PROMOTING THE PENGUIN: WHO IS ADVOCATING OPEN SOURCE SOFTWARE IN COMMERCIAL SETTINGS?

Oliver Alexy

TUM Business School

Technische Universität München

Arcisstr. 21, D – 80333 Munich, Germany

phone: +49-89-28925741, fax: +49-89-28925742, email: alexy@wi.tum.de

Joachim Henkel

TUM Business School

Technische Universität München

Arcisstr. 21, D – 80333 Munich, Germany

phone: +49-89-28925741, fax: +49-89-28925742, email: henkel@wi.tum.de

2nd affiliation: Center for Economic Policy Research (CEPR), London

PROMOTING THE PENGUIN: WHO IS ADVOCATING OPEN SOURCE SOFTWARE IN COMMERCIAL SETTINGS?

Acknowledgements

We thank Martin Bichler, Linus Dahlander, Lars Frederiksen, Simone Käs, and Francesco Rullani as well as conference and seminar participants at Boston University, ETH Zürich, EURAM, Harvard Business School, and Technische Universität München for helpful comments.

Author Biographies

Oliver Alexy holds a degree in management of information systems from the University of Regensburg, Germany. A PhD student at TUM Business School, Technische Universität München, he is engaged in research into the incentives that lead corporations and individuals to participate in, and the processes followed by, open source software development and open innovation.

Joachim Henkel is a professor at TUM Business School, Technische Universität München. He holds the Dr. Theo Schöller Chair in Technology and Innovation Management. After receiving his Ph.D. from the University of Mannheim in 1997, he worked for two years with the consulting firm Bain & Company. He is currently pursuing research into open source software, open innovation processes, patent infringement, and venture capital, and has been a visiting scholar at University College London and the Massachusetts Institute of Technology.

PROMOTING THE PENGUIN: WHO IS ADVOCATING OPEN SOURCE SOFTWARE IN COMMERCIAL SETTINGS?

ABSTRACT

Most firms that use or develop software today face the questions of whether and how to engage in open source software. Yet, little is known about the process of OSS adoption and diffusion within corporations. Guided by the theoretical frameworks of Rogers (innovation diffusion) and Davis (Technology Acceptance Model), we develop a model of how job function influences individuals' proclivity to support their employers' adoption of OSS and OSS practices. We argue that job function determines which tasks in the software development process are part of an individual's daily routine, and that different tasks are differentially affected by OSS.

Our study is based on interviews with 25 individuals and a large-scale survey distributed to 249 participants in the telecommunications department of a multinational company. The results, although consistent with theoretical considerations, are nevertheless surprising. Distinguishing between developers, software testers, software architects, project managers, and managers, we find greater involvement in OSS activities to be favored most strongly by software testers, followed by software architects and managers. Excepting project managers, developers, despite having the most experience with OSS, are the *least favorably* disposed to greater corporate engagement in OSS. A corporation interested in adopting OSS and open innovation processes should thus take into account the job function-related incentives of each individual as well as various organizational factors. More generally, we propose that models developed to predict IT adoption behavior be extended to account for the ways in which individual adopters interact with the innovation at hand, which we maintain will be determined largely by their job functions.

Keywords: open source software; open innovation; adoption; diffusion

PROMOTING THE PENGUIN: WHO IS ADVOCATING OPEN SOURCE SOFTWARE IN COMMERCIAL SETTINGS?

INTRODUCTION

Commercial use of open source software (OSS) passed through phases of curiosity, hype, and disillusionment before arriving at pragmatism (Driver et al. 2005a). Although the once dazzling stock market valuations of companies such as Red Hat and VA Linux have settled at more sober levels, OSS has nevertheless gained so strong a foothold in commercial settings that removing it would occasion the breakdown of many firms' IT infrastructures (Driver et al. 2005b; Goulde 2006), and even, in the electronics industry, of numerous products. Many researchers having consequently addressed the issues of OSS business models,[1] collaboration between firms and the OSS community,[2] and inter-firm OSS-based collaborative innovation processes,[3] the pros and cons of commercial OSS engagement are now quite well understood.

Little, however, is known about why and how firms become OSS adopters. Within firms, who promotes OSS, and do particular job functions and personal characteristics favorably dispose individuals to lobbying for its adoption? Anecdotal evidence (Henkel 2004; Moody 2001; Raymond 2001a; Raymond 2001b) of the importance of grassroots initiatives by developers is supported by survey results that find that software professionals tend to champion OSS (Henkel 2006a), but systematic evidence is lacking. Because active engagement in OSS might be vital to firms in terms of efficiency gains, standard setting, defining the rules of competition, and responding to competitive threats generated by OSS, IT profession-

[1] Bonaccorsi and Rossi 2006b, Dahlander 2005, Feller and Fitzgerald 2001a, Grand, von Krogh, Leonard and Swap 2004, Hecker 1999, Lerner and Tirole 2002, Raymond 2001a, Raymond 2001b, West 2003, Wichmann 2002.

[2] Bessen 2005, Bonaccorsi, Giannangeli and Rossi 2006a, Dahlander and Magnusson 2005, Dahlander and Wallin 2006, Henkel 2006a, Osterloh and Rota 2005, Shah 2006.

[3] Henkel 2006b, Nuvolari 2001, West and Gallagher 2006.

als' disposition towards and willingness to promote its adoption are highly relevant. Yet, understanding of the process of adoption and diffusion within firms remains largely a black box.

In the interest of making it less so, we dissect the question of who advocates OSS in commercial settings in a way that distinguishes three levels of engagement. Specifically, we ask who advocates *using* existing OSS, who advocates *contributing* to existing OSS projects, and who advocates *releasing* proprietary software as OSS? Note that levels of contributing and releasing amount to a process innovation in software development, which would imply that employees influence organizational structure and processes. As our main research question, we ask whether particular job functions and personal characteristics favorably incline an employee towards OSS.

Because we attempt to analyze through these questions the alleged grassroots aspect of corporate OSS adoption, we exclude OSS initiatives by top management in order to focus explicitly on the level of IT professionals. Guided by a framework based on the Technology Acceptance Model (TAM) (Davis 1989; Davis et al. 1989) and concept of innovation diffusion (Rogers 2003), we develop a model that links individuals' attitudes towards commercial OSS adoption to their job functions. We maintain that job function determines an individual's tasks (and thus daily routines), and that different tasks are differentially affected by the introduction of OSS. Software professionals whose daily routines are more strongly or more negatively affected by its introduction are expected to be less favorably disposed to the adoption of OSS. Our findings further suggest that the TAM and similar models be extended in the manner suggested by Venkatesh (2006) to take into account the adopter's job function, or, more generally, the way in which the adopter interacts with the innovation at hand.

We pursued answers to these questions over a period of a year and a half in the telecommunications department of a multinational corporation that is not among the early and outspoken corporate proponents of OSS (e.g., IBM). The company so far has no company-wide OSS policy and is in a relatively early phase of commercial adoption and diffusion of

OSS. Thus, not being prejudiced by existing corporate strategy governing OSS, the company was well-suited to our study, which explored the role of employees in OSS adoption through interviews with 25 individuals and a survey involving 249 participants.

Our main findings regarding the impact of respondents' job functions are consistent with theoretical considerations, yet still rather surprising. Among developers, software testers, software architects, project managers, and managers, software testers are generally most favorably disposed to increasing corporate involvement in OSS activities, followed by software architects and managers. Excepting the (small) group of project managers, developers, despite having the most experience with OSS, were the *least favorably* disposed to greater corporate engagement.

Since developers represent both the basic level of software development and the largest of the studied groups, our results lead us to question anecdotal evidence that OSS adoption, at least when it goes beyond the use of existing OSS, is generally driven by a broadly supported grassroots movement. This finding has consequences for companies' management of OSS engagement. Given a 46% probability that a randomly selected developer from our sample is neutral (26%) or even unfavorably disposed (20%) towards releasing proprietary software as OSS, developers should probably be given the opportunity to self-select into OSS-related projects. We can make the same recommendation for project managers, and, more generally, that in the context of corporate OSS engagement, the job function-related incentives of each affected individual need to be considered.

IT professionals are nevertheless, *on average*, "somewhat" positive about their employer increasing its OSS activities. Differentiating by type of activity, we find "using existing OSS more often" to be most strongly supported, followed by "contributing to OSS projects" and "releasing proprietary software as OSS." A favorable disposition to all types of OSS activity is strongly dependent on, apart from an individual's job function, prior exposure to OSS, identification with the OSS community, acceptance of reciprocity norms, and age.

The remainder of the paper is organized as follows. We first compare OSS development with proprietary closed source software (PCSS) development. We then provide the theoretical background and develop our hypotheses for corporate OSS adoption and development. Our data and methods are presented next, followed by the results of the study. Finally, implications for theory and practice are derived and limitations of the findings discussed.

BACKGROUND: PCSS VERSUS OSS DEVELOPMENT

Software Development Process

In PCSS development, firms build almost exclusively on their employees' knowledge. Developers write source code according to use-cases specified by software architects without any significant interaction with the outside (see Figure 1). Before a product is released, product testing ascertains that it adheres to both initial requirements and company quality standards (Jones 2003; Lehman 1980; Royce 1987; Senyard et al. 2004). Outside influence is limited to the requirements articulated by the customers (which might be internal to the firm) at the beginning of the development process, licensed-in commercial third-party software, and beta testing towards the end of the development process. This description is clearly of the waterfall model of software development, which, although more advanced models are in use by many firms, continues to be widely employed (Cusumano et al. 2003; Jones 2003) . It was the model of choice in the firm we studied.

--- Insert Figure 1 about here ---

--- Insert Figure 2 about here ---

Boundaries between a firm and its environment become more permeable with OSS development (see Figure 2). Consider first the case of a firm that releases internally developed software as OSS in order to launch a public OSS project. The first release, usually done in the same way as in a PCSS environment, is typically a prototype that is good enough to solve the initial problem (Senyard et al. 2004). But thereafter development changes significantly. Whereas PCSS would enter the maintenance phase, which frequently consumes more than half of all development resources (Banker et al. 1991), outsiders are now encouraged to report bugs, suggest new features, and contribute source code to further improve the software (see Figure 3). Additional source code that meets a project's quality standards is made available to the community and included into subsequent releases. Substituting OSS for PCSS development has been shown to reduce maintenance costs dramatically (Lakhani et al. 2003; Raymond 2001a; Senyard et al. 2004), and significant changes in the development process are observed, particularly in the daily routines of developers, project managers, and managers (see Figure 2).

--- Insert Figure 3 about here ---

The above describes a particular case of corporate OSS engagement. More generally, software development can embrace OSS and OSS practices to varying degrees. For purposes of this paper, we distinguish three levels of corporate OSS activity: using existing OSS, contributing to existing OSS projects, and releasing proprietary software under an OSS license.[4]

Existing OSS is widely used both within and outside the IT industry, and has often become an integral part of corporate software architectures and even commercial software offerings. Prominent examples include the Linux operating system, Apache web server, and Eclipse programming environment (Driver et al. 2005b; Goulde 2006; Grand et al. 2004; Varian et al. 2003). In these cases, OSS is basically treated just as any other third-party software,

[4] Grand, von Krogh, Leonard and Swap 2004 use a related classification that distinguishing four levels: using existing OSS; adapting and extending OSS; acting as core OSS developers; and providing development and distribution services related to OSS projects (thus running an OSS compatible business model). The first levels in this and our classification are identical. The second and third levels in this classification largely correspond to "contributing to existing OSS" in our classification. The top levels differ because Grand, von Krogh, Leonard and Swap 2004 focus on "pure play" OSS firms, whereas our focus is on established corporations.

mostly without or with only limited modifications.[5] Only one-way interaction between the company and the environment takes place, such that clear boundaries between the two exist.

Yet, there are instances in which corporations fix existing bugs or adapt the software to their needs, and contribute the modified source code back to the OSS project. The latter typically happens selectively, that is, modifications are contributed back only when this is deemed advantageous to the company (Henkel 2006b). Giving back conforms to the idea of reciprocity promoted by OSS and Free Software proponents (Bonaccorsi et al. 2006b; Raymond 2001a; Shah 2006), but typically contradicts established corporate policies that dictate that innovation take place within firms' boundaries and that intellectual property leave the corporation, if at all, only under a licensing contract. It is thus a significant step, especially for established firms, to move from using OSS to contributing to OSS projects. The greater the extent of such activities, the more blurred boundaries are likely to become between a company and its environment (Chesbrough 2003; West et al. 2006). Regular, two-sided interaction with the outside environment is rare apart from high-profile cases such as IBM's engagement in Linux. Companies that contribute internally developed source code are clearly focused on leveraging it externally with the aim, for example, of improving their reputation or influencing a standard.

