2010

CONCEPTUALIZING THE COMMONS-BASED PEERPRODUCTION OF SOFTWARE: AN ACTIVITY THEORETIC ANALYSIS

Pavel Andreev
Sami Shamoon College of Engineering, pavelan@sce.ac.il

Joseph Feller
University College Cork, jfeller@afis.ucc.ie

Patrick Finnegan
University of New South Wales, p.finnegan@unsw.edu.au

Jeffrey Moretz
University of Ontario Institute of Technology, jeff.moretz@uoit.ca

CONCEPTUALIZING THE COMMONS-BASED PEER-PRODUCTION OF SOFTWARE: AN ACTIVITY THEORETIC ANALYSIS

Completed Research Paper

Pavel Andreev
Sami Shamoon College of Engineering
pavelan@sce.ac.il

Joseph Feller
University College Cork
JFeller@afis.ucc.ie

Patrick Finnegan
Australian School of Business
University of New South Wales
p.finnegan@unsw.edu.au

Jeffrey Moretz
University of Ontario Institute of Technology
jeff.moretz@uoit.ca

Abstract

Commons-based peer-production (CBPP), as exemplified by community-based open source software (OSS) development, has been posited by Yochai Benkler as an alternative to hierarchies and markets for organizing the production of information goods. This study seeks to conceptualize viable CBPP through an Activity Theoretic analysis of 524 peer-reviewed OSS research artifacts. The analysis reveals the reliance of peer-production communities on complex systems of inter-related tools, rules, and roles as mediating components enabling communities to (i) exploit the two theorized advantages of CBPP (resource allocation and information processing) and (ii) overcome the two theorized challenges associated with this mode of production (motivation and organization). The study clarifies and extends extant understanding of CBPP in several significant ways, and concludes that in order for CBPP to be viable, participants must operate in a sustainable fashion that both enhances the commons and leaves the community intact.

Keywords: Commons-Based Peer Production, Open Source Software, Activity Theory

Introduction

Information systems researchers have been strongly influenced by the economic theories of Coase (1937), Hayek (1945) and Williamson (1973; 1981; 1991). This work explains the governance mechanisms within which production takes place as being either a managed hierarchy or market, although hybrid governance has also been studied (e.g., Jacobides and Billinger 2006; Santos and Eisenhardt 2005). Research within the broad stream of agency theory (see Eisenhardt 1989 for a review) has utilized a similar distinction between hierarchies and markets (behavioral and outcome contracting, respectively). Recently, the exclusivity of firms and markets as modes of production was challenged, when Benkler (2002; 2006) drew on early manifestations of the open source software (OSS) phenomenon to propose "a third mode of production" that he labeled "Commons-based peer-production" (CBPP). Arguing that OSS projects represented a significant departure from the normal signals that characterize hierarchy- and market-based models of production, Benkler generalized from the OSS phenomenon to characterize how large-scale collaborations in a digitally networked environment could produce a range of information goods (Benkler 2002,2006; Benkler and Nissenbaum 2006).

Commons-based peer-production refers to "a model of social production, emerging alongside contract- and market-based, managerial-firm based and state-based production" (Benkler and Nissenbaum 2006, p. 400). It is, primarily, an "Internet-based effort whereby volunteers contribute project components, and there exists some process to combine them to produce a unified intellectual work" (Krowne 2005, p. 5). Many other terms have been applied to similar phenomena. For example, Felin et al (2009) identifies "communities of practice", "collaborative communities", "peer and social production", and "governance in community forms" as organizational forms that emphasize community action within a knowledge economy. Sawhney (2003) refers to "cooperative design" to illustrate social/community elements required for peer production of complex, knowledge-intensive objects. "Bazaar governance" has also been applied to refer to peer collaborations (Demil & Lecocq 2006). Haythornthwaite (2009) bifurcates the domain into "crowdsourcing" and a "virtual community model" which represent "lightweight peer-production" and "heavyweight peer-production" respectively.

Specific examples of peer-production activity abound. Benkler (2002) and Benkler & Nissenbaum (2006) provided a list of peer production examples including the GNU/Linux operating system, Apache web server, Perl and BIND, SETI@home, NASA Clickworkers, Wikipedia, Slashdot, Kuru5hin, etc. This list has been extended by numerous studies investigating similar phenomena. Given the lack of broadly accepted, specific definition of CBPP, we endeavored to clarify the domain and to more precisely identify our area of focus for this study. We found the term peer production applied to a variety of activities that we classified into four distinct types:

- **Peer (SW) development projects** – free/libre/open-source software projects are the most prominent example of peer production with tens of thousands of projects (e.g., Linux, Apache, Perl, Mozilla, OpenOffice, and Moodle)
- **Peer resource sharing activities** – this category includes peer distribution, processor sharing, and communications infrastructure sharing (e.g., P2P file sharing/distribution, SETI@home, Skype, and WiFi daisy-chaining)
- **Peer filtering/assessment endeavors** – these endeavors include consumer ratings/reviews as well as micro-participation in parsing text, graphics, etc. like CitySearch, Amazon's product ratings, YouTube's video ratings, NASA's Clickworkers, and Games With A Purpose (aka GWAP) such as the ESP Game ("Google Image Labeler")
- **Peer content collaborations** such as Wikipedia, Slashdot, Wikichains, OpenStreetMap, or Kuro5hin

It might even be argued that academic research, itself, is an example of the peer production of knowledge. At the very least it is an example of open innovation. In addition there are on-going attempts to apply peer production approach to physical goods (e.g. Bug Labs, Openmoko project). This study focuses on more "heavyweight" peer development projects as exemplified by open source communities engaged in the production of software.

Benkler (2002) emphasizes the emerging phenomenon of commons-based peer production and calls for research. Although the importance of investigating this new production model is widely acknowledged and numerous studies have begun investigating different aspects of peer production, further examination of the phenomenon is needed. In response to Benkler's call to examine the enabling and operational elements of CBPP, some studies (e.g., Bhadauria 2007; Haythornthwaite 2009; Pouwelse et al. 2008; Wubishet 2009) have highlighted discrete questions concerning

social norms, technologies, and motivational issues. Researchers have also examined the organizational aspects of peer production (e.g., Demil and Lecocq 2006; Viégas et al. 2007) and have considered broader questions of the success of peer production (e.g., Demil and Lecocq 2006; Wubishet 2009). However, there are to date no comprehensive responses to Benkler's call to better understand CBPP as a viable mode of production; specifically to understand how the two advantages that CBPP offers over hierarchy and market models (information processing and resource allocation) can be leveraged in the context of the two prevailing problems with the phenomenon (organization and motivation) such that peer production can match "the best available human capital to the best available information inputs in order to create information products" (Benkler 2002, p. 444).

We contend that as CBPP relies on the largely independent actions of a large number of dispersed participants rather than normal hierarchical and market structures, an understanding of the articulated advantages and problems can only be gained by studying how participants in a CBPP environment deal with such issues over time. We argue that participants in such environments will devise (in a peer-produced manner) practices and tools that leverage the advantages of the mode of production as well as overcome its inherent problems. The objective of this paper is to *conceptualize viable commons-based peer-production through an analysis of the social production of open source software*. In doing so, we seek to understand how OSS communities, as the most prevalent form of CBPP, (i) leverage the information processing and resource allocation advantages and (ii) overcome the organization and motivation problems.

The next section presents a theoretical grounding of peer-production focusing on information processing and allocation advantages, as well as on motivation and organization challenges. This is followed by a description of the research design, which is an analysis of 524 papers published between 1998 and 2009 using Activity Theory (see e.g., Engeström 2000; Leont'ev and Hall 1978) as a conceptual lens. Our analysis reveals that (i) resource allocation is a potential, rather than actual advantage, which is inherently dependent on the health of the "commons", and that (ii) the information processing advantage represents an actualization of the resource allocation advantage and is dependent on broad participation and effective filtering. In addition, our analysis (iii) clarifies that nature of the motivation problem by revealing that motivational elements are scalable and contextual, and (iv) illustrates the necessity of coordination in the context of the organizational problem in order to facilitate the integration of individual effort necessary to produce a unified information good. The paper concludes that although the degradation of the commons in the traditional sense (Hardin 1968) is not a major issue in CBPP (due to the non-rival nature of resources), failure to continually nurture and improve the commons is a major threat to the viability of this mode of production.

Commons-based Peer-Production

Unlike hierarchical and market modes of production, CBPP "relies on decentralized information gathering and exchange to reduce the uncertainty of participants" (Benkler 2002, p. 375), and is characterized by (i) decentralization of action and decision-making and (ii) utilization of social cues instead of prices or managerial authority as a means of motivating and organizing efforts (Benkler and Nissenbaum 2006). These characteristics are also evident in other organizational forms that emphasize community action within a knowledge economy, such as 'communities of practice', 'collaborative communities', 'peer and social production', 'governance in community forms' (Felin et al. 2009), 'bazaar governance' (Demil and Lecocq 2006) and 'cooperative design' (Sawhney 2003), and are common to both 'lightweight' peer-production models such as crowdsourcing and more involved or 'heavyweight' models such as virtual communities (see Haythornthwaite 2009).

Benkler (2002) argues that three structural characteristics are necessary in order to facilitate peer production: modularity, granularity, and low-cost integration. Modularity means that the object of production must be divisible into components (modules), which can be independently produced and improved. Granularity means that such modules should be sufficiently small scale to enable a large number of participants to complete them without exhausting their intrinsic motivation to do so. Finally low-cost integration means that the modules can be integrated in a meaningful way to create overall value at a low cost.

Given these structural characteristics, CBPP potentially offers two major advantages over hierarchies and markets; *resource allocation* and *information processing* (Benkler 2002). The *allocation* advantage is intimately related to the commons-based nature of peer-production; when the resources necessary for production are placed in the commons, any agent may work with them, thereby avoiding the constraints associated with allocation in hierarchical and market models. Prior research has noted the advantage of unfettered allocation in open knowledge production. For

example, Uhlir (2003) argues that commons-based regimes are akin to the "republic of science" and that they produce better outcomes than proprietary knowledge regimes in many circumstances, including academic research. It is therefore not surprising that Benkler (2002) notes that the broad domain of information/knowledge work subsumes nearly all examples of commons-based peer-production, due to the ability for near costless distribution of resources for widespread utilization. The *information processing* advantage derives from the ability of myriad distributed agents both to self-identify for tasks and to carry out those tasks utilizing idiosyncratic information/knowledge. The products of such distributed work are then peer reviewed, again often in distributed fashion, to filter for acceptable solutions that best fit the overall project.

