6-18-2013

"Release Early, Release Often"? An Empirical Analysis of Release Strategy in Open Source Software Co-Creation

Wei Chen
University of California, wei.chen@rady.ucsd.edu

Vish Krishnan
University of California, vkrishnan@ucsd.edu

Kevin Zhu
University of California, kxzhu@ucsd.edu

"RELEASE EARLY, RELEASE OFTEN"? AN EMPIRICAL ANALYSIS OF RELEASE STRATEGY IN OPEN SOURCE SOFTWARE CO-CREATION

Wei Chen, Rady School of Management, University of California, San Diego, La Jolla, CA, USA, wei.chen@rady.ucsd.edu

Vish Krishnan, Rady School of Management, University of California, San Diego, La Jolla, CA, USA, vkrishnan@ucsd.edu

Kevin Zhu, Rady School of Management, University of California, San Diego, La Jolla, CA, USA, kxzhu@ucsd.edu

Abstract

"Release early, release often" is becoming a popular new product introduction strategy in open source software development. We study the influence of release strategies on the download market share of open source projects. Using a panel data set collected from Sourceforge.net, we find that while more frequent releases are associated with better subsequent download market share, the relationship is curvilinear. Too frequent releases could backfire due to the subtle effects on the demand and supply sides of open source software production. From the demand side, we find that releasing frequently may work less effectively in projects with higher adoption costs. From the supply side, fast releases may work less effectively in projects with weak community contributions. Even when the community contributions are strong, the restrictiveness of open source license moderates the effectiveness of releasing early and often. These results have implications for managing open source projects and research on open source software, open innovation, and software adoption.

Keywords: Open Source Software, Community Contribution, License, Adoption Cost, Download.

1 INTRODUCTION

Open source software (OSS) has become an important approach to organize software development in the past decade, and is continuously transforming the software industry (The Economist 2009). OSS differs from traditional proprietary software by distributing the source code openly (at no or very little cost), and allowing others to modify or enhance it (OSI 2011). In a typical OSS project, a single developer or a small team start an OSS project, which then grows as it attracts community contributions (Setia et al. 2012). Incorporating these contributions, the team distributes the software through online channels such as Sourceforge.net (SF). Users download the software under one of various types of license arrangements that govern how the software can be subsequently developed and monetized. Some of the users may join the co-creation of OSS production by reporting bugs or even writing patches for the project.

The co-creation in OSS does not always unfold smoothly. Fogel (2005) estimates that 90% to 95% of the OSS projects fail, which means they attract no attentions or the developers stop working on it. Chengalur-Smith and Sidorova (2003) find that about 80% of projects on Sourceforge.net (SF) have no activity at all. To fully understand the social mechanisms behind OSS, the motivations and governance of OSS stands out as important research topics (Von Krogh and Von Hippel 2006). While empirical studies identify project characteristics affecting OSS success (Grewal et al. 2006; e.g. Stewart et al. 2006; Subramaniam et al. 2009), the OSS production process and its social environment are usually ignored (Singh et al. 2011). Our study considers OSS projects in a competitive context and examine one crucial factor that a project development team can control: the release strategy, which refers to the release frequency and quality improvements by the OSS teams. We also study the influence of factors such as community contributions, OSS licenses and product adoption cost in shaping the release strategy.

Raymond (1999) argues that an OSS project should "release early, release often" to produce software with higher quality, which helps the project succeed under competition. Over the years, this has been accepted as a common belief in OSS development. However, there has been no empirical study testing this approach. In this study, we investigate the relationship between release strategy and project success. which is normally defined with user interest and developer interest (Grewal et al. 2006; e.g. Stewart et al. 2006; Subramaniam et al. 2009). We focus specifically on the user interest, which refers to the download market share of a project.