Releasing proprietary software under an OSS license, as illustrated at the beginning of this section, is an even more radical departure from established practice. Doing so to initiate co-development with the OSS community completes the transition from closed to open, collective, or private-collective innovation (Allen 1983; Chesbrough 2003; Osterloh et al. 2005; von Hippel et al. 2003). As the term co-development suggests, releasing proprietary software as OSS implies a process innovation: team organization, project management, and coding must be organized and executed differently than in the conventional mode of development

[5] Franke and von Hippel 2003, found that only 19% in a sample of 131 Apache webmasters had modified the web server source code, even though they likely were expert users. In general, this percentage will be even lower.

(Grand et al. 2004; von Hippel et al. 2003; West et al. 2006). Moreover, the full bidirectional interaction that follows is likely to blur or even erase boundaries between a corporation and its environment as relate to a project and the knowledge associated with it.

Advantages and Disadvantages of Corporate OSS Engagement

Advantages attendant on *using OSS* include savings on licensing fees and development effort, reduced lock-in, shorter time-to-market, and higher quality and performance (Goldman et al. 2005; Hecker 1999; Raymond 2001a).

Contributing code to existing projects might yield both technical and marketing benefits. Technology-wise, the company may influence the standard version of the software, thereby eliminating the need to redo the changes for each update of the OSS and to take into account possible incompatibilities, new security issues, etc. Moreover, others might be induced to contribute improvements to the contributed code. From a marketing point of view, contributed code that is well received can boost the contributor’s reputation and visibility (Behlendorf 1999; Gruber et al. 2006; Hecker 1999; Henkel 2004; Koenig 2004; Raymond 2001a; Raymond 2001b). *Releasing proprietary software as OSS* can yield much the same benefits but to a greater magnitude (Goldman et al. 2005; Lakhani et al. 2003; Shah 2006). Moreover, new business models such as the sale of complementary goods or services might become viable, and commoditizing a particular layer of the software architecture might help to shift competition to an area in which the corporation has competitive strengths (Raymond 2001b; West 2003).

The foregoing benefits must be weighed against a number of potential drawbacks. The principal disadvantage of *using* OSS is that if the project that produced the software loses momentum or goes in an undesired direction, the burden of development shifts, unexpectedly, back to the corporate user. *Contributing to an OSS project* and *releasing PCSS as OSS* expose the contributor to start-up costs associated with modularizing and sanitizing the source code as well as to requirements that additional resources be allocated to support (Hecker 1999; Henkel 2004; Raymond 2001b). The most obvious risk of opening proprietary software is, of course, the loss of intellectual property and, consequently, of competitive advantage.

The net benefit of deliberate and selective participation in OSS activity has been found in many instances to be positive (Bessen 2005; Bonaccorsi et al. 2006a; Dahlander et al. 2005; Dahlander et al. 2006; Henkel 2006a; Osterloh et al. 2005; Shah 2006), but potential benefits and drawbacks must nevertheless be weighed and addressed in a preceding business case (West et al. 2006).

RESEARCH QUESTIONS AND HYPOTHESES

In this section, we develop a theoretical framework for describing the drivers of IT professionals' attitudes towards corporate OSS engagement. The framework is built on a three-step model of organizational adoption that distinguishes (1) initiation of the innovation from (2) the organization's (i.e., management's) adoption decision and (3) diffusion through individual employees' adoption decisions (Damanpour 1991; Leonard-Barton et al. 1988; Rogers 2003). We then derive hypotheses regarding the influence of respondents' job functions on their attitude towards corporate engagement in OSS, using the Technology Acceptance Model (TAM) (Davis 1989; Davis et al. 1989) as a guideline. To control for effects other than those associated with job function, we additionally employ the concept of diffusion of innovations (DOI) (Rogers 2003).

The Process of Organizational Adoption

Organizational innovation is defined as an organization's adoption of a new idea or behavior (Daft 1978; Damanpour 1991; Damanpour et al. 1984). The adoption process can be segmented into initiation, adoption, and diffusion (Damanpour 1991; Kwon et al. 1987; Leonard-

Barton et al. 1988; Rogers 2003; Zmud 1982). Usually triggered by an organizational need or new technology, the process is initiated by scanning the organization for opportunities or solutions to problems (Cooper et al. 1990). Adoption is the favorable decision by management to commit the required resources to an identified solution, diffusion the dissemination and acceptance of that solution (i.e., the innovation) by members of the organization (Cooper et al. 1990; Rogers 2003).

Following Daft (1978), organizational innovations can be classified as technical and administrative. Technical innovations "occur in the technical system of an organization and are directly related to the primary work activity of the organization. […] Administrative innovations are defined as those that occur in the social system of the organization" (Damanpour et al. 1984). Although more specific adaptations exist (e.g., Swanson 1994), this general distinction is well suited to IT innovations (Zmud 1982; Zmud 1984).

How each type of innovation is introduced to an organization is a central issue in the present context. As organizational structure is set by management (Burgelman 1983) and modified by means of administrative innovations, the latter are likely to be introduced in top-down fashion (Daft 1978). Technical innovations, on the other hand, are more likely to be bottom-up initiatives because individuals disposed to adopting technical innovations often will do so even without management support (Leonard-Barton et al. 1988). Such activity might culminate in a bottom-up adoption process in which management only *subsequently* decides that the corporation as a whole should adopt the technology, long after its employees have decided to do so (Grover 1997; Swanson 1994). For both technical and administrative innovations, however, the most effective path of introduction has been found to be a two-step process that begins top-down, with management deciding on adopting an innovation overall, and employees then making their personal adoption decisions individually (Cooper et al. 1990; Grover 1997; Swanson 1994; Zmud 1982).Thus, whether adoption begins bottom-up or top-down, successful diffusion of an innovation ultimately relies on the individual adoption decisions of employees (Agarwal 2000; Zmud 1984).

Because anecdotal evidence that associates OSS with grassroots adoption suggests an inherent potential for bottom-up organizational innovation (Henkel 2004; Moody 2001; Raymond 2001a), it is important to understand what motivates individual employees to support such adoption. As we have seen, employees can drive corporate OSS engagement in two ways, (1) by opting, themselves, for OSS in decisions that are in their own discretion and (2) by lobbying for broader, organization-wide OSS engagement.

However, individuals with different job functions will, in general, have different attitudes towards the adoption of new information technology (Agarwal et al. 1999; Baldridge et al. 1975), and in particular towards engagement in OSS. Specifically, employees' attitudes will reflect their perception of how adoption will affect the daily routines associated with their role in the software development process. Two questions thus become vitally important, (1) who can be expected to lobby for greater corporate engagement in OSS, which is to say, where is adoption initiated and diffusion supported, and (2) who is likely to oppose engagement in OSS development and pose a bottleneck to its introduction? Given that OSS practices affect all steps in the software development process, the weakest link in this chain, that is, the least supportive job function, limits the overall effectiveness of OSS development.

Modeling Individuals' Adoption Decisions

To ensure that we are, in fact, measuring the effect of individuals' roles, we control for other elements that might influence their adoption decision. We conducted our study within one firm in order to create a constant environment by eliminating such external factors that strongly influence adoption as firm size, diversity, slack resources, IT application portfolio, specialization, and professionalism (Damanpour 1991; Swanson 1994). We model individuals' attitude towards corporate OSS engagement, as an antecedent to both their individual adoption decisions and their lobbying for OSS, using the two most widely accepted concepts in the IS literature (Gallivan 2001), namely, TAM and DOI. Our research model follows the approach suggested by Kwon and Zmud (1987) in using variables from TAM and DOI to represent characteristics of the individual, task, and organization.[6] The resulting research model is as depicted in Figure 4.

--- Insert Figure 4 about here ---

Our primary interest is in measuring the effect of individuals' job functions on their attitudes towards their employers' further engagement in OSS development. This largely corresponds to employing as the dependent variable the TAM's "attitude towards using" [26, 27]. "Using" here refers to all three levels of OSS engagement (not just to "using existing OSS"), and is not restricted to (but comprises) using by the individual. We deliberately employ "attitude towards using" and not "intention to use" as the dependent variable, since (1) the latter can not be extended to use by others and (2) is not really applicable to the third level of corporate OSS engagement, "releasing proprietary software as OSS." Also, we aim at analyzing drivers of both individual OSS adoption and lobbying for OSS, and "attitude towards using" seems more appropriate as a determinant of lobbying than "intention to use." [7]

The two main factors that drive an individual's attitude towards a new a technology, according to the TAM, are perceived usefulness and perceived ease of use. Attitude towards using a new technology affects the behavioral intent to use it, which, in turn, determines whether one becomes a user. Perceived usefulness is highly influential on both attitude towards using and behavioral intent to use. Perceived ease of use has an indirect effect on both

[6] Kwon and Zmud build their model on Rogers' DOI and extend it by integrating task and environmental characteristics. As described above, we have deliberately excluded (external) environmental characteristics from our study by focusing on one firm. Task-related characteristics are captured by individuals' job functions.

[7] As an indicator for this, a respondent's attitude towards releasing proprietary software as OSS is highly correlated ($p < 0.01$, $r_{sp} = 0.22$) with the number of current software products of the corporation that this respondent suggested, in an open question, as potential candidates for a release as OSS. The act of suggesting constitutes lobbying for OSS, even if with a limited effort. Similarly, the attitude towards using OSS is highly correlated ($p < 0.01$, $r_{sp} = 0.23$) with the extent to which the individual was currently trying to use OSS in corporate software projects.

through perceived usefulness: an individual perceives a new technology as more useful if it previously has been identified as easy to use (Davis 1989; Davis et al. 1989).

We employ the TAM in the following section to derive perceived usefulness and perceived ease of use from job function (and some control variables). In the model as well as the subsequent econometric analysis, perceived usefulness and perceived ease of use are captured by the job function variables, such that the TAM items themselves do not appear as explanatory variables. Deviating in this way from the standard TAM approach allows us to directly juxtapose the analysis of attitudes by job *groups* with a multivariate analysis controlling for other characteristics of the respondent.

Following Rogers, we assume the diffusion of an innovation to depend on the innovation itself, communication and communication channels, time, and social systems (Rogers 2003). Rogers' measures for the first of these dimensions, "innovation," include relative advantage, compatibility, complexity, trialability, and observability. The greatest importance attaching to the first two measures (Rogers 2003; Tornatzky et al. 1982), we decided to drop the other three, and because relative advantage refers to the adopter's *perception* and is thus highly dependent on job function, relative advantage was not measured as an independent variable. Instead, we argue that the latter is captured by an individual's job function.

Job Functions

For purposes of this paper, we distinguish five job functions in the software development process. Ordered (roughly) by their sequence in the waterfall model, these are software architects, developers, software testers, project managers, and (general) managers. We describe each of these roles, in particular, with respect to how it would likely be affected by increased corporate engagement in OSS development.

Software architects translate user needs into a set of (high-level) software requirements and subdivide these into subsystems that are coded by developers (Bass et al. 2003;

Royce 1987). As their title implies, they determine product architecture, that is, the linkages among different (possibly modular) components of a software system (Baldwin et al. 2000; Henderson et al. 1990).

Even in PCSS development, software architects interact with outsiders, receiving customer input and selecting and integrating third-party modules into the design of proprietary software systems. Perceived ease of use with regard to outside engagement in the OSS process should thus be high for this job function, which might be expected to view existing OSS as just another external source of building blocks. Perceived usefulness should also be high given that (1) the availability of OSS expands design choices, and (2) the suitability of OSS for a particular purpose can more easily be assessed owing to availability of the source code. Access to the source code also facilitates and reduces the cost of adaptations, and contributing code to influence the architecture of an OSS project (Goldman et al. 2005; Lakhani et al. 2003) could make the contributor party to the eventual establishment of an industry standard (Bonaccorsi et al. 2006a; Henkel 2004). We thus expect software architects' attitudes towards corporate OSS engagement to be rather positive.

For the *developers*, whose job it is to code the subsystems that comprise a system design, the changes entailed by engagement in OSS are perhaps best described by the terms not-invented-here, OSS development style, and coding for re-use. The share of external software with which a developer works will typically be much higher in OSS than in PCSS development (see Figure 2). Re-using existing OSS or accepting external contributions to a corporate OSS project, instead of writing one's own code, may be subject to the Not-Invented-Here (NIH) syndrome (DiBona 2005; Katz et al. 1982), and the potential for internal development to be scaled down by using existing OSS might be perceived to put developers' jobs in jeopardy. At least in the short run, however, the potential for re-using OSS to simplify and speed up their work should be viewed positively by developers.

