership and decision-making distribution, and reciprocity of the appropriation regime. The intersection between these two dimensions defines four OS governance configurations: open source, sponsored, reciprocity-based, and tolerant dictatorship. Furthermore, to explore whether those configurations might have some implications in terms of a community's performance, we conducted an ANOVA[1] with respect to the vitality score and the vitality score calculated on the major releases. In both cases, the configuration results are statistically significant.

---

1 Analysis of Variance: in its simplest form, ANOVA provides a statistical test of whether the means of several groups are all equal.

The structure of the paper is the following. In the next section, we address the concept of OS governance. We focus on two major aspects. The first is the purpose of OS governance. We argue that the main purpose of OS governance is to leverage participation and contribution. The second is whether and how the objective of governance changes over time. From this second point of view, we argue that the set of resources required for an OS project to succeed changes over time. Therefore, the objective of OS governance is expected to change compatibly. In section 4, we analyze the specific dimensions of OS governance. Sections 5 and 6 are devoted to methodology and data analysis, respectively. Results are discussed in section 7. Finally, section 8 is devoted to concluding remarks and further research.

The Purpose of OS Governance

Although OS governance is a rather recent issue in the literature, identifying some differences in the way this category is conceptualized is already possible (Markus 2007).

The first difference is in the purpose assigned to OS governance. Contributions, from this perspective, can be divided into three categories. The first category refers to those who believe that the main purpose of OS governance is to solve collective action dilemmas (e.g., Demil & Lecocq 2006; Franck & Jungwirth 2003; Bonaccorsi & Rossi 2003; Rossi & Bonaccorsi 2005; Osterloh & Rota 2007). Those who subscribe to this view tend to focus more on the lack of private incentives and on the way this lack is overcome in those communities. Differently, those who believe that the main purpose of OS governance is to sustain coordination and control highlight the lack of hierarchical power due to the voluntary nature of participation and, in contrast, the specific coordination mechanisms set up by these communities to secure the quality of the collective good produced (Lee & Cole 2003; Kogut & Metiu 2001; Crowston 1997; Scacchi 2004). Finally, a third perspective refers to those who believe that OS governance serves to create a better climate for contributors (Shah 2006a; West & O'Mahony 2008; Baldwin & Clark 2003). Those who subscribe to this latest perspective argue that governance does not have only a passive role. Governance is not only a solution to an existing problem but is also strategic for the type or resources attracted and the quality of the social relationships prevailing in a community.

In this paper, we subscribe to the third perspective. Therefore, we assume that the purpose of OS governance is not only to support collective action or to secure the quality of the collective good produced but also to secure a sufficient supply of people to sustain development and innovation. The basic idea behind this view is that the capacity of a community to attract and retain people willing to collaborate on development depends on the quality of the social atmosphere and indeed of the social relationships prevailing in the community. The hypothesis is that people, if they can choose, prefer to work in a workplace that corresponds to their interests, attitudes, and competencies. In the case of OS, this hypothesis is even more justified because most people are not obliged to participate in an OS project but choose to participate[2] (e.g., Raymond 1999; Bonaccorsi & Rossi 2003). However, their question is not so much whether they want to participate in an OS project but in which one they want to participate (Markus 2007). Furthermore, people are not even obliged to remain on a project. The costs of switching from one community to another are low. Therefore, if people are dissatisfied with their experience of being part of a given community, they will choose another community or, at worst, definitively leave the OS movement.

If we assume that OS governance is strategic for the quality of the people attracted and retained in a community, the focus is no longer on OS governance per se and how this is different from other forms of governance, but also on the differences existing among alternative solutions/configurations of OS governance and how those differences impact the decision to participate in an OS community.

Extensive literature addresses participants' motivations for participating in an OS community (Bonaccorsi & Rossi 2003; Rossi & Bonaccorsi 2005; Lerner & Tirole 2000; Franck & Jungwirth

[2] Someone could argue that a number of employees are obliged to work for an OS project. This is true. However, the labor market in the software industry is very competitive. Therefore, someone who is not willing to work for an OS company can relatively easily change jobs. Furthermore, empirical analysis shows that most people participate in OS projects to enhance their career opportunities.

2003; Shah 2006a; Lakhani & von Hippel 2003; Feller & Fitzgerald 2002). It is not an objective of this paper to discuss this topic in-depth. Otherwise, we simply summarize those arguments useful to our scope. First, there is a distinction among three categories of motivations: economic, social, and technological (Feller & Fitzgerald 2002). Economic reasons are traced back to a classical cost-benefit framework. Social motivations are often tied to the hacker culture that originally inspired the free/open source software development. Finally, technological motivations are tied to the opportunity to learn from others' contributions and feedback. These categories are applicable to individuals and firms. However, the content changes accordingly to the type of actor considered (Feller & Fitzgerald 2002; Rossi & Bonaccorsi 2005; Bonaccorsi & Rossi 2003; Dahlander & Magnusson 2005). Economically speaking, individuals are motivated by monetary rewards, low opportunity costs, and gaining a reputation among peers for future career benefits. Differently, firms are motivated by being independent of the price and license policies of the large software companies, making money on complementary services or products, and hiring good IT specialists. Individuals' social motivations are the fun of programming, altruism and gift economy, sense of belonging, and fighting against proprietary software. Firms, from this perspective, tend to conform to the values of the hacker culture either to reciprocate or not dislike the other members of the community. However, a number of firms behave parasitically. Finally, from a technological point of view, individuals are motivated by the opportunity to learn, to contribute and receive feedback, to work with leading-edge technology, and to follow a personal compulsion. Firms, differently, are more motivated by exploiting feedback in terms of cost reduction and innovation, diffusion and win adoption, and promotion of standardization.

However, to design a governance architecture capable of enhancing the capacity of a community to attract and retain qualified people, knowing that people and individuals participate for multiple and different reasons is not sufficient. It is also necessary to know how those attitudes translate into preferences in terms of community environment and to what extent these preferences are compatible with each other.

From this perspective, Franck and Jungwirth (2003) proposed to distinguish participants in two groups: rent-seeker and donators. Rent-seekers are mainly motivated by economic reasons. Donators, on the contrary, are more driven by social/idealistic motivations, such as contributing for free to the production of a collective good. Therefore, donators do not expect to receive anything in exchange for their contributions. According to those authors, the basic institutional innovation in OS has been the crafting of a governance structure, which enables rent-seeking without crowding out donations. The core of the authors' argument lies in the peculiar nature of free/open source software licenses. These licenses safeguard the value of the software's reputation and, therefore, the investments made by rent-seekers in the community. In addition, these licenses secure donators such that nobody can turn donations into private profit before they have contributed to the production of the public good. The objective of these licenses is not to prevent rent-seekers from making money out of donatives' resources, but to buy this right by contributing to the development of the public good.

