THE INNOVATION PARADOX UNDER CONTROL: THE ROLE OF CITIZENSHIP BEHAVIORS IN REDUCING SOFTWARE DEVELOPMENT BACKLOGS

Completed Research Paper

Inchan Kim
Price College of Business
University of Oklahoma
Norman, OK 73072
ick881@ou.edu

Shaila Miranda
Price College of Business
University of Oklahoma
Norman, OK 73072
shailamiranda@ou.edu

Jama Summers
Price College of Business
University of Oklahoma
Norman, OK 73072
jama.summers@ou.edu

Abstract

In this inter-disciplinary theoretical paper, we analogize software development to the innovation process, specifically to the generation of novel and useful ideas (i.e., conceptualization) and the subsequent implementation of those ideas. We then note that factors that promote one of these two activities typically impede the other. To resolve this paradox in software development and thus to reduce software development backlogs, we propose that researchers consider teams' organizational citizenship behaviors as antecedents of successful conceptualization and implementation. Specifically, we argue that teams whose members exhibit social, functional, and advocacy participation—in conjunction with loyalty and obedience—are likely to both conceptualize quality software products and implement those products on time.

Keywords: Software development, backlogs, innovation, organizational citizenship behavior, innovation paradox

Introduction

Software (or system) development backlogs have concerned IT researchers from the very inception of our academic discipline (e.g., Brancheau et al. 1996; Brancheau and Wetherbe 1987; Dickson et al. 1984; Niederman et al. 1991). Software development backlogs refer to the amount of approved development projects waiting to be kicked off (Cheney et al. 1986). The size of such visible backlogs (i.e., approved waiting projects) as well as invisible backlogs (i.e., projects desired, but not approved) is indicative of IT team and department success (Hamilton and Chervany 1981).

Researchers have proposed several solutions to backlog problems. These include increasing internal IT spending (Galliers and Sutherland 2008), creating new organizational resources such as information centers (Fuller and Swanson 1992), outsourcing application development (Baldwin et al. 2001), complementing or replacing conventional methodologies with joint application design (e.g., Avison and Fitzgerald 2003), and relying on end-user development (e.g., Amoroso and Cheney 1991; Fitzgerald 2006; Rivard and Huff 1984).

Unfortunately, these solutions may only address the issue at the broad-spectrum level, similar to addressing symptoms rather than the causes of the problem. Essentially, in order to mitigate backlog problems, a development team should expedite software development processes while maintaining development quality. Regardless of software development methodologies (e.g., traditional and agile), software development typically entails two main sets of activities: conceptualization (e.g., requirement gathering, analysis, and design) and implementation (e.g., construction, debugging, and testing). Then, a development team should ensure that it conducts, in a speedy and quality manner, not only creative conceptualization but also effective implementation.

The challenge is, however, that innovation such as software development (Austin and Devin 2009; Zmud 1983) is naturally susceptible to the innovation paradox (Miron-Specktor et al 2011; Obstfeld 2005). The innovation paradox suggests that creative actors (e.g., individuals and teams) are often poor implementers and vice versa. This is because the attributes required for creative thinking often suppress those needed for effective implementation and the attributes required for effective implementation tend to suppress those required for creative thinking. As a consequence, if a development team stresses conceptualization in order to reduce backlogs, its implementation efforts may suffer. In contrast, if the development team emphasizes implementation, its conceptualization often becomes mundane. The big challenge for software development teams is, then, to find a way to promote both conceptualization and implementation.

Thus, in this paper we aim to shed light on this phenomenon. More specifically, we aim to propose a way to mitigate the innovation paradox in software development. We propose that organizational citizenship behaviors (OCBs) can play an important role (e.g., Bolino et al. 2012). OCBs are conceptualized as a "class of pro-organizational behaviors that can neither be enforced on the basis of formal role obligations nor elicited by contractual guarantees of recompense" (Organ 1990: 46). We develop a theoretical model that describes how OCBs influence conceptualization and implementation in software development. In doing so, we make three main contributions to the software development literature, the innovation paradox literature, and the OCB literature, respectively.

First, we enrich the software development literature. Taking into account the innovation paradox lens, we proffer a novel and more specific solution to the decades-old backlog problem. Second, we enrich the innovation paradox literature. We extend the applicability of the innovation paradox notion to the IS literature and propose, through an OCB lens, a way to resolve the team-level innovation paradox. Third, we enrich the OCB literature. We propose a potential theoretical link between OCBs and organizational effectiveness: the mitigation of the innovation paradox. In doing so, we answer the call for additional theoretical explications of the relationship between OCBs and the indicators of organizational effectiveness (Bolino et al. 2002; Mackenzie et al. 2011; Podsakoff et al. 2009). Our theoretical model suggests that OBCs can mitigate the innovation paradox in software development (i.e., the timely launch of innovative software products), which enhances firm performance (Bharadwaj 2000).

Below, we first describe backlog problems in software development. We then discuss our theoretical lens, the innovation paradox. Next, we propose volunteerism—in particular, OCBs—can be a solution to the innovation paradox in software development. We then explicate the role OCBs play in mitigating the innovation paradox. Lastly, we discuss our contributions as well as suggest directions for future research.

Software Development Backlogs

In traditional plan-based software development, such as the waterfall approach (e.g. Harris et al. 2009; Hong et al. 2011), development activities are separated into sequential stages. With increasing movement toward flexible software development (Harris et al. 2009; Lee et al. 2010), e.g., agile development, the stages begin to blur with frequent iteration (Austin and Devin 2009). Regardless of the process structure, however, the types of tasks can generally be categorized as either conceptualization tasks (e.g., requirements gathering, analysis, and design) or implementation tasks (e.g., construction, debugging, and testing).

If inefficiencies occur during any of those tasks and accumulate over time, then waiting projects stack up—i.e., backlogs increase. Software development backlogs refer to the amount of approved development projects waiting to be kicked off (Cheney et al. 1986). The size of organizations backlogs is viewed as a good indicator of IT team and department success (Hamilton and Chervany 1981). Approved waiting projects, however, represent only visible backlogs, and such backlogs exacerbate invisible backlogs—i.e., failure of desired projects to be even considered for approval (Cheney et al. 1986). The very existence of visible backlogs tends to deter organizational members from submitting requests (Cheney et al. 1986). It is therefore important that software development teams address visible backlog problems.

One set of proposed solutions is to muster more resources inside the organization. For example, a software development organization can increase IT spending internally (Galliers and Sutherland 2008) and create new organizational resources such as an information center (Fuller and Swanson 1992). Organizations can also employ outsourcing strategies. They can outsource software development to third party development teams (Baldwin et al. 2001), rely on end-user development (e.g., Amoroso and Cheney 1991; Rivard and Huff 1984), and take advantage of open source development strategies (e.g., Fitzgerald 2006). Finally, organizations can enhance their application development approaches. They can complement or replace conventional methodologies with joint application design, CASE tools, or agile (e.g., Avison and Fitzgerald 2003).

Despite these various solutions, however, backlogs seem to persist (Galliers and Sutherland 2008). We propound that the innovation paradox lens can shed novel light on the solutions to such stubborn backlogs problems.