Developers unfamiliar with OSS will, of course, be unfamiliar with its development method, design principles, and meritocratic style (Scacchi 2004). The need to interact intensely with the outside world and to aggregate as well as write source code and, possibly, maintain it in a public project will be new territory even for developers who have used external components in PCSS development, as communication with the originators of the source code would likely not have been required. Because they occur concurrently, these activities demand a great deal of coordination. New responsibilities also accrue to developers, who become to varying degrees moderators and managers of users and contributors (Kogut et al. 2001; Senyard et al. 2004). Adapting to these changes takes considerable effort. Developers who cannot make the transition to this model (due either to a general lack of skills or to an inability to see how it relates to their existing knowledge) or who feel pressured to do so by outside forces will be inclined to evaluate negatively, and exhibit a negative attitude towards their employer's engagement in, OSS use and development (Deci et al. 1985; Ryan et al. 2000). On the other hand, being able to communicate with outside experts might help to resolve problems their companies might be having difficulty solving on their own (Constant et al. 1996; Lakhani et al. 2007) or identify problems that might otherwise have been overlooked.

The importance of modularity and coding for reuse are far greater in OSS than in PCSS development (Bonaccorsi et al. 2003; Goldman et al. 2005; Raymond 2001a; Senyard et al. 2004),[8] and although engaging in OSS might generate positive signaling effects and enhance peer recognition (Lerner et al. 2002; McLure Wasko et al. 2005), it also exposes internally generated code to greater scrutiny by external as well as internal experts. Our interviews revealed that especially less skilled developers fear losing face when mistakes are now

[8] More modern software development methods such as the spiral model, extreme programming, and agile methodologies by their nature rely more heavily on modularization and coding for reuse than the classical waterfall model and V-model and, thus, are more "compatible" with OSS development (see, for example, Feller and Fitzgerald 2001a, Goldman and Gabriel 2005, and Hang, Hohensohn, Mayr and Wieland 2004).

more easily found and attributed. We thus expect that, relative to software architects, developers will exhibit a less positive attitude towards OSS overall and, in particular, a strongly negative attitude towards contributing to existing OSS projects and releasing proprietary software as OSS.

Testing being a highly routinized task executed by technical specialists supported by dedicated software applications, its nature is not substantively different for OSS and PCSS development. In their role as end control, *software testers* scan code for semantic and syntactic errors, which they report back to the developers (Royce 1987). Even with PCSS development, external actors are engaged in so-called beta testing, whereby selected users are provided with a copy of a release candidate of a software program for testing purposes. That OSS development introduces for testers (as for software architects) no fundamentally new activities should translate into high perceived ease of use, and that it avails testers access to a community of developers and users and concomitant significant increase in frequency of testing and numbers of test designs should translate into high perceived usefulness. Moreover, incorporating OSS components that likely have been heavily scrutinized by the OSS community might be expected to reduce the number of bugs (Raymond 2001a), and no appreciable degree of employee redundancy is indicated as final quality inspections will still need to be performed before a product is released. Testers are thus expected to generally exhibit positive attitudes towards their employers' engagement in OSS.

Project managers plan, execute, and monitor software projects, coordinate tasks and personnel, allocate resources, and set milestones (Kirsch 2000). Both in PCSS and OSS development, they perform boundary-spanning tasks involving bringing together organization members with different backgrounds performing different tasks (Tushman 1977; Tushman et al. 1981). That OSS development entails working with an additional, external boundary increases uncertainty in the coding realm (Goldman et al. 2005). How is a project manager who doesn't even know who is working on it to set milestones for and allocate resources to a project? The "kindness of strangers" (Constant et al. 1996) might be helpful, but cannot be taken for granted. The resulting uncertainty introduces risk and a concomitant need for greater coordination, which increase the project manager's workload (Kirsch 2000).

Our interviews further revealed that contributing source code to existing OSS and releasing proprietary software as OSS upon completion of a project adds work that yields no immediate and clearly visible benefits for the project manager or the project. Potential benefits will be extremely difficult to quantify and project managers will scarcely be evaluated on them. The transformation process for projects not planned as OSS from the outset, moreover, can be costly (Hecker 1999; Henkel 2004). Perceived usefulness and perceived ease of use might thus be expected to be rather low, and attitude towards corporate engagement in OSS rather negative, for project managers.

(General) managers define and enact corporate strategy. They decide which projects to pursue and allocate the requisite resources (Burgelman 1983). Their attitude towards OSS development might thus be expected to be much the same (albeit less intense, not being directly responsible for meeting deadlines) as that of project managers. But the need of managers, "diversified" by oversight of a plurality of projects, to think beyond individual projects mitigates the effects of risk-aversion. Long-term benefits such as reuse of OSS adapted to local needs should thus offset negative short-term effects, and the benefits of releasing proprietary software as OSS, being generally strategic in nature, might be expected to be recognized by managers (Dahlander 2005; Feller et al. 2001b; Grand et al. 2004; Hecker 1999; Raymond 2001a; Raymond 2001b; West 2003). They must nevertheless weigh such benefits against possible disadvantages such as loss of competitive advantage consequent to relinquishing intellectual property or the risk of forking. Managers' attitudes towards corporate engagement in OSS might be expected to be more positive than project managers' and less positive than software architects' and testers'.

We find the effect of job function on attitude to depend to some extent on type of OSS engagement (i.e., using, contributing, or releasing), but no indication that ranking of job functions with respect to affinity for OSS should be contingent on the type of engagement. We thus arrive, for each type of engagement, at the following predictions regarding the impact of job function on attitude towards OSS: software architects and testers should be most favorably disposed to OSS, followed by managers and developers. Based on our theoretical discussion we are unable to predict whether and how the attitudes of software architects and testers, and managers and developers, might differ. Project managers, finally, should be least favorably disposed to OSS. These findings give rise to the following hypotheses (letters in square brackets indicate to which groups each hypothesis refers).

H1[TM]: Testers' attitudes towards OSS are more positive than managers'.

H2[TD]: Testers' attitudes towards OSS are more positive than developers'.

H3[TP]: Testers' attitudes towards OSS are more positive than project managers'.

H4[AM]: Architects' attitudes towards OSS are more positive than managers'.

H5[AD]: Architects' attitudes towards OSS are more positive than developers'.

H6[AP]: Architects' attitudes towards OSS are more positive than project managers'.

H7[MP]: Managers' attitudes towards OSS are more positive than project managers'.

H8[DP]: Developers' attitudes towards OSS are more positive than project managers'.

As indicated in the derivation of our hypotheses, we expect differences between the roles to become more significant with increased OSS engagement.

Control Variables

Compatibility. Compatibility in the context of innovation diffusion might, but does not necessarily, imply compatibility in a technical sense. More important is that compatibility encompasses the degree to which an innovation is coherent with existing norms and premises (Rogers 2003), which, in turn, at least in part, determines its perceived ease of use (Davis

1989; Venkatesh 2000). In our context, goals of the OSS community (e.g., freedom and reciprocity) constitute such norms and premises. Identification with these goals probably drives some, but not most, individuals' engagement in OSS development (Hars et al. 2002; Lakhani et al. 2005; McLure Wasko et al. 2000; McLure Wasko et al. 2005). For these individuals, the community's norms represent unifying aspects that constitute the underlying philosophy of the OSS movement (Hertel et al. 2003; Stewart et al. 2006); identification with this philosophy might motivate "giving back" to the community by contributing to an OSS project or releasing proprietary software as OSS (Venkatesh 2000).

Organizational characteristics. It having been shown that people who belong to a group are likely to take actions based on a frame of reference created by the group (Merton et al. 1949), innovation diffusion might be expected to be influenced by embeddedness in *social systems* (Granovetter 1985; Rogers 2003). Whether individuals are favorably or unfavorably disposed to OSS engagement tends to be influenced by their immediate environment. Such peer influence is also important in the context of interaction and knowledge exchange with others both within and outside an organization (Granovetter 1985; Granovetter 1973; Katz et al. 1982; Katz et al. 1985; Rogers 2003). This study combines social systems and *communication*—as a mediator of peer influence, a means to bridge gaps in compatibility, and a valuable source of innovation in itself (Chakrabarti et al. 1977; Ebadi et al. 1984; Katz et al. 1985)—to measure the influence of *organizational factors* on individuals' attitudes towards corporate OSS engagement,

Individual characteristics. Kirton proposed an Adoption-Innovation index (KAI) to measure creative style or innovativeness, which he maintains is an important determinant of how a person copes with change. A person who scores low on this index (i.e., who prefers no, or at most very little, change to the status quo) is classified as an "adopter" rather than an "innovator" (Kirton 1976; Kirton 2003). Using OSS, contributing to existing OSS projects, and releasing proprietary software as OSS, because they involve considerable change, should be more likely to be viewed favorably by individuals who score higher on the index. We might also expect individuals who have previously been engaged in OSS to view increased corporate engagement in OSS more favorably (Davis 1989; Davis et al. 1989).

DATA AND METHODS

Our study was conducted in the telecommunications department of a multinational electronics company in which software development plays a key role. Hence, most (if not all) employees were involved in some way with software development in the course of their everyday work. At the time of the study, the department had no officially communicated strategy regarding OSS. Rules and initiatives promoting the optimal use of OSS, despite widespread OSS adoption within the corporation, were known only to a minority of employees. Nor was there any general policy governing contributing to existing OSS projects or releasing proprietary software as OSS, although instances of both had occurred.

Sample

Current involvement in and practices related to OSS were assessed in 25 interviews of 45 minutes average duration. These interviews, conducted with employees in different countries and at different hierarchical levels as well as with experts outside the company, were recorded, transcribed, and categorized using the software NVivo 7.

A large-scale online survey that built on these interviews was disseminated to the department's IT employees in early 2006. Participants were asked to share their general opinion of, and current experience with and exposure to, OSS as well as their perceptions of their peers with respect to OSS. Additionally, a measure of personal innovativeness was included.

The survey, in both English and German, was distributed at several of the corporation's international sites. The validity of the survey and consistency of the translation were confirmed in pre-tests. Addressees were invited to participate during a three-week time span. Participants received from their respective superiors invitational e-mails containing text composed by the survey's authors and a general user-password combination valid for all employees that protected the survey from unauthorized participation. Questions were answered anonymously. A reminder was sent out halfway through the three-week period to encourage additional participation.

Approximately 800 people in five countries were contacted and 249 valid replies received, yielding a response rate of 31%.[9] Response rates among sites varied from 24% to 80%. By job function, usable replies were received from 37 software testers, 23 software architects, 27 managers, 153 developers, and 9 project managers. This breakdown was nearly identical to the distribution of job functions in the company as a whole. All computations were performed using Stata 9.2.

--- Insert Table 1 about here ---

Dependent Variables

Attitude towards engaging in OSS. Our assessment of employee attitude distinguished among three levels of corporate OSS engagement: using existing OSS, contributing to OSS projects, and releasing proprietary software as OSS.[10] Respondents were asked to indicate their agreement, on a 5-point Likert scale ranging from 1 "strongly disagree" to 5 "strongly agree", to the following statements.

[9] We originally received 404 replies from approximately 1,100 people in seven countries (yielding a response rate of 37%). Legal restrictions, however, prevented the collection of demographic information in some countries thus reducing the number of usable observations. We analyzed for significant differences between the two groups (included vs. dropped observations), but found none. Nor were any differences observed with respect to individuals from different countries in both groups. To avoid single source bias, we also validated the survey results against the results of the interviews.

[10] We argued in the section on theory that attitude towards *personal* OSS engagement should be closely related to attitude towards *corporate* OSS engagement. This is confirmed empirically: agreement with the statement, "I would like to use more OSS in my job" is highly correlated with agreement with the statement, "[Company] could benefit from using OSS more often" (Spearman rank correlation: 0.51), and agreement with the statement, "I would like to develop more OSS in my job" is highly correlated with agreement with the statements, "[Company] could benefit from contributing modified OSS back to the public" (0.44) and "[Company] could benefit from making some of its own proprietary software public under an OSS license" (0.52).

I think that [company department] could benefit from...

using existing OSS more often

contributing modified OSS back to the public

making some of its own proprietary software public under an OSS license

Note that whether individuals are able to *correctly* assess how OSS will affect their employer is irrelevant in our context. It is their *perception* that determines their attitude towards corporate engagement in OSS, and consequently whether they adopt and lobby for OSS.

Explanatory Variables

Most explanatory variables are indices made up of items derived from theory. A complete list is given in Tables A.1 and A.2 in the Appendix. Descriptive statistics and correlations between explanatory variables are provided in Tables A.3 and A.4 in the Appendix.

Job functions are coded by dummy variables. We use testers as the reference group (1) because they are the most positive about OSS according to both our theoretical considerations and descriptive statistics (see Table 4), and (2) because they are a sufficiently large group, accounting for 14.9% of respondents (see Table 1).

