Toward the Comparison of Open Source Commons Institutions

Charlie Schweik
University of Massachusetts, Amherst

Paper prepared for presentation at the Workshop on Cultural Commons,
NYU School of Law, New York, New York.

(Note to reviewer – apologies! A VERY rough draft, version 1.0)

Introduction

In this day and age of networked computers and the Internet, any effort to understand commons and culture needs some attention to what is happening in the digital world, and in particular, collaboration over the Internet. In this paper, I report some of our findings from a five-year empirical study of open source software commons.

In his seminal work *The Wealth of Networks*, Benkler coined the phrase "commons-based peer-production" (2006, 63) to describe situations where no centralization exists, no hierarchical assignments occur, and individuals self-select what to work on. In the best of circumstances, large numbers of individuals, working over the Internet, search for the right project to contribute to (Benkler 2005, 178). One of the key incentives driving this search, according to Eric von Hippel (2005a, b), is user-driven need.

This idea of people searching the Internet for activities that interest them and that they might be able to contribute to extends to a variety of digital commons situations (consider the phenomenon of people contributing to Wikipedia entries, for example), it is not at all a stretch to argue that computer programmers were probably the first community to do this. After all, in the earliest days of computing right up to about 1980, it was standard practice in the academic and computer industry to share and collaborate on software code (Cambell-Kelly, 2003). This practice continued as Internet infrastructure was being built and up until around 1980 when an amendment to the Copyright Act of 1976 treated software code and their corresponding binary distributions as trade secrets (Schwarz and Takhteyev, 2009). This privatization of software, as

many in this conference know, led to the free/libre and open source software "movement," initiated by Richard Stallman and colleagues in the 1980s and the brilliant use of copyright law to create software licenses that promoting sharing and collaboration – an approach referred to as Copyleft (FSF, 2009). Benkler (2006: 63) has referred to the collaborative principles that have emerged through this history and have evolved on the Internet in free/libre and open source software as the "quintessential instance" of commons-based peer-production.

Experiments following open source-like commons-based peer production have now emerged in other settings. The obvious example is the open digital content effort Wikipedia. But this idea has expanded into other collaborative areas of writing, such as in the area of course curriculum (e.g., Rice Connexions, MIT Open Courseware) and in many areas where digital products can be shared and remixed (see CreativeCommons.org for many examples). Moreover, peer-production commons experiments have appeared in some surprising new areas. Consider, for example, the use of this production model in national intelligence gathering (Howard, 2010) or the borrowing of open source production and crowdsourcing to spur design innovation and efforts to produce "openly designed physical (not digital) products, sometimes in an effort to solve market failure problems.[1] My particular interest in understanding open source peer-production motivation is to see how these collaborative principles might be applied to encourage sharing and dialog between public sector academics and practitioners (see Schweik, et al., 2011) or in efforts to solve problems faced across the world in environmental management (see Schweik, Evans and Grove, 2005).

[1] For a few interesting examples, see LocalMotors (2011) for an example in the automotive industry in an effort to crowdsource innovative design ideas, or Jha and Nerurkar (2010) and Kuniholm (2011) for examples in health care where market failures exist.

Given that the open source collaborative paradigm was born in the context of computer programming, and that it is in this space that it has been around the longest, it makes sense to study open source software commons empirically. Historically, there has been a great deal of research on open source software from a wide range of disciplines. Prominent book-length related-studies include Weber (2004), von Hippel (2005a), Benkler (2006) and Fogel (2006). There are also a vast number of studies in variety of outlets (see for example, Feller et al. 2005, for an edited volume with a variety of empirical studies). Many more individual studies on particular elements of open source software commons can be found in a variety of journals.

What I report in this paper is some results of a study that we initiated in 2005 in an effort to gain insight into the socio-technical aspects of open source software collaboration. This study took over five years to complete, and involved a team of researchers at the University of Massachusetts, Amherst. The major product of the study is a book-length manuscript (Schweik and English, forthcoming) that summarizes our efforts to capture what was happening in a close approximation to the population of open source. Much of the research prior to this study focused on high-profile open source software cases, such as the Linux operating system. The goal of this study was to get a better understanding of what was happening in the broader population of open source projects, the vast majority of which are small, much less visible projects. The study has two primary goals: (1) to identify factors that appear to affect whether an open source software commons is successful in terms of maintaining ongoing collaboration – or becomes abandoned; and (2) to focus more attention on how open source projects are governed, and to explore methods for the more systematic study of the institutions

that govern open source commons.[2] In this paper, I will provide first a few of the findings from #1, and then focus attention on #2 with the goal of creating a dialog in the Cultural Commons Workshop on how we move toward systematic study and articulation of open commons cases.

A Summary of the Open Source Commons Success and Abandonment Study

A major incentive driving the initiation of our study was that, at the time we started in 2005, there was very little empirical work trying to capture and document the population of open source software commons, and also looking at these projects as socio-technical systems of production. The overarching research question is "What factors lead some open source software commons to achieve ongoing collaborative success while others become abandoned?"

Over the next five years we: (1) established an overarching theoretical framework to guide the research; (2) conducted literature reviews in a number of relevant disciplines – Information Systems, Virtual Teams and Distributed Work, Environmental Commons, and the emerging literature specifically on open source – in an effort to identify testable hypotheses or, in some cases, "sub-research questions" where no testable hypothesis could be identified from the existing literature. Multiple people on our team worked for a year and a half and identified a variety of variables aligned along the components of the Elinor Ostrom and Colleague's Institutional Analysis and Development (IAD) framework (Ostrom, 2005; Hess and Ostrom, 2007). Ultimately, more than 40 hypotheses and research questions were identified, each of which we later investigated empirically

[2] In this paper "open source commons" and "open source projects" are used interchangeably.

(Schweik and English, forthcoming). Figure 1provides a quick summary of some of the variables we researched, organized following the structure of the guiding theoretical Institutional Analysis framework (see Hess and Ostrom, 2007).[3]

*** Figure 1 about here ***

Like other areas of digital culture, open source software is a space that is changing rapidly. At the time we initiated this study, a vast majority of the literature was focused on the central question of why people (volunteers) would contribute their ideas and effort to a public commons, rather than protect their work as private, proprietary property. This "gift culture" (Raymond, 2000) was a particular puzzle to economists and, over time, the motivations driving these contributions became better understood (e.g., self learning, user centered need, self promotion, contributing to a movement, and being part of a community) (David and Shapiro, 2008; Schweik and English, forthcoming).