The ability for CBPP communities to leverage theses advantages is dependent on overcoming two problems associated with CBPP; motivation and organization. The *motivation* problem stems from the fact that participants in CBPP to not typically receive the material rewards (e.g. payment, profit) that typify hierarchies and markets. Benkler (2002) posits two potential remedies to the motivation problem. The first proposes that material rewards can coexist with intrinsic hedonic and socio-psychological rewards. Indeed, Benkler suggests that there are situations where projects with no financial rewards attract people that might otherwise be unlikely to be interested in participation. The second remedy is related to the fine-grained modularity of CBPP, where low levels of motivation may be sufficient to complete many modularized tasks.

The *organization* problem of CBPP emerges with the need to (i) coordinate human effort and (ii) integrate individual output in a low cost manner, while also preserving the motivational characteristics noted above. Organizing effort in CBPP differs from the classical hierarchy and market modes of production. Governance in hierarchies is achieved through the *visible hand* of internal authority via the employment relationship that governs interactions between agents by means of behavioral contracts. In market governance, economic interactions are regulated by the *invisible hand* that governs the relationships between agents/firms by means of the price mechanism. In contrast to these two modes, the governance of CBPP is a challenge since it requires governing a largely volunteer community in which neither the *visible hand* of internal authority nor the *invisible hand* of price and demand signals are practical (cf., Benkler 2002).

Research Design

The objective of this study is to *conceptualize viable commons-based peer-production through an analysis of the production of open source software.* In doing so, we seek to understand how OSS communities, as the most prevalent form of peer-production (i) leverage the information processing and resource allocation advantages and (ii) overcome the organization and motivation problems. We thus seek to identify the central elements of OSS projects and communities that facilitate and mediate their activities in producing digital artifacts, especially software, but also service, support, and development methodologies related to that software. We seek to describe how these elements combine in such a way to produce viable commons-based peer-produced digital artifacts. We contend that as commons-based peer production does not rely on internal authority or market signals, an understanding of these issues can only be gained by studying how participants in a commons-based peer-production environment deal with such issues over time. In particular we contend that participants in such environments will devise (in a peer-produced manner) practices and tools that leverage the advantages of the mode of production as well as overcome its inherent problems.

The study was operationalized through the analysis of 524 peer-reviewed OSS research artifacts from 1998 to 2009. Similar approaches have been used at both a disciplinary level - for example within Information Systems (Alavi and Carlson 1992; Chen and Hirschheim 2004; Claver et al. 2000; Farhoomand and Drury 1999; Orlikowski and Baroudi 1991), Software Engineering (Glass 2003), and Computer Science (Ramesh et al. 2004) - and at a thematic or sub-field level (e.g., Crowston et al. 2008; Ghosh et al. 2002; Romano and Fjermestad 2001). Previous analyses of extant research have limited their sample to a specific number of outlets (e.g. Chen and Hirschheim 2004; Farhoomand and Drury 1999) or to the paradigmatic focus into which the research falls (e.g. Chen and Hirschheim 2004; Orlikowski and Baroudi 1991). In order to take into account the multi-disciplinary nature of research in the area of OSS, our literature search was not confined to specific publications within any one discipline. Rather, a strategy of exploring a range of outlets was employed, following similar efforts by Romano and Fjermestad (2001).

Candidate papers were discovered through a combination of (i) keyword searches of citation indices (e.g. EBSCO, Science-Direct, IEEE, ACM Portal), (ii) existing bibliographies of OSS research, and (iii) citation recursion. In addition to a range of journals from various disciplines, a variety of international conferences and books were also

reviewed. In total, 524 research artifacts were reviewed. Of these, 265 were journal papers, 223 were conference papers and the remaining 36 consisted of various books, reviews, reports, commentaries etc.

Classification systems facilitate knowledge sharing and the communication of the contents of a field of study, thereby "allowing researchers and practitioners to generalize, communicate, and apply the findings." (Vessey et al. 2005, p. 249). For the purpose of this analysis, we used Activity Theory (AT) as a framework for exploring, categorizing, and analyzing OSS phenomena. We contend that AT provides a useful classification system for the myriad factors associated with OSS production, allowing more focused yet comprehensive analysis of the phenomena of interest in this examination of OSS as peer production. Thus, activity theory provides the basic structure around which we develop our conceptualization of the peer production of software.

Activity Theory has its origins in the concept of artifact-mediated and object-oriented action, whereby a human being's interactions with their environment are not direct, but are mediated by physical and cognitive tools (Leont'ev and Hall 1978; Vygotsky 1978). Engeström (1987) expands on this interaction-oriented view of human activity to reflect the collective and collaborative nature of such activity by incorporating two additional mediators: (i) the rules and norms and (ii) the roles and responsibilities that together mediate the relationships between individuals, their community and their shared objectives. Engeström (1999) argues that the study of mediated activity should consider how elements interact with each other within an activity system. Thus, while an "activity" is the basic unit of analysis, AT encompasses participating social actors, the technological and non-technological tools they employ, the rules and norms of the social or socio-technical context, and the roles and responsibilities of participating actors. As AT is focused on human *inter*-action rather than isolated human action, it is particularly well suited to an analysis of commons-based peer-production (cf., Engeström 2008), particularly OSS development (cf., Hemetsberger and Reinhardt 2009) due to the computer-supported cooperative nature of the work (Barthelmess and Anderson 2002).

Using the search principles of systematic reviews (cf. Kitchenham 2004), a thorough search of the existing OSS literature (described above) was followed by structured content analysis (cf. Smith et al. 1991) undertaken by two of the authors. This analysis was performed using codes derived to represent the conjunction of activity theory, open source software, and commons-based peer-production. Codes focused on the elements of activity theory that mediate subject interactions with objects within the context of a community – specifically the tools, rules, division of labor (roles) that facilitate the performance of the activities that comprise community-based OSS projects. To ensure coding uniformity and to reduce coding ambiguity, coding was undertaken in an independent fashion by the two coders. Coding comparisons showed a high degree of inter-rater reliability (cf. Gwet 2001). However, all disagreements were referred to a third author as mediator, with analysis only being accepted when all three agreed, thus increasing trustworthiness (validity) and replicability (reliability) (cf. Denzin and Lincoln 2000; Gay et al. 2006). This analysis used existing theoretical and empirical research and leveraged activity theory to develop a clearer conceptualization regarding CBPP as exemplified by OSS. In particular, our analyses focused on the elements of AT that mediate subject interactions with objects within the context of a community – specifically the tools, rules, division of labor (roles) that facilitate the performance of the activities that comprise OSS projects.

Analysis

Figure 1 presents an overview of the flow of peer production processes derived from our analysis of OSS research. Our analysis reveals that resource allocation is a potential, rather than actual advantage, which is inherently dependent on the health of the "commons". This potential is realized through the information processing advantage of commons-based peer-production, which is dependent on broad participation and the effective filtering of knowledge and effort. However, both broad participation and effective filtering require the ability to overcome the motivation problem by leveraging diverse, scalable, and contextually embedded motivations. However, the resulting action only ceases to be anarchic when the information processing advantage of commons-based peer-production is leveraged to coordinate and integrate action, thus overcoming the organization problem. However, while our analysis reveals the non-rival nature of information resources means that degradation is not a major issue, failure to continually nurture and improve the commons is a major threat to the viability of commons-based peer-production.

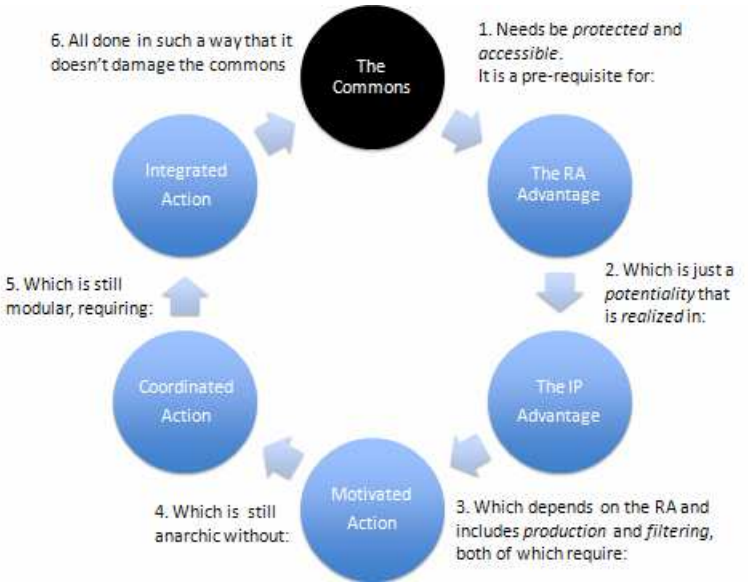

Figure 1. Analysis

The resource allocation advantage

In line with Benkler's theorization of CBPP, the allocation advantage is clearly observable in OSS production, where the universal availability of resources to all agents allows OSS projects to avoid allocation barriers traditionally associated with other forms of production. Our analysis also revealed, again in line with Benkler, that this advantage is inextricably linked to the Commons-Based nature of OSS projects. Specifically, we observed two inter-related systems of tools and rules which mediate OSS project activities in such a way to ensure that the commons on which development depends is (i) protected and (ii) made universally accessible to all agents.

In terms of protecting the OSS commons, legal tools (in the form of licensing and other intellectual property rights mechanisms) and social rules (norms) both help to ensure that project resources (such as code, documentation, knowledge-bases and project brand) are not misappropriated. License choice is a critical decision for an OSS project, providing the legal foundation for protecting common property and reinforcing the ideology and norms of the community (Bonaccorsi and Rossi 2003; German 2003). Legal tools allow intellectual property be made freely accessible and redistributable while applying desired restrictions (O'Mahony 2003), while social rules, such as normative sanctions, also prove useful in defending the identity of a project (Dahlander and Magnusson 2005; O'Mahony 2003), and preventing the misappropriation of a project's output in a manner contrary to the norms and values of the community (Gambardella and Hall 2006).