This scenario, according to our perspective, is oversimplified. It assumes that those who are motivated by technological reasons do not display any preferences in terms of work environment (Feller & Fitzgerald 2002; Bonaccorsi & Rossi 2003; Rossi & Bonaccorsi 2005; Dahlander & Magnusson 2005). However, this is not the case. Their motivation to participate is tied to the opportunity to learn and experience leading-edge technologies. Therefore, it unlikely that these people are going to be attracted by communities that are more oriented toward efficiency, standardization, and diffusion. Furthermore, these people value their freedom to create more (Florida 2002). Therefore, they prefer to work in contexts that are open and where they do not have to ask for permission. Finally, they tend to be ego-centric. Therefore, they prefer a workplace where their cleverness can be displayed and appreciated. Thus, their profile is not equal to that of rent-seekers, who are more concerned with their profit opportunities, and that of donators, who are more concerned with the public nature of the source code. However, those who are driven by technological reasons are often critical for overcoming the initial phases in the development of a collective good.

Nonetheless, this perspective does not consider that firms can also be motivated by different reasons to participate depending on the firm's specific business model. From this, Dahlander and Magnusson (2005) distinguish among three approaches: symbiotic, commensalistic, and parasitic. Parasitic firms

exploit collective resources while avoiding, as far as possible, conflict with the community. Commensalistic firms also exploit collective resources. However, these firms try to gain acceptance from the community for using its resources in commercial applications. Symbiotic firms build up a win-win relationship with the community. These firms not only respect the rule of the communities but also are involved in their development. Therefore, these firms share their code and infrastructures with the community. However, these firms' behavioral attitude toward the communities will depend on the firms' business model. From this perspective, West and Gallagher (2006) distinguish between four OS business models: pooled R&D/product development, spinouts, selling complements, and donating complements. Only when the business model is based on the selling of complements might there be tension between firms and communities. Furthermore, the degree of this tension depends on the kind of complements the firms sell. If the business model is service-oriented, then this tension will be lower than when the source code is integrated into a commercial software application.

Managing the Trade-Off between Incentives

In the previous section, we argued that the main purpose of OS governance is to attract and retain an adequate combination of people, firms, and resources. Furthermore, we also suggest that people respond differently to different types of incentives and work environments. Therefore, someone might be more concerned about the risk that his or her contribution is exploited for economic purposes. Others are more concerned with the type of technology they are using and their freedom to create. Finally, others participate because the company they are working for asks them to do it. The main problem is that the success of an OS community depends on the contributions of all these people and the appropriate combination of these people changes with time. Therefore, the setup of a governance structure should take into account all these needs at the same time and how those evolve with time.

Managing the trade-off between incentives: static analysis

Bonaccorsi and Rossi (2003), from this perspective, have highlighted participants' heterogeneity as strategic for the success of an OS community. This enables a more efficient allocation of tasks and the initial phases of the supply of a collective good to be overcome. In OS communities, due to the lack of hierarchal power, tasks are not centrally allocated but self-selected by each participant according to his or her interests and competencies. Therefore, a wider range of interests and competencies secures that a larger number of tasks will be selected and accomplished. This is particularly relevant in the case of what Bonaccorsi and Rossi define as "non-sexy" activities. Those activities are not attractive to skilled developers. However, these activities, whose development does not require advanced skill, are critical for the functionality, usability, and diffusion of a technology. Therefore, they may be attractive either to skilled users or developers employed by firms to contribute to OS development. Therefore, OS governance should secure an appropriate degree of heterogeneity to optimize the efficiency of the development process.

Managing the trade-off between incentives: a life cycle perspective

Second, the quality of this trade-off should change over time. From this second perspective, Bonaccorsi and Rossi highlight how the evolution of an OS project can be characterized in three phases: invention, innovation, and diffusion. In the invention phase, the main problem to overcome is that of bringing together a small group of strongly motivated and resourceful developers capable of producing a collective good without the cooperation of others. In this phase, therefore, governance should provide participants with the opportunity to learn and share knowledge in the leading-edge context of technological development. In the innovation phase, the work of this initial group has to be complemented by the work of a second and larger group that focuses more on enhancing technological efficiency and usability. In this phase, therefore, governance should extend its portfolio of incentives in order to attract leading users and firms. In this phase, the bifurcation of the release process in stable and unstable, and the two-tier structures, highlighted by Lee and Cole (2003), enables the community to stimulate the cooperation between the two groups, keeping, at the same time, separated the sets of incentives in place. Finally, in the diffusion phase, a great role is played by firms that can leverage their distribution networks to enhance distribution. Therefore, governance, in this third phase, should

be more oriented toward supplying incentives to firms. This may lead part of the participants to the first group to leave the community because of their commitment to free information accessibility. However, this may happen in any case because the project is no longer technologically challenging. An evolutionary perspective of OS governance is compatible with more recent contributions explaining how OS governance emerged in OS communities (O'Mahony & Ferraro 2007; de Laat 2007).

Dimensions of OSS Governance

So far, we have argued that OS governance is ever more critical in shaping the capacity of a community to attract the appropriate combination of resources required over time, securing development efficiency, effectiveness, and innovation. However, we have not yet addressed the portfolio of mechanisms available and how those impact the attractiveness of those communities with respect to the different targeted resources. We will develop a comparative analysis of a panel of OS communities in order to understand if there is a correlation between the governance mechanisms adopted and whether those mechanisms are combined together according to a specific configuration. Therefore, our focus is not on the factors affecting the decision to participate in an OS project but on which OS projects participate. Only a few contributions have addressed this specific question. However, most have focused on the mechanisms either open source communities have used to safeguard their community-model or sponsors have implemented to open up their business model and retain some degree of control of the development process. In so doing, however, these authors take a specific stance, either that of the community or that of the sponsor. Our objective is different. We do not want to take ex-ante a specific stance, but to understand, ex-post, whether there are differences in the way those mechanisms are configured. Therefore, in this section we limit ourselves to developing a list of those mechanisms and ranking them according to their degree of openness.

In the literature, two models can be applied to develop such a comparative analysis. The first is Markus's (2007) model. This is the result of an extensive literature review. The model defines the main dimensions of OS governance discussed in the literature. The second is West and O'Mahony's (2008) model. This is the result of a long series of works empirically exploring the differences between sponsored and community managed projects. The two models, in our research, are not so different. Markus considers two additional dimensions: communication media and conflict resolution. However, we decided not to consider these two dimensions. The first because the communication media used in those communities are similar. This argument would not apply to communication processes and conventions. However, mapping those differences in a comparative model is difficult and incompatible with the attempt to consider a large sample of projects. The second because communities use many methods to settle conflict. Therefore, ranking those processes in terms of openness and transparency is not always possible. Part of this aspect is contained in the way decisions are made in the community. However, we do not apply West and O'Mahony's model. In their model, in fact, governance dimensions are already bundled together as a way to characterize the distinction between community versus sponsored managed OS projects.