Compatibility was measured with two items. Identification with the OSS community was measured with a single-item construct,[11] reciprocity by degree of agreement with three statements that reflect the community's give-and-take philosophy (see Table A.1). These items were taken from existing studies and slightly adjusted (Henkel 2006b; Lakhani et al. 2005). The three items that measure reciprocity load higher than 0.8 on a single factor, explaining 71% of the total variance. Cronbach's α of the index is 0.79.

[11] The respective survey question asked to what degree the participant agreed with the statement, "I identify with the OSS community."

Organizational factors that potentially influence attitude towards OSS were measured using seven questions based on statements collected in our interviews. These items capture mainly how respondents' peers and supervisors think about, and the degree to which they are familiar with, OSS. The items were distributed throughout the questionnaire to minimize social desirability bias. All items load higher than 0.5 on one factor with eigenvalue larger than one, explaining 44% of the total variance (see Table A.1). Cronbach's α of the index (Organizational Factors) is 0.77.

Previous OSS Exposure. To account for the effect of previous exposure to OSS, we included a dummy variable ("Did OSS") for whether a participant had worked on OSS code before.

Personal Innovativeness. We assessed personal innovativeness by scoring the individual items of the KAI index on a Likert scale ranging from 1 ("very adopter-like") to 5 ("very innovator-like"). The KAI items load on three factors. The first factor, originality, describes how well a person can deal with new ideas, the second, efficiency, a person's need for efficient processes and desire to execute tasks in great detail, and the third, conformity, a person's adherence to rules and authorities. Because the items loading on the efficiency and conformity scales must be scored in reverse, the resulting indices correspond to inefficiency and nonconformity, and higher scores in fact indicate higher innovativeness (Kirton 1976; Kirton 2003). Because the factor structure of the original 32-item scale has been disputed, the 13-item version of the KAI was used (Foxall et al. 1992b; Taylor 1989a; Taylor 1989b).

A number of studies have demonstrated the KAI inventory's relevance to IT and IT adoption. Gallivan (2003) found that people who scored higher on the originality factor had higher overall job performance and people who scored higher on the (in-)efficiency scale better communication skills. Foxall and Hackett (1992a) show that managers with higher levels of computer use as indicated by number of software applications used were more innovative with respect to the originality and conformity scales but more adopter-like with respect to efficiency.

Principal component analysis retains three factors with eigenvalues larger than 1, explaining 58% of the total variance. After varimax rotation, all items load higher than 0.64 on the factors predicted by the KAI model and lower than .21 on any other (see Table A.2). Cronbach's α for the factors originality, (in-)efficiency, and (non-) conformity are 0.82, 0.79, and 0.68, respectively.

Further individual characteristics. Participant age was solicited and this variable included in the regressions. In line with the findings of Agarwal and Prasad (1999), we checked as well for highest level of education attained, country in which the degree was awarded, and major. We also recorded at which site an individual was working. In no specification did any of the latter control variables show significance (either individually or jointly), so we only report specifications without these variables.

RESULTS

Job Functions – Descriptive Analysis

Respondents' attitudes towards corporate OSS engagement must be viewed in light of their OSS experience. Table 2 displays the means of various OSS-related characteristics.[12] Taking into account both use and development, we find that architects and developers qualify as most experienced in OSS.[13] In particular, developers have the highest share of respondents who have worked on OSS code. These facts are important for the following analysis. As we show below, developers are significantly less positive than software testers and architects about

[12] For comparison, variables that describe respondents' general programming activity are also displayed.

[13] The most experienced OSS users, in terms of both number of applications used and years working on OSS code, are software architects. Developers follow. In terms of share of respondents who have worked on OSS code, developers clearly lead (48%) architects (39%). In terms of hours spent per week writing and testing OSS code (at work and at home), project managers (5.6h/w) lead testers and developers (each 3.1h/w).

corporate OSS engagement. The data on OSS experience reported above allows us to rule out a simple explanation of this finding based on lack of OSS-related experience.

--- Insert Table 2 about here ---

We now turn to our main question. As Table 3 shows, respondents, on average, exhibit a "somewhat positive" attitude towards increased corporate engagement in OSS. When we probe deeper by distinguishing the *type* of OSS engagement, a richer picture emerges. Using a scale from 1 ("strongly disagree") to 5 ("strongly agree"), we obtain a mean value of 4.25 for *using OSS*, decreasing to 3.90 for *contributing to OSS projects* and to 3.53 for *releasing proprietary software as OSS*. The share of respondents that ticked "strongly agree" or "somewhat agree" yields an even clearer picture, declining from 85.1% (*using*) to 69.9% (*contributing*) to 56.2% (*releasing*).

Note also that the standard deviation monotonically increases from using OSS to contributing to OSS projects to releasing proprietary software as OSS. This finding holds both for the pooled sample and for each of the five job functions independently (see Table 2 and Table 3). This larger variation in the level of agreement, going hand in hand with the decrease in its mean, reflects the fact that the higher-involvement forms of corporate OSS engagement are yet unknown to employees and, hence, attended by higher uncertainty and higher perceived risk.

Regarding the influence of respondents' job functions, both the descriptive analysis presented here and the multivariate analysis in the following section matter. On the one hand, we need to know how testers, architects, developers, project managers, and managers, taken as *groups*, behave with respect to, and think about, OSS, which we analyze using univariate analysis. On the other hand, we want to isolate the effect of the *job function* net of other respondent characteristics with which it might be correlated.

-- Insert Table 3, Table 4 about here ---

As predicted, we find the level of support for OSS engagement to be highest for the group consisting of testers and architects, followed by the group made up of managers and developers (see Table 4). Project managers are the least positive about OSS. This finding is consistent across all three levels of OSS engagement, as is the ranking of attitude towards OSS by job function: testers, architects, managers, developers, and project managers (the only exception being that developers are more positive than managers with respect to *releasing*). The number of respondents being large (153) only for developers, it should come as no surprise that a Mann-Whitney test on the equality of medians fails to reject the Null hypothesis in a number of cases. Table 5 shows the results of this test.[14]

--- Insert Table 5 about here ---

Using OSS receives similar (high) levels of agreement from all job functions such that only H2[TD], H3[TP] and H6[AP] are supported. For *contributing to public OSS projects*, we find significant differences between testers/architects and developers (H2[TD] and H5[AD]), and between testers/architects and project managers (H3[TP] and H6[AP]). The largest, most significant differences between job functions are found in attitude towards *releasing proprietary software as OSS*, H1[TM], H2[TD], H3[TP], H4[AM], H6[AP], and H8[DP] being supported. In fact, quite a number of our hypotheses are supported. In particular, the difference between developers and the "leading" groups, testers and architects, is significant four times out of six.

Job Functions – Multivariate Analysis

The results presented above are informative about the attitudes of the five groups. But although understanding their attitudes towards OSS is relevant for managing these groups of IT professionals, the univariate results might be due not so much to the respondents' job func-

[14] Given that our dependent variable is ordinal (not interval) scaled, a t-test on the equality of means would be inappropriate. The Mann-Whitney test has the additional advantage of being non-parametric (i.e., of not presuming a particular distribution of the data).

tions as to other characteristics that, for whatever reason, are correlated with the task a person performs. To separate these intertwined effects, we employ multivariate analysis, specifically, ordered probit regression to account for the ordinal nature of our dependent variables. Table 6 shows the regression results, using level of agreement with the three statements above as dependent variables.[15] The first four lines show the estimation coefficients of the dummy variables coding respondents' job functions, with testers as the reference group (i.e., their coefficient is implicitly set to zero). Coefficients thus indicate differences between the attitudes of testers and the respective other group. Post-estimation analyses run to compare the displayed coefficients with each other, yielded the results reported in Table 7. Note that significance levels here refer to our hypotheses, which are directed (as opposed to the undirected significance levels given in Table 6).

--- Insert Table 6 and Table 7 about here ---

The first box of Table 7, regarding attitudes towards *using OSS*, shows significant differences between testers and developers (H2[TD]), but no support for other hypotheses, indicating that the differences found to be significant in Table 5 are mostly accounted for by characteristics other than job function. For *contributing to public OSS projects*, we find significant differences between the job functions of developer/project manager and architect/tester, which supports H2[TD] and H5[AD] as well as H3[TP] and H6[AP]. This finding is consistent with the univariate analysis presented above. In addition, the coefficient describing the difference to testers is larger for developers than for any other group. Hence, performing the job function of developer has, *ceteris paribus* (i.e., after correcting for characteristics potentially correlated

[15] To assure that multicollinearity was not an issue, we also ran regressions dropping one of the more strongly correlated explanatory variables ("Identification with OSS community") and obtained results largely identical to those presented. We also controlled for the influence of tenure at the company, but dropped the variable due to its high correlation ($r = 0.84$) with age. Using tenure with the corporation instead of age does not have an effect on the sign or level of significance of the other explanatory variables. Tenure itself is insignificant for the first two regressions (using, contributing) and significantly negative ($p < 0.05$) for the third regression (releasing).

with it), an even more negative effect on attitude towards contributing than is suggested by the univariate analysis.

For *releasing software as OSS*, we find significant differences between testers on the one hand and managers, developers, and project managers on the other, providing support for H1[TM], H2[TD], and H3[TD]. We also observe a significant difference between architects and managers (H4[AM]).

We thus find that the differences in attitudes towards corporate OSS engagement between the five groups defined by job function can be explained only partly by other characteristics of the respondents. In particular, in four out of six pairwise comparisons the job function of developer implies, ceteris paribus, significantly lower support for corporate OSS engagement than the job function of architect or tester.

Control Variables

A full discussion of all control variables being beyond the scope of this paper, we comment briefly on a few salient points. To summarize, all significant coefficients carry the expected sign.

Especially compatibility, as measured by identification with the OSS community and opinion on reciprocity, is highly significant in all regressions. Experience with OSS ("Did OSS") has a significant positive influence on attitude towards using OSS and a highly significant effect on attitude towards contributing to existing OSS projects. Age has an inverse effect on attitude towards contributing to existing OSS projects or releasing proprietary software as OSS. The finding that younger persons are more likely to perceive such behavior to be beneficial to the firm is in line with interviewees' statements that university training in IT used to value writing one's own code much higher than re-using code, and that only in the last decade or two were students trained to draw on existing code where possible. Somewhat surprisingly, the constructs derived from the KAI index matter little; even joint insignificance cannot be rejected for any of our three regressions.

DISCUSSION AND IMPLICATIONS

Job Functions and OSS Adoption

We have developed and tested a model that links individuals' attitudes towards commercial OSS adoption, for various types of OSS engagement, to their job function. On an aggregate level, we find that corporate OSS engagement is viewed at least "somewhat" positively by most of the people in the company we studied. With respect to type of OSS engagement, we find that *using existing OSS more often* is seen by a majority of respondents (85% agreed "somewhat" or "strongly") as potentially beneficial to the company. For more intense types of engagement, agreement decreases: *contributing to OSS projects* is seen as advantageous by 70% of respondents, *releasing proprietary software as OSS* by only 56%. At the same time, the variance of the agreement level goes up, reflecting the higher perceived uncertainty and risk associated with more intense types of OSS engagement.

To understand how OSS fits into corporate software development processes, however, an even more differentiated view is required that takes into account, in addition to the type of OSS engagement, an individual's job function. Attitudes towards OSS are influenced by the fact that engaging in OSS projects and releasing proprietary software as OSS represent, to varying degrees for different job functions, a significant deviation from ingrained routine. Guided by the theoretical frameworks of TAM and DOI and based on an analysis of the advantages and drawbacks of OSS engagement for each job function, our model predicts that testers and architects would be the most favorably disposed to OSS, managers and developers less so, and project managers the least.

The empirically revealed differences between job functions are in line with our hypotheses, and turn out to be significant in half of all pairwise comparisons. In particular, developers were found, in 8 out of 12 cases, both accounting and not accounting for other individual characteristics, to be significantly less favorably disposed to OSS than either architects or testers. For developers, OSS seems to approximate what Lyytinen and Rose (2003) term a "disruptive IT innovation." The inherent organizational change thus disposes developers, on average, to react less than enthusiastically to increased corporate commitment to OSS.[16] Generally, the more OSS development differs from the current development model and the less skilled developers consider themselves to be,[17] the less supportive they will be. Three quotes from our interviews illustrate the changes OSS engagement occasions, in particular, for developers.

> *"Yes, I think documentation is an important prerequisite [for making software public as OSS] that we are currently not yet meeting." (Translated from German by the authors)*
>
> *"Following the license is somewhat hard […]. That takes a lot of effort and people don't really know what to do."*
>
> *"[Among developers] there is a not-invented-here syndrome, you know, that people feel they need to build on own [their] developments." (Translated from German by the authors)*

Our findings seem at odds with anecdotal evidence of developers' supposedly positive attitude towards OSS. Most likely, this evidence relates to *individual* developers who advocated or perhaps even launched isolated OSS efforts in their firms. Although such efforts do

[16] This finding must be seen in light of the given corporate environment, in which OSS does not play a central role. Henkel 2006a, in contrast, studies firms, many of which are rather small and young, active in or even dedicated to the development of embedded Linux. In his sample of 197 commercial OSS developers, 49.7% of respondents stated that they either make suggestions to their supervisor as to what code could be released or even that this decision is within their own discretion.