Motivated by the issues identified above, we seek to study the following research questions: (1) How would release frequency in OSS projects affect their success? Is it true that more frequent release is better? (2) How do factors like adoption cost and community contributions impact the release strategy? (3) What is the effect of OSS license in this process? To address these questions, we first develop a theoretical framework which specifies release strategy as a factor associated with the success of OSS projects, with other project attributes as control or moderate variables. We then test the model using a panel data set of 1092 projects in 15 categories from SF. Our data analysis identifies release strategy as an important factor that is associated with market share of OSS projects, and reveals other variables that affect the community-enabled product development and release process.

2 THEORY AND HYPOTHESES

We view the OSS development as a co-creation process of the OSS team and the community, which von Hippel and von Krogh (2003) call a "Private-Collective" innovation model. This is unique to open source software in contrast to traditional proprietary software development. We investigate the co-creation process through one factor that the team could control – the release strategy, and other variables of the production process, the community contributions and the software license types.

Several important questions have been asked in the OSS literature. First, why do people contribute to OSS for free? Developers contribute to OSS projects mainly with two types of motivations: intrinsic (e.g. altruism, reciprocity, and hobbies) and extrinsic (e.g. reputation, signaling, learning, and self-need) (Lakhani and Wolf 2007; Roberts et al. 2006). However, utilizing the motivations does not necessarily lead to success of OSS projects, because most of normal users do not know the effort of developers unless they can see and use the product. Seeking to bridge this gap, we propose to examine release strategy as an important but missing linkage between motivations and success of OSS. Second, in considering the competition with proprietary software, studies have asked when software should go open source and which license should a project choose (August et al. 2009; Lerner and Tirole 2005). However, projects sometimes have to follow certain license or are affected by social influence (Singh and Phelps 2012). For example, the GNU General Public License (GPL) requires that any derived work should open their source code under GPL. In this paper, we investigate the effect of release strategy on the download market share given the software license of the project.

From the software adoption perspective, the users choose among competing OSS projects by comparing the utilities they could obtain from the products. Since OSS is normally distributed for free, quality of OSS becomes an important determinant of project success. In this study, we draw on related literature to formulate our hypotheses. Figure 1 presents the conceptual framework we developed for this study.

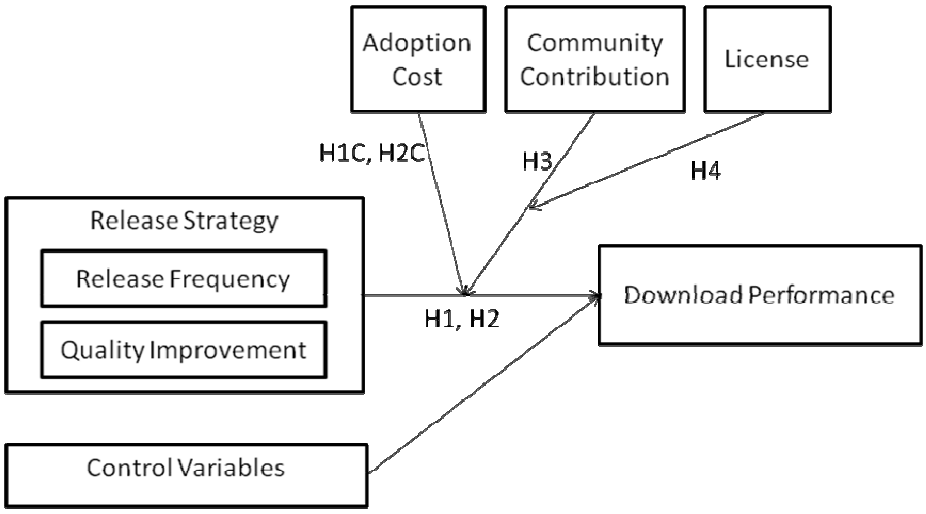

Figure 1. Conceptual Framework

2.1 Release Strategy

How and why would releasing early and often help an OSS project in its download market share? First, a project can deliver quality improvement quickly to users with frequent releases. Research on software releasing shows that due to the fixed cost nature of investment in patching, a software may release a buggier product early and patch it later in a larger market (Arora et al. 2006; Ji et al. 2005). Second, OSS projects rely heavily on community contributions. A faster release frequency may attract more community contributions, which help improve the quality of the software. Faster release frequency would also reward the contributors by incorporating their suggestions (Raymond 1999). Lastly, higher release frequency may signal energy and momentum in development and can give users more confidence in adopting the software. It is argued in the literature that under-provision of introductory quality (release early) could signal high externality, and upgrades could serve as the mechanism to implement the signaling strategy (Padmanabhan et al. 1997). With the above logic, we conjecture that:

H1A: *Higher release frequency is associated with larger subsequent download market share.*

However, there may be a down side with high release frequency. First, there are always costs associated with software adoption, both in the case of users upgrading and switching software. When a project releases too fast but does not deliver enough quality improvements, customers may hold back by weighing the benefit and cost of upgrading. Therefore, customers may be more likely to hold back when the adoption cost is high. Second, releasing a large number of versions in a short time period may exaust the team and community and create version fragmentation and overload. The project may have to maintain a lot of versions, which creates additional burden for the project to absorb the contributions and makes it hard for the community to keep up with the speed. Therefore, we expect the release frequency to have a curvilinear relationship (inverse U-shape) with subsequent download market share.

H1B: *Release frequency has a curvilinear relationship with subsequent download market share.*

Users will natually weigh the benefit and cost of adopting the new versions. Even though most of the OSS products are distributed for free, there are costs associated with adoption anyway. Whether the user is installing or updating, it takes time and effort to implement and learn the new versions, especially for software that affects a whole organization. If the release is of limited scope and contains only a few small bug fixes, the upgrade cost would exceed the benefit and users could choose to wait until the next release. Therefore, the adoption cost may outweigh the benefit from adopting the new version. This leads to our next hypothesis:

H1C: *The positive effect of release frequency on download market share is lower in projects with higher adoption cost.*

2.2 Community Contributions

We now investigate the supply side of OSS production. If a project has unlimited resources, it could definitely improve the quality by fast and frequent releases. However, projects seldom have unlimited resources, and most OSS projects have a small core development team (Raymond 1999). Even if the OSS project is backed up by a big company, the firm has to evaluate the investments carefully since it is difficult to sell the software product directly. Moreover, adding developers to the project may have diminishing marginal productivity boosting effect (Brooks 1995). On the other hand, there is a whole community working voluntarily for the project, though the team often has no control over the features and bugs that volunteers want to work on (Michlmayr et al. 2007). But they do have control over the release plan for new versions. Our second research question thus is: Given the contributions of the community, what release strategy should an OSS project choose?

The importance of contributions from the community (either individuals or firms) has been recognized in OSS studies (e.g. Grewal et al. 2006; e.g. Lerner et al. 2006). Raymond (1999) argues that the OSS community could help in rapid improvement and effective debugging. von Krogh et al. (2003) analyze the innovation process of Freenet and discuss the community joining and specializing of open source community. von Hippel (2001) use OSS as an important example of innovation by user community. Following West and Lakhani (2008), we define the community of an OSS project as a voluntary association of users and developers without prior common organizational affiliation but united by creating and adopting the OSS program. That is, we define the community contributions as the work emerged from those users and developers outside the OSS team. Given the benefits brought by community contributions, we expect that the impact of a certain release is bigger when the community contributions are higher controlling the quality improvement effort by the team.

H2: *Effect of release frequency on subsequent download market share is higher when the community contributions are greater.*