The most important instrument in the governance structure of open source projects is the licensing agreement (e.g., Demil & Lecocq 2006; Franck & Jungwirth 2003). Many different licenses are defined OS. However, to be recognized as such, those agreements should satisfy the ten requirements of the OSI (Open Source Initiative) definition. For the matter of governance, licensing agreements are often categorized in two broad categories: viral/free and permissive/open (e.g., Bonaccorsi & Rossi 2003). Viral code is defined as such because it "infects" the source code with which it comes in contact. This means that if a piece of private code is integrated with a piece of free code, the first has to be made freely available. Free licenses are designed to achieve two major objectives. The first is to prevent somebody from conditioning, ex-post, development by claiming property rights on the part of the source code. The second is to extend the knowledge-sharing base. Free licenses, however, are largely incompatible with the participation of firms, which may want to maintain some degree of control of their internal development and source code. Therefore, open source licenses, which do not oblige users to share their own source code, propose a more pragmatic approach. However, the value of collective contributions and individual reputations is often safeguarded by forcing users to quote

original developers and preventing, if not agreed otherwise, the use of any trademark or brand associated with the original community for marketing purposes (O'Mahony 2003). Therefore, these licenses enable firms to make commercial use of open source code in an exchange of credit and reputation. Furthermore, the licenses provide users with incentives to negotiate the users' participation in the community in terms of human, material, and financial resources to get access to the communities' brand names and reputation. A third licensing strategy is often used by companies to stimulate external collaboration. This is named dual licensing and consists of making a distinction between commercial and non-commercial users. Commercial users are required to pay a fee to get access to the source code. Conversely, the source code is freely available for non-commercial purposes. The variable licenses as treated as follows. We distinguish among four license categories: recursive/viral, partially recursive, permissive, or pragmatic, dual. Viral licenses have been defined as the most open and the dual licensing scheme as the most closed. A second governance arrangement that has received growing attention in the literature is the institutionalization of non-profit foundations (O'Mahony 2003). These foundations were originally institutionalized to strengthen the capacity of the OS community to secure future accessibility to the source code. Therefore, they have been entrusted mainly with a legal mission. Few, however, have evolved to the next stage, where they are deeply involved in the governance and management of the communities. For instance, part of the mission of the GNOME[3] foundation is to set up common standards and guidelines for the development process and for selecting contributions, to manage the release process, and to negotiate commercial agreements with external parties. In these few cases, the institutionalization of a non-profit foundation responds not only to the need to secure property rights but also to institutionalize the original business model. However, non-profit foundations, such Mozilla and Qt, have been prompted by sponsoring firms. But they were moved by a different purpose, that is, to strengthen their commitment to supply open accessibility to the source code in the future. Therefore, source code ownership is often transferred to those foundations. However, this does not mean that neither management of the project nor control of the source code is shared with the community. This depends on how decision-making rights are distributed in the community (see below). Therefore, we distinguish among three states. The highest rank of openness has been attributed to a community governed by a foundation. The rationale is that those foundations reflect an attempt to make openness and accessibility institutional. The lowest rank, conversely, has been attributed to communities that are still governed by an informal leader.

Decision-making rights have been distributed in the community in different ways. Open source licenses grant owners with the exclusive right to decide on the authenticity of the source code. Therefore, legally speaking, they are the only ones who can decide which functionalities and lines of code should be included in each software release. However, to make sure that the community is going to collaborate in the future, owners are required to negotiate and share the content of this decision (West & O'Mahony 2008). Owners implement different strategies to structure and coordinate this negotiation, such as dealing directly with the community or institutionalizing roles that are entrusted with the power to decide. If the latter is the case, the question is to what extent those roles are collective, and who can be appointed to perform those roles and how. In our model, we focused on two roles: project leadership and release authority. The highest rank has been assigned to collective authorities democratically elected. The rationale is that the more those authorities are open and accessible, the more people trust the authorities and are willing to participate and collaborate. Related to decision making, there is the issue of membership. Members, in fact, are often entrusted with the power to directly make changes in the source code in the community repository. However, membership is often a precondition for being elected to an institutional role in the community. Therefore, the rules governing membership are critical for understanding to what extent leadership is accessible. The highest rank of openness has been attributed to communities where the rules for becoming a member are formalized and membership is individual. Formalization has been evaluated positively because it enables potential participants to evaluate and verify the respect of those rules.

---

3 (Abbreviation of **G**NU **N**etwork **O**bject **M**odel **E**nvironment) is a desktop environment—a graphical user interface that runs on top of a computer operating system—composed entirely of free and open source software.

The decision to allow firms to become members has been penalized in terms of openness because these companies could be granted with more power to control the development of the project. Furthermore, the companies may also discourage individuals' participation. Differently, we did not attribute any value to the content of those rules. There are mainly two reasons. First, most of these rules are merit oriented. Thus, membership is granted to those who provide major contributions in terms of quantity, quality, and continuity to the development of the community. Second, defining a common scale to evaluate the ranking of those rules is difficult. A third aspect that has been used to characterize decision making is membership.

The last dimension we consider is participation in the development process (Lee & Cole 2003). The extent users are involved in the development is relevant to shaping how users' motivation evolves (Shah 2006b). The more the production process is open to external contribution, the more participation is long-term and driven by technological and social reasons. Conversely, this is not the case in gated communities, where development control is much tighter. We characterize participation in the development in terms of source code availability, changes committed, and sub-project creation (West & O'Mahony 2008). Making source code available is a must in open source communities. However, this can be made more or less accessible by regulating the way people get access to the source code. For instance, accessibility may be constrained to the receipt of a formal request. Second, the process of committing changes in the source code repository may be centralized or distributed. In this second case, users are more stimulated to participate because they have more influence on development. However, commercial firms may prefer to maintain tight control over this process to prevent damage, and enhance compatibility and efficiency. Finally, users may be authorized to start up their own independent projects to address unmet needs. However, this may turn out to be a loss of opportunities for the source code owner. A fourth dimension is also relevant to shape participation in the development process. This is the degree of modularity. Baldwin and Clark (2003) show that a modular architecture enables work to be equitably distributed among developers and makes joining and staying in the collective development effort a strictly dominant strategy compared to coding in isolation. However, the main reasons we decided not to include this dimension is that technical modularity is only a condition sine qua non. It is not sufficient that the architecture of a code base is modular to stimulate participation. The political will is also required, and this is reflected in the other three dimensions included. In Table 1, we summarize the description of the eight OS governance variables we used to map governance dimensions.

Methodology

The objective of this contribution is to identify the main forms of governance in OS communities. To achieve these objectives, we conducted an exploratory analysis of 40 open source projects. The list of the projects to analyze was extracted randomly from a dataset of projects listed on Freshmeat.net. As Feller and Fitzgerald (2002) point out, Freshmeat is a gathering place for the "grassroots" OSS community; thus, we expected that most of the projects we observed there would be of a more "traditional" nature rather than those that are sponsored by large companies (e.g., Netscape) or that are already established and successful (e.g., Linux). We referred to this database because of the availability of additional data on project productivity, creativity, and perceived quality such as the vitality score. Vitality is calculated using the number of announcements about a project and the time since its last release. All the analyses in this paper were developed with R 2.9.0 software.[4] FLOSSmole (Free/Libre/Open Source Software mole) does not collect data on project governance. Therefore, these data were collected by visiting projects' websites. The sample also takes into account project origin. West and O'Mahony argue that the distinction between sponsored and autonomous communities reflects a different project leadership purpose. Therefore, we expect that project origin, defined as community founded versus sponsor founded, has some impact on governance.