However, in 2005 when we started this study a shift was occurring moving open source software from what had largely been described as all-volunteer efforts to a more complex ecosystem (Figure 2). Some in private industry, such as IBM, embraced the open source collaborative paradigm in an effort to advance their business. A variety or business models and motivations connected to the open source commons production exist (see Krishamurthy, 2005; Riehle, 2007; and Deek and McHugh, 2007: 272–279 for more detail). Governments embraced open source in efforts to break the vendor lock-in problem and, in some cases, to jump-start their own country's software industry (Simon,

[3] We are not the only ones to consider the utility of this framework for the study of culturally-related commons. See Madison, Frischmann and Standburg (year needed) for another adaptation of it in cultural commons research.

2005; Lewis 2010). Not-for-profit organizations became active in this space both in their own efforts to cut information technology costs and also in some instances to promote and help open source projects operate through overarching foundations (McQuillan, 2003; NOSI, 2008). Scientific and academic organizations continued, in some cases as they always had, to share and collaborate on software code, but in sometimes new areas in an effort to move their own IT needs forward (see Courant and Griffiths, 2006). In short, when we started in 2005 it was apparent that open source was moving from a less complicated setting of volunteers collaborating to one where a variety of organizations could be involved, suggesting that the governance and institutional design (meaning in our case rules-in-use guiding collaboration) could be more complex and fluid in its evolution. We set out in our study to investigate this possibility.

*** Figure 2 about here ***

The Dependent Variable:
Success and Abandonment in Open Source Commons

After the task of researching relevant theory and empirical work related to factors that contribute to the successful collaboration or abandonment of open source projects, we then turned our attention to how to conceptualize and, ultimately, operationalize this important measure for our study. Space limits us from going into a deep discussion on

this component of the project. Here, we'll only summarize a few key aspects of this work.[4]

First, in this work we focused on the longitudinal aspects of these commons and identified two key stages in their evolution. As we see it, open source commons go through an "Initiation Stage" where they start up and have yet to produce a first public release, and a "Growth Stage" capturing the period after a first public release of code. We hypothesized that the factors affecting collaborative success and abandonment would be different in these two stages.

Second, with these stages identified, we set out to develop solid metrics for collaborative success and abandonment in each of these stages. The measures are conservative in that they see both projects that grow and gain large development teams and ones that begin and remain as very small collaborations (even 1 developer and a small community of users) as collaborative success stories. Our measures build on project "life and death metrics" (Robles-Martinez and colleagues, 2003) as well as "project popularity metrics" (Weiss, 2005).

Datasets/Methods

As I stated above, a key interest at the outset of this project was to analyze a more representative dataset of open source software commons, rather than to focus on more high profile, larger ones. In 2005 – and arguably continuing to this day – a very important location where these commons "reside" is the project hosting site

[4] A full discussion on this dependent variable can be found in English and Schweik (2007; also in Schweik and English, forthcoming, Chapter 7) and the work was generally replicated by Wiggins and Crowston (2010) helping to confirm that a solid measure of these concepts was established.

Sourceforge.net (SF). For those unfamiliar with this site, it provides a free web-based platform that allows open source software developers to store and manage their code and projects. It is a website widely known in the software programming world and is a hub on the Internet where users can find open source software projects for potential use. It is also understood that it is a somewhat "noisy" dataset in that it is used by programmers to store, for example, projects they developed for a college course. That said, in Schweik and English (forthcoming) we provide a discussion on why we think SF still would provide the best data at the time if a researcher wanted to attempt to characterize the unknown population of open source software commons.

To get SF project data, we turned to an open access dataset of SF projects called "FLOSSMole" built and provided by researchers affiliated with Syracuse University (Howison, Conklin and Crowston, 2004). The initial FLOSSMole database had 107,747 projects and represented SF in the fall of 2006. After analyzing the 2006 data alone, we then developed an online survey to capture theoretical concepts related to community and institutional variables that are not found in the SF project metadata. In an effort to ensure enough data on abandoned projects, we invited a large random sample of SF developers in the Fall of 2009 resulting in over 1400 usable surveys returned (Schweik and English, forthcoming). We then connected our survey data to another "snapshot" of SF project data provided by Madey (2010) for a more conceptually complete dataset representing, roughly, yearend 2009. Summarizing several years of work in one sentence, we carefully categorized each project in these datasets based on our definitions of success and abandonment (Table 1) and with the dependent variable in place, we analyzed and investigated the importance of theoretically identified

independent variables using contingency tables, classification and regression tree and logit regression. But rather than present statistical results, we will simply extract a few findings we think are important for discussion at the cultural commons workshop.

*** Table 1 about here ***

Finding 1. The vast majority of open source projects have very small development teams.

Figure 3 provides a histogram of the team size distribution of the 174,000 projects in SF in the fall of 2009. Table 2 shows team size numerically. Together they show that the vast majority of SF projects have only one or two developers involved. This finding isn't new – Krishnamurthy (2002) was one of the first to make this observation. But this leads to two points to be made related to this. First, much of the case research in open source over the last decade has focused on high profile projects with larger team compositions. But looking at open source from a "cultural commons" standpoint, this demonstrates that these commons are often very small teams. A second point to be made is that even though these are often very small development teams, even the 140,000 1-developer projects could be collaborations. It is well understood in open source that the end users of the software sometimes contribute to these projects in the form of bug reporting, testing, or even the creation of supporting documentation.

*** Figure 3 about here ***

*** Table 2 about here ***

Finding 2. We have statistical support for "conventional wisdom" of how open source projects operate.

Our broad statistical analysis described in detail in Schweik and English (forthcoming) tells a story of how open source projects, in general, operate. In the Initiation (pre-first release) stage, the most important factors for success originate with the designated leader of the project, who is often the single developer on the project. Projects that are successful in this stage – meaning they continue to be worked on – are ones where project leaders who devote larger numbers of hours to the project. In addition, putting in place plans for architecture and functionality, project goals, and project documentation as well as a good project web presence appears to be important, for it helps get contributions from volunteers or potential end-users before the first release. We found that "putting in the hours" and various elements of leadership are not simply correlated with success but appear to be causes of success in the Initiation stage.