2.3 OSS Licenses

OSS projects use open source license schemes to maintain their openness, which refers to free access, free distribution, and free modification of the source code (OSI 2011; Zhu & Zhou 2012). Though all OSS licenses follow the Open Source Initiative (OSI) definition, they vary in their relative restrictiveness, based on whether the derivative works should follow the same license (e.g. GPL) and whether to allow the mixing of open and closed source software (e.g. GNU Lesser General Public License, LGPL) (Lerner and Tirole 2005). A highly restrictive license such as GPL requires the derived works (even if they just use the project instead of modifying it) to follow the same license. A less restrictive license (e.g. Berkeley Software Distribution, BSD) is less restrictive in the sense that the community contributors have less limitations in using or modifying the work. Regarding the determinants and consequences of OSS licenses, Lerner and Tirole (2005) suggest that restrictive licenses protect the project from "hijacking" by commercial software firms, which means the firm may add some proprietary code to the project and hijack it with an open source approach. Colazo and Fang (2009) find that restrictive licenses are associated with more developers, higher coding activity, and faster project speed. Stewart et al. (2006) argue that restrictive licenses keep the visibility of the developer's contributions and remain the customizability of the source code (so they don't need to pay for the software that comes from their own efforts). Belenzon and Schankerman (2008) find that developers are strongly sorted by the license type. Summarizing these studies in OSS literature, restrictive license could (a) promise customizability; (b) protect from hijacking; (c) sustains visibility; and (d) encourage identification.

Even though some authors argue that license would affect download (Stewart et al. 2006), we view it as a device to coordinate the efforts of the team and the community. Therefore, it should work through the community contributions. Because restrictive licenses are likely to attract better developer contributions, the effect of community contributions in restrictive projects may be higher. Thus release faster may help improve the quality of the project faster, and enhance its competition with other projects.

H3: *The moderate effect of community contributions is higher in projects with restrictive licenses.*

2.4 Network Effects and Product Diffusion

Besides the effect of release strategies, we expect that some other variables also have an influence on the market share of OSS projects. Two important factor that will affect the adoption of OSS project are network effects and product diffusion. Network effects indicate that the value of a product increases with the size of the network. It has been documented in many technology adoption studies (Kauffman et al. 2000; Zhu et al. 2006). The product diffusion literature finds that consumer adoption is influenced by the current user base and the number of potential users (e.g. Bass 1969). There are two diffusion processes in our context of OSS. One process happens on the category level. The users of a certain category of software may be increasing or shrinking. We control this by doing our analysis in each category. Another process happens on each product. As the quality of various software differs, the market share will increase or decrease accordingly. We will control this by decomposing the product diffusion effect into product fixed effect and its age. We control the network effect and product diffusion following Duan et al. (2009). The details will be provided in Section 4.

3 DATA AND METHODS

We gather the data of this study from SourceForge (SF), which is a major platform for OSS development, distribution and maintenance since 1999. It has been an important source of data for OSS studies (Belenzon and Schankerman 2008; Grewal et al. 2006; Lerner and Tirole 2005; Singh et al. 2011).

We developed a Java program to collect all the project and release information on SF. The data was collected in Nov, 2010. There were more than 157,720 projects hosted on SF at that time, about 80,000 of which have more than one download. We collected the release information of relatively popular projects with more cumulative download than the average download (27,804) at the time. Our sample contains 3995 OSS projects. We then constructed an unbalanced monthly panel for all the projects, and downloaded all the related data such as download, bug reports, patches contributions etc. in each month. The key variables and their descriptive statistics are presented in Table 1.

Variable	Description and Measure	Mean	Std. Dev
$log\_mktshare_{jt}$	log transformed market share	-6.30	2.03
$releases_{jt}$	number of releases	0.25	1.17
$releases2_{jt}$	squared term of number of releases	1.42	89.54
bug_fixed_{jt}	number of bug fixes since last release	0.98	3.58
$comm_{jt}$	sum of bugs reported and patches since last release	6.08	13.17
$strong\_comm_{jt}$	dummy variable, whether comm is above the mean in the category.	0.27	0.44
$weak\_comm_{jt}$	dummy variable, whether comm is below the mean in the category.	0.73	0.44
$highly\_restrictive_{j}$	dummy variable, whether project j has a highly restrictive license	0.62	0.48
$less\_restrictive_{j}$	dummy variable for whether project j has a highly restrictive license	0.38	0.48
$high\_cost_{j}$	dummy variable for whether project j is in a high upgrading/switching cost category	0.69	0.46
$user\_oriented_{j}$	dummy variable for whether project j is user-oriented	0.47	0.50
log_cu_{jt}	log transformed number of cumulative download	10.33	2.50
age_{jt}	days since the project registered on SF	1498.56	949.70
$age2_{jt}$	days since the project registered on SF squared	3147612.42	3293006.15

Subscripts *j* stands for project j, *t* stands for month t.