The Innovation Paradox

Innovations are not necessarily those objects or ideas that are novel to all industrial fields or to the entire world (Damnpour and Evans 1984). Innovations are products, processes, or services that are new to a focal set of actors (Moran and Ghoshal 1999). In this sense, software development itself is inherently innovation (Austin and Devin; Zmud 1983). Software development teams conduct a number of analysis and design activities in order to meet customer demands, which are often quite unique. Development teams therefore frequently have to come up with fresh ideas and completely new software designs. In fact, most software development teams are known to leverage creativity-spurring techniques such as brainstorming, nominal groups, and visioning (Kettinger et al. 1997).

As the foregoing paragraph argues, software development is inherently innovation. Innovation entails not only conceptualization of creative ideas but also implementation of those ideas (Amabile et al. 1988; Burt 2004). Naturally, software development is then typically comprised of these two main activities. The innovation paradox lens (Miron-Spektor et al. 2011; Obstfeld 2005; West 2002b) would view backlog problems as stemming from the tension between conceptualization and implementation.

The innovation paradox describes the situation in which teams or individuals with creative thinking skills are not necessarily effective implementers and vice versa (Miron-Spektor et al. 2011; Obstfeld 2005). Accordingly, software development teams adept at conceptualization are likely to short change implementation. For example, creative teams' and individuals' conceptualization often ignore organizational constraints (e.g., avaible resources, existing practices, and established routines), hindering implementation (Levitt 2002; Miron-Specktor et al. 2011). In contrast, teams with effective implementation skills tend to lack the ability to conceptualize creative useful ideas. In sum, it is rare that a development team is good at both creative conceptualization and effective implementation. This is where, the innovation paradox literature would argue, backlog problems brew.

Unfortunately, mitigating an existing innovation paradox is not easy. The innovation paradox occurs primarily because actors' (e.g., teams and individuals) attributes and environments that are favorable for conceptualization tend to suppress those that are favorable for implementation, and vice versa (Miron-Specktor et al. 2011; Levitt

2002). Attributes favorable for creative conceptualization are spanning around out-of-the-box thinking such as exploration, risk taking, and tolerance of mistakes (Miron-Spektor et al. 2011). In contrast, the attributes favorable for implementation stem from inside-the-box thinking such as conforming to organizational constraints and respecting established rules and routines. These contradictory attributes make it challenging to balance creative conceptualization and effective implementation (Miron-Specktor et al 2011).

Mitigating a potential innovation paradox is not easy either. A development team with insufficient conceptualization and implementation skills may desire to develop both sets of skills. Then, the inherent contradiction in the attributes required for conceptualization and implementation tend to stymie such efforts (Miron-Spektor et al. 2011).

The bottom line is that drawing on an innovation paradox lens, we argue backlogs can be reduced by mitigating the existing and the potential innovation paradox. However, careful consideration is required when a team attempts to balance and promote creative conceptualization and effective implementation. Rather than focus on attributes that enhance conceptualization and implementation abilities, a software development team may consider individual attributes unrelated to creativity that can unblock conceptualization and implementation.

In the next section, we argue that volunteerism—in particular OCBs, known to increase organizational effectiveness (Organ 1988; Possakoff et al. 2009)—can be an appealing candidate for mitigating the innovation paradox in software development.

Solutions to the Innovation Paradox in Software Development

Volunteerism in Software Development

The open source movement epitomizes volunteerism in software development. Stewart and Gosain (2001: 292) observed that "OSS teams are often composed of volunteers working without financial remuneration directly tied to their contributions, and their output (e.g., source code) is generally made available to any interested users with little or no charge." There is now a widespread recognition of the "gift culture" underlying the open source movement. "Open source software innovation hinges on contributors giving gifts in the form of code" (Von Krogh et al. 2003: 1233).

Von Krogh and colleagues found that newcomers entered an open source community by contributing whole software modules and features, rather than modifying or enhancing those under development by the community. These "gifts" were an important source of community innovation, facilitating both new conceptualization and implementation within the community (Von Krough et al. 2003). Volunteers also benefited from the community's knowledge (Shah 2006), further stimulating creative conceptualization.

The benefits of these ideas of "gift giving" to software development have been demonstrated in organizations adopting open source methods. Martin and Hoffman (2007) observed that helping behaviors often occurred when an organization adopted open source development techniques. Users subscribed to a message board and coordinated to solve issues by posting problems and potential solutions. Developers subscribed to the same message board to counter bad advice and offer additional solutions.

In sum, prior research on volunteerism—in particular, open source development—suggests that volunteerism may be a way to mitigate the innovation paradox. In traditional organization contexts, OCBs have been representing such voluntary behaviors (Bolino 1999; Bolino et al. 2012). In the next section, we review the relevant OCB literature.

OCBs as an Innovation Paradox Solution in Software Development

While conceptualizations of OCBs vary (see LePine et al., 2002; Podsakoff et al., 2009; Podsakoff et al. 2000 for review), most are based on Organ's (1988, 1990) conceptualization (LePine et al. 2002; Podsakoff et al. 2009; Coleman and Borman 2000). Organ's conceptualization was subsequently synthesized into a theory of organizational citizenship (Graham 1991; Van Dyne et al. 1994). We take this theory as our starting point, focusing on the five dimensions of organizational citizenship identified there—i.e., loyalty, obedience, social participation, functional participation, and advocacy participation.

Loyalty refers to "identification with and allegiance to an organization's leaders and the organization as a whole, transcending the parochial interests of individuals, work groups, and departments" (Van Dyne et al., 1994: 767). Loyalty relates to sportsmanship, civic virtue (LePine et al., 2002), and job dedication (e.g., Van Scotter and Motowidlo, 1996). Obedience refers to the acceptance of "the necessity and desirability of rational rules and regulations governing organizational structure, job descriptions, and personnel policies" (Van Dyne et al., 1994: 767). This conceptualization is similar to the notion of generalized compliance (e.g., Bolino, 1999) or conscientiousness (Morrison, 1994).

Social participation represents "employee participation that is interpersonal or involves social contact, such as attending non-mandatory meetings and being involved in social activities within the organization" (Bolino et al., 2002: 508). Social participation is similar to the concept of civic virtue (e.g., Podsakoff, Ahearne, and MacKenzie, 1997). Functional participation is conceptualized as "behavior that goes above and beyond the call of duty in the execution of one's job" (Bolino et al., 2002: 514). This concept is similar to the concept of individual initiative (e.g., Bolino and Turnley, 2005). Advocacy participation refers to behavior geared toward "speaking up with constructive suggestions and encouraging colleagues to do so as well" (Bolino et al., 2002: 515). This is rather conceptually similar to challenging citizenship behavior (e.g., Grant and Mayer, 2009; Mackenzie et al., 2011) and voice extra-role behavior (e.g., Van Dyne and LePine, 1998). In a broader sense, advocacy participation also speaks to taking-charge behavior (e.g., Morrison and Phelps, 1999).

Theoretical Model

In this section, we elaborate on how organizational citizenship may mitigate the innovation paradox in software development settings. Figure 1 summarizes the arguments we develop in this paper. We do not posit direct effects of participation on conceptualization and implementation. That participation enhances task performance is tautological, i.e., true by definition, and effects of the specific participation behaviors on innovation previously noted (e.g., Ancona and Caldwell 1992). Because we are interested in explaining conceptualization and implementation, we also do not posit direct effects of loyalty and obedience on participation, though those undoubtedly exist.