4 R is an open source software environment for statistical computing and graphics.

Table 1: Description of the Variables

Variable		*Score*	*Description*
Foundation	X1	0	No foundation
		1	The foundation provides legal protection against copyright abuses or is not yet fully established.
		2	The foundation is also involved in standardization issues and strategic management.
Type of license	X2	0	Dual
		1	Permissive
		2	Partially recursive (e.g., LGPL)
		3	Recursive
Membership	X3	0	Membership is not clearly defined and regulated.
		1	Membership is planned for enterprise.
		2	Membership is defined and open only to individuals.
Changes to source code	X4	0	Proposed changes should be submitted to project leaders, who are the only ones with the authority to decide.
		1	The proposed changes are evaluated and discussed in public forums and/or under the supervision of a project leader.
		2	Committed members are authorized to commit changes directly in the source code repository.
Sub-project	X5	0	Sub-projects may not be set up.
		1	Sub-projects may be set up, but project leaders retain control over copyright.
		2	Sub-projects may be set up, but they are not included in the main distribution.
		3	Sub-projects may be set up, which are downloadable as versions or accessories.
Release authority	X6	0	There is no formal release authority and/or the process has not been formalized.
		1	There is no release authority, but the process is open to members who have attained a certain status or indirectly through suggestion systems.
		2	There is a release authority, and the process through which members can contribute to it is formalized and accessible.
Leadership and decision making	X7	0	There is no formal board.
		1	There is a board, but members are appointed by the project owner.
		2	The decision-making power is delegated to a board of directors whose members are not elected but chosen by the sponsor or by the founders.
		3	There is a board, and members are elected by the community.
Access to the code and bug reporting	X8	0	Access to source code, although guaranteed, is limited to subscribers or to the previous release, or made arbitrarily complex. To protect the sponsor's benefits, obtaining access requires sending a formal request to get access to the repository.
		1	Access to the source code is granted and made simple in order to attract new participants in the development process.

Therefore, our sample is formed by 28 community-founded projects and 12 sponsor-founded projects. Finally, the sample is made up of projects with a vitality score greater than 0 and a number of registered developers greater than 30. The vitality score is an index that measures the vitality of a project. It is 0 when not a single line of code has been issued since the project has been registered on Freshmeat.net, and its value increases as the number of releases increases. Therefore, choosing only projects with a vitality score greater than 0 serves to make sure that a version of the software has been released. There are two reasons for taking into account only projects with a number of registered developers greater than 30. The first is to make sure that the project has been capable of attracting a sufficient number of developers. The second is that the number of contributors is large enough to require making governance explicit.

Data Analysis

In Table 2, the main descriptive statistics are summarized. The first step of our analysis was to verify correlations between OS governance dimensions. Correlation analysis confirms that OS governance dimensions are strongly correlated (see Table 3). Therefore, the results are compatible with the hypothesis that OS governance is reducible to a single continuum (latent factor) between autonomous and sponsored. To further explore this hypothesis, we conducted an exploratory factor analysis[5] with a

[5] Reducing data to a smaller set of summary variables or latent variables.

single factor (Table 4). The resulting model is structurally sound, as confirmed by the values of the chi-square statistic (a chi-square value significantly greater than 1 and a p-value close to zero). However, the factor identified does not explain all the variables in the same way. The factorial weight of the variables (loadings), as also confirmed by the uniqueness values, shows that the model explains in particular the variance of x1, x3, x4, x6, and x7. Actually, calculating Cronbach's alpha on only these variables, we obtain a value equal to 0.868, significantly higher than the value calculated with all the variables. The difficulties encountered in reducing the set of variables to a single factor led us to explore the possibility of referring to two latent factors (Table 5). The results show that the two factors better describe the variable set. In fact, the factors explain the variables x1, x3, x4, x6, and x7 and the variables x2 and x5, respectively, while only x8 is not explained by the model. The variable x2 (with residual variance close to zero) is the best explained, while, in contrast, x4 and x8 are the worst explained. Though the two latent factors models' chi-square statistical results are less than the factor model's one, the value and respective p-value value appear statistically acceptable.

In addition, the uniqueness of one latent factor model is over 90% (Table 4) for three variables x2, x5, x8; in the second model (Table 5) the highest residual variance variable x8 does not exceed 90%. Therefore, we could conclude that the two-latent-factor model makes the whole variable set clearer. If we consider that each factor represents only the variables with a factorial weight above 0.4 and we calculate the Cronbach alpha on these variables to verify if indeed the two factors significantly explain such variables, we obtain the following: Factor 1 interpolates significantly the variables "Foundation," "Membership," "Authority," and "Leadership," resulting in a Cronbach alpha equal to 0.869; Factor 2 interpolates significantly the variables "License" and "Subproject," resulting in a Cronbach alpha equal to 0.670. Finally, we conducted a factor analysis with three factors to verify whether we could reduce our model to West and O'Mahony's (2008). However, this hypothesis, with a p-value of 0.95 and a chi square of 2.17, is rejected.

Discussion

Our objective in this paper was to verify whether there is a correlation between OS dimensions and, if there is, which were the main configurations available. Our analysis confirms that those dimensions are correlated. However, differently from what is commonly suggested, there are four, rather than two, configurations of governance. Therefore, OS governance is not reducible to a single continuum between autonomous and sponsored. The factor analysis, in fact, has identified the existence of two latent factors along which the configurations of governance are defined. The first describes the leadership and decision-making structure, the second the intellectual property rights regime.

Definition of the Two Dimensions

The first dimension has been defined as the leadership and decision-making structure. The variables described by this latent factor are foundation, membership, code change, release authority, and leadership. Foundations have been institutionalized for two reasons. The first is to stimulate firms' participation. Firms are moved by the opportunity to make profit out of their participation in an OS community. Therefore, their participation is often constrained to rules that make room for some degree of proprietary appropriation. However, risks are also associated with stimulating companies' participation.