In the Growth or post-first release stage, the story seems to be that once a project has achieved a first release then the leadership abilities of members of the development team, coupled with the utility of the software itself, begins to attract new users and eventually, at least one new development team member or other "community" contributors. Developers continue to make contributions to the software leading to a virtuous circle of continuing improvements and continued collaborative success. In this stage, we found that aspects of project leadership, slight growth in the development team and project financing are causal influences for success.

These findings are aligned with what we would call “conventional wisdom” about open source software commons, and may seem obvious to some who have thought about these commons. However, our findings are based on a large empirical dataset (170,000 projects) along with our own survey data of over 1400 developers. In other words, our study confirms what is commonly thought about open source with strong statistical evidence.

Finding 3. Statistical findings support the contention that the ecosystem is now more complex than the earlier “all-volunteer” days of open source.

In our analysis of the quantitative data, we had several hypotheses that were related to the emergence of the more complex ecosystem (Figure 2). Our SF dataset had nearly 50 categorical variables that classified each project in some form. For example, project administrators could identify which operating system it was built for. Or they could classify the project along a large set of “software types” (e.g., databases, end user desktop applications, etc.). Our expectation was that if open source was still being motivated by the more philosophical open source or “free/libre movement,” we’d expect to find projects aligned with this movement to be more successful in Initiation and Growth compared to projects not aligned with this movement. In other words, contingency tables would show statistically significant distributional differences for these variables between successful and abandoned projects.

In our data analysis of our 1400 surveyed projects (Schweik and English, forthcoming), we found no evidence suggesting that projects affiliated with more “pure open source” were more successful compared to projects that did not have this

affiliation. We also found no distributional differences between projects licensed with the General Public License (GPL; the license more often connected to the philosophical free/libre movement). We found success and abandoned cases evenly distributed across GPL and non-GPL cases. Finally, cases of success are found throughout all different types of software classes. There were no classes that stood out in terms of having higher numbers of successful collaborations. Further, in our survey we investigated motivations for participating in open source projects and while a vast majority felt that software should be open, it was clear that there was a wide mix of motivations driving participation and that many of the participants were paid by firms. Finally, von Hippel's (2005) concept of user driven need is widely found in all projects (successful and abandoned) and our survey shows evidence that this need driven is both a motivation for why individuals participate but also why their employers encourage them to participate (because their organization needs the software). Taken together, we think these findings provide statistical evidence that while open source is still driven by an ideology that software should be open, the lack of distributional differences in our data suggest that open source is now more mainstream and found in many areas of software. Moreover, it is not just about volunteers anymore but rather reflects the footprint of the more complex ecosystem represented in Figure 2.

Finding 4. SF and search engines like Google act as global intellectual matchmakers.

This is one of our most interesting empirical findings. In our analysis of the 2006 SF dataset, we found that the developer team in successful growth stage projects grows

slightly while abandoned projects did not. Armed with that knowledge, we investigated this further in our 2009 survey to developers. We asked them two questions related to this issue. First (paraphrasing), if their development team had more than one individual, did they find that individual on the Internet (yes, no)? Second, we asked which of the following best describes the geographic location of all project team members: very close; same city; same state/province; same country; same continent or multi-continental. Through these questions we found that successful projects find new team members on the Internet more than abandoned projects, and in the growth stage, 52% of the successful growth stage projects had team members from multiple continents (Schweik and English, forthcoming). Third, we asked respondents about the frequency of face-to-face interaction on their teams and found that while face-to-face indeed helps to build social capital, there are many successful projects where some team members have never met in person.

Taken together, what these findings suggest is that SF, perhaps coupled with search engines like Google, act as a “power law hub” (Karpf, 2010) allowing people with similar interests, passions and skills to connect with each other and begin to feel each other out as potential collaborators, and eventually joining in co-production together. In the SF case this is largely multi-continental collaborations between North America and Europe. I have not seen statistical evidence anywhere else that suggests this is happening but this appears to be important in open source and I think this is a very important finding and something that is happening in other digital commons situations (e.g., Wikipedia). In short, in open source it is often not about whether you can build a large development team. It is about finding “just the right person” with similar interests, skills and passions

somewhere in the world and developing a collaborative relationship with that person. I think this is a potentially very important insight for our understanding of cultural commons.

Finding 5. Institutions provide "friction" but they also evolve.

The governance and institutional design is one of the areas of open source commons that has been studied the least, although there are a few scholars who have contributed important work (see for example O'Mahony, 2005, 2007; O'Mahony and Ferraro, 2007; Markus, 2007). A driving motivation for our study was to begin to shed more light in institutional structure of open source as well as the evolution of open source institutions.

In his famous essay on open source *The Cathedral and the Baazar* (2001), Eric Raymond argued potential contributors to open source software commons will be less apt to participate if the "number of hoops" they have to deal with is higher. In other words, more formalized rules guiding the process of contributing code will reduce the likelihood of participation. Programmers want to program, not deal with rules guiding collaboration. In our research, we found strong statistical and qualitative evidence suggesting that this is indeed the case, and this is one area that differs quite sharply from more traditional environmental commons I have studied in the past. In most of the SF projects we surveyed, the operational rules are "very informal" social norms. What operational rules that do exist are often embedded in the online collaborative system used to coordinate work (e.g., CVS or Subversion). Moreover, we found that in the SF

survey data the dominant governance model was a “benevolent dictator” rather than a model with more democratic processes.[5]

This all said, we did also see some indication – as we expected – that institutions evolve and become more formalized as projects grow in terms of developers. However, in our survey data, this tended to be more along the lines of “we moved from a *very* informal institutional structure to simply an informal structure,” that is, the “very” descriptor was removed. But it did suggest that institutions do evolve toward some level of formalization.

What this tells us

Two conclusions come out of the discussion above that are important for where we are headed in this paper.

First, looking back at Figure 3, our interest in understanding and articulating the population of open software source commons leaves us with conclusions that are, in a way, weighted toward smaller development projects. This is true because the population we studied, SF, is vastly dominated by projects driven by very small development teams and a majority that is only one developer. But even in cases of 1-developer teams, the software is still a form of commons (more accurately a common-property regime, see Schweik and English, forthcoming), and can still involve collective action between the developer and a user community. However, what we didn’t do in our data analysis – and what should be done – is an examination of the larger developer team projects that

[5] We think this finding was so strong because we were trying to understand the population of open source and so many of the projects in our dataset were small development team projects.

reside in the longer tail in Figure 3. In our 2006 data, the largest project was on the order of 320 developers, but it was only one of 107,747 in our dataset. Still (while at the time of this writing I don't have the actual number) I estimate that there are several thousand projects in these SF data that have a developer team of ten people or more. The institutional design of these projects is likely different than the vast majority of others in SF, and their attributes are drowned out in our statistical analysis because they represent such a small proportion of the population.