Legal tools and social rules also play a role in ensuring that the resources within the "protected" commons are widely accessible. Appropriate legal tools help to promote wider circulation of resources, and both norms and licensing support coordination mechanisms (Bonaccorsi and Rossi 2003; Gambardella and Hall 2006), which in turn facilitate allocation advantages, particularly in the context of cross-project coordination and the creation of a broader commons. Additionally, a wide variety of technological tools (together with the social rules dictating their use) serve to enhance the accessibility of the commons, enabling collaboration to take place both within the core of a project community as well as at the projects periphery, for example the involvement of users in defect detection and reporting (Porter et al. 2006). The use of web-based tools promotes widespread availability of all project artifacts/resources (Desouza et al. 2007; German 2003) producing a collective infrastructure (Jensen and Scacchi 2005b) and stimulating broader involvement (Lane et al. 2004b). Such tools ensure that the discourse and knowledge of the community – not just the code artifacts of the project – are accessible as resources in the commons (e.g., German 2003; Lee and Cole 2003; Robbins 2005). In addition to facilitating development, such availability also enables the provision of community support (Lakhani and von Hippel 2003). The commons can thus be understood as a persistent store of information that is accessible by any project member at any time (Sowe and Stamelos 2008), thus enabling the distributed, collective action, knowledge sharing and learning that characterize

open source software development (Markus et al. 2000; Moon and Sproull 2000; Osterloh and Rota 2007; Ye and Kishida 2003). Finally, the OSS development norm of "bootstrapping" means that the tools used in OSS production are also themselves often products of OSS projects (Robbins 2002; 2005), allowing for their widespread and nearly costless dissemination.

The information processing advantage

Manifestations of the CBPP information processing advantage (the ability of myriad distributed agents both to self-identify for tasks and to carry out those tasks utilizing idiosyncratic information/knowledge) were readily observable in the context of OSS development. Just as the resource allocation benefit was found to be dependent on a protected and accessible commons, our analysis revealed the information processing advantage to be dependent on maximizing the breadth of participation in OSS and on effectively filtering distributed knowledge and effort. Again, these goals are achieved in the OSS context through inter-related systems of rules, tools and roles.

Broad Participation

A wide variety of tools are utilized in OSS projects to stimulate and enable broad participation. Our analysis highlights two key tool characteristics as particularly central to this goal. First, the modular character of the software artifacts themselves enables the decomposition of project effort into smaller divisions of work and legitimizes peripheral participation (Ye and Kishida 2003). Modularity enables self-selection, allowing for greater fit between a task and a developer's interests/skills (Asklund and Bendix 2002; Basnet and Lane 2005; Crowston et al. 2005b). Furthermore, modular design facilitates the parallel development of multiple solutions to any given problem (Baldwin and Clark 2006; Ye and Kishida 2003), making solution space searching more effective and efficient (Baldwin and Clark 2006) and easing the communication barriers to large team effectiveness (Capiluppi and Adams 2009). Second, broad participation is enabled by the use of readily available, often web-based common tool kits, which lower entry costs and enable quick onboarding of new community members (Hemetsberger and Reinhardt 2009; Robbins 2005). Version control systems (VCS) are a critical enabler of distributed development activities, providing a centralized, controlled repository of code for the project (Fogel 2005; Robbins 2005). Such systems enhance the transparency of the development process while providing universal access to the code (Robbins 2002,2005), facilitating wider participation in information processing tasks of all kinds. Newsgroups, discussion forums, and email are critical communication tools for knowledge exchange among users (Lakhani and von Hippel 2003; Lee and Cole 2003; von Krogh et al. 2003). These tools may be used to enhance project activities in conjunction with other tools. For example, tools such as Bugzilla structure bug reporting activities (Robbins 2005), though early reports of problems may show up in user forums or mailing lists prior to being documented in bug tracking systems (Iivari 2009).

Individually, members generally deal with single questions, answers, and enhancement ideas. Through the use of a common set of tools that provides a persistent, searchable collection of these individual elements, communities are able to enhance their collective knowledge and capabilities (Hemetsberger and Reinhardt 2004,2006; Sowe and Stamelos 2008). Such tools also help to increase the project's absorptive capacity, it's ability to acquire and use more new knowledge (Daniel 2006; Daniel et al. 2006). Communication tools provide persistent stores of information, making it accessible to any project member at any time (Sowe and Stamelos 2008), thus enabling the distributed development and collective utilization of knowledge that characterize open source software (Markus et al. 2000; Moon and Sproull 2000; Osterloh and Rota 2007; Ye and Kishida 2003). Thus, as noted by Benkler (2002), the capacity of the community to increase the collective pool of knowledge is predicated upon the fact that it is internet-based, enabling the distributed production of collective knowledge (Asklund and Bendix 2002; Daniel 2006; Hemetsberger and Reinhardt 2004).

Rules of behavior also play a significant role in facilitating information processing. Rules such as reciprocity, reputation/meritocracy, users as co-developers, and transparency/peer review help to create a community dynamic that is conducive to information exchange and the development of new knowledge. The existence of such rules encourages the participation of members of the wider project community (German 2003; Sowe et al. 2006). For example, treating users as co-developers encourages broad participation that enables the massive, parallel user testing and feedback (Franck and Jungwirth 2003; German 2003; Lee and Cole 2003) that are such critical components of open source development (Bagozzi and Dholakia 2006; Daniel 2006; Lee and Cole 2003; Midha 2008; Sowe et al. 2006; Sowe and Stamelos 2008). Activities such as user provision of support (Lakhani and von

Hippel 2003) also leverage rules including reciprocity and reputation/meritocracy through the inclusion of signatures and email addresses with posts, enabling reputation effects (Bergquist and Ljungberg 2001; Lakhani and von Hippel 2003; Roberts et al. 2006; Zeitlyn 2003). Many developers begin their involvement in projects as peripheral participants assisting with bug tracking (Fang and Neufeld 2009) while developing their skills and credentials. This kind of distributed action has been characterized as legitimate peripheral participation (Lave and Wenger 1991), with participant learning opportunities as among the central reasons for continued participation (Fang and Neufeld 2009). Community norms regarding knowledge sharing are also important in development activities (e.g., Asklund and Bendix 2002), and such knowledge sharing facilitates learning (Hemetsberger and Reinhardt 2004; Sowe and Stamelos 2008) and improves group performance (Au et al. 2009; Daniel et al. 2006).

Common, accessible infrastructures reinforce social rules by supporting observation of community practices by newcomers (Hemetsberger and Reinhardt 2004), and their effectiveness is enhanced by rules, such as those regarding transparency, that promote work practices that make all project work products widely available in order to foster shared activity (Bagozzi and Dholakia 2006). Together, these infrastructural elements enable OSS communities to reconcile potential losses in efficiency which stem from duplication of effort (cf. Porter et al. 2006; Sowe and Stamelos 2008) with the advantages gained from the existence of a large, diverse and knowledgeable pool of participants (Daniel et al. 2006; Porter et al. 2006).

OSS projects promote widespread involvement through the use of a variety of roles, sometimes formally defined but often informal in nature, through which members can participate. Numerous roles facilitate user involvement (Koch and Schneider 2002; Lakhani and von Hippel 2003; Scacchi 2004; Ye and Kishida 2003) covering a range of engagement from minimal, occasional contribution in limited domains to extensive, regular involvement across multiple project activities. Numerous developer roles have also been identified (Glass 2003; Shaikh and Cornford 2009; Tsiavos and Hosein 2003; Ye and Kishida 2003), increasing the feasibility of participants with varied backgrounds and different skill sets to participate productively, while also facilitating diverse degrees of involvement (Sawhney 2003; Tsiavos and Hosein 2003). Such breadth of modes of participation facilitates the active involvement of disparate contributors who may have substantially different interests and skill-sets. Thus, when the number of users becomes quite large (e.g., Jensen and Scacchi 2005b; Lee and Cole 2003), the capabilities provided by various tools, rules, and roles allow even low rates of active participation to yield substantial net contributions to projects. Though some mechanism must be in place for integrating a variety of capabilities and perspectives (see next section), such broad, distributed deliberation and action represents precisely the information processing advantage proposed by Benkler.

Filtering

Filtering of project contributions is a significant element of OSS development efforts. Such filtering is amenable to the same distributed work processes that characterize other development tasks. Indeed, in OSS production filtering is in essence another information processing task, and therefore many of the same tools, rules, and roles identified above as facilitating information processing also facilitate filtering, through the process generally referred to as peer review (Franck and Jungwirth 2003; Robbins 2005; Wijnen-Meijer and Batenburg 2007). The fact that filtering, in addition to other production activities, can be provided via extremely broad peer review is among the key benefits claimed by open source advocates (e.g., OSI 2009) and supported by the empirical literature (Asundi and Jayant 2007; Lee and Cole 2003; Waguespack and Fleming 2009).

Tools such as VCS make the most current code base readily accessible to users (Robbins 2005; Scacchi 2004). OSS peer review is particularly reliant upon the common data store provided by such coordination tools (Robbins 2005) and is facilitated by communication tools that enable rich, complex discussion by a potentially widely dispersed group of contributors pursuing diverse tasks and leveraging varied skills and perspectives (German 2003; Jensen and Scacchi 2005b; Lee and Cole 2003; Long and Yuan 2005; Nichols and Twidale 2006). In addition, modularity and granularity facilitate filtering as project participants can engage in review activities regarding more focused elements that better fit their expertise and interests (Demil and Lecocq 2006; Lattemann and Stieglitz 2005).

Rules, particularly the expectation of peer review, are central to enabling broad peer filtering to function effectively. Participants in commons-based projects should be open to constructive criticism and comments (Franck and Jungwirth 2003; Lee and Cole 2003; Waguespack and Fleming 2009). The openness of the code base permits any participant to engage in peer review activities (Franck and Jungwirth 2003; Sowe and Stamelos 2008), leading to improved group outcomes (Basnet and Lane 2005). Transparency rules facilitate work practices such as knowledge and code reuse, helping to speed development and increase quality (Haefliger et al. 2008; von Krogh et al. 2005),

and also play a role in ensuring widespread availability of the code for anyone to review, comment, and provide suggestions for improvement (Franck and Jungwirth 2003). Rule driven openness and transparency, coupled with tool-mediated immediate availability of code artifacts facilitates faster feedback loops between core an periphery (Porter et al. 2006), including the parallel use and evaluation of the artifacts as they are produced that is a key value proposition of open source projects with large and active communities (Feller and Fitzgerald 2002; Hemetsberger and Reinhardt 2009).