[17] Our interviewees consistently indicated the share of developers with the necessary skills (programming, social, and management) to work in a corporate OSS project to be around 25%.

affect corporate OSS adoption, serious corporate engagement relies on championing and sponsoring efforts (Chakrabarti et al. 1977; Hauschildt et al. 1989; Hauschildt et al. 2001; Howell et al. 1990; Markham et al. 1991; Rothwell et al. 1974). The project managers who would seem, due to their boundary-spanning role, to be ideally suited to act as champions turn out to be the least favorably disposed towards OSS. Based on our analysis of the job-specific pros and cons of OSS, we suggest solutions to this dilemma below.

Our study is in line with findings by Sherif et al. (2006) that similar conflicts arise for developers in software reuse, for which they suggest as a solution management intervention. Fichman and Kemerer's (1993) earlier reported slow adoption of software reuse practices by developers was later found by Kim and Stohr (1998) to be caused by lack of (mandatory) organizational support (including required resources, training, and rewards) and difficulty measuring economic impact. We have shown that similar issues arise with OSS, but the degree to which they hinder the adoption and diffusion of corporate OSS engagement is likely higher inasmuch as software reuse occurs within the boundaries of a firm.

We further suggest that the results of our study are not limited to the context of OSS, that they have broader implications. The segmentation of roles according to a design-build-test cycle is not unique to the software development industry (Wheelwright et al. 1994). Comparable open and distributed innovation efforts in other industries (Chesbrough 2003; von Hippel 2005) are thus likely to face similar challenges and even resistance from the respective counterparts of developers and project managers.

Finally, our results suggest an extension of existing theory and avenues for further research. As Venkatesh (2006) has suggested, models that predict the adoption of IT innovations should take into account an individual's job function, or, phrased more generally, the way in which the adopter interacts with the innovation at hand. Job function will strongly influence an individual's perception of and attitude towards an innovation, and will be an important, even decisive, factor in an individual's adoption decision. Especially in cases in which innovations significantly affect existing processes, the moderating effect of job function cannot be ignored. Recent extensions of the TAM and similar models (Venkatesh et al. 2000; Venkatesh et al. 2003) include items that measure the job relevance of an innovation; extending the extensions by explicitly including job function could further increase their applicability.

Limitations and Possible Extensions

Some limitations of our study need to be mentioned. First, it was conducted within a single firm. This has the advantage of simplifying comparisons between respondents because firm-specific effects are kept constant. But the level of support for OSS engagement might be higher or lower in other corporations depending on corporate culture, previous exposure to OSS, industry (Klevorick et al. 1995), and home country. *Differences* in attitudes towards OSS among testers, developers, and others, however, result from the professional activities of these groups, which should be largely independent of firm or location. Hence, we are confident that our findings regarding the impact of job function generally hold.

As our respondents were located in seven countries, we were able to check for national idiosyncrasies, but found no significant differences between countries. Still, conducting a similar study in different countries, in one or more other firms, possibly in different industries, could provide valuable insights.

Finally, the firm's software development method might have influenced our results. At the time of our survey, the firm studied was still relying heavily on the waterfall model. Agile software development had been introduced early in 2005, but was not yet being widely used. Consequently, OSS represents to developers in this firm an entirely new model of software development. Developers in firms that have experience in extreme programming, agile methodologies, or the spiral model should thus be more favorably disposed towards OSS development.

Recommendations for Practice

Introducing OSS and OSS development implies changes, in particular, for developers and project managers, precisely the groups that turned out to be least favorably disposed to OSS. Possible steps towards solving this dilemma include training and step-by-step introduction of corporate OSS engagement. Training programs should be established for new hires and advanced training programs for developers, managers, and IP and legal staff. One might also consider brown-bag seminars for developers, incorporating the open source policy in employee handbooks, and online seminars or training (Fan et al. 2004).

A first step suggested by many of our survey and interview participants might be the introduction of a “corporate source program” (Dinkelacker et al. 2001). Corporate source initiatives that mimic the OSS development style within the boundaries of an organization might be a good way to familiarize staff with the OSS development style while minimizing risks with respect to loss of intellectual property and sociological issues such as the not-invented-here syndrome.

Finally, individuals should be given the opportunity to self-select into OSS-related activities. Even among project managers, the group most skeptical of OSS, one-third of our respondents considered an OSS engagement on all three levels as “somewhat” beneficial to the corporation. It should thus be possible to staff pilot OSS projects, in all job functions that are required, with OSS supporters, provided the staffing is done diligently.

No matter what its business, every firm active in software development needs to confront the questions of whether and how to engage in OSS. It is striking that even Microsoft, a long-time opponent of OSS, supports OSS practices among licensees of Windows CE 6.0, granting them to access the full kernel source code and permitting them to modify it and share the modified code with other licensees.[18] Still, among our survey participants, OSS failed to find support among a considerable share of developers and project managers. In their own time, the practices of software reuse and object oriented programming faced similar obstacles, but became widely adopted once their benefits came to be realized throughout the IT industry. We believe the question should not be *whether* firms should engage in OSS, but rather when, how, and to what extent. Attempts to answer these questions need to take into account the impact of OSS both on the firm's software development processes and on the individuals involved. Our study is an attempt to shed light on these issues in order to enable broader, more informed use of OSS and the OSS development style.

[18] See http://msdn2.microsoft.com/en-us/embedded/aa714518.aspx (accessed May 23, 2007).

