Knowledge Integration in Outsourced Software Development: The Role of Sentry and Guard Processes

By: Nikhil Mehta and Anandhi Bharadwaj

Mehta, N. and Bharadwaj, A. (2015). Knowledge Integration in Outsourced Software Development: The Role of Sentry and Guard Processes, *Journal of Management Information Systems*, 32(1), 82-115.

This is an Accepted Manuscript of an article published by Taylor & Francis in *Journal of Management Information Systems* on 06 July 2015, available online: http://www.tandfonline.com/10.1080/07421222.2015.1029381.

Abstract:

We examine the role of sentry and guard activities in outsourced software development. Sentry activities are designed to regulate the inflow of external information to the project teams and guard activities are designed to manage the outflow of teams' information and resources to external sources. The use of sentry and guard activities has been examined in teams in other contexts such as new product development, but their role and relationship to performance in software development teams is not well understood. We hypothesize and test curvilinear relationships between these activities and knowledge integration in vendor development teams. We also examine how these effects vary under conditions of greater project uncertainty. We tested the hypotheses using data from 139 vendor development teams drawn from sixteen Indian software companies. Results highlight complex curvilinear associations among sentry and guard activities, and knowledge integration, which are further impacted by the level of uncertainty that the project team faces. We recommend that carefully calibrating sentry and guard processes will help vendor development teams enhance project outcomes.

Keywords: guard processes | knowledge integration | project uncertainty | sentry processes | software development | software development teams | software projects | team boundary

Article:

The past decade has witnessed a sharp rise in the outsourcing of software development. During this time, the accompanying academic literature on information systems (IS) has examined the reasons, goals, problems, and success factors associated with outsourced software development projects [26, 48]. Much less attention, however, has been focused on team-level processes, such as coordination and knowledge integration processes associated with outsourced software development [70]. Software development, whether carried out internally within the firm, or outsourced to an external vendor, is widely recognized as a knowledge-intensive activity

requiring the integration of knowledge from diverse sources [cf. 31, 39, 47, 58, 67]. In an outsourced setting, a team comprising of vendor personnel, who fulfill a range of roles (e.g., domain expert, technical lead, programmer, analyst, system tester, database administrator, etc.) is assembled for a system development project to be executed on behalf of a client firm [17]. The team relies on the collective skills and knowledge of its members because the scope of the effort required and the inherent complexity of the task usually exceed the ability of any single individual [22]. Effective *team members' knowledge integration*, which refers to actively combining the individually held expertise, know-how, and ideas of team members to jointly achieve project outcomes, is therefore seen as key to successful project completion.

Software development is a mix of formal coordination processes (e.g., coordination by formal plans, standards, and periodic reports) and informal coordination mechanisms [70, 71]. Informal processes include various intra- and interteam activities, such as interactions among team members, discussions about how relevant information will be gathered, sharing of information, and resolution of conflicts that arise in the course of working together [42, 50]. These processes are therefore seen as critical to the team's knowledge integration efforts [62].

The broader literature on teams, especially in the context of new product development,[1] discusses the importance of boundary-spanning processes in managing the informal interactions across team boundaries [2, 3, 8, 29]. Boundary-spanning activities refer to the actions that members of the team make to connect to external stakeholders or people who are outside of the team, such as sponsors, customers, and specialists [56]. Five boundary-spanning processes, namely, scout, ambassador, sentry, guard, and coordinator, are identified in new product development teams as representing external activities that members of the team perform. Among these, guard and sentry activities are designed to "protect" the team by buffering the boundary, thereby allowing the team members to work with minimal distraction [71, 82]. Sentry processes are aimed at actively monitoring and controlling the inflow of information from external entities, and deciding what type of information or resources to acquire from external sources [71]. Guard activities on the other hand, are intended to actively control the outflow of internal information and resources by preventing their unauthorized or unwarranted access by external entities [35, 64]. Thus, guard and sentry activities help teams "filter" the inputs to and from the team's members [71]. In this study, we focus on vendor teams as our unit of analysis to examine how sentry and guard activities performed by them influence their efforts to integrate knowledge within the team.