Second, in our analysis, we have found evidence that the governance of projects do indeed evolve. This is important to understand better, in particular for larger projects that involve participants from a variety of organizations. For example, recently I gave an invited talk on my forthcoming book at an open source software conference focused on software in the U.S. Military (http://mil-oss.org/wg3-agenda). Participants at this conference support the contention that open source software is now "more mainstream" and involves a much more complex configuration of actors. Projects presented involved university partners, private contracting firms, and government agencies. Other presenters, such as Alex Voultepsis of the Office of the Director of National Intelligence, talked about forming a software sharing and development effort that would cross homeland security agencies and involve private contractors. Understanding the evolution of how these collaborations work, and how they are structured institutionally, is important for future larger-scale software projects (and other open-type collaborations outside of software) to succeed.

In other words, mid- to large-sized projects are interesting and an important subset of open source software commons that need to be understood and studied more

carefully. They are an important subset of open source commons “success stories” and represent more “mature” collaborations.

The key questions that we must ask then are:

> *How do we systematically document the institutional designs of larger collaborations or collaborations between organizations?*
>
> and,
>
> *How do we study systematically the evolution of open source-like commons?*

Rich case description is helpful and important. However the question I am asking is how do we compile and analyze, side by side, rich case studies of open source commons institutional designs as well as their evolution? How do we compare, to use to famous projects as an example, how the Debian Linux project has evolved compared to how the Eclipse project has evolved? In the next section – the last part of this paper – I build on work by Elinor Ostrom (2005) and summarize an attempt I have made to move toward more systematic study of the institutional designs of larger-scale open source software commons.

Toward a Systematic Analysis of Institutions: The OSGeo Case Study

In parallel to the quantitative research briefly reported above, we also were interested in understanding open source commons in more depth, for two reasons. First, we wanted to complement our quantitative analysis with a more rich study to see if what we were learning from the quantitative work aligned with what we could learn from case research. Second, given we had read a number of more descriptive papers on specific, usually high-profile open source projects, we wanted to take a step forward toward

developing a possible systematic comparative approach to analyzing open source governance and institutional design.

We selected the Open Source Geospatial Foundation and seven of its associated projects for study. We chose this case because it provided an example of a more complex open source ecosystem given nonprofits and private firms are involved, and its international scope was an added benefit. A practical reason for choosing this case is that I am involved with this foundation, currently acting as the chair of its education and curriculum group. This participation, I think, gave me more credibility as we moved ahead to interview developers in OSGeo-related projects.

Full descriptions of the analysis of the OSGeo case can be found at Schweik and Kitsing (2010) and Schweik and English, forthcoming (chapters 5 and 6). For our interests here, my goal is to briefly describe my experience using IAD as a guiding framework for institutional analysis, coupled with Ostrom's seven categories of rules (Ostrom, 2005).

A primer on the three institutional levels of IAD and Ostrom's seven rule classes

Summarizing what is more fully described in Ostrom (2005), within the "institutional attributes" component of the IAD framework are three nested institutional levels: operational, collective-choice and constitutional.

The operational level is a general label to describe the general sets of rules (formal or informal) that influence the daily behavior and actions of commons participants. For example, in an open source software setting, these can be rules established for how

computer code gets accepted into the next version of the software being developed or the process for releasing a new version.

The collective-choice level involves a different set of rules that (a) define who is eligible to undertake operational-level activities and (b) defines who has authority to change operational-level rules and the procedures for how these changes come about. An example of a collective-choice rule in the context of open source software commons would be a change in the process of how code is committed to the repository, or a change in how code is reviewed prior to adding to a new release library.

Finally constitutional level rules specify how the commons is structured or organized constitutionally, but, in addition include specifications on who can change collective-choice arrangements and also the procedure for how those changes can be made. In open source software commons, an obvious constitutional issue is the choice of license to attach to the software (e.g., a GPL or non-GPL type license). But it can also be related to whether the project is associated with a non-profit foundation or whether there are particular requirements related to an oversight board, etc.

These three levels of rules in IAD have been guiding commons researchers for at least two decades, particularly in the context of natural resource commons. But what was lacking in case analysis was more specificity on how to articulate rules that exist in any of these levels. In her book *Understanding Institutional Diversity* (2005), Elinor Ostrom tried to add more clarity by proposing seven classes of rules that can exist in any of these three levels, summarized in Table 3.

*** Table 3 about here ***

The OSGeo Case

OSGeo is a nonprofit foundation that provides overarching support to a number of open source software projects working in the area of geographic information systems. Broadly speaking, these are technologies that, in some fashion, deal with data that has positional ties to the earth. OSGeo's mission is to "support the development of open-source geospatial software, and promote its widespread use" (OSGeo 2009). At the time of our research there were ten software projects treated as "full members" with the foundation, and several others in "incubation." We chose to study only fully affiliated projects because their institutional designs would be fully formed. We contacted representatives of all ten projects for interviews, but only seven were willing to participate in the study. Here, we only report some of the findings related to institutional design and structure.

Institutional analysis of this case requires not only attention to the three levels and classes of rules that Ostrom presents, but also a realization that there is an institutional design at the foundation level, and then also institutional designs for each project. Moreover, there are mandates established by the foundation that each project must comply with.

At the foundation level, we found examples of Ostrom's rule classes, some of which are summarized in Table 4.

*** Table 4 about here ***

At the project level, our interviews and interpretation of online documentation led us to a characterization of rules at this level as well (Table 5).

*** Table 5 about here ***

I present Tables 3 and 4 not with the expectation that readers will have enough information to understand the institutional design of the OSGeo foundation or associated projects. These tables are included simply to demonstrate that it is possible to articulate open source commons institutional designs using Ostrom's seven rule framework.

In another section of our book-length study (Schweik and English, forthcoming, Chapter 5, Table 5.4), I was able to take the description of the evolution of institutions in the Debian Linux case articulated by O'Mahony and Ferrarro (2007) and associate key rules described to Ostrom's seven rule classes. In their paper, O'Mahony and Ferrarro describe the evolution of governance structures in this high profile open source software project, and we were able to articulate in a two-column table the various seven rule classes described at different institutional levels (constitutional, collective-choice, or operational) for two longitudinal stages. (For space reasons I decided not to include this table in this paper). The important point to be made is that it is possible to do this.