Table 1. Key Variables and Summary Statistics

We classify the restrictiveness of OSS licenses following the literature (Belenzon and Schankerman 2008; Lerner and Tirole 2005). We construct a variable $highly\_restrictive_j$ to indicate whether project *j* has a highly restrictive license such as GPL.

To calculate the market share of each project, we followed Duan et al. (2009) to define the market of an OSS project as the category it is listed in. We then choose the 15 most popular categories in our data[1]. For those projects that are listed in multiple categories, we choose the category that has the largest number of projects in it. Including market share as a dependent variable has several advantages. First, it removes unobservable influences such as weekend and holiday effects (Duan et al. 2009). Second, it controls for the trend of the whole category. Third, it addresses to a certain extent the concern of new installs and upgrades.

[1] These contains projects from categories of Games/Entertainment, Editors, File Sharing, Chat, Software Development, Integrated Development Environments (IDE), Systems Administration, Security, Firewalls, Database, Database Engines/Servers, Enterprise, Accounting, ERP, CRM.

We create a dummy variable $cost_j$ to represent the adoption cost of project *j* according to our classification[2]. We also use the audience of the projects as an indicator of adoption cost to confirm the result. End-user oriented programs usually have lower adoption costs than projects for developers and system administrators.

Since not all projects on SF use the tracker system (e.g. bug reports, patches, and feature requests, etc.) provided by SF, we use only projects that have those systems on SF. Our sample ended up containing 1092 projects from 15 categories.

As suggested by our conceptual framework, we are interested in the consequence of releases strategy of OSS projects. Technically, our estimation model is

$$log\_mktshare_{jt} = \alpha_j + \beta X_{jt} + \gamma Z_{jt} + \varepsilon_{jt} \qquad (1)$$

where Z_{jt} is a vector of control variables. Here we control for the network effect and product diffusion by adding the cumulative downloads, age of the project, and age-squared in Z_{jt}.

4 RESULTS

4.1 Effect of Release Strategy

We examine how release strategy relates to the download market share of OSS projects. The results are shown in Table 2, and the model-fit indices are shown in the bottom rows. With the effect of the fixed effect excluded, the dependent variables have R^2 of 39%, which is deemed acceptable. We then proceed to test each hypothesis by examining the magnitude and significance of the coefficients.

In Model (1), we test the effect of release frequency on download market shares while controlling the quality improvement efforts since last release (bug_fixed_{jt}). We find that release frequency is positively associated (p<0.01) with subsequent download market share, which supports our H1A. The significant negative coefficient (p<0.01) of $releases2_{jt}$ supports our H1B that there exists a curvilinear relationship between release frequency and subsequent download market share. Coefficients of $releases_{jt} \times high\_cost_j$ (*relXcost*) and $releases_{jt} \times user_j$ (*relXuser*) in model (2) and (3) support our H1C in that when adoption cost is high, the effect of release frequency decreases. Note that the dummy $user_j$ stands for enduser-oriented projects, which usually have lower adoption costs than software targeting developers or system administrators. The opposite signs of the coefficients of the two confirm our H1C.

	$log\_mktshare_{jt}$		
	(1)	(2)	(3)
L.releases	.095***	.15***	.084***
L.releases2	-6.4e-04***	-5.1e-04***	-7.0e-04***
L.bugs_fixed	.01***	.01***	.01***
L.log_cu	.45***	.45***	.45***
L.age	-.0019***	-.0019***	-.0019***
L.age2	1.9e-07***	1.9e-07***	1.9e-07***

[2] Among the 15 categories, Games/Entertainment, Editors, File Sharing, and Chat are classified as low cost categories.