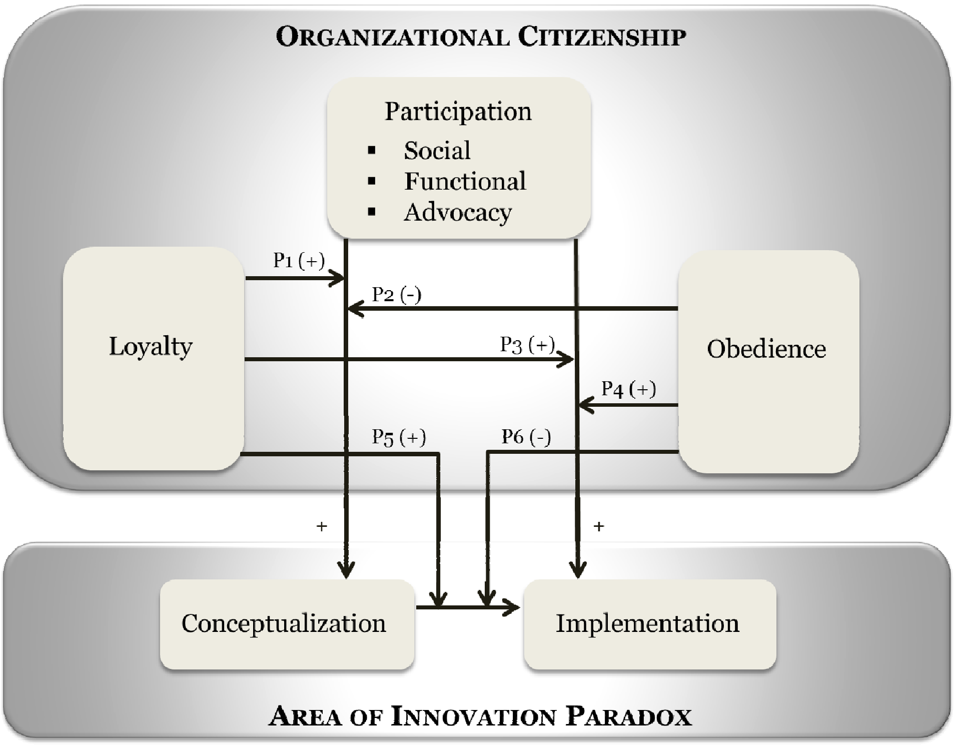

Figure 1: OCBs as a Solution to the Innovation Paradox in Software Development

We propose that participatory behaviors, moderated by loyalty and obedience, help resolve the innovation paradox. While participatory behaviors refer to developers' extra-role enactments that have the potential to enhance conceptualization quality and expedite implementation, we view loyalty and obedience as governance frames within which those enactments occur and through which others' enactments are viewed and responded to.

This perspective of loyalty and obedience as parallel governance frames has been developed in the dual motivation model proposed by Bendapudi and Berry (1997). This model suggests that customers' or employees' relationships with a firm are mainly governed by dedication (i.e., I want to stay in the relationship) or constraint (i.e., I have to stay in the relationship) (Bendapudi and Berry 1997; Dunham et al. 1994). This perspective has been appropriated to IS phenomena as well, such as continuous use of IT (Kim and Son 2009) and antecedents of developers' intention to develop smartphone applications (Kim et al. 2010).

This dual motivation model suggests that at least one of two mechanisms comes into play while an employee is engaging in organizational activities. When an employee likes working at the company, his/her behavior is likely to be governed by loyalty. Underlying the loyalty- or dedication-based behaviors is organizational control through employees' cognitive processes of identification. On the other hand, if the employee feels (s)he has to work at the company, his/her behavior is likely to be motivated by mere obedience. Underlying obedience- or constraint-based behaviors is organizational control through employees' compliance with formal and informal rules (Van Dyne et al., 1994).

These cognitive and regulative structures circumscribe developers' participatory enactments and developers' receptivity to (interpretations of and responses to) team members' enactments. Specifically, loyalty shapes behaviors and responses to behaviors through the cognitive pillar of organization-as-institution and obedience through the normative and regulative pillars of organization-as-institution (Scott 1994; Zucker 1987).

Software Conceptualization

We now examine the structuring role of loyalty and obedience on effects of participatory behaviors on conceptualization. Creativity in conceptualization becomes increasingly problematic as teams collaborate repeatedly on projects (Skilton and Dooley 2010). In general, OCBs tend to boost creativity (Alge et al. 2006). We propose that loyalty unlocks the value potential of participatory behaviors with regard to conceptualization. In contrast, we argue that obedience has a detrimental effect on the relationship between participatory behaviors and conceptualization.

Moderating Effects of Loyalty

In the OCB literature, loyalty has been associated with behaviors such as boosterism (Farh et al. 2004; Moorman and Blakely 1995), spreading good will (George and Brief 1992), and endorsing, supporting, and defending organizational objectives (Borman and Motowidlo 1997). As noted earlier, however, a key facet of loyalty is identification (Van Dyne et al. 1994). While identification has been associated with pro-social behaviors (e.g., Klandermans 2002; Tyler and Degoey 1995), researchers have noted that "it is difficult to conceive that strong team identification can be the vital origin from which newfangled and groundbreaking ideas arise" (Janssen and Huang 2008: 72). This expectation resonates with sociologists' critique of identification as ideological control that homogenizes employee perspectives and precludes creativity (e.g., Alvesson and Willmott 2002; Kärreman and Alvesson 2004). Nonetheless, researchers have found identification to induce citizenship behaviors, which in turn heightened creativity (Janssen and Huang 2008). While we are not concerned with the effects of loyalty (or obedience) on participatory behaviors, Janssen and Huang's findings speak to the plausibility of creativity in the presence of loyalty. Research also found that supervisors who perceived employees to be loyal tended to encourage employee creativity (Wang and Casimir 2007). Consequently, sociologists' concerns notwithstanding, loyalty has the potential to channel participatory behaviors toward creative output. We now consider specific effects of each type of citizenship behavior on creativity in the presence and absence of loyalty.

Team members are exposed to diverse knowledge, opinions, and behaviors by attending optional meetings and social events (Bolino et al., 2002). Through such social participation, individuals become aware not only of current company situations, but also of future directions (Organ 1988), contributing to their capabilities to generate innovative ideas. While engagement with dedicated boundary spanners was found to be more useful in diffusing knowledge about software development methodologies than engagement with casual boundary spanners (Zmud

1983), social participation has generally been found to provide developers with knowledge of emergent technologies and current development practices (e.g., Nilakanta and Scamell 1990; Rai and Patnayakuni 1996). Social participation increases teams' knowledge of both organizational issues that represent problems requiring technology solutions or opportunities for efficiency or market growth via technology, as well as about specific technologies and development practices that can be leveraged toward those problems and opportunities. Structural holes theory (Burt 2000) also suggests that individuals at the confluence of diverse social domains have the capability to generate quality ideas by combining diverse views and knowledge. Thus, teams consisting of individuals who engage in high levels of social participation have the capacity to conceptualize novel and useful software products by combining the diverse knowledge and perspectives available to them.

However, individually-held knowledge need not be appropriated to the benefit of the collective. A key concept of structural holes theory is that of the *tertius gaudens*, i.e., the one "who benefits from brokering the connection between others" by *strategically* moving information that is "accurate, *ambiguous, or distorted*" between contacts (Burt 1997: 342, italics added). Appropriating individually-held knowledge for the benefit of the collective rather than one's self therefore requires individuals identify with and experience loyalty to the collective. Without such loyalty, individuals are liable to leverage their brokerage across a structural hole toward private, rather than collective, advantage. Loyalty engenders concern for organizational outcomes, even when those might conflict with an individual's idiosyncratic interests (Bolino et al. 2002). Developers who are loyal to their organization and to their teams are therefore less likely to hoard or distort individually-held knowledge, in an attempt to benefit themselves. As noted earlier, loyalty entails organizational identification (Van Dyne et al. 1994). Research has demonstrated that expertise diversity on teams with low collective identification was negatively related to team learning and performance. In the presence of strong collective identification, on the other hand, expertise diversity was positively related to team learning and performance (Van Der Vegt and Bunderson 2005).