Conclusions and Future Research

The take home message is that the OSGeo case study experience has convinced me that it is possible to articulate more systematically the institutional designs of open

source software commons (and, by extension, other types of open, digital commons where collaboration occurs). Related to the study of cultural commons, this is vitally important, for increasingly we are seeing digital commons appear on the Internet where people are trying to collaborate. In many cases, similar to what we have shown with open source software commons, projects will be very small teams with extremely lean institutional designs. One thing that we've learned from our quantitative work is in a very large percentage of these, it is not about the building a large team of developers. Much can get done, driven in part by user-or organizational-centered need if even two people with a similar passion, interest and skills can find each other and begin to build a working relationship.

At the same time, I would argue we are just at the cusp of a new period of open collaboration that involves much more complicated organizational arrangements where it is likely that institutional designs will be more complex and perhaps fluid over time. The Debian Linux case described by O'Mahony and Ferrarro (2007) is a case in point. Increasingly there is a need to understand the structure of institutional designs in these larger, more complex commons, and a need to understand how they evolve. Our empirical work on SF projects revealed some evidence that formalized institutions do emerge as projects get larger, but our findings along these lines are limited because of the large number of small projects in our SF dataset.

The question this leads me to is how to best analyze institutional structure and its evolution in a comparative fashion? We need methods for systematically articulating these structures and Ostrom's IAD levels (operational, collective-choice, constitutional) coupled with her seven rule categories provides an initial framework to do this. Our work

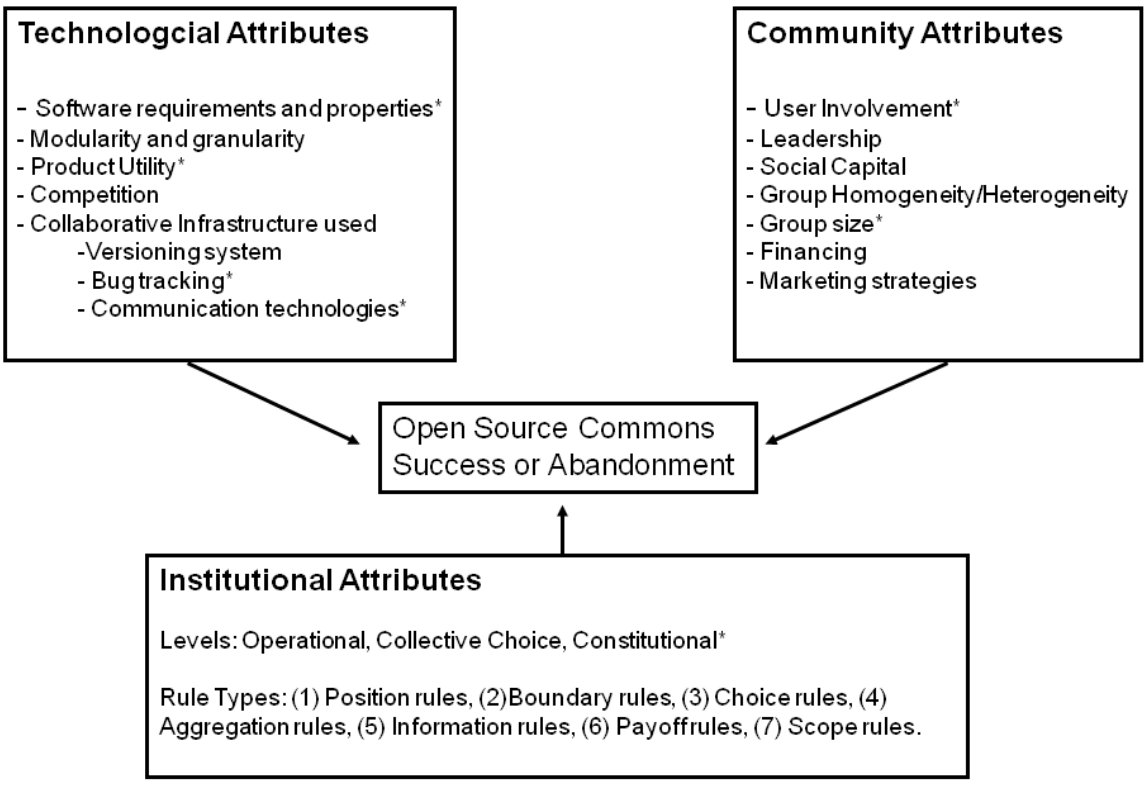
provides an existence proof that this is possible. However, I've found that even with these common classification schemes, comparative analysis in table format between cases or across time periods is still difficult.

I mentioned earlier that I gave a talk on this study recently at a U.S. Military open source software conference (http://mil-oss.org/wg3-agenda). Sessions were devoted to understanding various open source licenses as they apply to government and government contracting situations (Wheeler, 2011) and issues related to government policy related to the adoption of open source technologies. These examples provide more proof in my view that we are moving toward a much more complicated open source commons setting, and we need to develop analytic methodologies to document project structure and evolution. In these examples, clearly, the legal scholarly community has much insight to provide. It is my hope that some of the other discussions in the Cultural Commons workshop might help to connect analytic approaches like Ostrom's to other methods and approaches in other disciplines like law to help us articulate the evolution of these kinds of digital commons cases.

Acknowledgments

Support for this work was provided by a grant from the U.S. National Science Foundation (NSFIIS 0447623). The findings, recommendations and opinions expressed are those of the authors and do not necessarily reflect the views of the funding agency. The work this paper builds upon is the result not just the author but a team of highly capable individuals: Bob English, Sandra Haire, Meng-Shiou Shieh, and Meelis Kitsing. Of course, any mistakes are my responsibility alone.