While wide participation is highly encouraged, and reputation may also play a role in determining who is allowed to contribute code (Iivari 2009; Ye and Kishida 2003), roles also play an important role in peer review activities. Core developers occupy a particularly prominent position in such activities because of their role as committers, regulating the admission of code to the project's central repository (Asklund and Bendix 2002; Mockus et al. 2002; von Krogh et al. 2003). These participants in peer review serve a gate-keeping function, ensuring that only the highest quality submissions are accepted for use in the community's products (AlMarzouq et al. 2005; Lane et al. 2004b)

Motivation challenge

Hierarchies and markets both rely heavily on financial reward to motivate and direct action, with particular focus on the need to deal with agency concerns regarding divergent goals (see Eisenhardt 1989). This focus on financial rewards to overcome agency concerns is reflected in the many criticisms of CBPP regarding the lack of monetary incentives as a source of motivation. Our analysis of the OSS literature reveals three approaches to answering the motivation challenge: leveraging diverse, scalable, and contextually embedded motivations.

Diverse motivations accommodated

One of the chief advantages of OSS peer production is the ability to leverage multiple sources of motivation. "*The diversity of motivations allows large-scale collaborations to convert the motivation problem into a collaboration problem*" (Benkler 2002, p.434). Benkler defines three types of preferences for rewards: monetary rewards, intrinsic hedonic rewards, and social-psychological rewards. Both intrinsic and extrinsic motivations are important in facilitating the knowledge sharing that underlies much OSS production (Kaiser and Müller-Seitz 2008), though care must be taken that the use of some types of rewards does not diminish other motivations (cf. crowding-out, see Feigenbaum et al. 2009; Osterloh and Frey 2000 for OSS application).

The review of OSS literature reveals numerous attempts to categorize the diversity of motivations (e.g., 2000; Hars and Ou 2001; 2003; 2003; 2005; 2002). Motivational categories including intrinsic and extrinsic, immediate and delayed, and personal, social, reward, and identification have been put forward. Research in OSS has documented these myriad potential motivating factors (e.g., Asklund and Bendix 2002; Basnet and Lane 2005; Crowston et al. 2005b), in addition to the possibility of financial motivations through paid development work (Capra et al. 2008), confirming the potential for leveraging diverse sources of motivation.

Many of the tools utilized in OSS projects facilitate leveraging disparate motivations. For example, highly modular designs (a 'soft' tool) coupled with the ability of participants to self-select which element of the project to work on promotes intrinsic motivations such as fun (German 2003; Roberts et al. 2006) or learning (Franck and Jungwirth 2003; Lakhani and von Hippel 2003; Ye and Kishida 2003) and allow greater fit between the necessary tasks and developer interests and skills yielding more effective motivation and control (Asklund and Bendix 2002; Jensen and Scacchi 2005a; Wang 2005). Participants in OSS projects may also be motivated by the use value of the product being developed, though studies have found learning to be a stronger motivation in many cases (Subramanyam and Xia 2008; Ye and Kishida 2003). In addition to myriad self-interested motivational elements, altruistic motivation is also accommodated (Demil and Lecocq 2006), though such motivations may interact with social motivations such as belongingness, identity, and feelings of kinship (Bagozzi and Dholakia 2006; Lakhani and Wolf 2005; Zeitlyn 2003), and altruistic actions need not be motivated by altruism (Gleave et al. 2009).

Scalable motivation

Fine-grained modularity allows projects to utilize small-scale contributions that entail relatively minimal effort. The ability to leverage smaller motivations for peripheral participation that requires relatively little effort is a significant advantage. However, the critical advantage of OSS PP is that projects are able to leverage diverse sources of

motivation *and* diverse magnitudes of motivation, as well. Many tasks require substantial commitment of time and effort, requiring greater motivation. For example, some tasks, often though not always those that are less glamorous or attractive to volunteer participants, might be undertaken by paid participants who generally receive a salary from a corporate sponsor (Bonaccorsi and Rossi 2003), though paid developers are also often drawn from those who have volunteered (German 2003) or have self-selected though they are being paid (Raymond 2000). Roles may be used to facilitate motivation, with meritocratic role appointment serving as a reputation enhancing reward for participants who have provided valuable service to a project, including programming and non-programming contributions as well as tenure with the project (German 2003; Jensen and Scacchi 2005a; O'Mahony and Ferraro 2007). Attaining high status positions may also lead to eventual fiscal reward (Hann et al. 2002), likely via signaling effects (Lerner and Tirole 2002; Roberts et al. 2006).

Contextually embedded motivation

There is an underlying social character to all motivations in OSS peer production. Motivations based on belonging, kinship, or identification rely on a social, community context. Some participants may be motivated by a sense of identification with OSS broadly or with specific projects that give rise to distinct community identities (David and Shapiro 2008; Elliott and Scacchi 2008; Felin et al. 2009; Hars and Ou 2001), such as that of Linux with its well known mascot and large number of passionate users and developers (e.g., Hertel et al. 2003). Reputation rewards are reliant upon some degree of continuity of the underlying project community in order for reputational benefits to have meaning. Firm participation, including paid participation by the employees of a sponsor firm, needs to take place within the broader community context if a viable OSS project community is to be fostered and preserved (Raymond 2000; Shaikh and Cornford 2009).

Many of the tasks that involve greater degrees of interaction among project members leverage motivations that are social in nature. Such motivations are often complementary to intrinsic motivations such as fun or learning. Reputation effects are a key extrinsic reward for OSS participation when members of the community value the judgments of their peers or the broader community. Rules regarding transparency regarding the authors of submissions facilitate the invocation of the meritocracy and reputation rules that contribute to motivation in a variety of project tasks. The tools utilized by OSS projects often facilitate the invocation of community based motivations by ensuring that the contributions of community members are visible. For example, the signatures and email addresses of users are typically included with posts, enabling reputation effects (Bergquist and Ljungberg 2001; Elliott and Scacchi 2008; Lakhani and von Hippel 2003; Roberts et al. 2006; Zeitlyn 2003).

In addition, many projects develop a pervasive community character that inspires a sense of belonging among participants. Participants often develop strong identification with specific projects and a feeling of kinship with members of the community. Such feelings of belonging and kinship may also augment the activation of norms regarding reciprocity. Thus participants may be motivated by social ties and feelings of kinship (Zeitlyn 2003; Zhang 2006), their identification or sense of involvement with a particular project or community (Bagozzi and Dholakia 2006; Xu et al. 2009; Zhang et al. 2007), or a feeling that they are indispensable to a project's success (Hertel et al. 2003). Participants may be motivated by direct reciprocity, where their contributions are directed at a specific member from whom they have received some specific value, or by indirect reciprocity in which their contributions are in response to value derived from the community in general (e.g., Lakhani and von Hippel 2003). The relationships among community members have been discussed as social capital that is critical for OSS production and which is both drawn upon and created as members participate in communities to create OSS products (Okoli and Oh 2007; Tan et al. 2007; Wang 2005).

It is widely accepted that maintaining motivation among project participants requires trust (e.g., Gallivan 2001; Lane et al. 2004a; Lane et al. 2004b; Lattemann and Stieglitz 2005). Ensuring that community norms are followed is a necessary element in building the long-term trust (Lattemann and Stieglitz 2005) necessary for a sustainable OSS project. Adler (2001) goes so far as to include trust in the name of his third mode of production, referring to market/price, hierarchy/authority, and community/trust. As Adler's third mode implies, trust is fundamentally based in the community and is thoroughly contextual in nature. Reliance on reputation or merit of the individual, broadly construed, including his tenure with the project in determining who will fill certain project positions (O'Mahony and Ferraro 2007; Roberts et al. 2006; Ye and Kishida 2003) also leverages the social context of the community to facilitate motivation.

Organization Challenge

The organizational problem of CBPP emerges with the need to coordinate human effort keeping it motivated in order to accomplish different project elements in an effective way and integrate them in a low cost manner into the complete product. We distill five critical elements within the organization challenge. CBPP projects must first determine *what* is worth doing – what work needs to be done and with what priority. Then projects must determine *who* will do what – a classic division of labor question and *how* it should be done. These three questions address the *coordination* of organization related to developing individual contributions. The remaining two critical elements refer to the *integration* issue of CBPP organization. As individual contributions are made available, the community must determine *which* contributions to use and *how* these contributions should be integrated.

Coordination

Given the necessity of distributed action, both initial production and subsequent filtering, for the realization of information processing advantages and the lack of authority in order to organize collaboration, enabling effective coordination is problematic from the perspective of the traditional hierarchical organization paradigm. In CBPP the cause of the challenge is also often the basis of the solution.

What needs to be done?

A project's objectives are dynamic and dependent on broader project context. As a project matures, community consensus becomes a driving force in determining its objectives. Hence, although the idea of the OSS is self-determined/assigned it is still full controlled by community. Overcoming this organizational challenge related to defining boundaries, objectives, and ideology of an OSS project requires mutual agreement within the community regarding macroculture norms and beliefs. By analogy with traditional employment viewpoint, the community members should 'sign' a contract on the subject of shared values, norms, and beliefs. This contract is an essential condition of self-regulation within the community providing the sense of belongings to the community and facilitating trust. A common set of values and beliefs helps to create a sense of community identity (Lattemann and Stieglitz 2005; Ljungberg 2000) and promotes greater trust within the community (Stewart and Gosain 2006). Transparency is an essential rule for achieving consensus, which is inherent to democratic decision making regarding project objectives (German 2003; Tsiavos and Hosein 2003).

Who will do it?

In contrast to hierarchy organizational structure where division of labor is a centralized process with a highly directive controlling and coordination mechanism, CBPP is typically characterized by decentralized structure with generally self-assignment and self-regulation. The division of workforces into diverse, specialized categories requires dividing production process into numerous tasks. As noted above, one of the substantial advantages of open source software development is that, given an identified issue, *modularity and granularity* allow parallel development of multiple, different solutions (Baldwin and Clark 2006; Ye and Kishida 2003) following the rule of self-assignment, which is a dominant form of governance in most OSS project activities (Crowston et al. 2005b). In order to enable the key advantage of CBPP regarding information processing, some mechanism for coordinating across multiple, parallel development streams must be realized. Our analysis of the division of labor in open source software communities identified five categories of participants: *passive users, active users, peripheral developers, core developers,* and *leaders*.

Passive users, in our categorization, are minimally involved with a project community. These users may form the largest group in projects that produce products with wide end-user appeal (e.g., OpenOffice, Firefox), but they often participate in the community only through downloading and using the product. We also include in the passive user category those users who only (or primarily) ask questions in support forums. Distinction between those who actively engage in the community and those who more passively use the community is supported in the existing literature and (e.g., Iivari 2009; Lakhani and von Hippel 2003; Nichols et al. 2001; Nichols and Twidale 2006; Raymond 2000; von Hippel 2001).