L.relXcost		-.076***	
L.relXuser			.021**
Observations	89114	89114	89114
R-squared	0.3870	0.3876	0.3871
categories	1092	1092	1092
="* p<0.10 ** p<0.05 *** p<0.01"			

Table 2. Effect of Release Strategy and Adoption Cost

4.2 Community Contributions and OSS Licenses

We now turn to the influence of release frequency strategy under community contributions, most closely related to Hypotheses 2 to 3. The results are presented in Table 3. Controlling the level of community contributions, we find that release frequency have a positive effect ($p < 0.01$) under strong community contributions and negative effect ($p < 0.01$) under weak community contributions (Column (1) and (2) in Table 3), which supports our H2. This suggests that if the project has strong community contributions, releasing early and often may help with the download market share. Otherwise, it might have opposite impact.

	$log\_mktshare_{jt}$				
	(1)	(2)	(3)	Highly Restrictive	Less Restrictive
L.releases	.15***	.086***	.086***	.073***	.16***
L.releases2	-5.8e-04***	-5.8e-04***	-5.8e-04***	-4.8e-04***	-.0057***
L.bugs_fixed	.0064***	.0064***	.0064***	-3.6e-04	.017***
L.log_cu	.45***	.45***	.45***	.45***	.44***
L.age	-.0019***	-.0019***	-.0019***	-.002***	-.0016***
L.age2	1.9e-07***	1.9e-07***	1.9e-07***	2.2e-07***	1.4e-07***
L.weak_comm	-.073***				
L.relXweak	-.062***				
L.strong_comm		.073***	.072***	.083***	.036**
L.relXstrong		.062***	.023	.098***	.02
L.relXstrongXhr			.069**		
Observations	89114	89114	89114	55611	33503
R-squared	0.3879	0.3879	0.3880	0.3943	0.3839
projects	1092	1092	1092	697	395
="* p<0.10 ** p<0.05 *** p<0.01"					

Table 3. Effect of Community Contributions & License

Column (3) in Table 3 presents the results related to OSS license. We further interact the term $releases_{jt} \times strong\_comm_{jt}$ with the highly restrictive dummy, it turns out the total effect of $releases_{jt} \times strong\_comm_{jt}$ (relXstrong) is only significantly positive in projects with highly restrictive licenses. For projects with less restrictive licenses, this effect is not significant. This means even when the community contributions are strong, the motivations of the contributors still matter. Releasing fast seems to work when the license is restrictive. To confirm this, we run the regression in the subsample of projects with highly restrictive license. The results are presented in the last two columns of Table 3.

In the restrictive subsample, the moderate effect is significant while in the less restrictive subsample we do not see significant effect of the interaction term.

5 CONCLUSIONS

Using a unique longitudinal data set, we test the effect of release strategy on download market share of OSS projects, and investigated the moderation effect of adoption cost, community contributions, and OSS license. Our empirical results show the importance of community contributions and the license type of the project in shaping the effect of release strategies.

The software release strategy literature and the wisdom in the OSS community both indicate that fast and frequent releases would benefit the project. While we found that release frequency is positively associated with download market share, the relationship is not linear. Releasing too frequently may not only not product a significant positive benefit but also have a negative effect on the download market share.

We analyze the reasons from the demand and the supply sides of the OSS production process. From the demand side, users of OSS incur costs while adopting the new releases. If a project releases too frequently, the accumulated adoption cost may offset the benefit from quality improvement. This would leave the project in a worse position in competition.

From the supply side, we consider the co-creation of software product by the OSS team and the community. We find that community contributions have a moderate effect on the effect of release frequency. When community contributions are weak, frequent releases might have a negative moderate effect on download market share. Two reasons may be behind this result. First, as the release frequency increases, the project team has less time to incorporate the community contributions. Therefore the quality that the project gets from the community contributions actually decreases, which leads to lower market share. Second, as the team speeds up the release frequency, the community might not be able to keep up with the fast release iterations. Even though they still contribute, the contribution quality actually decreases. Therefore, releasing too fast may backfire.