Figure 1: A simplified Institutional Analysis Framework with independent variables identified (adapted from Schweik and English, Forthcoming)



Note: "*" denotes concepts that we could operationalize using the Sourceforge.net dataset

Figure 2: The Open Source Software Commons Ecosystem
(from Schweik and English, forthcoming)

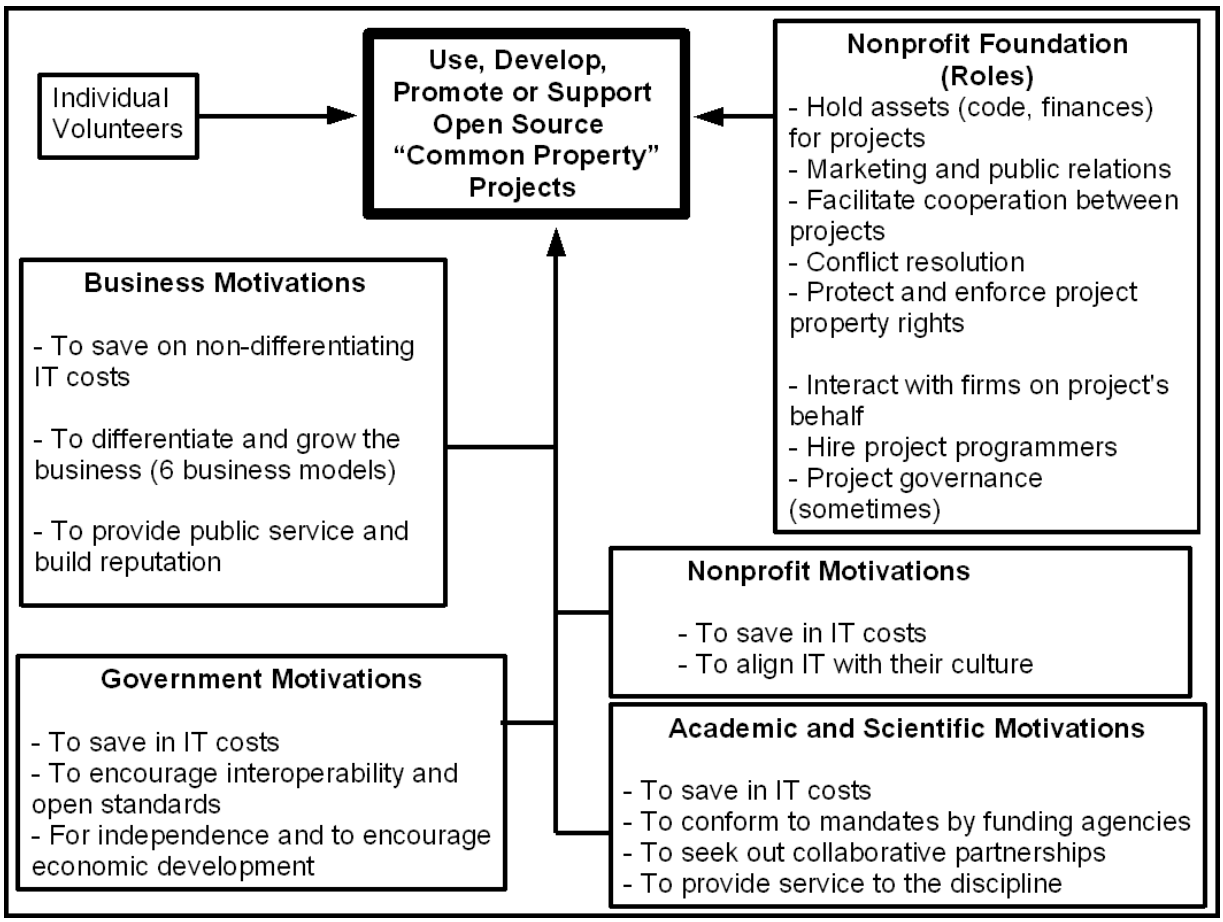


Figure 3.

Distribution of 2009 Sourceforge Projects by Development Team Size

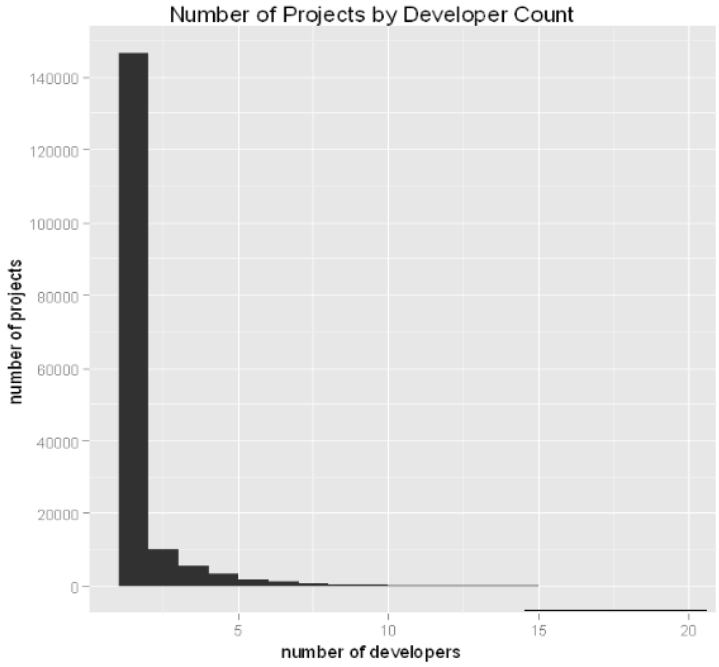


Table 1. Success and Abandonment Classification Results for the SF 2006 and 2009 Datasets

	SF 2006 Data	SF 2009 Data
Class	# of projects	# of projects
Abandoned Initiation	37,320	67,126
Indeterminate Initiation	13,342	16,806
Indeterminate in Growth	10,711	12,052
Abandoned Growth	30,592	53,450
Successful Growth	15,782	24,899
Total	107,747	174,333

Table 2. Developer Counts for the May 2009 Sourceforge Population for Abandoned (AG) and Successful (SG) Growth Stage Projects (percentage of total* in parentheses)

Developer Count	SF Population: Class AG	SF Population: Class SG
1	37259 (48)	11765 (15)
2	8329 (11)	4420 (6)
3	3236 (4)	2380 (3)
4	1573 (2)	1515 (2)
5	921 (1)	1076 (1)
6	525 (1)	731 (1)
7	316 (0)	514 (1)
8	198 (0)	424 (0)
9	141 (0)	309 (0)
10	90 (0)	234 (0)
11–20	305 (0)	990 (1)
>20	53 (0)	457 (0)
Totals	52,946 (68)	24,815 (32)

Table 3: Ostrom's Seven Rule Classes

(These can apply to any or all of the three nested levels: operational, collective-choice or constitutional)

(Adapted from Ostrom, 2005: 193-210)

Position rules	Articulate what roles people play in the project
Boundary rules	Define who is eligible for a position, the process of how he or she is assigned to that position, and rules related to how the person leaves that position.
Choice rules	Define actions that can, cannot, or must be done.
Aggregation rules	Articulate the process for how conflict should be resolved. Within this category, there are three sub-classes: symmetric (e.g., unanimity), non-symmetric (where a leader can make decisions for a group) and lack-of-agreement rules.
Information rules	Specify how and what kind of information flows between project members and other interested parties, as well as how information is archived through the project life cycle.
Payoff rules	Assign some kind of reward or sanction to specific actions or outcomes.
Scope rules	Specify which outcomes may, must or must not be affected or produced in a given situation.