Active users are among those referred to users as co-developers (Raymond 2000). In an open source community, non-technical users can be very effective co-developers as well (German 2003). Active users may provide knowledge in community support forums (Lakhani and von Hippel 2003), report issues and bugs in project artifacts (Ye and Kishida 2003), drive broader adoption of the community's product (Midha 2008; Miralles et al. 2005; Ven and Verelst 2008).

Developers in an open source software community produce the digital artifacts that represent the product that is the objective of the community's effort (Glass 2003). Developers are often classified into peripheral developers and core developers. A primary differentiator between the two groups is that peripheral developers have limited coordination roles (Feller et al. 2008; von Hippel and von Krogh 2003), focusing most of their efforts within the community on producing solutions to identified issues or bugs or providing new features roles (Asklund and Bendix 2002; von Hippel and von Krogh 2003). Core developers play a central coordination role in project development activities. Core developers have considerable control within OSS projects and determine what requirements will be implemented and in what order (Gallivan 2001; German 2003). These developers typically have commit access to the VCS (von Krogh et al. 2003) and are responsible for ensuring effective division of the work to multiple developers or developer teams. Some empirical work has found that projects with more core developers tend to be more successful (e.g. Long 2005), but research also finds that core groups still tend to be small and suggests that the use of computer mediated communication does not increase the maximum size of free-forming groups (Crowston et al. 2006).

The category of *leaders* includes project and product management roles that often cross code modules and associated development group boundaries, in addition to broader community-focused leadership roles such as providing vision, building group identity and providing a public face for the project. Project leaders help to attract developers to the project and publicize the project and its products. They also play a role in conjunction with core developers in preventing project splintering through forking. We place release managers and release teams in the leadership category because of their broad scope of responsibilities across most of a project's development activities. These leaders have higher level coordination responsibilities that subsume multiple development streams and modules (German 2003; Jensen and Scacchi 2005b).

Participants of an OSS project are free to self-assign themselves to any available task or a number of tasks from the pool of project actions and thus self-identify for different roles. However, self-assignment does not necessarily lead to a division of labor that will satisfy the needs of the project since it requires obtaining community agreement regarding distribution of project tasks and associated roles. For instance, though a developer might deliver a solution to fix a specific bug, the community may decide to accept another solution. A developer can proclaim herself as a leader, however, the broader community might decide not to follow her and instead select another leader. Working from this list of issues, bug fixers can self-select which issues they will seek to resolve, though in some cases specific developers are assigned to fix issues, particularly issues identified as critical (Glance 2004). For example, some developers may be assigned to specific tasks by their employer company if that company is a participant in an OSS project. Notwithstanding that such developers are paid to fill specific roles, their contribution to the project is embedded in the project context and conditional on community acceptance.

How it should be done?

CBPP projects typically deal with coordination issues utilizing much the same approach that enables the information processing advantage. Self-assignment, driven by a variety of motivational factors, allows projects to leverage distributed information in the hands of individual agents regarding what needs to be done and who is capable of doing it. Open source communities typically have work structures that are self-assigned, flat, flexible, and distributed. Self-assignment is one of the signature elements of most open-source software communities (Crowston et al. 2005a) with self-imposed deadlines and self-control predominating (Robbins 2005; Zhao and Deek 2005). Given the volunteer nature of many contributions in open source projects (von Krogh et al. 2005; Xu et al. 2005), the positive relationship between self-assignment and motivation is critical (Asklund and Bendix 2002). It has also been argued that self-assignment leads to task specialization (Lattemann and Stieglitz 2005) and is central to the effectiveness of developer efforts (Basnet and Lane 2005; Crowston et al. 2005a; von Krogh et al. 2005).

Efficient task specialization reflects in *modularity* that leads to increasing of coordination costs associated with coordination of work on numerous actions, or coordination of a large number of developers working on one task, or motivating filtering of quality control, or motivation and coordination of working on unwanted tasks as well finalizing of uncompleted tasks, and eventually integrating and combination of the project puzzle. On the other, modularity is a vital principal of OSS production and represents shared belief that it is essential to production of OSS by distributed community (Bonaccorsi and Rossi 2003). The rules related to the coordination of actions include some values of *macroculture* such *as common ideology, shared validity criteria, reciprocity, reputation, transparency, release early – release often, consensus*. Release managers/project leaders in most OSS projects have no authority to assign tasks to other developers or require them to complete an assignment (Jensen and Scacchi 2005a). However, they announce critical issues to attract developers to particular development tasks (Crowston et al.

2005b; Jensen and Scacchi 2005a), for example by possibility to increase their reputation in community (Dahlander and Magnusson 2005; Lerner and Tirole 2002). Governance often relies on *transparency* in open source projects, where the openness of decision processes helps to maintain community cohesion and motivation (German 2003; Lane et al. 2004a). In addition, transparency facilitates work process elements such as knowledge and code reuse, helping to speed development and increase quality (German and González-Barahona 2009; Haefliger et al. 2008; von Krogh et al. 2005).

Given the diversity of actions that must be done within the OSS project, a wide variety of tools is required. For the purpose of the study we created an inventory of tools described/discussed in the reviewed collection of papers, and then we derived four broad categories of tools: *communication, coordination, construction,* and *legal/IPR*.

All *communication* tools serve to facilitate discussion, deliberation, decision-making and knowledge sharing. However, our analysis reveals that synchronous and asynchronous communication tools have different mediating effects and thus support different types of activities. Synchronous tools, such as Web-based chat, are typically used for time-sensitive communication, to enable real-time streaming of information, and to support informal exchanges that facilitate trust and act as a substitute for face-to-face communication (German 2003; Lane et al. 2004b). In contrast, asynchronous tools, such as mailing lists, support more reflective exchanges and decision-making processes (Hemetsberger and Reinhardt 2004), and provide a publically visible record of transparent coordination processes (German 2003). *Coordination* tools provide structure for collaborative activities, serving as mechanisms to ensure common formats, approaches, and designs. Version control systems (VCS) are centralized systems that serve as a collaboration mechanism for code viewing and altering and provide controlled repository for the community project code. Executive tools such as ToDo lists, HOWTO, and FAQ ensure common design/implementation methods and facilitate accuracy and coherence of the organizing (or self-organizing) and execution of the OSS development process (Robbins 2005; Ye and Kishida 2003). *Construction* tools structure the way developers and users interact with the digital artifacts with which they work and provide mechanism for writing, editing, and saving code. *Legal / intellectual property rights (IPR)* tools provide for the open dissemination of the artifacts produced by a community/project while also enabling a degree of control/protection of the underlying intellectual property (Bonaccorsi and Rossi 2003; de Laat 2005; O'Mahony 2003) preventing others from unduly appropriating the value created by the community (Dahlander and Magnusson 2005; Franck and Jungwirth 2003; West 2003).

Integration

Determination of which solutions to use and how to integrate them are often managed through social mechanisms including consensus, democratic processes, and consensual, socially embedded, and often temporary leadership positions. Self-assignment plays a central role also in these broader, long-term project management matters since project participants are free to join or leave projects (within broader socially defined boundaries for OSS participation) and even form competing projects if entrenched leadership is problematic.

Which solutions to include in product?

Given potentially quite numerous solutions to any identified defect or feature gap, determination of what to include in any particular release can be a significant issue. This determination is most often based on the perceived merit of the code, often involving a democratic process whereby the community of developers determines collectively which solutions are the best (Jensen and Scacchi 2005b). Transparent processes facilitate arriving at consensus by ensuring that the members of the community are comfortable that all relevant contributions have been considered and that the best solution (or at least an acceptable solution) has been selected.

One area where duplication of effort is likely significant and filtering of contributions is important is in the identification of bugs, since multiple users are likely to report commonly encountered bugs. However, even this duplication may produce information gains as numerous reports may indicate bugs that impact more users. However, developers within the community must make sense of these reports/requests by identifying faulty or duplicate reports, constructing a rationalized list of issues, and conducting initial prioritization. Contributors may also offer feature gifts or create new functionality in response to identified gaps in the existing feature set (von Krogh et al. 2003).

One of the critical rules allowing reaching consensus regarding best solutions is peer-review. Peer review serves a gate-keeping function, ensuring that only the highest quality submissions are accepted for use in the community's products (AlMarzouq et al. 2005; Lane et al. 2004b). Version control systems may facilitate transparency in

decision-making in open source projects as when they are used for voting (Tsiavos and Hosein 2003), as do tools such as email (Fielding 1999; German 2003; Lundell et al. 2006).

While some well established OSS projects underlying processes of decision-making tend to be consensus based and democratic, project leaders also play a role as gatekeepers for a project and final decision-making arbiter (Asklund and Bendix 2002; German 2003; Wubishet 2009; Ye and Kishida 2003) to define the status of other members and decide regarding which appropriate solutions should be accepted (Scacchi 2002; Tsiavos and Hosein 2003).

How solutions should be integrated?

Many projects have a release manager who is granted some degree of authority over the coordination of integrating components from different development streams. Typically such positions are filled based upon the reputation or merit of the individual, broadly construed, including his tenure with the project (O'Mahony and Ferraro 2007; Roberts et al. 2006; Ye and Kishida 2003). Project leaders engage in formal release activities. For example, some projects use release manager/s to coordinate release submissions and timing across modules (German 2003; Tsiavos and Hosein 2003).

One of the responsibilities of the project leaders/release manager is ensuring that community rules regarding consensus and transparency are adhered to while also making sure that any agreed upon or promulgated timelines are observed. Ensuring that community norms are followed is a necessary element in building the long-term trust (Lattemann and Stieglitz 2005) necessary for a sustainable OSS project, while deadlines may provide a critical motivational element for the submission of code (Francalanci and Merlo 2008; Rossi et al. 2009). These responsibilities may conflict when consensus regarding integration is difficult to reach, but the common norm of "release-early release-often" means that any code submissions excluded from a particular release cycle as infeasible may be picked up in a near-term future release. The reputational effects of code appearing in a release (Hann et al. 2004; Lee and Cole 2003; Raymond 2000) may be enhanced by this practice, as well.