Table 2: Descriptive Statistics

	Variables	Sample Size	Mean	Median	Mode	S.D.	Min	Max
1	Foundation	40	0.55	0	0	0.84	0	2
2	Type of license	40	1.62	1	3	1.17	0	3
3	Membership	40	0.45	0	0	0.71	0	2
4	Code changes	40	1.05	1	1	0.71	0	2
5	Subproject	40	1.42	2	2	0.90	0	2
6	Authority	40	0.55	0	0	0.78	0	2
7	Leadership	40	1.12	1	1	1.16	0	3
8	Code access	40	0.90	1	1	0.30	0	1
9	Vitality score	40	39099.17	5157.22	-	81209	2	295210
10	Vitality score 2	40	1475919	20516.39	-	7270000	2.46	44300000
11	Major release	40	10.94	9	1	11.59	1	60

Table 3: Descriptive Statistics and Bivariate Correlations Matrix

	Variables	1	2	3	4	5	6	7	8	9	10	11
1	Foundation	1.00										
2	Type of license	0.03	1.00									
3	Membership	0.68	0.11	1.00								
4	Code changes	0.29	0.36	0.51	1.00							
5	Subproject	0.19	0.49	0.21	0.28	1.00						
6	Authority	0.65	0.29	0.69	0.41	0.24	1.00					
7	Leadership	0.63	0.15	0.70	0.61	0.19	0.66	1.00				
8	Code access	0.22	0.25	0.22	0.26	0.16	0.13	0.18	1.00			
9	Vitality score	-0.14	0.18	-0.20	0.15	0.20	-0.12	-0.10	0.14	1.00		
10	Vitality score 2	0.27	0.20	0.11	-0.01	0.12	0.10	-0.03	0.06	0 57	1.00	
11	Major release	0.15	0.30	0.10	0.10	0.23	0.07	-0.07	0.23	0.47	0.73	1.00

Table 4 (left) and Table 5 (right): Factor Analysis Results

```
R console

Call:
factanal(x = open1, factors = 1, scores = "regression")

Uniquenesses:
   x1    x2    x3    x4    x5    x6    x7    x8
0.418 0.949 0.265 0.649 0.920 0.355 0.301 0.938

Loadings:
   Factor1
x1 0.763
x2 0.227
x3 0.857
x4 0.592
x5 0.283
x6 0.803
x7 0.836
x8 0.249

                Factor1
SS loadings       3.205
Proportion Var    0.401

Test of the hypothesis that 1 factor is sufficient.
The chi square statistic is 42.76 on 20 degrees of freedom.
The p-value is 0.0115
```

```
R console

Call:
factanal(x = open1, factors = 2, scores = "regression")

Uniquenesses:
   x1    x2    x3    x4    x5    x6    x7    x8
0.382 0.005 0.247 0.602 0.725 0.341 0.306 0.899

Loadings:
   Factor1 Factor2
x1  0.784
x2          0.997
x3  0.857   0.137
x4  0.508   0.374
x5  0.158   0.500
x6  0.752   0.306
x7  0.816   0.170
x8  0.187   0.258

                Factor1 Factor2
SS loadings       2.898   1.595
Proportion Var    0.362   0.199
Cumulative Var    0.362   0.562

Test of the hypothesis that 2 factors are sufficient.
The chi square statistic is 22.55 on 13 degrees of freedom.
The p-value is 0.0649
```

For instance, firms may leverage their market power to exploit the value created by the community or to take control of project management. Therefore, the second reason for institutionalizing a foundation is to safeguard public accessibility and control of the project management, project development, and source code. However, institutionalizing a foundation is not sufficient to ensure public accessibility and control. The rules governing these foundations, such as membership and the right to perform changes directly in the source code repository, are also relevant.

The second dimension has been defined as the intellectual property right regime. It refers to two variables. The first has to do with the type of license adopted to distribute the software. The second has to do with the possibility of setting up sub-projects, and with the ownership of these sub-projects.

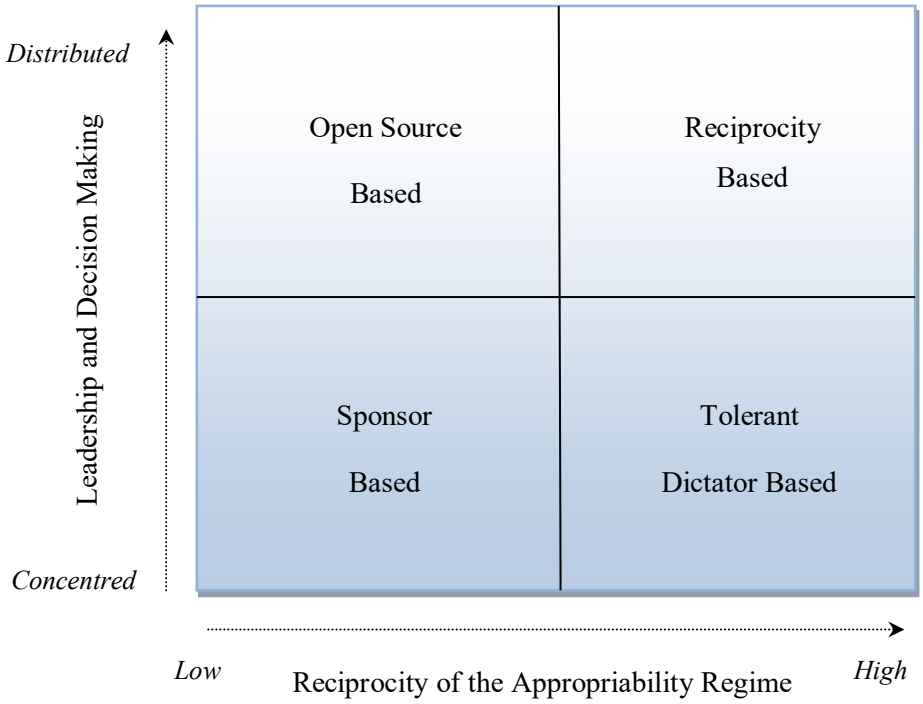
The Matrix of OS Governance Form

The combination of these two dimensions defines four models of OS governance (see Figure 1). *Sponsor Based*. These are communities sponsored and controlled by a company or a pool of companies. In these communities, founders/sponsors retain a large part of the control power per se with little community involvement. The intellectual property right regime is pragmatic. Therefore, it safeguards the sponsor's capacity to appropriate value by either selling accessibility to the source code for commercial purposes or to retain control of parallel or complementary project development. Furthermore, the opportunity to set up sub-projects is either bound or directly controlled by claiming property rights on these sub-projects.

An example of a sponsored community is the Sun Java platform SE (see Figure 2). This case has been studied by Yamauchi et al. (2000). The Java platform is an Internet-based programming environment originally developed by Sun. Java represented the opportunity for Sun to reposition itself as a leading player within the wider post-PC, Internet-based information technology (IT) field. However, to achieve this object, Sun had to cope with Microsoft, which had already announced its intention to release a comparable platform and could exploit its dominant position to impose its platform as the standard, and the risk that others could develop their own comparable technology. Sun chose an open system strategy to try to impose Java as the technological standard in the emerging field of Internet-based technologies. This strategy consists of making it easier for rivals and vendors of complementary products to get access to the sponsor's proprietary technology. For instance, the license scheme implemented by Sun required paying an up-front fee and royalties on unit sales of Java-based products but allowed licensees to modify the technology as long as they shared these modifications freely with Sun and other licensees. This strategy turned out to be successful as long as the technology did not achieve its stage of maturity. Most of Sun's partners, in fact, started to feel threatened by Java's control of the technology and Sun's introduction of Java products that competed with those offered by other members of the Java collective. Sun's decision to progressively open the source part of Java

technology reflects an attempt to manage the trade-off between the risk of losing the support of a large part of the community and securing control of the core technology to prevent fragmentation, forking, and indeed Sun's centrality in the Internet-based IT field.