In sum, by engaging in social participation, individuals gain diverse views and knowledge, which are good sources of novel ideas. When individuals are also loyal, they are more likely to appropriate those ideas toward the benefit of their teams and organizations. We therefore offer the following proposition:

Proposition 1a: The relationship between members' social participation and the quality of ideas generated by software development teams will be more positive in the presence of members' loyalty to their organization.

While social participation has the potential to enhance individuals' social capital, functional participation has the potential to enhance individuals' human capital. Functional participation entails taking on additional work activities or volunteering for special projects in addition to one's normal job duties (Bolino et al., 2002). The diverse experiences gained through such activities have been shown to be good inputs to idea generation (Smith et al. 2005). For example, while working on extra-role tasks, developers might encounter books and research journals that they would not otherwise read. Such references can be useful knowledge sources for devising new methodologies for software development (Nilakanta and Scamell 1990). Like social participation, however, human capital gained through functional participation might not be leveraged to benefit one's work group. In line with this, Bolino (1999) suggests that individuals sometimes participate in such extra activities simply to look good to others.

Again, loyalty is key to unlocking the benefits of functional participation to the collective. Loyalty to the collective (e.g., team and organization) enhances the value of functional participation to conceptualization in three ways. First, loyal individuals are more likely to strive to learn from extra activities undertaken. Second, such individuals are more likely to engage in functional participation activities that contribute to their organization-specific human capital, rather than to their industry-specific or generic human capital alone. Third, they are more likely to actively leverage knowledge gleaned toward the good of their collective. Such behaviors are predicated on loyalty because employees who identify themselves with and show allegiance to their organization and leaders (i.e., being loyal) might believe that developing themselves can also possibly help improve organizational effectiveness at least in the long run. Consequently, through such loyalty behaviors, team members who engage in functional participation are more likely to assimilate new knowledge as well as retaining it, which can add to quality ideas generated by their team. Hence, we propose the following relationship:

Proposition 1b: The relationship between members' functional participation and the quality of ideas generated by software development teams will be more positive in the presence of members' loyalty to their organization.

When employees participate in advocacy, they voice their opinions on how to improve the status quo and encourage their colleagues to do so too (LePine and Van Dyne, 1998). Challenging status quo has been found to enhance teams' generation of quality ideas (De Dreu and West 2001; Farh et al. 2010). Such behaviors also socially construct a work environment in which other members feel comfortable sharing ideas and knowledge (Bolino et al., 2002). Team members in such environments can subsequently benefit from each other's knowledge and ideas, in turn enhancing innovation. In line with this, Gino et al. (2010) found that information exchange between team members helped them generate creative ideas. Amabile et al. (1996) found that teams were more creative in the presence of supportive co-workers.

Yet, researchers have found that the viability of peers critique and feedback were contingent on employee commitment to the organization. When employees were dissatisfied with their jobs, coworker feedback positively impacted creativity in the presence of high continuance commitment. Absent such commitment, coworker feedback had a negative impact on creativity (Zhou and George 2001). Accordingly, we propose the following:

Proposition 1c: The relationship between members' advocacy participation and the quality of ideas generated by software development teams will be more positive in the presence of members' loyalty to their organization.

Moderating Effects of Obedience

Earlier, we associated loyalty with cognitive control of employees by organizations and obedience with normative and regulative control. While research has suggested that creativity *can* occur in the presence of cognitive control, there is little to suggest that creativity is possible in the presence of normative and regulative control. In fact, Woodman et al. (1993: 300) noted that "interventions such as evaluations and reward systems may adversely affect intrinsic motivation toward a creative task because they redirect attention away from the heuristic aspects of the creative task and toward the technical or rule-bound aspects of task performance." We can consider effects of normative and regulative control in terms of the presence of formal and informal rules in organizations versus employees' attentiveness to those rules.

The *presence* of rules in organizations is a matter of organizational structure. Since Burns and Stalker (1961) proposed that mechanistic structures were ill-suited to innovation, researchers have examined effects of organizational structure on creativity and innovation. Early research observed that bureaucratic structures were antithetical to creativity because they discourage conflicts of opinion, free exchange of resources, and value uniformity and stability, rather than variation and change (Cummings 1965; Thompson 1965). Strict upper management control, red tape and formal procedures have been found to be detrimental to creativity in R&D teams (Amabile and Gryskiewicz 1987). Centralization has been found to reduce knowledge sharing in organizations (Tsai 2002). Centralization and formalization have been found to impede learning and creativity in alliances (Bucic and Gudergan 2004). Thus, centralization limits the value of social capital within and across organizations. As formalization increased, the relationship between supervisors' perceptions of employee loyalty and supervisors' encouragement of creativity diminished (Wang and Casimir 2007). Most recently, researchers found the negative effects of centralization and formalization on creativity to be exacerbated by strong employee goal orientations (Hirst et al. 2011). The most optimistic finding relating the presence of rules and creativity observed that the relationship between formalization and creativity was curvilinear, with creativity initially increasing with increases in formalization, but subsequently decreasing with higher levels of formalization (Andrews and Smith 1996). Even informal rules constrain employees' freedom because "norms have an 'ought' quality; they confer legitimacy and reward value upon certain modes of action, thought, and emotion, while condemning others" (Levinson 1959: 174). And freedom is an important determinant of creativity (Amabile and Conti 1999). Amabile (1983) reported that while positive evaluations enhanced creativity in the short run by increasing individuals' self-efficacy, in the long run, they inhibited creativity by creating paralyzing expectations of future evaluation. Research on control and creativity suggest *internalization* of organizational rules has a more insidious negative effect on creativity than the rules themselves.

Employees vary in their *internalization* of rules. Attention to time pressures has been found to relate negatively to creativity (Andrews and Smith 1996; Kelly and McGrath 1985). Adherence to rules is related to employee conscientiousness, which has been found to relate negatively to creative behavior in the presence of close supervisory monitoring (George and Zhou 2001). Adherence to rules, as manifest in a "systematic problem-solving style" has been found to relate negatively to innovative behavior (Scott and Bruce 1994). Thus prior evidence strongly suggests that constraint through formal or informal rules suppresses creativity. Pervasiveness of the realization that the presence and observance of organizational rules impedes creativity is visible in the skunk works approach to organizational innovation, wherein teams charged with innovation are quarantined from the rest of the organization (Dougherty 1992; Fosfuri and Rønde 2009; Van de Ven 1986).

Earlier, we noted the potential for participatory behaviors to enhance creativity in software development teams. When framed within obedience to existing organizational rules and standards, the positive effects of these behaviors will dissipate as members selectively attend to ideas that are congruent with existing organizational structures, ignoring those that could possibly break new paths. We therefore propose the following:

Proposition 2a: The relationship between members' social participation and the quality of ideas generated by software development teams will be less positive in the presence of members' obedience to their organization.

Proposition 2b: The relationship between members' functional participation and the quality of ideas generated by software development teams will be less positive in the presence of members' obedience to their organization.