Release-early release-often (RERO), a well known and widely acknowledged OSS development norm, is especially important for releasing action of the activity system. This rule accelerates the development cycle, which enables timelier user feedback that helps to keep the software updated and implement latest and best features leading to the consumer satisfaction as a result increasing project popularity. RERO rule builds developers confidence in the project success (Robbins 2002; Sridhar et al. 2005), which is critical for enhancing trust in the project. RERO is an essential part in developing sustainable OS products.

The question of *how* different solutions should be integrated into the final product is highly interrelated with the question of *how* the work on this project should be done. Consequently, rules related to the coordination of actions are also relevant to integration of these actions. These rules include macroculture such as common ideology, shared validity criteria, reciprocity, reputation, transparency, release early – release often, consensus.

Conclusion

This paper provides a rich conceptualization of viable commons-based peer-production (a mode of production first described by Benkler (2002)) through an activity theoretic analysis of 11 years of peer-reviewed research on the social production of open source software. This conceptualization reveals the reliance of peer-production communities on complex systems of inter-related tools, rules, and roles as mediating artifacts enabling them to (i) exploit the two theorized advantages of common-based peer-production (resource allocation and information processing) and (ii) overcome the two theorized challenges associated with this mode of production (motivation and organization). The work significantly enhances and extends our understanding of CBPP in several ways.

First, we have clarified the operational nature of the allocation advantage; highlighting its reliance on a shared commons of resources, and identifying the legal and technological tools and social norms and values that serve to protect the commons and ensure its resources are universally accessible to all participants. Furthermore, we reveal that although the resource allocation advantage is a necessary but not sufficient pre-requisite for commons-based peer-production, it is not sufficient in itself. Rather, the allocation advantage is a potential advantage, which is actualized/realized in CBPP efforts through the active consumption, manipulation and sharing of resources (i.e. the information processing advantage).

Second, we have clarified the nature of the information processing advantage. The analysis revealed this advantage to be dependent on (i) maximizing broad participation and (ii) effectively filtering the output of this distributed

activity. Again, we have identified the variety of tools, rules and roles which mediate activity in ways that enable such participation and filtering.

Third, we have extended extant understanding of the motivation problem and how it is overcome. The analysis reveals that CBPP depends on the ability for participation to satisfy a diverse and scalable array of individual motivations; thus expanding on the micro-motivation explanation proffered by Benkler (2002). More importantly, we have demonstrated the ways in which individual motivations become culturally and socially contextualized.

Fourth, we have extended extant understanding of the organization problem, decomposing it into two related problems – coordinating effort and integrating the output of effort. Again, we have identified the various tools, rules and roles that mediate coordination and integration activities. Furthermore, we have revealed that in CBPP, just as individual motivations are socially contextualized, individual autonomy (i.e. the self-selection of work) is exercised in the context of community consensus and shared goals.

Each of these contributions has specific implications for the IS research community due to the prevalence of CBPP in information goods. The work highlights the need for further investigation of how CBPP communities – particularly in contexts other than open source software - maximize participation; accommodate diverse, scalable and socially embedded motivations; efficiently coordinate action; and effectively integrate the results of action. This represents a significant research agenda.

Overall, we argue that the central implication of the work is the need for research into how CBPP communities exploit the two advantages and overcome the challenges in a sustainable fashion that both enhances the commons and leaves the community intact. Due to the non-rival nature of information resources, we posit that the information commons at the heart of the CBPP mode of production is not like a physical commons, subject to degradation through abusive over-consumption. However, we contend that a different "tragedy of the commons" is still possible. CBPP communities must operate in such a way that they actively improve the commons through (i) the injection of new knowledge (e.g. capturing knowledge created through community discourse in a visible and persistent way) and (ii), more subtly, by sustaining and improving the socio-technical system of tools, rules and roles that enable the community to motivate and organize collective action.

Acknowledgements

This research was funded by the Irish Research Council for the Humanities and Social Sciences (IRCHSS), through the Open Code, Content and Commerce (03C) Business Models project.

References

Adler, P.S. 2001. "Market, hierarchy, and trust: the knowledge economy and the future of capitalism," *Organization Science* (12:2), pp. 215-234.

Alavi, M., and Carlson, P. 1992. "A review of MIS research and disciplinary development," *Journal of Management Information Systems* (8:4), pp. 45-62.

AlMarzouq, M., Li, Z., Guang, R., and Grover, V. 2005. "Open Source: Concepts, Benefits, and Challenges," *Communications of AIS* (2005:16), pp. 756-784.

Asklund, U., and Bendix, L. 2002. "A Study of Configuration Management in Open Source Software Projects," *IEE Proceedings - Software* (149:1), pp. 40-46.

Asundi, J., and Jayant, R. 2007. "Patch Review Processes in Open Source Software Development Communities: A Comparative Case Study," *Proceedings of the 40th HICSS – 2007*: Computer Society Press.

Au, Y.A., Carpenter, D., Chen, X., and Clark, J.G. 2009. "Virtual organizational learning in open source software development projects," *Information and Management* (46:1), pp. 9-15.

Bagozzi, R.P., and Dholakia, U.M. 2006. "Open Source Software User Communities: A Study of Participation in Linux User Groups," *Management Science* (52:7), pp. 1099-1115.

Baldwin, C.Y., and Clark, K.B. 2006. "The Architecture of Participation: Does Code Architecture Mitigate Free Riding in the Open Source Development Model?," *Management Science* (52:7), pp. 1116-1127.

Barthelmess, P., and Anderson, K.M. 2002. "A view of software development environments based on activity theory," *Computer Supported Cooperative Work (CSCW)* (11:1), pp. 13-37.

Basnet, P., and Lane, M. 2005. "Informal Control in Open Source Project: An Empirical Assessment," *ACIS 2005 Proceedings. Paper 47.*

Benkler, Y. 2002. "Coase's Penguin, or, Linux and The Nature of the Firm," *Yale Law Journal* (112:3), pp. 369-446.

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

Benkler, Y., and Nissenbaum, H. 2006. "Commons-based Peer Production and Virtue," *Journal of Political Philosophy* (14:4), pp. 394-419.

Bergquist, M., and Ljungberg, J. 2001. "The Power of Gifts: Organising Social Relationships in Open Source Communities," *Information Systems Journal* (11:4), pp. 305-320.

Bhadauria, V. 2007. "An Integrative Theoretic Approach to Open Source Software Development," *AMCIS 2007 Proceedings. Paper 421.*

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

Capiluppi, A., and Adams, P.J. 2009. "Reassessing Brooks' Law for the Free Software Community," in: *Open Source Ecosystems: Diverse Communities Interacting,* C. Boldyreff, K. Crowston, B. Lundell, and A.I. Wasserman (eds.). IFIP Advances in Information and Communication Technology: Springer Berlin Heidelberg, pp. 274-283.

Capra, E., Francalanci, C., Merlo, F., and Rossi Lamastra, C. 2008. "Firms' Involvement in Open Source Projects: A Controversial Role," *MCIS 2008 Proceedings. Paper 45.*, Hammamet, Tunisia.

Chen, W., and Hirschheim, R. 2004. "A paradigmatic and methodological examination of information systems research from 1991 to 2001," *Information Systems Journal* (14:3), pp. 197-235.

Claver, E., González, R., and Llopis, J. 2000. "An analysis of research in information systems (1981-1997)," *Information & Management* (37:4), pp. 181-195.

Coase, R. 1937. "The Nature of the Firm," *Economica* (4:16), pp. 386-405.

Crowston, K., Annabi, H., Howison, J., and Masango, C. 2005a. "Effective Work Practices for FLOSS Development: a Model and Propositions " *Proceedings of the 38th HICSS – 2005*: Computer Society Press.

Crowston, K., Kangning, W., Qing, L., and Howison, J. 2006. "Core and Periphery in Free/Libre and Open Source Software Team Communications," *Proceedings of the 39th HICSS – 2006*: Computer Society Press.

Crowston, K., Wei, K., Howison, J., and Wiggins, A. 2009. "Free/Libre Open Source Software: What We Know and What We Do Not Know," *Under Review*), http://floss.syr.edu/content/freelibre-open-source-software-development-what-we-know-and-what-we-do-not-know.

Crowston, K., Wei, K., Li, Q., Eseryel, U., and Howison, J. 2005b. "Coordination of Free/Libre Open Source Software Development," *ICIS 2005 Proceedings. Paper 16.*

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

Daniel, S. 2006. "An Absorptive Capacity Perspective of Open Source Software Development Projects," *AMCIS 2006 Proceedings. Paper 24.*

Daniel, S., Agarwal, R., and Stewart, K. 2006. "An Absorptive Capacity Perspective of Open Source Software Development Group Performance," *ICIS 2006 Proceedings. Paper 59.*

David, P.A., and Shapiro, J.S. 2008. "Community-based production of open-source software: What do we know about the developers who participate?," *Information Economics and Policy* (20:4), pp. 364-398.

de Laat, P.B. 2005. "Copyright or copyleft?: An analysis of property regimes for software development," *Research Policy* (34:10), pp. 1511-1532.

Demil, B., and Lecocq, X. 2006. "Neither Market nor Hierarchy nor Network: The Emergence of Bazaar Governance," *Organization Studies* (27:10), October 1, 2006, pp. 1447-1466.

Denzin, N.K., and Lincoln, Y.S. 2000. "The Discipline and Practice of Qualitative Research," in: *Handbook of Qualitative Research,* N.K. Denzin, and Y.S. Lincoln (eds.). Thousand Oaks, CA: Sage Publications.

Desouza, K., Awazu, Y., Wecht, C., Kim, J., and Jha, S. 2007. "Roles of Information Technology in Distributed and Open Innovation Process," *AMCIS 2007 Proceedings. Paper 285.*

Eisenhardt, K.M. 1989. "Agency theory: An assessment and review," *Academy of management review* (14:1), pp. 57-74.

Elliott, M., and Scacchi, W. 2008. "Mobilization of software developers: the free software movement," *Information, Technology and People* (21:1), pp. 4-33.

Engeström, Y. 1987. *Learning by expanding: An activity-theoretical approach to developmental research*. Helsinki, Finland: Orienta-Konsultit Oy.

Engeström, Y. 1999. "Expansive Visibilization of Work: An Activity-Theoretical Perspective," *Computer Supported Cooperative Work (CSCW)* (8), pp. 63-93.

Engeström, Y. 2000. "Activity theory as a framework for analyzing and redesigning work," *Ergonomics* (43:7), pp. 960-974.