Figure 1: OS Governance Matrix Model



Therefore, Sun's sponsor-based open source strategy reflects the trade-off between the need to build up credibility as an institutional third party and the need to defend its position as a commercial party. To achieve this objective, Sun leveraged the appropriability regime, by allowing, for instance, the Java collective to establish working groups to extend Java into new areas and determine when these new specifications would be released to the public, to enhance credibility, and in community management to control the direction of development.

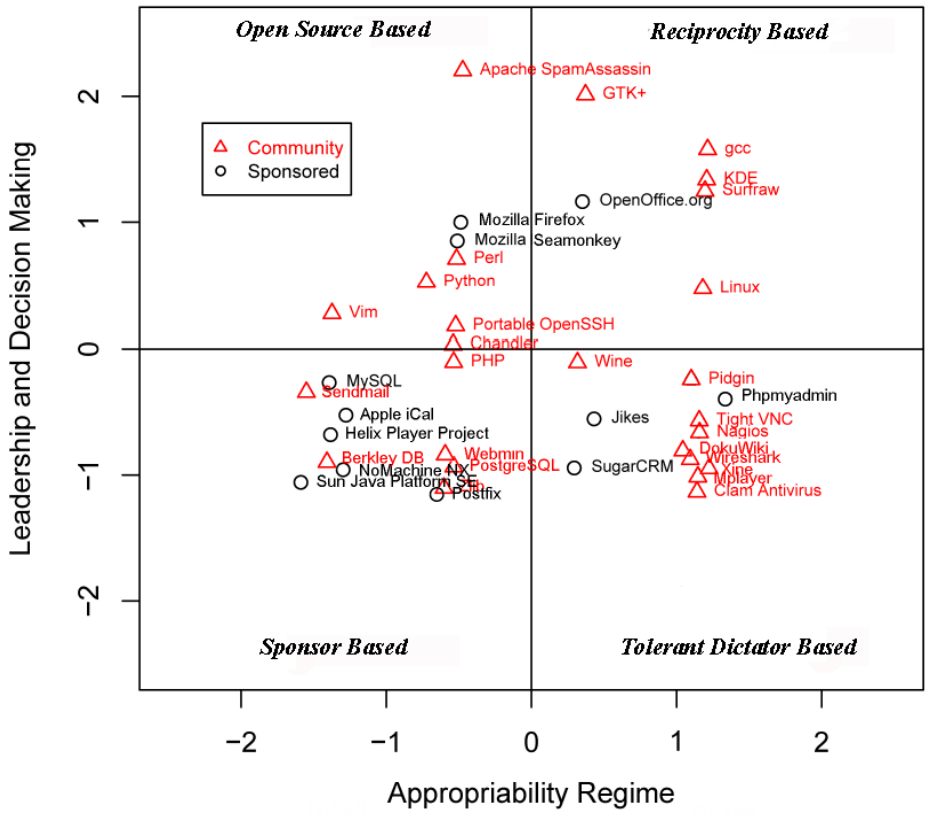
Open Source Based. These communities have developed a pragmatic attitude toward the contribution given by enterprises to OS development. Actually, these communities recognize the potential for diffusion and activation of an extended feedback network. At the same time, they are aware of the possible risks that may result from opening too much to enterprises. The communities handle this trade-off by ensuring companies have the opportunity to pursue autonomous development strategies and by ensuring public access to source code and decision making. An example of an open-source-based community is Mozilla. This case has been studied by Mockus et al. (2002). Mozilla was started in 1998 as a sponsor-based community by Netscape that, in the face of stiff competition, announced that the Communicator product and source code would be available free of charge. The group mozilla.org was meant to act as a benevolent dictator for the open source effort. Members of the group were mainly employees of Netscape Inc. entrusted with the mission to develop Mozilla for testing purposes and not for use by end users. This led to a fork meant to provide the documentation and support necessary for an end-user project. In 2003, when AOL (Netscape's parent) started to disinvest from Mozilla, a non-profit foundation was founded. AOL donated the property rights, the technological development infrastructure, and a grant of US$2 million to the foundation to support its startup. Now, the members of the board are not elected but appointed. The source code is licensed out in MPL (Mozilla Public License). That is an open/free license. The project is coordinated by two wholly owned subsidiaries. Development coordination and control are delegated to a number of roles, such as module owners, peers, and super reviewers. Persons are appointed to those roles. However, the experience and competencies required for being appointed to those roles are codified. Furthermore, those roles' decision-making rights and responsibilities are also identified. Finally, users are also allowed to set up their own sub-project. Leadership and decision making in the Mozilla community

are not as open as in the Apache one. The board of directors and the main organizational roles are not elected but appointed. However, those roles have been defined in terms of the decision-making experience required to be appointed to the role.

Reciprocity Based. These communities are linked to the traditional model of free development. They have a radical attitude toward IPR, which discourages firms' participation. They have a radical attitude also toward the institutionalization of foundations, membership, and decision-making, which are viewed as constraints to self-organization and emerging cooperation. Decision-making power is de facto shared between contributors who are commonly recognized as leaders within the community. An example of a reciprocity-based community is the GNU GCC. This project has its origin in the C compiler Stallman created to make free Unix. Now, the project maintains various compilers and libraries. GCC, according to Yamauchi et al. (2000), has experienced two discontinuities. The first was architectural. The second was tied to the governance of the community. The main enabler for this second change was the impossibility of incorporating the work of many programmers in the main trunk of the code because of the orientation of the project to stability. Therefore, according to our perspective, there is a risk of loss of attractiveness in terms of the quality and quantity of people attracted. Therefore, according to what Lee and Cole also suggested, an experimental development project was established (EGCS – Experimental GNU Compiler System) and appointed as the official maintainer of the GNU compiler, which was also renamed GCC. The decision-making structure was also revised in order to enlarge participation. A steering committee of 13 members, each representing a certain community of users (e.g., Fortran), has been set up. In addition, 134 programmers and 35 testers joined the development team.

Tolerant Dictator Based. This model differs from the previous ones only because of the concentration of decision-making power in the hands of a single tolerant dictator, as Linus Torvald used to be in the early stages of Linux development. The power of the dictator is de facto counterbalanced by his or her weak control of the source code. Therefore, dictators are always required to renegotiate their technical leadership within the community in order to canalize resources efforts and prevent versioning and forking.