FIGURES & TABLES

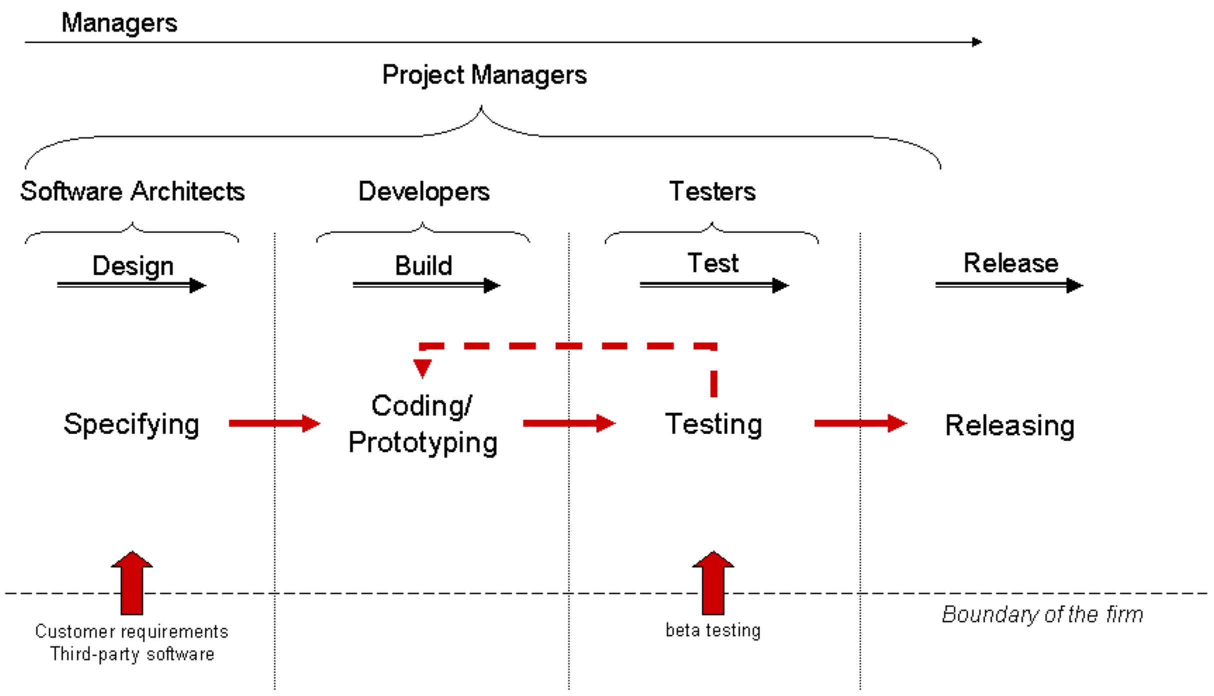

Figure 1: Proprietary Closed Source Software (PCSS) Development

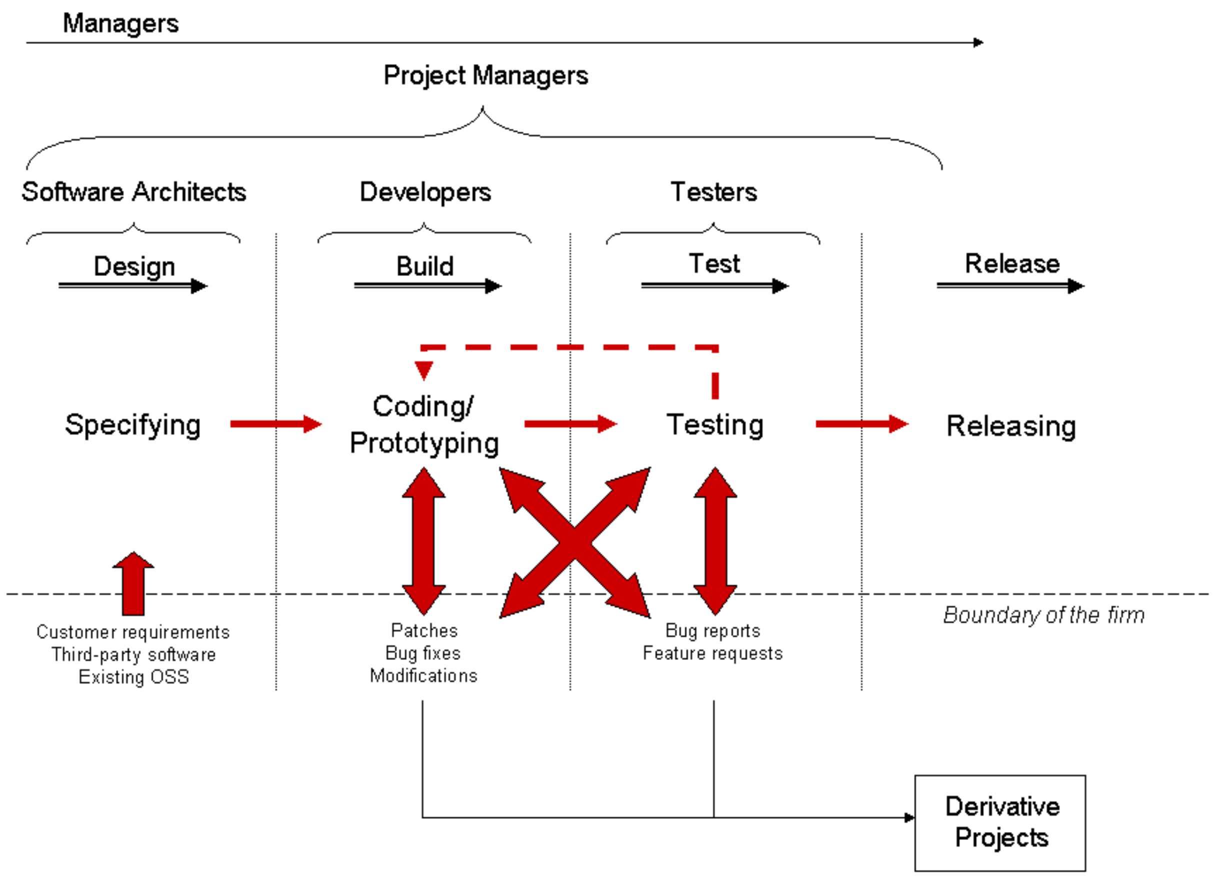

Figure 2: Open Source Software (OSS) Development

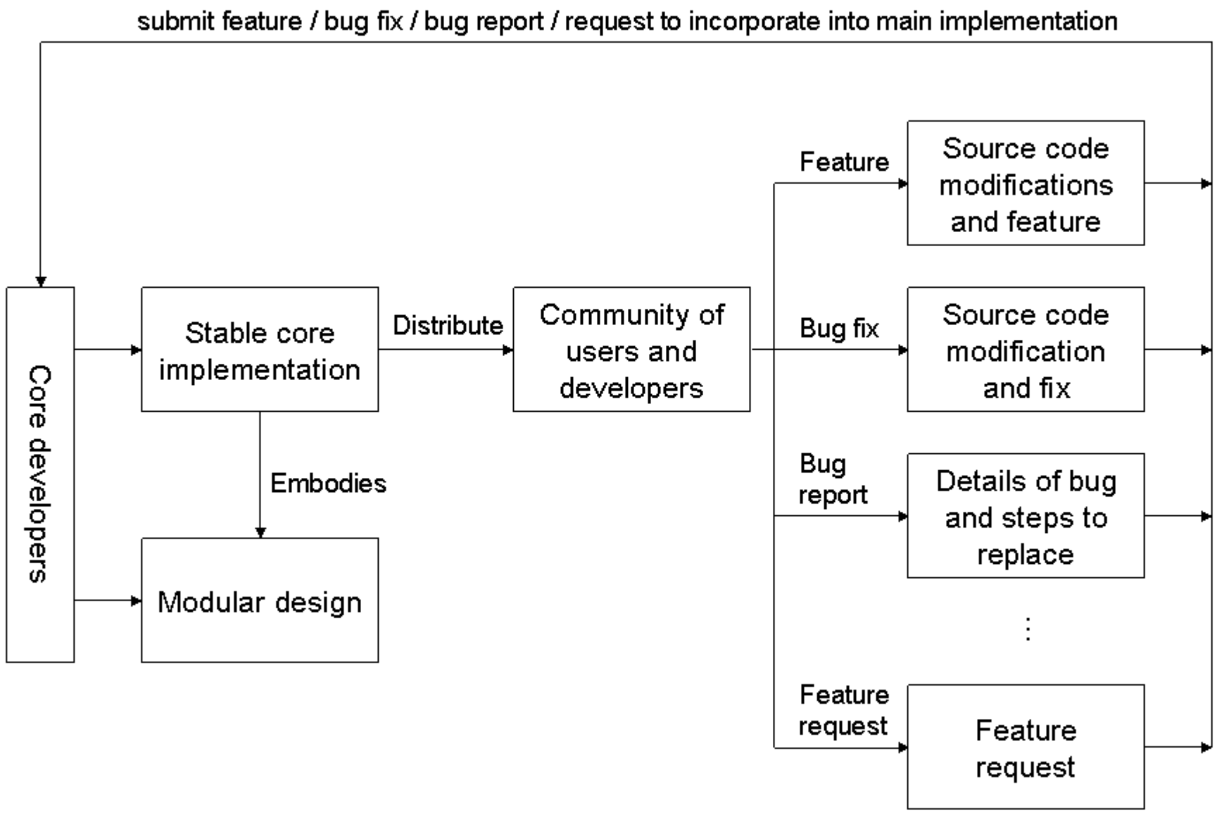
submit feature / bug fix / bug report / request to incorporate into main implementation
Core developers
Stable core implementation
Distribute
Community of users and developers
Embodies
Modular design
Feature
Source code modifications and feature
Bug fix
Source code modification and fix
Bug report
Details of bug and steps to replace
⋮
Feature request
Feature request

Figure 3: OSS Development Cycle (Senyard et al. 2004)

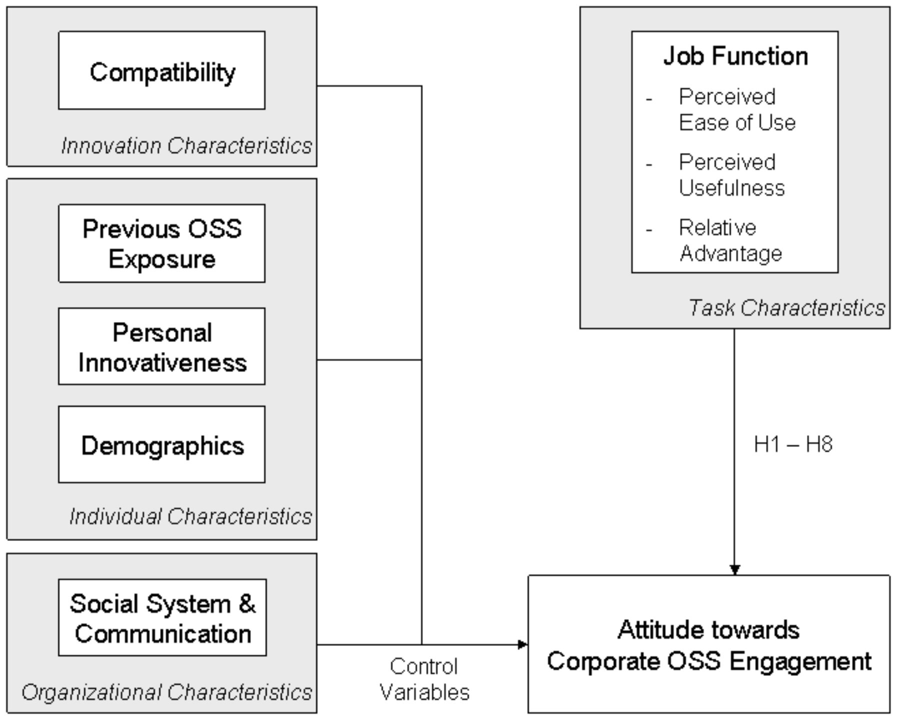

Figure 4: Research Model

Job Function	**Frequency**	**Percent**	**Cumulative**
Testers	37	14.86	14.86
Architects	23	9.24	24.10
Managers	27	10.84	34.94
Developers	153	61.45	96.93
Project Managers	9	3.61	100.00
Total	249		

Table 1: Survey participants by job function

Variable	**Testers**	**Architects**	**Managers**	**Develop-ers**	**Project Managers**
N	37	23	27	153	9
Number of OSS Applications used (in total)	2.486	**3.652**	2.741	**3.046**	2.444
Number of OSS Applications used (out of 6 suggestions)	2.297	**3.478**	2.556	**2.627**	2.444
Years working on OSS source code	1.541	**2.391**	1.815	**2.373**	0.889
Has worked on OSS code (1: Yes, 0:No)	0.297	**0.391**	0.259	**0.484**	0.333
Hours per week spent on pro-gramming at work (incl. testing, documentation)	**16.784**	7.043	5.852	**27.193**	13.389
Hours per week spent on pro-gramming at home	**5.270**	4.130	3.296	**4.412**	2.222
Hours per week spent on pro-gramming OSS at work (incl. testing, documentation)	2.514	0.696	0.593	**2.601**	**5.611**
Hours per week spent on pro-gramming OSS at home	**0.622**	**0.783**	0.074	0.471	0.000
I would like to use more OSS at [FIRM]	**4.344**	3.909	**4.043**	3.979	3.333
I would like to develop more OSS at [FIRM]	**4.069**	**3.864**	3.632	3.806	2.833

Bold: largest and second largest value in each line.

Table 2: Means of OSS- and programming-related characteristics, by job function

Variable	**N**	**Me-dian**	**Mean**	**Std. Dev.**	**Min**	**Max**	**Share 4 & 5**
Corporation could benefit from using existing OSS more often	249	4	4.25	0.85	1	5	85.14%
Corporation could benefit from contrib-uting to existing OSS projects	249	4	3.90	0.98	1	5	69.88%
Corporation could benefit from releas-ing proprietary software as OSS	249	4	3.53	1.14	1	5	56.22%

Table 3: Descriptive statistics of dependent variables

Variable	**Testers**	**Archi-tects**	**Manag-ers**	**Devel-opers**	**Project Managers**
Corporation could benefit from using existing OSS more often	**4.46** (0.73)	**4.30** (0.82)	4.30 (0.95)	4.20 (0.88)	4.00 (0.5)
Corporation could benefit from con-tributing to existing OSS projects	**4.22** (0.75)	**4.17** (0.83)	3.89 (1.01)	3.80 (1.05)	3.56 (0.53)
Corporation could benefit from releas-ing proprietary software as OSS	**4.14** (0.92)	**3.74** (0.92)	3.15 (1.29)	3.46 (1.14)	2.89 (1.05)

Bold: largest and second largest value in each line. Standard deviation in parentheses.

Table 4: Mean values of the dependent variables by job function

Using	**Testers**	**Architects**	**Managers**	**Developers**	**Project Mgrs**
Mean	4.46	4.30	4.30	4.20	4.00
Median	5	4	5	4	4
Testers	---				
Architects	0.227	---			
Managers	0.332	0.403	---		
Developers	**0.049****	0.313	0.210	---	
Project Mgrs	**0.014****	**0.073***	**0.061***	0.102	---
Contributing	**Testers**	**Architects**	**Managers**	**Developers**	**Project Mgrs**
Mean	4.22	4.17	3.89	3.80	3.56
Median	4	4	4	4	4
Testers	---				
Architects	0.460	---			
Managers	0.111	0.169	---		
Developers	**0.018****	**0.060***	0.370	---	
Project Mgrs	**0.005*****	**0.014****	0.125	0.135	---
Releasing	**Testers**	**Architects**	**Managers**	**Developers**	**Project Mgrs**
Mean	4.14	3.74	3.15	3.46	2.89
Median	4	4	3	4	5
Testers	---				
Architects	**0.039****	---			
Managers	**0.001*****	**0.049****	---		
Developers	**0.000*****	0.172	0.112	---	
Project Mgrs	**0.001*****	**0.025****	0.306	**0.066***	---

* significant at 10%; ** significant at 5%; *** significant at 1%.

Table 5: Mann-Whitney Test on differences in medians (p)

	Corporation could benefit from… (1-5 scale, ordered probit)		
	… using	**… contrib.**	**… releasing**
Job Architect	-0.516	-0.015	**-0.572***
	(0.348)	(0.278)	(0.308)
Job Manager	-0.194	-0.268	**-1.002*****
	(0.351)	(0.291)	(0.329)
Job Developer	-0.404	**-0.511****	**-0.752*****
	(0.249)	(0.208)	(0.214)
Job Project Manager	-0.307	-0.456	**-0.920****
	(0.298)	(0.314)	(0.395)
Identification with OSS community	**0.561*****	**0.284*****	**0.325*****
	(0.085)	(0.097)	(0.093)
Reciprocity		**0.432*****	**0.373*****
		(0.119)	(0.119)
Organizational factors	0.077	-0.011	0.006
	(0.124)	(0.117)	(0.117)
Did OSS	**0.439*****	**0.401*****	0.094
	(0.154)	(0.147)	(0.141)
KAI Originality	0.181	0.091	0.171
	(0.142)	(0.133)	(0.130)
KAI Efficiency	-0.101	-0.120	-0.149
	(0.142)	(0.126)	(0.114)
KAI Conformity	-0.165	-0.058	0.095
	(0.152)	(0.123)	(0.121)
Age	-0.003	**-0.019****	**-0.019****
	(0.008)	(0.008)	(0.008)
Observations	249	249	249
Pseudo R-squared	0.14	0.13	0.13
Pseudo Likelihood	-240.862	-285.031	-320.875
Wald's chi-squared	73.694	62.806	88.274
Degrees of freedom	11	12	12

* significant at 10%; ** significant at 5%; *** significant at 1%;
Robust standard errors in parentheses

Table 6: Results of the ordered probit regressions

Using	**Testers**	**Architects**	**Managers**	**Developers**	**Project Managers**
Coefficient Value	0	**-0.516***	-0.194	**-0.404***	-0.307
Testers	---				
Architects	**0.069***	---			
Managers	0.291	0.175	---		
Developers	**0.053***	0.345	0.356	---	
Project Managers	0.151	0.247	0.356	0.321	---
Contributing	**Testers**	**Architects**	**Managers**	**Developers**	**Project Managers**
Coefficient Value	0	-0.015	-0.268	**-0.511****	**-0.456***
Testers	---				
Architects	0.479	---			
Managers	0.179	0.195	---		
Developers	**0.007*****	**0.019****	0.163	---	
Project Managers	**0.073***	**0.092***	0.289	0.42	---
Releasing	**Testers**	**Architects**	**Managers**	**Developers**	**Project Managers**
Coefficient Value	0	**-0.572****	**-1.002*****	**-0.752*****	**-0.920****
Testers	---				
Architects	**0.032****	---			
Managers	**0.001*****	**0.093***	---		
Developers	**0.000*****	0.229	0.174	---	
Project Managers	**0.01****	0.198	0.423	0.319	---

* significant at 10%; ** significant at 5%; *** significant at 1%

Table 7: Ordered probit and probit post-estimation: test of equality of coefficients (p-values)

APPENDIX

Reciprocity	Factor Loadings
[FIRM] has an obligation of giving back to the OSS community	**0.807**
I would release code because I consider it fair to give back to the community, since the company benefits from it	**0.882**
I would release code because in the long run, you only get something when you gave something before	**0.823**