Engeström, Y. 2008. "The Future of Activity Theory: A Rough Draft," in: *Learning and Expanding with Activity Theory,* A.L. Sannino, H. Daniels, and K.D. Gutierrez (eds.). Cambridge, UK: Cambridge University Press.

Fang, Y., and Neufeld, D. 2009. "Understanding Sustained Participation in Open Source Software Projects," *Journal of Management Information Systems* (25:4), Spring2009, pp. 9-50.

Farhoomand, A., and Drury, D.H. 1999. "A historiographical examination of information systems," *Communications of the AIS* (1:5es), p. 4.

Feigenbaum, J., Parkes, D.C., and Pennock, D.M. 2009. "Computational challenges in e-commerce," *Communications of the ACM* (52:1), pp. 70-74.

Felin, T., Zenger, T.R., and Tomsik, J. 2009. "The knowledge economy: emerging organizational forms, missing microfoundations, and key considerations for managing human capital," *Human Resource Management* (48:4), pp. 555-570.

Feller, J., Finnegan, P., and Nilsson, O. 2008. "We Have Everything to Win": Collaboration and Open Innovation in Public Administration," *ICIS 2008 Proceedings. Paper 214.*

Feller, J., and Fitzgerald, B. 2002. *Understanding Open Source Software Development*. London, UK: Addison-Wesley.

Fielding, R.T. 1999. "Shared Leadership in the Apache Project," *Communications of the ACM* (42:4), pp. 42-43.

Fogel, K. 2005. *Producing Open Source Software: How to Run a Successful Free Software Project*. Sebastopol, CA: O'Reilly Media, Inc.

Francalanci, C., and Merlo, F. 2008. "The Impact of Complexity on Software Design Quality and Costs: An Exploratory Empirical Analysis of Open Source Applications," *European Conference on Information Systems,* W. Golden, T. Acton, K. Conboy, H. van der Heijden, and V.K. Tuunainen (eds.), Galway, Ireland, pp. 1442-1453.

Franck, E., and Jungwirth, C. 2003. "Reconciling Rent-Seekers and Donators--The Governance Structure of Open Source," *Journal of Management and Governance* (7:4), pp. 401-421.

Gallivan, M.J. 2001. "Striking a Balance between Trust and Control in a Virtual Organization: A Content Analysis of Open Source Software Case Studies," *Information Systems Journal* (11:4), pp. 277-304.

Gambardella, A., and Hall, B.H. 2006. "Proprietary versus public domain licensing of software and research products," *Research Policy* (35:6), pp. 875-892.

Gay, L.R., Mills, G.E., and Airasian, P. 2006. *Educational Research: Competencies for Analysis and Applications*. Upper Saddle River, NJ: Pearson Merrill Prentice Hall.

German, D.M. 2003. "The GNOME project: a case study of open source, global software development," *Software Process: Improvement and Practice* (8:4), pp. 201-215.

German, D.M., and González-Barahona, J.M. 2009. "An empirical study of the reuse of software licensed under the GNU General Public License," in: *Open Source Ecosystems: Diverse Communities Interacting,* C. Boldyreff, K. Crowston, B. Lundell, and A.I. Wasserman (eds.). IFIP Advances in Information and Communication Technology: Springer Berlin Heidelberg, pp. 185-198.

Ghosh, R.A., Glott, R., Krieger, B., and Robles, G. 2002. "Free/libre and open source software: Survey and study," in: *Maastricht Economic Research Institute on Innovation and Technology*. Maastricht, The Netherlands: University of Maastricht, p. http://www.flossproject.nl/report/.

Glance, D.G. 2004. "Release criteria for the Linux kernel," *First Monday* (9:4)

Glass, R.L. 2003. "A Sociopolitical Look at Open Source," *Communications of the ACM* (46:11), pp. 21-23.

Gleave, E., Welser, H.T., Lento, T.M., and Smith, M.A. 2009. "A Conceptual and Operational Definition of 'Social Role' in Online Community," *Proceedings of the 42nd HICSS – 2009*: Computer Society Press.

Gwet, K. 2001. *Handbook of inter-rater reliability: How to estimate the level of agreement between two or multiple raters*. Gaithersburg, MD: STATAXIS Publishing Company.

Haefliger, S., von Krogh, G., and Spaeth, S. 2008. "Code Reuse in Open Source Software," *Management Science* (54:1), pp. 180-193.

Hann, I.-H., Roberts, J., and Slaughter, S. 2004. "Why Developers Participate in Open Source Software Projects: An Empirical Investigation," *ICIS 2004 Proceedings. Paper 66.*

Hann, I.-H., Roberts, J., Slaughter, S., and Fielding, R. 2002. "Economic Incentives for Participating Open Source Software Projects," *ICIS 2002 Proceedings. Paper 33.*

Hardin, G. 1968. "The Tragedy of the Commons," *Science* (162:3859), December 13, 1968, pp. 1243-1248.

Hars, A., and Ou, S. 2000. "Why Is Open Source Software Viable? - A Study of Intrinsic Motivation, Personal Needs, and Future Returns," *AMCIS 2000 Proceedings. Paper 379.*

Hars, A., and Ou, S. 2001. "Working for free? - Motivations of participating in Open Source Projects," *Proceedings of the 34th HICSS – 2001*: Computer Society Press.

Haythornthwaite, C. 2009. "Crowds and Communities: Light and Heavyweight Models of Peer Production," *Proceedings of the 42nd HICSS – 2009*: Computer Society Press.

Hemetsberger, A., and Reinhardt, C. 2004. "Sharing and Creating Knowledge in Open-Source Communities: The case of KDE," *European Conference on Organizational Knowledge, Learning, and Capabilities*, Innsbruck, Austria.

Hemetsberger, A., and Reinhardt, C. 2006. "Learning and Knowledge-building in Open-source Communities: A Social-experiential Approach," *Management Learning* (37:2), p. 187.

Hemetsberger, A., and Reinhardt, C. 2009. "Collective Development in Open-Source Communities: An Activity Theoretical Perspective on Successful Online Collaboration," *Organization Studies* (30:9), September 1, 2009, pp. 987-1008.

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

Iivari, N. 2009. "'"Constructing the users" in open source software development: An interpretive case study of user participation," *Information Technology and People* (22:2), pp. 132-156.

Jacobides, M.G., and Billinger, S. 2006. "Designing the boundaries of the firm: From" make, buy, or ally" to the dynamic benefits of vertical architecture," *Organization Science* (17:2), p. 249.

Jensen, C., and Scacchi, W. 2005a. "Collaboration, Leadership, Control, and Conflict Negotiation and the Netbeans.org Open Source Software Development Community," *Proceedings of the 38th HICSS – 2005*: Computer Society Press.

Jensen, C., and Scacchi, W. 2005b. "Process modeling across the web information infrastructure," *Software Process: Improvement and Practice* (10:3), pp. 255-272.

Kaiser, S., and Müller-Seitz, G. 2008. "Leveraging Lead User Knowledge in Software Development—The Case of Weblog Technology," *Industry & Innovation* (15:2), pp. 199-221.

Kitchenham, B. 2004. "Procedures for Performing Systematic Reviews," Keele University, Keele, UK, from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.122.3308&rep=rep1&type=pdf.

Koch, S., and Schneider, G. 2002. "Effort, co-operation and co-ordination in an open source software project: GNOME," *Information Systems Journal* (12:1), pp. 27-42.

Krowne, A. 2005. "The FUD-based encyclopedia," *Free Software Magazine* (n. 2), pp. 1-8.

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

Lakhani, K.R., and Wolf, R.G. 2005. "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.A. Hissam, and K.R. Lakhani (eds.). Cambridge, MA: MIT Press.

Lane, M.S., Van der Vyver, G., and Basnet, P. 2004a. "Trust in Virtual Communities involved in Free/Open Source Projects: An Empirical Study," *ACIS 2004 Proceedings. Paper 106.*

Lane, M.S., Van der Vyver, G., Basnet, P., and Howard, S. 2004b. "Interpretative Insights into Interpersonal Trust and Effectiveness of Virtual Communities of Open Source Software (OSS) Developers," *ACIS 2004 Proceedings. Paper 67.*

Lave, J., and Wenger, E. 1991. *Situated Learning: Legitimate Peripheral Participation*. Cambridge UK: Cambridge University Press.

Lattemann, C., and Stieglitz, S. 2005. "Framework for Governance in Open Source Communities," *Proceedings of the 38th HICSS – 2005*: Computer Society Press.

Lee, G.K., and Cole, R.E. 2003. "From a Firm-Based to a Community-Based Model of Knowledge Creation: The Case of the Linux Kernel Development," *Organization Science* (14:6), pp. 633-649.

Leont'ev, A.N., and Hall, M.J. 1978. *Activity, consciousness, and personality*. Englewood Cliffs, NJ: Prentice-Hall.

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

Ljungberg J 2000 Open Source Movements as a Model for Organizing. In Proceedings of the Eighth European Conference on Information Systems (Hansen HR, Bichler M, Mahrer H eds.), 501-508, Vienna.

Long, J., and Yuan, M.J. 2005. "Are All Open Source Projects Created Equal? Understanding the Sustainability of Open Source Software Development Model," *AMCIS 2005 Proceedings. Paper 435.*

Long, Y. 2005. "Social Structure for Open Source Software Projects," *AMCIS 2005 Proceedings. Paper 76.*

Lundell, B., Lings, B., Ågerfalk, P., and Fitzgerald, B. 2006. "The distributed open source software development model: observations on communication, coordination and control," *European Conference on Information Systems,* J. Ljungberg, and M. Andersson (eds.), Goteborg, Sweden, pp. 683-694.

Markus, M.L., Manville, B., and Agres, C.E. 2000. "What Makes a Virtual Organization Work?," *Sloan Management Review* (42:1), Fall2000, pp. 13-26.

Midha, V. 2008. "Does Complexity Matter? The Impact of Change in Structural Complexity on Software Maintenance and New Developers' Contributions in Open Source Software," *ICIS 2008 Proceedings. Paper 37.*

Miralles, F., Sieber, S., and Valor, J. 2005. "CIO Herds and User Gangs in the Adoption of Open Source Software," *European Conference on Information Systems 2005 Proceedings. Paper 140.*

Mockus, A., Fielding, R.T., and Herbsleb, J.D. 2002. "Two case studies of open source software development: Apache and Mozilla," *ACM Transactions on Software Engineering and Methodology* (11:3), pp. 309-346.

Moon, J.Y., and Sproull, L. 2000. "Essence of Distributed Work: The Case of the Linux Kernel," *First Monday* (5:11).