Figure 2: OS Governance Matrix



Do those dimensions have an impact on community performances? An exploratory test was conducted on the vitality score. One of the reviewers suggested that typologies and classification are important only as stepping stones to develop meaningful theoretical statements on the relationship between the dimension identified and some variable of interest. Even if we recognize completely the merit of this argument, answering this question would basically require writing another paper. That is why we originally addressed this issue as part of future research directions. However, in an attempt to give even a partial response to this suggestion, we tried, with the available data, to test whether there were differences between the configurations identified and the variable of interest. The variable that we chose is the vitality score. This is the score Freshmeat uses to measure projects' vitality. The vitality of a project is calculated as:

$$VitalitySc\,ore = \sqrt{\frac{releases * age}{last\ release}}$$

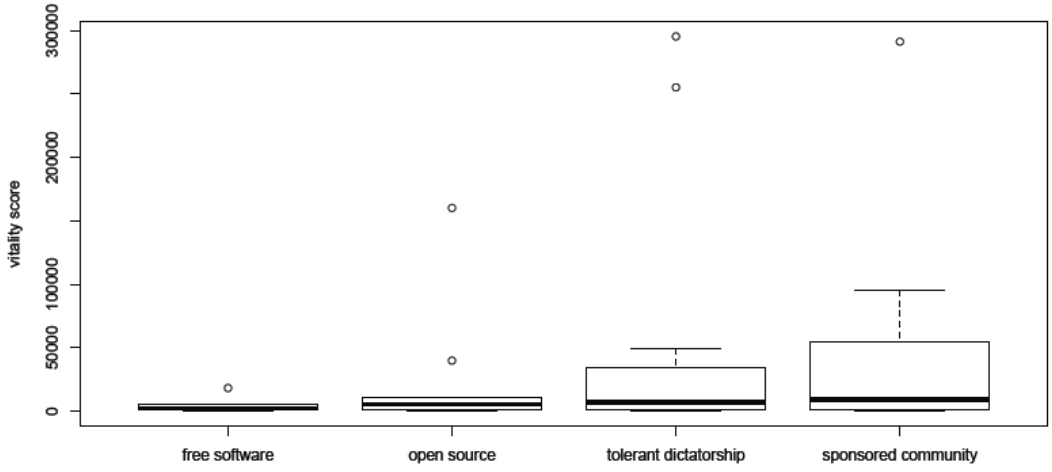
This way, projects with lots of announcements that have been around for a long time and have recently come out with a new release earn a high vitality score, and old projects that have been announced only once get a low vitality score. To test whether a configuration scores better in terms of vitality, we used one-way ANOVA. This is a general linear model (GLM) used to test for differences among two or more independent groups (in our case, configuration groups). This analysis shows that the four groups score significantly differently in terms of vitality (see Table 6). The ones that scored best are sponsored and tolerant dictatorship communities. Therefore, this result seems to suggest that communities with a more centralized model of leadership and decision making are more vital than the ones with a more distributed model of leadership.

Table 6: ANOVA of Groups with Respect to the Vitality Score

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Factor(x)	3	0.20373	0.067910	3.3324	0.02384 *
Residuals	36	1.54876	0.020378		

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Figure 3: Boxplot with Respect to the Vitality Score



We found this result counterintuitive. The reason is that we expected open leadership and decision making to be positively associated with a better capacity to attract skilled and strongly self-motivated developers and users. Those latter aspects are usually positively associated with higher creativity and indeed innovativeness of the process of development. This data, conversely, seems to suggest that a more centralized model of leadership and decision making is positively associated with a more

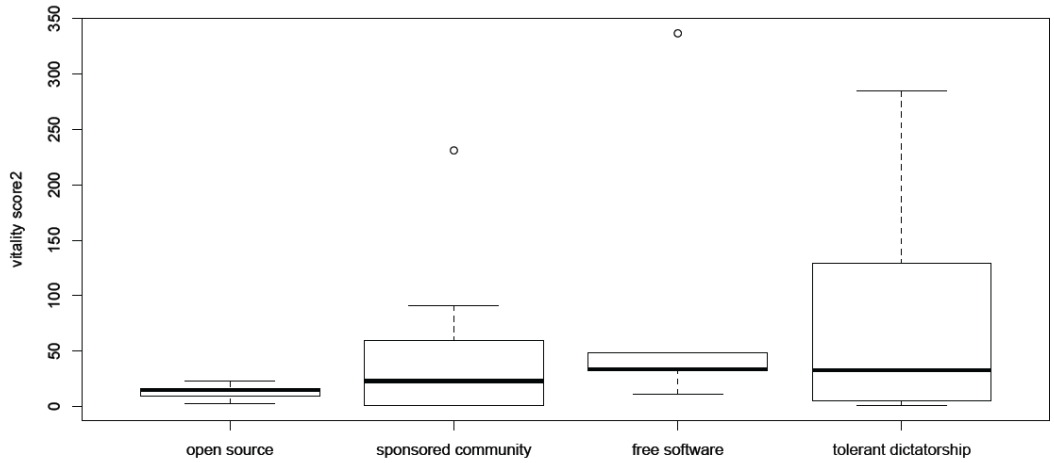
coordinated process of development and indeed a faster process of development. The main problem is that the vitality score is not a measure of communities' creativity. The vitality score, in fact, does not take into account the innovative content of a release. Therefore, in order to consider this aspect, we recalculated the vitality score of these projects for major releases (see Table 7). These data were collected from SourceForge.net. The basic idea behind this attempt is that the label major should be a better proxy of the development discontinuity contained in those releases. The ANOVA conducted on this index shows that reciprocity-based and tolerant dictatorship communities score significantly better than sponsored and open source communities. Therefore, in the case of a major release, the appropriability regime seems to be the discriminating variable. Therefore, if confirmed, this result seems to confirm Shah's findings that talented and self-motivated people prefer to work on projects where the source code is freely available.

Table 7: ANOVA of Groups with Respect to the Vitality Score 2

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Factor(x)	3	0.21352	0.113512	35.221	0.02175*
Residuals	33	134.222	0.018273		

*Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1*

Figure 4: Boxplot with Respect to the Vitality Score 2



This is only a preliminary analysis. To develop a credible thesis about the relationship existing among governance, creativity, and innovation, a deeper understanding of these variables and their connections is required. The state of maturity of a project, for instance, will have great influence on the frequency of releases and the innovative content of those releases. However, these results provide some initial ground for claiming that those configurations might be relevant for understanding the existence of differences in productivity, creativity, and innovation in OS communities.

Conclusion

Our main contribution in this paper has been to show that the configuration of OS governance can be reduced to two main factors. The first is the structure of the leadership and of the decision-making process. The second is the appropriability regime. The intersection of these two dimensions defines four configurations of OS governance: open source, sponsor, reciprocity, and tolerant dictatorship based. Open source and sponsored-based differ mainly in their degree of accessibility to leadership and decision making. In an open source community, leadership and decision making are a highly distributed process. In a sponsored community, on the contrary, sponsors retain control. Both communities are characterized by a permissive regime of appropriability, even if a sponsored community attempts to maintain control over sub-project content and development. Reciprocity and

tolerant dictatorship communities, in contrast, are characterized by a radical regime of appropriation. Therefore, these communities are grounded on the principle of reciprocity in the access to and usage of the source code. Reciprocity and tolerant dictatorship differ in the distribution of power in the community. In a reciprocity-based community, community leadership and decision making are highly distributed, whereas in a tolerant dictatorship community, they are concentrated in the hands of the copyright holder.