Proposition 2c: The relationship between members' advocacy participation and the quality of ideas generated by software development teams will be less positive in the presence of members' obedience to their organization.

Software Implementation

Moderating Effects of Loyalty

We anticipate loyalty to play a positive role in shaping participatory behaviors and interpreting those behaviors in software implementation efforts. Because loyal employees identify with the organization and its success, they have a great interest in an organization succeeding in its implementation efforts (Hamel 2007). In software development projects, a collectivist ideology favoring collaborative values has been found to foster trust, which in turn enhances effort and task completion in open source teams (Stewart and Gosain 2006). A collective identity increased participation in open source projects (Bagozzi and Dholakia 2006). Likewise, organizations adopting similar techniques to open source development show that trust and collaboration between users and developers increased software development effectiveness (Chakraborty et al. 2010).

Nonetheless, effects of loyalty and identification have not been uniformly positive in developer teams. More specifically, Hertel and colleagues found that while identification with an area of technical specialization increased voluntary contributions in open source communities, identification with the community at large *decreased* voluntary contributions (Hertel et al. 2003). These findings can be attributed to the fact that a narrowly-defined identity tends to coincide more closely with participants' needs and capabilities, motivating behavior via self-interest and self-efficacy than does a more broadly-specified identity (e.g., Simon et al. 1998).

We argue, however, that possible downsides of loyalty/identity can be mitigated by participatory behaviors. Software development teams encounter two main challenges when implementing: mobilizing sustained developer effort and coordinating those efforts. Social participation is useful in addressing both of these challenges. Developers' ties have been found to be important in co-opting voluntary implementation efforts on open source projects (Hahn et al. 2008). Social ties have been found to facilitate coordination within and across organizational units (Tsai 2002). Consequently, ties within the development team and between developers and users can facilitate early identification of errors.

Nonetheless, individuals need not harness the benefits of their social participation toward the good of the collective. When social participation is coupled with loyalty, however, knowledge gleaned is more likely to be leveraged toward the software development efforts at hand. Further, by broadening employees' sphere of interaction, social participation can mitigate the negative effects of loyalty to/identification with the organization rather than to a narrow specialty by increasing developers' awareness of the consequences of their efforts to the organization and to themselves and enhancing their sense of self-efficacy to contribute to organizational objectives. Consequently, we propose the following:

Proposition 3a: The relationship between members' social participation and the implementation by software development teams will be more positive in the presence of members' loyalty to their organization.

Functional participation directly addresses the software development challenge of having sustained developer effort. Coupled with loyalty, such participation can be leveraged toward project implementation. We therefore propose:

Proposition 3b: The relationship between members' functional participation and the implementation by software development teams will be more positive in the presence of members' loyalty to their organization.

Advocacy participation by one team member can enhance other members' self-efficacy. In both the OCB literature and software development literatures, an individual's heightened self-efficacy—their sense that they are responsible for outcomes and have the capability to achieve them—increases discretionary behaviors (Choi 2007; Hertel et al. 2003). Team members that display loyalty behaviors, especially the sportsmanship and civic virtue facets of loyalty (LePine et al., 2002), are likely to further augment each other's self-efficacy. We therefore propose:

Proposition 3c: The relationship between members' advocacy participation and the implementation by software development teams will be more positive in the presence of members' loyalty to their organization.

Moderating Effects of Obedience

While obedience may be detrimental to conceptualization, it has positive implications for implementation. For example, Sine et al. (2006) found that new ventures were more successful when their founding teams evinced high levels of formalization, specialization, and administrative intensity. Miron-Spektor et al. (2011) also emphasized the importance of obedience to idea implementation. They argued that in order to be implemented, ideas should be promoted through accepted channels and be prototyped and tested within organizational constrains. Without such attentiveness to organizational structure and rules, efforts to implement ideas may not materialize or may meet with failure.

Nonetheless, obedience without participatory behaviors is unlikely to meet with success either. Earlier, we noted the benefits of the three types of participatory behaviors to implementation. Additionally, social, functional, or advocacy participation during implementation can be useful in early detection of problems and errors, thereby reducing costs associated with errors. Just as organizational loyalty can unlock the benefits of participatory behaviors to implementation, so can too obedience. Without attentiveness to rules and order that accompanies obedience, novel ideas surfaced via social, functional, or advocacy participation during implementation can derail implementation efforts (Levitt 2011; Miron-Spektor et al. 2011). Consequently, we propose the following:

Proposition 4a: The relationship between members' social participation and the implementation by software development teams will be more positive in the presence of members' obedience to their organization.

Proposition 4b: The relationship between members' functional participation and the implementation by software development teams will be more positive in the presence of members' obedience to their organization.

Proposition 4c: The relationship between members' advocacy participation and the implementation by software development teams will be more positive in the presence of members' obedience to their organization.

Mitigating the Innovation Paradox

Innovation efforts often fail at the implementation stage despite the presence of novel ideas (Levitt 2002). The innovation paradox literature has noted that unless the idea creator actively promotes his/her ideas, quality ideas might fade away without being implemented (Levitt, 2002). One reason for this problem is that creative employees or teams are often unwilling to attempt to implement their ideas (Miron-Spektor et al., 2011). They perceive creative conceptualization or idea generation to be their unique talent, and implementation to be grubby details that do not require their advanced abilities (Levitt, 2002). It is also likely that the more creative a conceptualization, the less likely it is that it will be implemented. This occurs for three reasons. First, those responsible for implementation may not even be aware of the idea. This is because people tend to pay greater attention to the more familiar and ignore the less familiar in team communications (Hollingshead 1996). Second, people tend to resist the unfamiliar because of the cognitive and behavioral changes entailed (Strebel 1996). Third, even if team members charged with implementation are aware of the novel aspects of the conceptualization and are willing to embrace the effort entailed to implement it, they simply may not know how to implement it because implementing novel ideas may require a skill set they currently lack (Klein and Sorra 1996). Thus, quality ideas are not necessarily implemented automatically.

We argue that the loyalty of idea generators and other team members is an important key to addressing this paradox. Research has demonstrated that individuals are more apt to share information with others they perceive as being similar to themselves, i.e., with whom they identify (Miranda and Saunders 2003; Stasser et al. 1995). Consequently, loyalty is likely to heighten sharing of novel conceptualizations in teams. Loyal employees are more likely to try to implement innovative ideas because they look out for the interest of their organization (Bolino et al., 2002; Van Dyne et al., 1994). Such employees are more likely to set aside the inconvenience that challenging implementation may pose to them in favor of the benefits to their organization (Strebel 1996). Thus, Klein and Sorra (1996) noted employee commitment to play an important role in the success of implementation efforts. Thus, we propose:

Proposition 5: Novel conceptualizations are more likely to be implemented in the presence of loyalty by members of software development teams.

Unlike loyalty, however, obedience of idea generators and other team members will detract from sharing and implementation of novel conceptualizations. For example, research has found the presence of status differentials to reduce information sharing (Hollingshead 2006) and collaboration in ad hoc teams and across organizations (Levina and Vaast 2008; Pfeffer and Langton 1993). Implementing innovative ideas may require team members to follow unconventional approaches and/or bypass company rules, procedures, or norms. This is likely to be problematic to obedient individuals. We therefore propose:

Proposition 6: Novel conceptualizations are less likely to be implemented in the presence of obedience by members of software development teams.