Nichols, D.M., Thomson, K., and Yeates, S.A. 2001. "Usability and open-source software development," *Symposium on Computer Human Interaction,* E. Kemp, C. Phillips, and J. Kinshuk and Haynes (eds.).

Nichols, D.M., and Twidale, M.B. 2006. "Usability processes in open source projects," *Software Process: Improvement and Practice* (11:2), pp. 149-162.

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

O'Mahony, S., and Ferraro, F. 2007. "The Emergence of Governance in an Open Source Community," *Academy of Management Journal* (50:5), pp. 1079-1106.

Okoli, C., and Oh, W. 2007. "Investigating recognition-based performance in an open content community: A social capital perspective," *Information and Management* (44:3), pp. 240-252.

Orlikowski, W.J., and Baroudi, J.J. 1991. "Studying information technology in organizations: Research approaches and assumptions," *Information systems research* (2:1), pp. 1-28.

OSI. 2009. "Open Source Case for Business." Retrieved 04 December 2009, from http://www.opensource.org/advocacy/case_for_business.php

Osterloh, M., and Frey, B.S. 2000. "Motivation, knowledge transfer, and organizational forms," *Organization Science* (11:5), pp. 538-550.

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

Porter, A., Yilmaz, C., Memon, A.M., Krishna, A.S., Schmidt, D.C., and Gokhale, A. 2006. "Techniques and processes for improving the quality and performance of open-source software," *Software Process: Improvement and Practice* (11:2), pp. 163-176.

Pouwelse, J.A., Garbacki, P., Epema, D., and Sips, H. 2008. "Pirates and Samaritans: A decade of measurements on peer production and their implications for net neutrality and copyright," *Telecommunications Policy* (32:11), pp. 701-712.

Ramesh, V., Glass, R.L., and Vessey, I. 2004. "Research in computer science: an empirical study," *Journal of systems and software* (70:1-2), pp. 165-176.

Raymond, E. 2000. "The Cathedral and the Bazaar." Retrieved 2009-11-04, from http://catb.org/esr/writings/cathedral-bazaar/cathedral-bazaar/index.html

Robbins, J.E. 2002. "Adopting OSS Methods by Adopting OSS Tools," *International Conference on Software Engineering*, 19-25 May 2002, Orlando, FL.

Robbins, J.E. 2005. "Adopting Open Source Software Engineering (OSSE) Practices by Adopting OSSE Tools," in: *Perspectives on Free and Open Source Software,* J. Feller, B. Fitzgerald, S.A. Hissam, and K.R. Lakhani (eds.). Cambridge, MA: MIT Press, pp. 245-264.

Roberts, J.A., Il-Horn, H., and Slaughter, S.A. 2006. "Understanding the Motivations, Participation, and Performance of Open Source Software Developers: A Longitudinal Study of the Apache Projects," *Management Science* (52:7), pp. 984-999.

Romano, N.C., Jr., and Fjermestad, J. 2001. "Electronic commerce customer relationship management: An assessment of research," *International Journal of Electronic Commerce* (6:2), pp. 61-113.

Rossi, B., Russo, B., and Succi, G. 2009. "Analysis of Open Source Software Development Iterations by Means of Burst Detection Techniques," in: *Open Source Ecosystems: Diverse Communities Interacting,* C. Boldyreff, K. Crowston, B. Lundell, and A.I. Wasserman (eds.). IFIP Advances in Information and Communication Technology: Springer Berlin Heidelberg, pp. 83-93.

Santos, F.M., and Eisenhardt, K.M. 2005. "Organizational boundaries and theories of organization," *Organization Science* (16:5), pp. 491-508.

Sawhney, N. 2003. "Cooperative Innovation in the Commons: Rethinking Distributed Collaboration and Intellectual Property for Sustainable Design Innovation," in: *Media Arts and Sciences*. Cambridge, MA: Massachusetts Institute of Technology.

Scacchi, W. 2002. "Understanding Requirements for Developing Open Source Software Systems," *IEE Proceedings - Software* (149:1), pp. 24-39.

Scacchi, W. 2004. "Free and Open Source Development Practices in the Game Community," *IEEE Software* (21:1), Jan/Feb, pp. 59-67.

Shaikh, M., and Cornford, T. 2009. "Innovating with Open Sourcing: Governance Concerns for Managers," *AMCIS 2009 Proceedings. Paper 308.*

Smith, K.G., Grimm, C.M., Gannon, M.J., and Chen, M.J. 1991. "Organizational Information Processing, Competitive Responses, and Performance in the U.S. Domestic Airline Industry," *Academy of Management Journal* (34:1), pp. 60-85.

Sowe, S., Stamelos, I., and Angelis, L. 2006. "Identifying knowledge brokers that yield software engineering knowledge in OSS projects," *Information and Software Technology* (48:11), pp. 1025-1033.

Sowe, S.K., and Stamelos, I. 2008. "Reflection on Knowledge Sharing in F/OSS Projects," *The 4th International Conference on Open Source Systems*, 7-10 September 2008, Milan, Italy.

Sridhar, S., Altinkemer, K., and Rees, J. 2005. "Software Vulnerabilities: Open Source versus Proprietary Software Security," *AMCIS 2005 Proceedings. Paper 428.*

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

Subramanyam, R., and Xia, M. 2008. "Free/Libre Open Source Software development in developing and developed countries: A conceptual framework with an exploratory study," *Decision Support Systems* (46:1), pp. 173-186.

Tan, Y., Mookerjee, V., and Singh, P. 2007. "Social Capital, Structural Holes and Team Composition: Collaborative Networks of the Open Source Software Community," *ICIS 2007 Proceedings. Paper 155.*

Tsiavos, P., and Hosein, I. 2003. "Beyond Good and Evil: Why open source development for peer-to-peer networks does not necessarily lead to an Open Society, is as imbalanced as Copyright Law, and definitely is not going to make you a better person," *European Conference on Information Systems 2003 Proceedings. Paper 65.*

Uhlir, P.F. 2003. "Re-intermediation in the Republic of Science: Moving from intellectual property to intellectual commons," *Information Services and Use* (23:2/3), p. 63.

Ven, K., and Verelst, J. 2008. "The Organizational Adoption of Open Source Server Software: A Quantitative Study," *European Conference on Information Systems,* W. Golden, T. Acton, K. Conboy, H. van der Heijden, and V.K. Tuunainen (eds.), Galway, Ireland, pp. 1430-1441.

Vessey, I., Ramesh, V., and Glass, R.L. 2005. "A unified classification system for research in the computing disciplines," *Information and Software Technology* (47:4), pp. 245-255.

Viégas, F.B., Wattenberg, M., and McKeon, M.M. 2007. "The Hidden Order of Wikipedia," *Online Communities and Social Computing, HCII 2007,* D. Schuler (ed.), Berlin Heidelberg: Springer-Verlag, pp. 445–454.

von Hayek, F.A. 1945. "The use of knowledge in society," *American Economic Review* (35:4), pp. 519-530.

von Hippel, E. 2001. "Innovation by User Communities: Learning from Open-Source Software. (cover story)," *MIT Sloan Management Review* (42:4), Summer2001, pp. 82-86.

von Hippel, E., and von Krogh, G. 2003. "Open Source Software and the 'Private-Collective' Innovation Model: Issues for Organization Science," *Organization Science* (14:2), pp. 209-223.

von Krogh, G., Spaeth, S., and Haefliger, S. 2005. "Knowledge Reuse in Open Source Software: An Exploratory Study of 15 Open Source Projects," *Proceedings of the 38th HICSS – 2005*: Computer Society Press.

von Krogh, G., Spaeth, S., and Lakhani, K.R. 2003. "Community, joining, and specialization in open source software innovation: a case study," *Research Policy* (32:7), p. 1217.

Vygotsky, L.S. 1978. *Mind in society: The development of higher psychological processes*. Cambridge, MA: Harvard University Press.

Waguespack, D.M., and Fleming, L. 2009. "Scanning the Commons? Evidence on the Benefits to Startups Participating in Open Standards Development," *Management Science* (55:2), pp. 210-223.

Wang, J. 2005. "The Role of Social Capital in Open Source Software Communities," *AMCIS 2005 Proceedings. Paper 427.*

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

Wijnen-Meijer, M., and Batenburg, R. 2007. "To Open Source or not to Open Source: That's the Strategic Question. Results from a Survey Among Eight Leading Software Providers," *European Conference on Information Systems,* H. Österle, J. Schelp, and R. Winter (eds.), University of St. Gallen, St. Gallen, Switzerland, pp. 1019-1030.

Williamson, O.E. 1973. "Markets and hierarchies: some elementary considerations," *The American Economic Review* (63:2), pp. 316-325.

Williamson, O.E. 1981. "The Economics of Organization: The Transaction Cost Approach," *The American Journal of Sociology* (87:3), p. 548.

Williamson, O.E. 1991. "Comparative Economic Organization: The Analysis of Discrete Structural Alternatives," *Administrative Science Quarterly* (36:2).

Wubishet, Z.S. 2009. "Understanding the Nature and Production Model of Hybrid Free and Open Source Systems: The Case of Varnish," *Proceedings of the 42nd HICSS – 2009*: Computer Society Press.

Xu, B., Jones, D.R., and Shao, B. 2009. "Volunteers' involvement in online community based software development," *Information and Management* (46:3), pp. 151-158.

Xu, B., Xu, Y., and Lin, Z. 2005. "Control in Open Source Software Development," *AMCIS 2005 Proceedings. Paper 433.*

Ye, Y., and Kishida, K. 2003. "Toward an Understanding of the Motivation of Open Source Software Developers," *International Conference on Software Engineering*, 3-10 May 2003, Portland, OR.

Zeitlyn, D. 2003. "Gift economies in the development of open source software: anthropological reflections," *Research Policy* (32:7), pp. 1287-1291.

Zhang, C. 2006. "Impact of Collaborative Ties on Open Source Software Development," *AMCIS 2006 Proceedings. Paper 23.*

Zhang, D., Lowry, P.B., Zhou, L., and Fu, X. 2007. "The Impact of Individualism--Collectivism, Social Presence, and Group Diversity on Group Decision Making Under Majority Influence," *Journal of Management Information Systems* (23:4), Spring2007, pp. 53-80.

Zhao, L., and Deek, F.P. 2005. "Improving Open Source Software Usability," *AMCIS 2005 Proceedings. Paper 430.*