We find that OSS license plays an important role in this process. The moderate effect of community contributions may depend on the license type of the project. Even when contributions are strong, releasing often may only help when the license is highly restrictive. We interpret this result from the motivations of contributors in OSS. When the license is highly restrictive (like GPL), the team has to contribute any derivatives back to the community if they incorporate the codes from the community. Therefore, the community members have no concerns of their codes been hijacked. If the license is less restrictive (like BSD), the community may have concerns and thus releasing fast might not help receivecommunity contributions.

Overall, our results contribute to a richer understanding of the OSS project management process. We expect that our work will result in more research in this exciting area and lead to improvements in open source software practice.

References

Arora, A., Caulkins, J. P., and Telang, R. 2006. Sell First, Fix Later: Impact of Patching on Software Quality. *Management Science* **52**(3) 465 –471.

August, T., Shin, H., and Tunca, T. 2009. Licensing Decisions and Competition for Integration and Services in Open Source Software. *SSRN eLibrary*

Bass, F. M. 1969. A New Product Growth for Model Consumer Durables. *Management Science* **15**(5) 215 –227.

Belenzon, S., and Schankerman, M. 2008. *Motivation and Sorting in Open Source Software Innovation*

Berry, S. T. 1994. Estimating Discrete-Choice Models of Product Differentiation. *The RAND Journal of Economics* **25**(2) 242–262.
Boudreau, K. J., and Lakhani, K. R. 2009, July 1. How to Manage Outside Innovation. *MIT Sloan Management Review* **50**(4) 69–73.
Brooks, F. P. 1995. *The Mythical Man-Month: Essays on Software Engineering, Anniversary Edition*, (Anniversary,) Addison-Wesley Professional.
Chengalur-Smith, S., and Sidorova, A. 2003. Survival of open-source projects: A population ecology perspective. *ICIS*International Conference on Information Systems.
Colazo, J., and Fang, Y. 2009. Impact of license choice on Open Source Software development activity. *Journal of the American Society for Information Science and Technology* **60**(5) 997–1011.
Dogan, K., Ji, Y., Mookerjee, V. S., and Radhakrishnan, S. 2011. Managing the Versions of a Software Product Under Variable and Endogenous Demand. *Information Systems Research* **22**(1) 5–21.
Duan, W., Gu, B., and Whinston, A. B. 2009. Informational Cascades and Software Adoption on the Internet: An Empirical Investigation. *Management Information Systems Quarterly* **33**(1) 23–48.
Fogel, K. 2005. *Producing Open Source Software: How to Run a Successful Free Software Project*, O'Reilly Media.
Grewal, R., Lilien, G. L., and Mallapragada, G. 2006. Location, location, location: How network embeddedness affects project success in open source systems. *Management Science* **52**(7) 1043–1056.
Hahn, J., Moon, J. Y., and Zhang, C. 2008. Emergence of New Project Teams from Open Source Software Developer Networks: Impact of Prior Collaboration Ties. *Information Systems Research* **19**(3) 369–391.
Von Hippel, E. 2001. Innovation by user communities: Learning from open-source software. *Mit Sloan Management Review* **42**(4) 82–86.
Von Hippel, E., and Von Krogh, G. 2003. Open Source Software and the "Private-Collective" Innovation Model: Issues for Organization Science. *Organization Science* **14**(2) 209–223.
IDC. 2009. *Worldwide open source services 2009–2013 forecast*. Technical Report (219260),
Ji, Y., Mookerjee, V. S., and Sethi, S. P. 2005. Optimal Software Development: A Control Theoretic Approach. *Information Systems Research* **16**(3) 292 –306.
Kauffman, R. J., McAndrews, J., and Wang, Y. M. 2000. Opening the" Black Box" of Network Externalities in Network Adoption. *Information Systems Research* **11**(1) 61–82.
Kessler, E. H., and Chakrabarti, A. K. 1996. Innovation Speed: A Conceptual Model of Context, Antecedents, and Outcomes. *The Academy of Management Review* **21**(4) 1143–1191.
Krishnan, V., and Ramachandran, K. 2011. Integrated Product Architecture and Pricing for Managing Sequential Innovation. *Management Science* **57**(11) 2040–2053.
Von Krogh, G., and Von Hippel, E. 2006. The promise of research on open source software. *Management Science* **52**(7) 975–983.
Von Krogh, G., Spaeth, S., and Lakhani, K. R. 2003. Community, joining, and specialization in open source software innovation: a case study. *Research Policy* **32**(7) 1217–1241.
Lakhani, K. R., and Wolf, R. G. 2007. Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects. *Perspectives on Free and Open Source Software* (1st ed, Vol. 1) 3–21.
Lerner, J., Pathak, P. A., and Tirole, J. 2006. The Dynamics of Open-Source Contributors. *The American Economic Review* **96**(2) 114–118.
Lerner, J., and Tirole, J. 2005. The Scope of Open Source Licensing. *J Law Econ Organ* **21**(1) 20–56.
Mcfadden, D. 1974. Conditional Logit Analysis of Qualitative Choice Behavior. *Frontiers in econometrics* 105–142.
Michlmayr, M., Hunt, F., and Probert, D. 2007. Release Management in Free Software Projects: Practices and Problems. *Open Source Development, Adoption and Innovation*, J. Feller, B. Fitzgerald, W. Scacchi, and A. Sillitti (eds.), (Vol. 234) 295–300.
OSI. 2011. The Open Source Definition. *Open Source Initiative*