Discussion

In this paper, we have proposed that software development problems, described as a team-level innovation paradox, can be alleviated through careful consideration of OCBs. Specifically, we have argued that teams whose members show social, functional, and advocacy participatory behaviors, in conjunction with loyalty, but not obedience, are likely to conceptualize novel software products. In addition to this, we have argued that both loyalty and obedience behaviors unlock the value potential of participatory behaviors at the implementation stage of software development, leading to smaller backlogs. By building the proposed model, we have made six contributions.

First, we have offered a novel way to conceptualize software development and identified developer behaviors that enhance the success of software development. Specifically, by viewing software development as innovation, we have teased apart conceptualization and implementation tasks. By conceptualizing team behaviors through an OCB lens, we have demonstrated how specific developer behaviors might enhance teams' success with the two sets of tasks. On top of the prior IT research that considers how job design characteristics influence IT professionals' OCBs (Ang and Slaughter 2001) and how pro-social behaviors influence employees' willingness to share technical expertise and resources (Constant et al. 1994), this paper considers OCB as an antecedent to software development success. We believe this represents a novel contribution.

Second, we have suggested a way to mitigate the innovation paradox. The innovation paradox is a relatively recently-noticed, yet important, phenomenon that requires further examination (Miron-Spektor et al., 2011). Thus far, only a few scholars (e.g., Miron-Spektor et al. 2004; Obstfeld 2005) have suggested solutions to the innovation paradox. As a consequence, our understanding of the resolution of this paradox is still far from complete. This situation is exacerbated by extant literature's exclusive focus on the generation of quality ideas by individuals and teams—i.e., conceptualization (e.g., Amabile et al. 2004; Baer et al. 2010; Gong et al. 2009; Grant and Berry 2011; Hirst et al. 2009; Madjar et al. 2002; Oldham and Cummings 1996; Perry-Smith 2006; Shin and Zhou 2003; Zhou 2003), neglecting the other fundamental innovation activity of enacting those ideas (Levitt 2002; West 2002a).

While our analysis of the role of OCBs in resolving this paradox is specific to the software development context, OCBs are likely to be relevant to resolution of the paradox in other arenas.

Third, we have provided a new theoretical explanation for the relationship between OCBs and organizational effectiveness (here, reduction of backlogs). Even though the quest for the missing theoretical link between OCBs and organizational performance has been going on for a decade (Mackenzie et al., 2011; Podsakoff et al., 2009), few studies have explicated such links between OCBs and firm effectiveness. This paper has shown that OCBs can increase organizational effectiveness by resolving the innovation paradox in software development teams.

Fourth, we have argued that not every OCB has uniformly positive consequences for all organizational activities. Specifically, based on innovation research (e.g., Burns and Stalker 1961; Cummings 1965; Hirst et al. 2011; Thompson 1965; Wang and Casimir 2007), we have noted that obedience—as attentiveness and adherence to organizational rules—can be detrimental to the conceptualization phase of innovation. In fact, despite its significance, the dark side of OBCs has not yet received due attention (Bolino & Turnley 2005). Our analysis of the role of obedience in innovation suggests that perhaps a more dispassionate consideration of effects of OCBs on organizational activities is merited.

Fifth, we have demonstrated how dimensions of organizational citizenship interact in their effects on organizational performance. Extant OCB literature focuses on the direct effects of organizational citizenship constructs (Organ et al., 2006), ignoring potential interactions amongst the constructs. Using the dual model theory, we have showed that loyalty and obedience frame participatory behaviors differently, thereby moderating their impacts on performance.

Finally, this paper also bears several practical implications. Mitigating the innovation paradox is critically important to organizations. Understanding how organizations can leverage OCBs to foster innovation bears significant implications for organizational, team, and employee performance. The specific context of our investigation is software development teams. A normative interpretation of our proposed model of OCBs may suggest the ideal team composition for conceptualization (analysis and design) and implementation of software development. Specifically, it suggests that constituting software development teams with members whose participatory behaviors are based on loyalty rather than obedience may be the most effective in facilitating conceptualization and overcoming the innovation paradox. Our model also may offer training guidelines for software development teams regarding behaviors appropriate to conceptualization and implementation.

Future Research

Although we have bridged some research gaps by synthesizing the software development, innovation, and OCB literatures, there are still significant gaps we need to consider in future research.

First, individual-level factors such as the Big Five personality types may impinge on the work of software development teams (Hurtz and Donovan, 2000). Since early work concluding that a mix of cognitive styles were desirable on development teams (White 1984), there has been little attention to individual difference variables in the composition of software development teams. The management literature further suggests possible differences in team types (Hollenbeck et al. 2012) and types of team leaders (Yukl 1989) might be salient to innovation. Future IT research should explore the role of such variables in the performance of software development teams.

Second, like helping behavior (Organ et al. 2006), loyalty and obedience may be multi-dimensional constructs. In fact, this is implied in the OCB literature, and we have also shown several loyalty behaviors different from those identified in the existing OCB literature. Future research might want to examine the multi-dimensionality of loyalty and obedience.

Third, to the best of our knowledge, we have explicated interaction effects of OCBs for the first time. We speculate that OCBs likely have interaction effects in other contexts too. Future research might want to explore this possibility.

Fourth, although this paper is intended to be a pure theory paper—which senior IS scholars strongly promote (Xiao and Benbasat 2007; Zmud 1998), the model proposed here can be used to empirically investigate the behavioral antecedents of success in software development initiatives. Extant scales, such as those adapted by Van Dyne et al. (1994), can be used to assess the OCBs of software development teams. Quality of the conceptualized solution may be assessed in terms of users' perceptions of software quality, while implementation success may be assessed in terms of the extent to which the software delivered was on time and within budget (Zmud 1980).

Fifth, it is possible that the antecedents of the OCBs in software development are different from those of OCBs developed in general contexts. For example, the open source development literature suggests that software developer might more value symbolic capital such as reputation for generosity (Zeitlyn 2003) and status (Flynn 2003; Roberts et al. 2006) when partaking OCBs. Careful job design for such capital might increase the frequency and magnitude of developers' OCBs.

Finally, although maintenance tends to cause a great deal of backlogs, we did not explicitly incorporate its impact into our theoretical model. It is possible that OCBs also improve software maintenance work. It could be a fruitful domain for future research.

Conclusions

Software development backlogs have concerned IS researchers since the very inception of our discipline. To advance our knowledge on addressing backlogs, we have examined software development through an innovation paradox lens and have viewed team member behavior through an OCB lens. By doing so, we have offered a conceptual model that suggests how two different software development activities—i.e., conceptualization and implementation—can be optimized to reduce software development backlogs. It is our sincere hope that our research endeavor has contribute to the existing body of knowledge.

Acknowledgements

We are grateful to the two reviewers for their constructive comments.

References

Alge, B.J., Ballinger, G.A., Tangirala, S., and Oakley, J.L. 2006. "Information Privacy in Organizations: Empowering Creative and Extrarole Performance," *Journal of applied psychology* (91:1), p 221.

Alvesson, M., and Willmott, H. 2002. "Identity Regulation as Organizational Control: Producing the Appropriate Individual," *Journal of Management Studies* (39:5), pp 619-644.

Amabile, T. M. 1983. *The Social Psychology of Creativity*. New York: Springer-Verlag.

Amabile, T. M. 1988. A model of creativity and innovation in organizations. *Research in Organizational Behavior*, 10(1): 123-167.