Table 4. Examples of rules at the OSGeo foundation level	
Aggregation rules	Symmetric: Consensus in committees; Nonsymmetric: Board of Directors create committees
Information rules	Minutes meetings required. Meeting notifications required. Annual meetings required. Financial statements required.
Pay-off rules	Executive director and others can be paid. BOD members cannot be paid.
Scope rules	Specified to some extent in organizational and committee mission statements.

Table 5. OSGeo Affiliated Projects and Some of their Associated Rules							
Ostrom's Rule Category	**Project A**	**Project B**	**Project C**	**Project D**	**Project E**	**Project F**	**Project G**
Position rules	Project leader Project steering committee member Core developer (informal; often overlaps with the committee member) Developer	Project leader Project steering committee member Core developer (informal; often overlaps with the committee member) Developer	No formal project leader Informal lead team of three people Project steering committee member Committers	Project leader Project steering committee member Core developer (informal; often overlaps with the committee member) Developer	Project leader Project steering committee member Core developer (informal; often overlaps with the committee member) Developer	Project leader Project steering committee member Core developer (informal; often overlaps with the committee member) Developer	No formal project leader Informal lead team of four people Project management committee member Core developer (informal; often overlaps with the committee member) Developer
Boundary rules	Formal rules Community members elect to PSC No term limits	Formal rules.	Formal rules copied from another project	Formal rules copied from another project Almost never consulted	Formal rules	Formal rules exist but primarily depend on social norms	Formal rules, but not necessarily followed
Choice rules	Some formalized	Some formalized	Some formalized	Social norms	Social norms	Social norms	Formalized rules written down

Table 5. OSGeo Affiliated Projects and Some of their Associated Rules							
Ostrom's Rule Category	**Project A**	**Project B**	**Project C**	**Project D**	**Project E**	**Project F**	**Project G**
	Program steering committee makes some major rules Primarily social norms Open exchange in the list Mutual expectations	available in the wiki Primarily social norms	available in the wiki Primarily social norms				Program management acts if necessary Social norms important
Aggregation rules	Informal-symmetric: consensus in program steering committee and discussion including developers who are not in the committee Formal voting: rarely occurs, even though	Steering committee: almost all developers are on the committee Voting: if veto vote is used, discussion follows	Informal-symmetric: consensus in Program steering committee Formal voting: rarely occurs even though formal rules stipulate it Only PSC members can	Steering committee makes decision by consensus or voting All developers can vote as well but their votes do not count.	Informal-symmetric: consensus in program steering committee Formal voting: rarely occurs even though formal rules stipulate it Only PSC members can	Informal-symmetric: consensus in program steering committee and discussion but often back channels used before the decision is reached Voting is a last resort	Program management committee votes

Table 5. OSGeo Affiliated Projects and Some of their Associated Rules							
Ostrom's Rule Category	**Project A**	**Project B**	**Project C**	**Project D**	**Project E**	**Project F**	**Project G**
	formal rules stipulate it Only PSC members can vote		vote.		vote		
Information rules	Social norm: open exchange of information Unwritten rule that email list is the main communicatio n tool	Limited formal rules Most decisions are made in Internet Relay Chat and mailing list is used as well	Social norms Talking over email and weekly Internet Relay Chat meetings	Social norms Project leaders available on Internet Relay Chat almost all the time	Social norms	Social norms All communication is based on writing	Social norms Weekly Internet Relay Chat meetings; otherwise no clear rules
Pay-off rules	No rules	No rules	No rules	No rules	No rules	No rules	No rules
Scope rules	Design rules	Design rules	Design rules	Design rules	Design rules	Design rules	Design rules

References

Benkler, Y. 2006. *The Wealth of Networks: How Social Production Transforms Markets and Freedom*. New Haven, CT: Yale University Press.

Campbell-Kelly, M. 2003. *From Airline Reservations to Sonic the Hedgehog: A History of the Software Industry*, Boston, MA: The MIT Press.

Courant, Paul N., and Rebecca J. Griffiths. 2006. Software and Collaboration in Higher Education: A Study of Open Source Software. A Report Commissioned by the Andrew W. Mellon Foundation and the William and Flora Hewlett Foundation. Accessed May 15, 2008, http://www.ithaka.org/strategic-services/oss/OOSS_Report_FINAL.pdf.

David, Paul A., and Joseph S. Shapiro. 2008. Community-Based Production of Open-Source Software: What Do We Know about the Developers Who Participate? Information Economics and Policy 20:364–398.

Deek, Fadi P., and James A. M. McHugh. 2007. Open *Source Technology and Policy*. New York: Cambridge University Press.

English, Robert, and Charles M. Schweik. 2007. Identifying Success and Tragedy of FLOSS Commons: A Preliminary Classification of Sourceforge.net Projects. Upgrade: The European Journal for the Informatics Professional. VIII(6):54–59. URL: http://www.cepis.org/upgrade/files/full-VI-07.pdf.

Feller, Joseph, Brian Fitzgerald, Scott A. Hissam, and Karim R. Lakhani. ed. 2005. Perspectives on Free and Open Source Software. Cambridge, MA: The MIT Press.

Fogel, Karl. 2006. Producing Open Source Software: How to Run a Successful Free Software Project. Sebastopol, CA: O'Reilly Media. URL: http://producingoss.com/.

FSF (Free Software Foundation). 2009. What is CopyLeft? Accessed November 30, 2009, http://www.gnu.org/copyleft/.

Hess, C., and Ostrom, E. eds. 2007. Understanding Knowledge as a Commons: From Theory to Practice. Cambridge, MA: The MIT Press.