Popularity of OSS among Co-Workers (Organizational Factors)	Factor Loadings
Management promotes the use of existing OSS	**0.7**
Which of the following factors do you consider supportive of or an impediment to the wider use of OSS within [FIRM]? My supervisor	**0.727**
Which of the following factors do you consider supportive of or an impediment to the wider use of OSS within [FIRM]? My colleagues	**0.601**
My supervisor is familiar with OSS	**0.701**
Most programmers at [FIRM] are familiar with OSS	**0.638**
In case I had questions on OSS, I would know someone at [FIRM] I could turn to	**0.577**
Management sees the benefit of OSS	**0.64**

Table A.1: Questions underlying factor constructs (Part Three)

KAI Originality			
Would you consider yourself someone who…	**Originality**	**Efficiency**	**Conformity**
… has fresh perspectives on old problems	**0.689**	-0.216	-0.045
… copes with several new ideas at the same time	**0.679**	-0.116	-0.002
… is stimulating	**0.802**	-0.093	-0.016
… has original ideas	**0.802**	-0.1	0.075
… proliferates ideas	**0.77**	-0.07	-0.09

KAI Efficiency			
Would you consider yourself someone who…	**Originality**	**Efficiency**	**Conformity**
… enjoys detailed work	-0.085	**0.706**	0.214
… is thorough	-0.117	**0.786**	0.093
… masters all details painstakingly	-0.159	**0.766**	0.163
… is methodical and systematic	-0.119	**0.745**	0.113

KAI Conformity			
Would you consider yourself someone who…	**Originality**	**Efficiency**	**Conformity**
… conforms	0.152	0.184	**0.636**
… is prudent when dealing with authority	-0.092	0.044	**0.734**
… never acts without proper authority	-0.024	0.206	**0.662**
… fits readily into "the system"	-0.025	0.221	**0.75**

Table A.2: Questions underlying factor constructs (Part Four)

Variable		**Testers**	**Architects**	**Managers**	**Deve-lopers**	**Project Mgrs**	**Over-all**
Id. w. OSS community	Mean	3.40	3.68	3.32	3.25	2.44	3.29
	S.D.	(0.86)	(0.82)	(0.95)	(1.09)	(0.73)	(1.02)
	Median	3.00	4.00	3.00	4.00	3.00	3.00
Reciprocity	Mean	3.74	3.82	3.78	3.74	3.56	3.74
	S.D.	(0.54)	(0.64)	(0.6)	(0.85)	(0.88)	(0.77)
	Median	3.67	3.98	4.00	3.98	3.67	3.87
Organizational Factors	Mean	3.24	3.68	3.42	3.29	3.29	3.33
	S.D.	(0.70)	(0.56)	(0.72)	(0.69)	(0.70)	(0.69)
	Median	3.29	3.71	3.43	3.71	3.29	3.43
Did OSS	Mean	0.30	0.39	0.26	0.48	0.33	0.42
	Median	0.00	0.00	0.00	0.00	0.00	0.00
KAI Originality	Mean	3.75	4.24	4.03	3.77	3.75	3.84
	S.D.	(0.63)	(0.56)	(0.6)	(0.61)	(0.62)	(0.62)
	Median	3.80	4.40	4.00	4.40	3.60	3.80
KAI Efficiency	Mean	2.17	2.20	1.93	1.99	1.94	2.03
	S.D.	(0.65)	(0.76)	(0.86)	(0.62)	(0.54)	(0.67)
	Median	2.25	2.00	2.00	2.00	2.00	2.00
KAI Conformity	Mean	2.34	2.48	2.25	2.35	2.30	2.35
	S.D.	(0.6)	(0.81)	(0.64)	(0.66)	(0.46)	(0.66)
	Median	2.25	2.50	2.00	2.50	2.50	2.25
Age	Mean	35.76	44.78	45.56	38.62	44.00	39.71
	S.D.	(8.16)	(8.49)	(8.47)	(10.32)	(7.16)	(10.02)
	Median	34.00	45.00	48.00	45.00	45.00	40.00

Table A.3: Mean values, standard deviations, and medians of independent variables by job function

	Benefit of Using	Benefit of Contributing	Benefit of Releasing	Job Tester	Job Architect	Job Manager	Job Developer	Job Project Manager	Identification with OSS community	Reciprocity	Organizational Factors	Did OSS	KAI Originality	KAI Efficiency	KAI Conformity	Age
Benefit of Using	1															
Benefit of Contributing	0.60	1														
Benefit of Releasing	0.50	0.62	1													
Job Tester		0.13	0.23	1												
Job Architect				-0.13	1											
Job Manager			-0.11	-0.15	-0.11	1										
Job Developer		-0.11		-0.53	-0.40	-0.44	1									
Job Project Manager			-0.11				-0.24	1								
Id. with OSS community	0.45	0.46	0.43		0.13			-0.17	1							
Reciprocity	0.38	0.41	0.37						0.50	1						
Organizational Factors					0.17						1					
Did OSS	0.20	0.21	0.11			-0.11	0.17		0.18	0.14		1				
KAI Originality	0.15	0.13	0.12		0.23	0.10	-0.14			0.14			1			
KAI Efficiency													-0.34	1		
KAI Conformity											-0.20		-0.13	0.37	1	
Age		-0.19	-0.22	-0.16	0.16	0.21	-0.14		-0.17		0.17	-0.11		-0.12		1

Table A.4: Spearman rank correlation (displayed only for p<0.1)

REFERENCES

Agarwal, R. "Individual Acceptance of Information Technologies," in: *Framing the Domains of IT Management: Project the Future ... ... Through the Past,* R.W. Zmud (ed.), Pinnaflex, Cincinnati, OH, 2000, pp. 85-104.

Agarwal, R., and Prasad, J. "Are Individual Differences Germane to the Acceptance of New Information Technology?," *Decision Sciences* (30:2) 1999, pp 361-391.

Allen, R.C. "Collective Invention," *Journal of Economic Behavior & Organization* (4:1) 1983, pp 1-24.

Baldridge, J.V., and Burnham, R.A. "Organizational Innovation: Individual, Organizational, and Environmental Impacts," *Administrative Sciences Quarterly* (20:2) 1975, pp 165-176.

Baldwin, C.Y., and Clark, K.B. *Design Rules: The Power of Modularity* MIT Press, Cambridge, MA, 2000.

Banker, R.D., Datar, S.M., and Kemerer, C.F. "A Model to Evaluate Variables Impacting the Productivity of Software Maintenance Projects," *Management Science* (37:1) 1991, pp 1-18.

Bass, L., Clements, P., and Kazman, R. *Software Architecture in Practice*, (2nd ed.) Addison-Wesley, Boston, MA, 2003.

Behlendorf, B. "Open Source as a Business Strategy," in: *Open Sources: Voices from the Open Source Revolution,* C. DiBona, S. Ockman and M. Stone (eds.), O'Reilly, Sebastopol u.a., 1999, pp. 149-170.

Bessen, J. "Open Source Software: Free Provision of Complex Public Goods," 2005.

Bonaccorsi, A., Giannangeli, S., and Rossi, C. "Entry Strategies under Competing Standards: Hybrid Business Models in the Open Source Software Industry," *Management Science* (52:7) 2006a, pp 1085-1098.

Bonaccorsi, A., and Rossi, C. "Why Open Source Software Can Succeed," *Research Policy* (32:7) 2003, pp 1243-1258.

Bonaccorsi, A., and Rossi, C. "Comparing Motivations of Individual Programmers and Firms to Take Part in the Open Source Movement: From Community to Business," *Knowledge, Technology, and Policy* (18:4) 2006b, pp 40-64.

Burgelman, R.A. "A Model of the Interaction of Strategic Behavior, Corporate Context, and the Concept of Strategy," *Academy of Management Review* (8:1) 1983, pp 61-70.

Chakrabarti, A.K., and O'Keefe, R.D. "A Study of Key Communicators in Research and Development Laboratories," *Ground & Organization Studies* (2:3) 1977, pp 336-346.

Chesbrough, H.W. *Open Innovation: The New Imperative for Creating and Profiting from Technology* Harvard Business School Press, Boston, 2003.

Constant, D., Sproull, L., and Kiesler, S. "The Kindness of Strangers: The Usefulness of Electronic Weak Ties for Technical Advance," *Organization Science* (7:2) 1996, pp 119-135.

Cooper, R.B., and Zmud, R.W. "Information Technology Implementation Research: A Technological Diffusion Approach," *Management Science* (36:2) 1990, pp 123-139.

Cusumano, M., MacCormack, A., Kemerer, C.F., and Crandall, B. "Software Development Worldwide: The State of the Practice," *IEEE Software* (20:6) 2003, pp 28-34.

Daft, R.L. "A Dual-Core Model of Organizational Innovation," *Academy of Management Journal* (21:2) 1978, pp 193-210.

Dahlander, L. "Appropriation and Appropriability in Open Source Software," *International Journal of Innovation Management* (9:3) 2005, pp 259-285.

Dahlander, L., and Magnusson, M.G. "Relationships between Open Source Software Companies and Communities: Observations from Nordic Firms," *Research Policy* (34:4) 2005, pp 481-493.

Dahlander, L., and Wallin, M.W. "A Man on the Inside: Unlocking Communities as Complementary Assets," *Research Policy* (35:8) 2006, pp 1243-1259.

Damanpour, F. "Organizational Innovation: A Meta-Analysis of Effects of Determinants and Moderators," *Academy of Management Journal* (34:3) 1991, pp 555-590.

Damanpour, F., and Evan, W.M. "Organizational Innovation and Performance: The Problem of 'Organizational Lag'," *Administrative Science Quarterly* (29:3) 1984, pp 392-409.

Davis, F.D. "Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology," *MIS Quarterly* (13:3) 1989, pp 318-340.

Davis, F.D., Bagozzi, R.P., and Warshaw, P.R. "User Acceptance of Computer Technology: A Comparison of Two Theoretical Models," *Management Science* (35:8) 1989, pp 982-1003.

Deci, E.L., and Ryan, R.M. *Intrinsic Motivation and Self-Determination in Human Behaviour* Plenum, New York, 1985.

DiBona, C. "Open Source and Proprietary Software Development," in: *Open Sources 2.0: The Continuing Evolution,* C. DiBona, D. Cooper and M. Stone (eds.), O'Reilly, Sebastopol, CA, 2005, pp. 21-36.

Dinkelacker, J., Garg, P.K., Miller, R., and Nelson, D. "Progressive Open Source," Hewlett Packard Laboratories Palo Alto, 2001.

Driver, M., Drakos, N., Weiss, G.J., Claunch, C., Govekar, M., Feinberg, D., Maio, A.D., Hostmann, B., Igou, B., Cantara, M., Phifer, G., Enck, J., Pescatore, J., Latham, L., Gilliland, M., Silver, M.A., Haight, C., Valdes, R., Girard, J., Perkins, E.L., Lee, M., Hafner, B., Natis, Y.V., and Cain, M.W. "Hype Cycle for Open-Source Software, 2005," Gartner, p. 22.

Driver, M., and Weiss, G.J. "Predicts 2006: The Effects of Open-Source Software on the IT Software Industry," Gartner, p. 5.

Ebadi, Y.M., and Utterback, J.M. "The Effects of Communication on Technological Innovation," *Management Science* (30:5) 1984, pp 572-585.

Fan, B., Aitken, A., and Koenig, J. "Open Source Intellectual Property and Licensing Compliance: A Survey and Analysis of Industry Best Practices," 2004.

Feller, J., and Fitzgerald, B. "Open Source Software: Investigating the Software Engineering, Psychosocial and Economic Issues," *Information Systems Journal* (11:4) 2001a, pp 273-276.

Feller, J., and Fitzgerald, B. *Understanding Open Source Software Development* Addison-Wesley, Boston, MA, 2001b.

Fichman, R.G., and Kemerer, C.F. "Adoption of Software Engineering Process Innovations: The Case of Object Orientation," *Sloan Management Review* (34:2) 1993, pp 7-22.

Foxall, G.R., and Hackett, P.M.W. "Cognitive Style and Extent of Computer Use in Organizations: Relevance of Sufficiency of Originality, Efficiency and Rule-Conformity," *Perceptual and Motor Skills* (74:2) 1992a, pp 491-497.

Foxall, G.R., and Hackett, P.M.W. "The Factor Structure and Construct Validity of the Kirton Adaption-Innovation Inventory," *Personality and Individual Differences* (13:9) 1992b, pp 967-975.

Franke, N., and von Hippel, E. "Satisfying Heterogeneous User Needs via Innovation Toolkits: The Case of Apache Security Software," *Research Policy* (32:7) 2003, pp 1199-1215.

Gallivan, M.J. "Organizational Adoption and Assimilation of Complex Technological Innovations: Development and Application of a New Framework," *SIGMIS Database* (32:3) 2001, pp 51-85.

Gallivan, M.J. "The Influence of Software Developers' Creative Style on Their Attitudes to and Assimilation of a Software Process Innovation," *Information & Management* (40:5) 2003, pp 443-465.