Amabile, T.M., and Conti, R. 1999. "Changes in the Work Environment for Creativity During Downsizing," *Academy of Management Journal* (42:6), pp 630-640.

Amabile, T.M., Conti, R., Coon, H., Lazenby, J., and Herron, M. 1996. "Assessing the Work Environment for Creativity," *Academy of Management Journal* (39:5), pp 1154-1184.

Amabile, T.M., and Gryskiewicz, S.S. 1987. *Creativity in the RandD Laboratory*. Greensboro, NC: Center for Creative Leadership.

Amoroso, D.L., and Cheney, P.H. 1991. "Testing a Causal Model of End-User Application Effectiveness," *Journal of Management Information Systems* (8:1), pp 63-89.

Ancona, D.G. and Caldwell, D.F. 1992. "Bridging the Boundary: External Activity and Performance in Organizational Teams," *Administrative Science Quarterly*, 37(4): 634-665.

Andrews, J., and Smith, D.C. 1996. "In Search of the Marketing Imagination: Factors Affecting the Creativity of Marketing Programs for Mature Products," *Journal of Marketing Research*), pp 174-187.

Ang, S., and Slaughter, S.A. 2001. "Work Outcomes and Job Design for Contract Versus Permanent Information Systems Professionals on Software Development Teams," *MIS Quarterly* (25:3), pp 321-350.

Austin, R.D., and Devin, L. 2009. "Weighing the Benefits and Costs of Flexibility in Making Software: Toward a Contingency Theory of the Determinants of Development Process Design," *Information Systems Research* (20:3), pp 462-477.

Avison, D.E., and Fitzgerald, G. 2003. "Where Now for Development Methodologies?," *Communications of the ACM* (46:1), pp 78-82.

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, L., Irani, Z., and Love, P. 2001. "Outsourcing Information Systems: Drawing Lessons from a Banking Case Study," *European Journal of Information Systems* (10:1), pp 15-24.

Bharadwaj, A.S. 2000. "A Resource-Based Perspective on Information Technology Capability and Firm Performance: An Empirical Investigation," *MIS Quarterly* (24:1), pp 169-196.

Bolino, M.C., Harvey, J., and Bachrach, D.G. 2012. "A Self-Regulation Approach to Understanding Citizenship Behavior in Organizations," *Organizational behavior and human decision processes*).

Brancheau, J.C., Janz, B.D., and Wetherbe, J.C. 1996. "Key Issues in Information Systems Management: 1994-95 Sim Delphi Results," *MIS Quarterly* (20:2), pp 225-242.

Brancheau, J.C., and Wetherbe, J.C. 1987. "Key Issues in Information Systems Management," *MIS Quarterly* (11:1), pp 23-45.

Bucic, T., and Gudergan, S.P. 2004. "The Impact of Organizational Settings on Creativity and Learning in Alliances," *Management* (7:3), pp 257-273.

Burns, T., and Stalker, G.M. 1961. *The Management of Innovation*. London: Tavistock.

Burt, R.S. 1997. "The Contingent Value of Social Capital," *Administrative Science Quarterly* (42), pp 339-365.

Burt, R.S. 2004. "Strcutual holes and Good Ideas," *American Journal of Sociology* (110:2), pp 349-399.

Chakraborty, S., Sarker, S., and Sarker, S. 2010. "An Exploration into the Process of Requirements Elicitation: A Grounded Approach," Journal of the Association for Information Systems (11:4), pp 212-249.

Cheney, P.H., Mann, R.I., and Amoroso, D.L. 1986. "Organizational Factors Affecting the Success of End-User Computing," *Journal of Management Information Systems* (3:1), pp 65-80.

Choi, J.N. 2007. "Change-Oriented Organizational Citizenship Behavior: Effects of Work Environment Characteristics and Intervening Psychological Processes," *Journal of organizational Behavior* (28:4), pp 467-484.

Cummings, L. 1965. "Organizational Climates for Creativity," *Academy of Management Journal* (8:3), pp 220-227.

Dickson, G.W., Leitheiser, R.L., Wetherbe, J.C., and Nechis, M. 1984. "Key Information Systems Issues for the 1980's," *MIS Quarterly* (8:3), pp 135-159.

Dougherty, D. 1992. "Interpretive Barriers to Successful Product Innovation in Large Firms," *Organization Science* (3:2), pp 179-202.

Dunham, R. B., Grube, J. A., and Castañeda, M. B. 1994. "Organizational Commitment: The Utility of an Integrative Definition," Journal of Applied Psychology (79:3), pp. 370-380.

Faraj, S., and Sproull, L. 2000. "Coordinating Expertise in Software Development Teams," *Management Science* (46:12), pp 1554-1568.

Fitzgerald, B. 2006. "The Transformation of Open Source Software," *MIS Quarterly* (30:3), pp 587-598.

Flynn, F.J. 2003. "How Much Should I Give and How Often? The Effects of Generosity and Frequency of Favor Exchange on Social Status and Productivity," *Academy of Management Journal* (46:5), pp 539-553.

Fosfuri, A., and Rønde, T. 2009. "Leveraging Resistance to Change and the Skunk Works Model of Innovation," *Journal of Economic Behavior and Organization* (72:1), pp 274-289.

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

Fuller, M.K., and Swanson, E.B. 1992. "Information Centers as Organizational Innovation: Exploring the Correlates of Implementation Success," *Journal of Management Information Systems* (9:1), pp 47-67.

Galliers, R., and Sutherland, A. 2008. "Information Systems Management and Strategy Formulation: The 'Stages of Growth'model Revisited," *Information Systems Journal* (1:2), pp 89-114.

George, J.M., and Zhou, J. 2001. "When Openness to Experience and Conscientiousness Are Related to Creative Behavior: An Interactional Approach," *Journal of Applied Psychology* (86:3), p 513.

Hahn, J., Moon, J.Y., and Zhang, C. 2008. "Emergence of New Project Teams from Open Source Software Developer Networks: Impact of Prior Collaboration Ties," *Information Systems Research* (19:3), pp 369-391.

Hamilton, S., and Chervany, N.L. 1981. "Evaluating Information System Effectiveness-Part I: Comparing Evaluation Approaches," *MIS Quarterly* (5:3), pp 55-69.

Harris, M.L., Collins, R.W., and Hevner, A.R. 2009. "Control of Flexible Software Development under Uncertainty," Information Systems Research (20:3), pp 400-419.

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), pp 1159-1177.

Hirst, G., Van Knippenberg, D., Chen, C., and Sacramento, C.A. 2011. "How Does Bureaucracy Impact Individual Creativity? A Cross-Level Investigation of Team Contextual Influences on Goal Orientation–Creativity Relationships," *Academy of Management Journal* (54:3), pp 624-641.

Hollingshead, A.B. 1996. "The Rank-Order Effect in Group Decision Making," *Organizational behavior and human decision processes* (68:3), pp 181-193.

Hollingshead, A.B. 2006. "Information Suppression and Status Persistence in Group Decision Making the Effects of Communication Media," *Human Communication Research* (23:2), pp 193-219.

Hong, W., Thong, J.Y.L., Chasalow, L.C., and Dhillon, G. 2011. "User Acceptance of Agile Information Systems: A Model and Empirical Test," Journal of Management Information Systems (28:1), pp 235-272.

Janssen, O., and Huang, X. 2008. "Us and Me: Team Identification and Individual Differentiation as Complementary Drivers of Team Members' Citizenship and Creative Behaviors," *Journal of Management* (34:1), pp 69-88.