Padmanabhan, V., Rajiv, S., and Srinivasan, K. 1997. New Products, Upgrades, and New Releases: A Rationale for Sequential Product Introduction. *Journal of Marketing Research* **34**(4) 456–472.
Ramachandran, K., and Krishnan, V. 2008. Design Architecture and Introduction Timing for Rapidly Improving Industrial Products. *Manufacturing & Service Operations Management* **10**(1) 149–171.
Raymond, E. 1999. The cathedral and the bazaar. *Knowledge, Technology & Policy* **12** 23–49.
Roberts, J. A., Hann, I. H., and Slaughter, S. A. 2006. Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the Apache projects. *Management Science* **52**(7) 984–999.
Setia, P., Rajagopalan, B., Sambamurthy, V., and Calantone, R. 2012. How Peripheral Developers Contribute to Open-Source Software Development. *Information Systems Research* **23**(1) 144–163.
Singh, P. V., and Phelps, C. 2012. Networks, Social Influence, and the Choice Among Competing Innovations: Insights from Open Source Software Licenses. *Information Systems Research*
Singh, P. V., Tan, Y., and Mookerjee, V. 2011. Network Effects: The Influence of Structural Capital on Open Source Project Success. *Management Information Systems Quarterly* **35**(4) 813–829.
Stewart, K. J., Ammeter, A. P., and Maruping, L. M. 2006. Impacts of License Choice and Organizational Sponsorship on User Interest and Development Activity in Open Source Software Projects. *Information Systems Research* **17**(2) 126–144.
Subramaniam, C., Sen, R., and Nelson, M. L. 2009. Determinants of open source software project success: A longitudinal study. *Decision Support Systems* **46**(2) 576–585.
The Economist. 2009, May 28. Open-source software in the recession: Born free. *The Economist*
Watson, R. T., Boudreau, M. C., York, P. T., Greiner, M. E., and Wynn Jr, D. 2008. The business of open source. *Communications of the ACM* **51**(4) 41–46.
West, J., and Lakhani, K. R. 2008. Getting clear about communities in open innovation. *Industry and Innovation* **15**(2) 223–231.
Wooldridge, J. M. 2010. *Econometric Analysis of Cross Section and Panel Data*, (second edition,) The MIT Press.
Zhu, K., Kraemer, K., Gurbaxani, V., and Xu, S. X. 2006. Migration to Open-Standard Interorganizational Systems: Network Effects, Switching Costs, and Path Dependency. *Management Information Systems Quarterly* **30**(Special Issue) 515–538.
Zhu, K., and Zhou, Z. 2012. Lock-In Strategy in Software Competition: Open-Source Software vs. Proprietary Software. *Information Systems Research* **23**(2) 536–545.