Goldman, R., and Gabriel, R.P. *Open Source as Business Strategy: Innovation Happens Elsewhere* Morgan Kaufmann, San Francisco, CA, 2005.

Goulde, M. "Open Source Becoming Mission-Critical In North America And Europe," Forrester, p. 14.

Grand, S., von Krogh, G., Leonard, D., and Swap, W. "Resource Allocation Beyond Firm Boundaries: A Multi-Level Model for Open Source Innovation," *Long Range Planning* (37:6) 2004, pp 591-610.

Granovetter, M. "Economic Action and Social Structure: The Problem of Embeddedness," *American Journal of Sociology* (91:3) 1985, pp 481-510.

Granovetter, M.S. "The Strength of Weak Ties," *American Journal of Sociology* (78:6) 1973, pp 1360-1380.

Grover, V. "An Extension of the Tri-Core Model of Information Systems Innovation: Strategic and Technological Moderators," *European Journal of Information Systems* (6:4), Dec 1997, pp 232-242.

Gruber, M., and Henkel, J. "New Ventures Based on Open Innovation – an Empirical Analysis of Start-Up Firms in Embedded Linux," *International Journal of Technology Management* (33:4) 2006, pp 356-372.

Hang, J., Hohensohn, H., Mayr, K., and Wieland, T. "Benefits and Pitfalls of Open Source in Commercial Contexts," in: *Free/Open Source Software Development,* S. Koch (ed.), Idea Group, Hershey, PA, 2004, pp. 222-241.

Hars, A., and Ou, S. "Working for Free? Motivations for Participating in Open-Source Projects," *International Journal of Electronic Commerce* (6:3) 2002, pp 25-39.

Hauschildt, J., and Chakrabarti, A.K. "The Division of Labour in Innovation Management," *R&D Management* (19:2) 1989, pp 161-171.

Hauschildt, J., and Kirchmann, E. "Teamwork for Innovation - The 'Troika' of Promotors," *R&D Management* (31:1) 2001, pp 41-49.

Hecker, F. "Setting Up Shop: The Business of Open-Source Software," *IEEE Software* (16:1) 1999, pp 45-51.

Henderson, R.M., and Clark, K.B. "Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms," *Administrative Science Quarterly* (35:1) 1990, pp 9-30.

Henkel, J. "Open Source Software from Commercial Firms – Tools, Complements, and Collective Invention," *ZfB-Ergänzungsheft* (2004:4) 2004.

Henkel, J. "Champions of Revealing - The Role of Open Source in Commercial Firms," Technische Universität München, 2006a.

Henkel, J. "Selective Revealing in Open Innovation Processes: The Case of Embedded Linux," *Research Policy* (35:7) 2006b, pp 953-969.

Hertel, G., Niedner, S., and Herrmann, S. "Motivation of software developers in Open Source projects: an internet-based survey of contributors to the Linux kernel," *Research Policy* (32:7) 2003, pp 1159-1177.

Howell, J.M., and Higgins, C.A. "Champions of Technological Innovation," *Administrative Science Quarterly* (35:2) 1990, pp 317-341.

Jones, C. "Variations in Software Development Practices," *IEEE Software* (20:6) 2003, pp 22-27.

Katz, R., and Allen, T.J. "Investigating the Not Invented Here (NIH) Syndrome: A Look at the Performance, Tenure, and Communication Patterns of 50 R&D Project Groups," *R&D Management* (12:1) 1982, pp 7-19.

Katz, R., and Allen, T.J. "Organizational Issues in the Introduction of New Technologies," in: *The Management of Productivity and Technology in Manufacturing,* P.R. Kleindorfer (ed.), Plenum, New York, 1985, pp. 275-300.

Kim, Y., and Stohr, E.A. "Software Reuse: Survey and Research Directions," *Journal of Management Information Systems* (14:4) 1998, pp 113-149.

Kirsch, L.J. "Software Project Management: An Integrated Perspective for an Emerging Paradigm," in: *Framing the Domains of IT Management: Project the Future ... ... Through the Past,* R.W. Zmud (ed.), Pinnaflex, Cincinnati, OH, 2000, pp. 285-304.

Kirton, M.J. "Adaptors and Innovators: A Description and Measure," *Journal of Applied Psychology* (61:5) 1976, pp 622-629.

Kirton, M.J. *Adaption-Innovation: In the Context of Diversity and Change* Routledge, London and New York, 2003.

Klevorick, A.K., Levin, R.C., Nelson, R.R., and Winter, S.G. "On the Sources and Significance of Interindustry Differences in Technological Opportunities," *Research Policy* (24:2) 1995, pp 185-205.

Koenig, J. "Seven Open Source Business Strategies for Competitive Advantage," in: *IT Manager's Journal*, 2004.

Kogut, B., and Metiu, A. "Open-Source Software Development and Distributed Innovation," *Oxford Review of Economic Policy* (17:2) 2001, pp 248-264.

Kwon, T.H., and Zmud, R.W. "Unifying the Fragmented Models of Information Systems Implementation," in: *Critical Issues in Information Systems Research,* R.J. Boland, Jr. and R.A. Hirschheim (eds.), John Wiley & Sons, New York, 1987.

Lakhani, K., and von Hippel, E. "How Open Source Software Works: 'Free' User-to-User Assistance," *Research Policy* (32:7) 2003, pp 923-943.

Lakhani, K., and Wolf, B. "Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects," in: *Perspectives on Free and Open Source Software,* J. Feller, B. Fitzgerald, S. Hissam and K. Lakhani (eds.), MIT Press, 2005.

Lakhani, K.R., Jeppesen, L.B., Lohse, P.A., and Panetta, J.A. "The Value of Openness in Scientific Problem Solving," in: *HBS Working Paper Number: 07-050*, 2007.

Lehman, M.M. "Programs, Life Cycles, and Laws of Software Evolution," *Proceedings of the IEEE* (68:9) 1980, pp 1060-1076.

Leonard-Barton, D., and Deschamps, I. "Managerial Influence in the Implementation of New Technology," *Management Science* (34:10) 1988, pp 1252-1265.

Lerner, J., and Tirole, J. "Some Simple Economics of Open Source," *Journal of Industrial Economics* (50:2) 2002, pp 197-234.

Lyytinen, K., and Rose, G.M. "The Disruptive Nature of Information Technology Innovations: The Case of Internet Computing in Systems Development Organizations," *MIS Quarterly* (27:4) 2003, pp 557-595.

Markham, S.K., Green, S.G., and Basu, R. "Champions and Antagonists: Relationships with R&D Project Characteristics and Management," *Journal of Engineering and Technology Management* (8:3-4) 1991, pp 217-242.

McLure Wasko, M., and Faraj, S. "'It Is What One Does': Why People Participate and Help Others in Electronic Communities of Practice," *Journal of Strategic Information Systems* (9:2-3) 2000, pp 155-173.

McLure Wasko, M., and Faraj, S. "Why Should I Share? Examining Social Capital and Knowledge Contribution in Electronic Networks of Practice," *MIS Quarterly* (29:1) 2005, pp 35-57.

Merton, R.K., and Rossi, A.K. "Contributions to the Theory of Reference Group Behavior," in: *Social Theory and Social Structure,* R.K. Merton (ed.), Free Press, New York, 1949, pp. 225-275.

Moody, G. *Rebel Code - Inside Linux and the Open Source Revolution*, (1st ed.) Perseus Publishing, Cambridge, MA, 2001.

Nuvolari, A. "Open Source Software Development: Some Historical Perspectives," in: *Eindhoven Centre for Innovation Studies*, 2001.

Osterloh, M., and Rota, S. "Open Source Software Development–Just Another Case of Collective Invention?," 2005.

Raymond, E.S. "The Cathedral and the Bazaar," in: *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary,* E.S. Raymond (ed.), O'Reilly, Sebastopol, 2001a, pp. 19-63.

Raymond, E.S. "The Magic Cauldron," in: *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary,* E.S. Raymond (ed.), O'Reilly, Sebastopol, 2001b, pp. 113-191.

Rogers, E.M. *Diffusion of Innovations*, (5 ed.) Free Press, New York, NY, 2003.

Rothwell, R., Freeman, C., Horlsey, A., Jervis, V.T.P., Robertson, A.B., and Townsend, J. "SAP-PHO updated - project SAPPHO phase II," *Research Policy* (3:3) 1974, pp 258-291.

Royce, W.W. "Managing the Development of Large Software Systems: Concepts and Techniques," in: *Proceedings of the 9th International Conference on Software Engineering*, IEEE Computer Society Press, Monterey, California, United States, 1987.

Ryan, R.M., and Deci, E.L. "Self-Determination Theory and the Facilitation of Intrinsic Motivation, Social Development, and Well-Being," *American Psychologist* (55:1) 2000, pp 68-78.

Scacchi, W. "Free and Open Source Development Practices in the Game Community," *IEEE Software* (21:1) 2004, pp 59-66.

Senyard, A., and Michlmayr, M. "How to Have A Successful Free Software Project," 11th Asia-Pacific Software Engineering Conference (APSEC'04), 2004.

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

Sherif, K., Zmud, R.W., and Browne, G.J. "Managing Peer-to-Peer Conflicts in Disruptive Information Technology Innovations: The Case of Software Reuse," *MIS Quarterly* (30:2) 2006, pp 339-356.

Stewart, K.J., and Gosain, S. "The Impact of Ideology on Effectiveness in Open Source Software Teams," *MIS Quarterly* (30:2) 2006, pp 291-314.

Swanson, E.B. "Information Systems Innovation Among Organizations," *Management Science* (40:9) 1994, pp 1069-1092.

Taylor, W.G.K. "The Kirton Adaption-Innovation Inventory: A Re-Examination of the Factor Structure," *Journal of Organizational Behavior* (10:4) 1989a, pp 297-307.

Taylor, W.G.K. "The Kirton Adaption-Innovation Inventory: Should the Sub-Scales be Orthogonal?," *Personality and Individual Differences* (10:9) 1989b, pp 921-929.

Tornatzky, L.G., and Klein, K.J. "Innovation Characteristics and Innovation Adoption-Implementation: A Meta-Analysis of Findings," *IEEE Transactions on Engineering Management* (29:1) 1982, pp 28-45.

Tushman, M.L. "Special Boundary Roles in the Innovation Process," *Administrative Science Quarterly* (22:4) 1977, pp 587-605.

Tushman, M.L., and Scanlan, T.J. "Boundary Spanning Individuals: Their Role in Information Transfer and Their Antecedents," *Academy of Management Journal* (24:2) 1981, pp 289-305.

Varian, H.R., and Shapiro, C. "Linux Adoption in the Public Sector: An Economic Analysis," 2003.

Venkatesh, V. "Determinants of Perceived Ease of Use: Integrating Control, Intrinsic Motivation, and Emotion into the Technology Acceptance Model," *Information Systems Research* (11:4) 2000, pp 342-365.

Venkatesh, V. "Where To Go From Here? Thoughts on Future Directions for Research on Individual-Level Technology Adoption with a Focus on Decision Making," *Decision Sciences* (37:4) 2006, pp 497-518.

Venkatesh, V., and Davis, F.D. "A Theoretical Extension of the Technology Acceptance Model: Four Longitudinal Field Studies," *Management Science* (46:2) 2000, pp 186-204.

Venkatesh, V., Morris, M.G., Davis, G.B., and Davis, F.D. "User Acceptance of Information Technology: Toward a Unified View," *MIS Quarterly* (27:3) 2003, pp 425-478.

von Hippel, E. *Democratizing Innovation* MIT Press, Cambridge, MA, 2005.

von Hippel, E., and von Krogh, G. "Open Source Software and the 'Private-Collective' Innovation Model: Issues for Organization Science," *Organization Science* (14:2) 2003, pp 209-233.

West, J. "How Open is Open Enough? Melding Proprietary and Open Source Platform Strategies," *Research Policy* (32:7) 2003, pp 1259–1285.

West, J., and Gallagher, S. "Challenges of Open Innovation: The Paradox of Firm Investment in Open Source Software," *R&D Management* (36:3) 2006, pp 319-331.

Wheelwright, S.C., and Clark, K.B. "Accelerating the Design-Build-Test Cycle for Effective New Product Development," *International Marketing Review* (11:1) 1994, pp 32-46.

Wichmann, T. "Firms‘ Open Source Activities: Motivations and Policy Implications (Part 2)," Berlecon Research, Berlin.

Zmud, R.W. "Diffusion of Modern Software Practices: Influence of Centralization and Formalization," *Management Science* (28:12) 1982, pp 1421-1431.

Zmud, R.W. "An Examination of 'Push-Pull' Theory Applied to Process Innovation in Knowledge Work," *Management Science* (30:6) 1984, pp 727-738.