Kärreman, D., and Alvesson, M. 2004. "Cages in Tandem: Management Control, Social Identity, and Identification in a Knowledge-Intensive Firm," *Organization* (11:1), pp 149-175.

Kelly, J.R., and McGrath, J.E. 1985. "Effects of Time Limits and Task Types on Task Performance and Interaction of Four-Person Groups," *Journal of Personality and Social Psychology* (49:2), p 395.

Kettinger, W.J., Teng, J.T.C., and Guha, S. 1997. "Business Process Change: A Study of Methodologies, Techniques, and Tools," *Mis Quarterly* (21:1), pp 55-80.

Klandermans, B. 2002. "How Group Identification Helps to Overcome the Dilemma of Collective Action," *American Behavioral Scientist* (45:5), pp 887-900.

Klein, K.J., and Sorra, J.S. 1996. "The Challenge of Innovation Implementation," *Academy of Management Review* (21:4), pp 1055-1080.

Kraut, R.E., and Streeter, L.A. 1995. "Coordination in Software Development," *Communications of the ACM* (38:3), pp 69-81.

Lee, G., and Xia, W. 2010. "Toward Agile: An Integrated Analysis of Quantitative and Qualitative Field Data on Software Development Agility," MIS Quarterly (34:1), pp 87-114.

Levinson, D.J. 1959. "Role, Personality, and Social Structure in the Organizational Setting," *The Journal of Abnormal and Social Psychology* (58:2), p 170.

Martin, K., and Hoffman, B. 2007. "An Open Source Approach to Developing Software in a Small Organization," Software, IEEE (24:1), pp 46-53.

Metiu, A. 2006. "Owning the Code: Status Closure in Distributed Groups," *Organization Science* (17:4), pp 418-435.

Miranda, S., and Saunders, C. 2003. "The Social Construction of Meaning: An Alternative Perspective on Information Sharing," *Information Systems Research* (14:1), pp 87-106.

Moran, P., and Ghoshal, S. 1999. "Markets, Firms, and the Process of Economic Development," *Academy of Management Review* (24:3), pp 390-412.

Niederman, F., Brancheau, J.C., and Wetherbe, J.C. 1991. "Information Systems Management Issues for the 1990s," *MIS Quarterly* (15:4), pp 475-500.

Nilakanta, S., and Scamell, R.W. 1990. "The Effect of Information Sources and Communication Channels on the Diffusion of Innovation in a Data Base Development Environment," *Management Science* (36:1), pp 24-40.

Obstfeld, D. 2005. "Social networks, the tertius iungens orientation, and involvement in innovation," *Administrative Science Quarterly*, 50(1): 100-130.

Rai, A., and Patnayakuni, R. 1996. "A Structural Model for Case Adoption Behavior," *Journal of Management Information Systems*), pp 205-234.

Rivard, S., and Huff, S.L. 1984. "User Developed Applications: Evaluation of Success from the Dp Department Perspective," *MIS Quarterly* (8:1), pp 39-50.

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

Robillard, P.N. 1999. "The Role of Knowledge in Software Development," *Communications of the ACM* (42:1), pp 87-92.

Scott, W.R. 1994. *Institutions and Organizations: Toward a Theoretical Synthesis*. Thousand Oaks, CA: Sage.

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

Simon, B., Loewy, M., Stürmer, S., Weber, U., Freytag, P., Habig, C., Kampmeier, C., and Spahlinger, P. 1998. "Collective Identification and Social Movement Participation," *Journal of Personality and Social Psychology* (74:3), p 646.

Sine, W.D., Mitsuhashi, H., and Kirsch, D.A. 2006. "Revisiting Burns and Stalker: Formal Structure and New Venture Performance in Emerging Economic Sectors," *Academy of Management Journal* (49:1), pp 121-132.

Skilton, P.F., and Dooley, K.J. 2010. "The Effects of Repeat Collaboration on Creative Abrasion," *Academy of Management Review* (35:1), pp 118-134.

Stasser, G., Stewart, D.D., and Wittenbaum, G.M. 1995. "Expert Roles and Information Exchange During Discussion: The Importance of Knowing Who Knows What," *Journal of Experimental Social Psychology* (31:3), pp 244-265.

Stewart, K.J., and Gosain, S. 2001. "An Exploratory Study of Ideology and Trust in Open Source Development Groups," *MIS Quarterly* (30:2), pp 291-314.

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.

Strebel, P. 1996. "Why Do Employees Resist Change," *Harvard Business Review* (74), pp 86-94.

Thompson, V.A. 1965. "Bureaucracy and Innovation," *Administrative Science Quarterly*), pp 1-20.

Tsai, W. 2002. "Social Structure of “Coopetition” within a Multiunit Organization: Coordination, Competition, and Intraorganizational Knowledge Sharing," *Organization Science* (13:2), pp 179-190.

Tyler, T.R., and Degoey, P. 1995. "Collective Restraint in Social Dilemmas: Procedural Justice and Social Identification Effects on Support for Authorities," *Journal of Personality and Social Psychology* (69:3), p 482.

Van de Ven, A.H. 1986. "Central Problems in the Management of Innovation," *Management Science* (32:5), pp 590-607.

Van Der Vegt, G.S., and Bunderson, J.S. 2005. "Learning and Performance in Multidisciplinary Teams: The Importance of Collective Team Identification," *Academy of Management Journal* (48:3), pp 532-547.

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), pp 1217-1241.

Von Krogh, G., and Von Hippel, E. 2003. "Special Issue on Open Source Software Development," *Research Policy* (32:7), pp 1149-1157.

Wang, K.Y., and Casimir, G. 2007. "How Attitudes of Leaders May Enhance Organizational Creativity: Evidence from a Chinese Study," *Creativity and Innovation Management* (16:3), pp 229-238.

White, K.B. 1984. "Mis Project Teams: An Investigation of Cognitive Style Implications," *MIS Quarterly* (8:2), pp 95-101.

Woodman, R.W., Sawyer, J.E., and Griffin, R.W. 1993. "Toward a Theory of Organizational Creativity," *Academy of Management Review* (18:2), pp 293-321.

Wu, C.G., Gerlach, J.H., and Young, C.E. 2007. "An Empirical Analysis of Open Source Software Developers' Motivations and Continuance Intentions," *Information and Management* (44:3), pp 253-262.

Xiao, B., and Benbasat, I. 2007. "E-Commerce Product ecommendation Agents: Use, Characteristics, and Impact," *MIS Quarterly* (31:1), pp. 137-209.

Zeitlyn, D. 2003. "Gift Economies in the Development of Open Source Software: Anthropological Reflections," *Research Policy* (32:7), pp 1287-1291.

Zhou, J., and George, J.M. 2001. "When Job Dissatisfaction Leads to Creativity: Encouraging the Expression of Voice," *Academy of Management Journal* (44:4), pp 682-696.

Zmud, R.W. 1980. "Management of Large Software Development Efforts," *MIS Quarterly* (4:2), pp 45-55.

Zmud, R.W. 1983. "The Effectiveness of External Information Channels in Facilitating Innovation within Software Development Groups," *MIS Quarterly* (7:2), pp 43-58.

Zmud, R.W. 1998. "Editor's Comments: "Pure" Throey Manuscripts," *MIS Quarterly* (22:2) pp. xxix-xxxii.

Zucker, L.G. 1987. "Institutional Theories of Organization," *Annual Review of Sociology* (13), pp 443-464.