Howard, A. 2010. Connecting the Dots with Intellipedia. O'Reilly Radar. http://radar.oreilly.com/2010/06/connecting-the-dots-with-intel.html. Accessed September 8, 2011.

Howison, J., Conklin, M. and Crowston, K. 2006. FLOSSmole: A Collaborative Repository for FLOSS Research Data and Analyses. International Journal of Information Technology and Web Engineering 1(3):17–26.

Ja, S. K., and Nerurkar, A. 2010. Expanding Open Source Into Other Domains: Analysis of Open Source Biomedical Research. In Shulman and Schweik (editors), Conference Proceedings of JTIP 2010: The Politics of Open Source. Available at http://scholarworks.umass.edu/cgi/viewcontent.cgi?article=1000&context=jitpc2010. Accessed September 8, 2011.

Karpf, David. 2010. What Can Wikipedia Tell Us about Open Source Politics? In Proceedings of JITP 2010: The Politics of Open Source, comp. Stuart W. Shulman and

Charles M. Schweik, 2–30 Accessed August 10, 2010, http://scholarworks.umass.edu/jitpc2010/1/.

Krishnamurthy, Sandeep. 2002. Cave or Community? An Empirical Examination of 100 Mature Open Source Projects. First Monday 7(6) [online], http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/ fm/article/view/960/881.

Krishamurthy, Sandeep. 2005. An Analysis of Open Source Business Models. In Perspectives on Free and Open Source Software, ed. Joseph Feller, Brian Fitzgerald, Scott A. Hissam, and Karim R. Lakhani, 279–296. Cambridge, MA: The MIT Press.

Kuniholm, J. 2011. The Open Prosthetics Project. Presentation at the Military Open Source Software Conference. Available at http://mil-oss.org/wg3-speakers-and-presentations. August 31, 2011.

Lewis, James. 2010. Government Open Source Policies (Version 7). Accessed April 6, 2011, http://csis.org/files/publication/100416_Open_Source_Policies.pdf.

LocalMotors. 2011. How It Works. Accessed April 8, 2011, http://www.local-motors.com/howItWorks.php.

Madey, G. 2010. SourceForge.net Research Data. URL: http://www.nd.edu/~oss/Data/data.html. Accessed September 1, 2011.

Madison, M., Frischman, B.M. and Strandburg, K.J. year needed. Constructing Commons in the Cultural Environment. *Cornell Law Review* 95: 657-710.

Markus, M. Lynne. 2007. The Governance of Free/Open Source Software Projects: Monolithic, Multidimensional, or Configurational? *Journal of Management and Governance*. 11(2):151–163.

McQuillan, D. 2003. Open Source is on the Map. http://www.icthubknowledgebase.org.uk/opensourceonthemap. Accessed September 10, 2011.

NOSI (Nonprofit Open Source Initiative). 2008. NOSI's Survey of FOSS Use in the Nonprofit Sector. Accessed March 11, 2008, http://www.nosi.net/projects/survey.

O'Mahony, Siobhan. 2005. Nonprofit Foundations and Their Role in Community-Firm Software Collaboration. In Perspectives on Free and Open Source Software, ed. Joseph Feller, Brian Fitzgerald, Scott A. Hissam, and Karim R. Lakhani, 393–413. Cambridge, MA: The MIT Press.

O'Mahony, Siobhan. 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, Siobhan, and Fabrizio Ferraro. 2007. The Emergence of Governance in an Open Source Community. Academy of Management Journal 50(5):1079–1106.

OSGeo. 2009. OSGeo Mission Statement. http://www.osgeo.org/content/foundation/about.html. Accessed March 10, 2009.

Ostrom, Elinor. 2005. *Understanding Institutional Diversity*. Princeton, NJ: Princeton University Press

Raymond, E. 2000. Homesteading the Noosphere. Available at http://catb.org/esr/writings/homesteading/homesteading/. Accessed September 8, 2011.

Raymond, E. 2001. The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary. Sebastopol, CA: O'Reilly.

Riehle, Dirk. 2007. The Economic Motivation of Open Source Software: Stakeholder Perspectives. IEEE Computer 40(4):25–32.

Robles-Martínez, Gregorio, Jesús M. González-Barahona, José Centeno-González, Vicente Matellán-Olivera, and Luis Rodero-Merino. 2003. Studying the Evolution of Libre Software Projects Using Publicly Available Data. Portland, OR: ICSE International Conference on Software Engineering. May 3-11.

Schwarz, M. and Takhteyev, Y. 2009. Half a Century of Public Software Institutions: Open Source as a Solution to Hold-Up Problem. Working paper 14946. National Bureau of Economic Research. http://www.nber.org/papers/w14946.pdf?new_window=1. Accessed 9/6/2011.

Schweik, C.M., and English, R. forthcoming. *Successful Internet Collaboration: A Study of Open Source Software Commons*. Cambridge, MA: MIT Press.

Schweik, C., Tom P. Evans, and J. Morgan Grove. 2005. Open Source and Open Content: A Framework for Global Collaboration in Social-Ecological Research. Ecology and Society 10(1):33. http://www.ecologyandsociety.org/vol10/iss1/art33/.

Schweik, C. and Kitsing, M. 2010. "Applying Elinor Ostrom's Rule Classification Framework to the Analysis of Open Source Software Commons."Journal of Transnational Corporations Review. Available at http://www.tnc-online.net/pic/2010032809124697.pdf.

Schweik, C. M., Mergel, I., Sandfort, J. and Zhao, Y. 2011. Toward Public Administration Scholarship. Journal of Public Administration Research and Theory 21(2011):i175–i198.

Simon, Kimberly D. 2005. The Value of Open Standards and Open-Source Software in Government Environments. IBM Systems Journal 44(2):227–238.

von Hippel, E. 2005a. Democratizing Innovation. Cambridge, MA: The MIT Press. URL: http://web.mit.edu/evhippel/www/democ1.htm.

von Hippel, E. 2005b. Open Source Software Projects as User Innovation Networks. In *Perspectives on Free and Open Source Software*, ed. Joseph Feller, Brian Fitzgerald, Scott A. Hissam, and Karim R. Lakhani, 267–278. Cambridge, MA: The MIT Press.

Weber, Steven. 2004. The Success of Open Source. Cambridge, MA: Harvard University Press.

Weiss, Dawid. 2005. Measuring Success of Open Source Projects Using Web Search Engines. Paper presented at the First International Conference on Open Source Systems, Genova, Italy, July 11–15.