The results of our work have some interesting implications. First, the work provides a set of strategic insights into the key dimensions that have to be considered to evaluate the returns of an OS investment compared to an IPR investment. Second, our work provides an empirically grounded model to support firms' decision makers either in configuring an OS model of governance or in evaluating the opportunity to enter a communitarian-based OS community. The same results can be used by community leaders to evaluate the quality of governance safeguards in place and the potential activated by entering a specific company.

Three major directions can be pursued to extend the implications of this work. The first is to extend the experiment to a larger sample. Therefore, the results may change when a larger number of cases are considered. Furthermore, we identified other dimensions that could be interesting to explore in order to identify other factors that influence OS governance. For instance, in this paper the origin of the project has been defined as either sponsored or communitarian. However, projects have been started either as university spin-offs or company spin-offs. Furthermore, the institutionalization of a foundation is not the only institutional arrangement that autonomous communities have used to safeguard openness. Communities have set up associations or firms that have community members elected to the board to represent the community or that are constrained to reinvest profits in community development. Therefore, there might be a correlation between these dimensions and the solution to implemented governance.

A second development direction is to study the relationship between OS governance configurations and creativity. In this paper, we made a very preliminary attempt to verify whether those configurations could be relevant in explaining differences in creativity between communities. The results are encouraging. However, much more work is required to arrive at a theory linking governance to creativity. Our hypothesis so far is that governance is expected to influence the quantity and quality of the people attracted, and the quality of the cooperation. Talent and entrepreneurs, according to our perspective, are attracted by the context of governance that entrusts users with the power of self-determination and the capacity to influence decision making. The quality of the cooperation depends on the distribution of incentives between individuals and firms over time. Therefore, we expect to find a strong correlation between an open governance configuration and creativity.

Finally, a third direction is to explore the relationship between governance and the community climate. We argue that governance influences collaboration sustainability and membership satisfaction. The community climate is expected to improve as a consequence of the decision-making involvement and equity between the efforts produced and the opportunities available. Therefore, a governance configuration that privileges the interests of firms at the expense of that of individuals should produce dissatisfaction and fewer individuals' participation.

References

Baldwin, C.Y. & Clark, K.B., 2003. The Architecture of Cooperation: How Code Architecture Mitigates Free Riding in the Open Source Development Model. *Harvard Business School, 43*.

Bonaccorsi, A. & Rossi, C., 2003. Why Open Source software can succeed. *Research Policy, 32(7)*, 1243-1258.

Crowston, K., 1997. A Coordination Theory Approach to Organizational Process Design. *Organization Science, 2, 157 - 175*.

Dahlander, L. & Magnusson, M.G., 2005. Relationships between open source software companies and communities: Observations from Nordic firms. *Research Policy, 34(4), 481-493*.

Demil, B. & Lecocq, X., 2006. Neither market nor hierarchy nor network: The emergence of bazaar governance. *Organization studies, 27(10)*, 1447.

Feller, J. & Fitzgerald, B., 2002. *Understanding open source software development*, Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.

Florida, R.L., 2002. *The Rise of the Creative Class: And How It's Transforming Work, Leisure, Community and Everyday Life*, New York: Basic Books.

Franck, E. & Jungwirth, C., 2003. Reconciling investors and donators-The governance structure of open source. *Journal of Management and Governance*, 7, 401-421.

von Hippel, E. & von Krogh, G., 2003. Open Source Software and the" Private-Collective" Innovation Model: Issues for Organization Science. *Organization Science, 14(2*), 208-223.

Kogut, B. & Metiu, A., 2001. Open-source software development and distributed innovation. *Oxford Review of Economic Policy, 17(2),* 248.

de Laat, P.B., 2007. Governance of open source software: state of the art. *Journal of Management and Governance, 11(2)*, 165–177.

Lakhani, K.R. & von Hippel, E., 2003. How open source software works:?free? user-to-user assistance. *Research Policy, 32(6)*, 923-943.

Lee & Cole, 2003. From a firm-based to a community-based model of knowledge creation: The case of the Linux kernel development. *Organization Science*, 633–649.

Lerner, J. & Tirole, J., 2000. The Simple Economics of Open Source. *NBER Working Paper*.

Markus, M.L., 2007. The governance of free/open source software projects: monolithic, multidimensional, or configurational? *Journal of Management and Governance, 11(2)*, 151-163.

Mockus, A., Fielding, R.T. & Herbsleb, J.D., 2002. Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology (TOSEM), 11(3)*, 346.

O'Mahony, S., 2003. Guarding the commons: how community managed software projects protect their work. *Research Policy, 32(7)*, 1179-1198.

O'Mahony, S., 2007. The governance of open source initiatives: what does it mean to be community managed? *Journal of Management and Governance, 11(2)*, 139-150.

O'Mahony, S. & West, J., 2005. What Makes a Project Open Source? Migrating from Organic to Synthetic Communities. *Academy of Management Annual Meeting*. Honolulu.

O'Mahony, S., 2003. Guarding the commons: how community managed software projects protect their work* 1. *Research Policy, 32(7)*, 1179–1198.

O'Mahony, S. & Ferraro, F., 2007. The emergence of governance in an open source community. *The Academy of Management Journal (AMJ), 50(5),* 1079-1106.

Osterloh, M. & Rota, S., 2007. Open source software development?Just another case of collective invention? *Research Policy, 36(2)*, 157-171.

Raymond, E.S., 1999. The Cathedral and the Bazar, O'Reilly & Associates, Sebastopol.

Rossi, C. & Bonaccorsi, A., 2005. Intrinsic vs. extrinsic incentives in profit–oriented firms supplying open source products and services. *First Monday, 10(5)*.

Scacchi, W., 2004. Free and open source development practices in the game community. *Software, IEEE, 21(1),* 59-66.

Shah, S.K., 2006a. Motivation, governance, and the viability of hybrid forms in open source software development. *Management Science, 52(7)*, 1000.

Shah, S.K., 2006b. Motivation, Governance, and the Viability of Hybrid Forms in Open Source Software Development. *Management Science, 52(7)*, 1000-1014.

West, J., 2003. How open is open enough? Melding proprietary and open source platform strategies. *Research Policy, 32(7)*, 1259-1285.

West, J. & Gallagher, S., 2006. *Patterns of Open Innovation in Open Source Software*. HW Chesbrough & W. Vanhaverbeke & J. West (Eds.), Open Innovation: Researching a New Paradigm, 82-106.

West, J. & O'Mahony, S., 2008. The Role of Participation Architecture in Growing Sponsored Open Source Communities. *Industry & Innovation, 15(2)*, 145-168.

Williamson, O.E., 1999, Strategy research: governance and competence perspectives. *Strategic Management Review, 20(12)*, 1087-1108.

Yamauchi, Y. et al., 2000. Collaboration with Lean Media: how open-source software succeeds. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*.