Sentry and guard processes may be especially complex in outsourced software development projects as project stakeholders may be dispersed across multiple locations. Software development work may be split into multiple phases and carried out at different global centers for both cost and time leverage [13, 19]. A typical arrangement for an outsourced project is to have both onsite and offshore team members [31]. System design, software coding, and pilot testing are typically done at the offshore development centers, whereas transition planning, user training, and deployment activities are often carried out at the client's site. This arrangement requires significant amounts of information flow and coordination across the onsite and offshore project team members. The onsite members would work with the client during the day to capture the design of the process object, and at night, the offshore members would convert the design templates into a software configuration. The next day, the onsite team would test the software

configuration with the client and undergo a second iteration of the design, while the offshore team would develop the second iteration, and so on.

The intricacies of sentry and guard processes are also specific to outsourced software projects, insofar as there are typically three important boundaries for interactions between the core vendor team and various external entities [21, 31].[2] The most crucial boundary lies between the core vendor team and the client firm [40]. Vendor teams typically negotiate this boundary via a project manager who, as a member of the management, works closely with the client to build initial trust, understand project requirements, and carefully manage client expectations about the performance of the final product. The second boundary is between the vendor team and other vendor teams and technical specialists. For example, vendor teams may receive critical inputs about new and emerging technologies, or industry best practices from the external specialists [52]. Vendor teams might also be asked by other teams to share their specialists, such as their expert coders, with other project teams. The third boundary in outsourced projects lies between the vendor team and the senior management [31] with whom the team is expected to share information about project status and progress and also receive crucial inputs about project structure and staffing. Thus, vendor teams have to manage information flows across all these boundaries, requiring careful consideration of what information to bring in and what to share with external entities. The primary purpose of this study is to examine whether and how the boundary-spanning sentry and guard activities performed by the vendor team impact the team's ability to integrate knowledge effectively.

We also examine whether the effects of sentry and guard activities on team members' knowledge integration is impacted by the level of project uncertainty. Characterized typically by the inadequacy of critical knowledge about the project, uncertainty has been found to adversely impact various measures of software development performance such as software productivity, software quality, and cost and schedule performance [7, 59, 84]. Knowledge requirements of an uncertain project tend to be more fluid and dynamic, requiring more active collaboration among the team members [25]. We show that project uncertainty plays a particularly significant role in moderating the relationships between sentry and guard activities and a team's knowledge integration.

This paper makes two important contributions. First, we show that the manner and extent to which vendor development teams employ sentry and guard processes impacts their knowledge integration capabilities. Specifically, we highlight that sentry and guard processes exhibit complex curvilinear relationships with team members' knowledge integration, and that project teams cannot adopt simplistic rules for maximizing or minimizing information flows to achieve project success. Critical attention to issues such as "what type of information to acquire," "where to acquire it from," "what type of information can be shared, and with which external entities," will significantly impact the teams' ability to integrate knowledge, which is a critical precursor to project success [53]. The second contribution of this paper is its explication of the role of project uncertainty in shaping the influence of sentry and guard processes on team members' knowledge integration. We demonstrate that requirements uncertainty plays a moderating role by interacting with sentry and guard processes, such that the influence of these critical processes becomes even more salient in projects marked by greater uncertainty.

Team Members' Knowledge Integration in Vendor Development Teams: Theory and Hypotheses

The knowledge-based theory, permeating both team and organizational levels of analyses, holds that expertise, skills, and abilities are "owned" at the individual level and their integration into collective knowledge is necessary for productive outcomes [32, 33]. Project teams bring together members with diverse skill sets, know-how, and expertise, and enable the synthesis of individual knowledge resources into a shared stock of knowledge [14, 53, 69]. Teams then use this common stock of knowledge to develop a shared understanding of the project goals and to solve project problems at hand [42, 61]. This combining and aggregating of individually held expertise, know-how, and ideas of team members, to jointly achieve project outcomes is referred to as team members' knowledge integration [32, 61].[3]

The importance of knowledge integration in software development teams has been well-established in the literature [53, 79]. Software teams have been characterized as knowledge-based open systems that have excessive requirements for "rich" noncodified knowledge with multiple interdependent components [36, 46]. Prior research on social networks proposes that teams fulfill their requirements for such complex forms of knowledge by ensuring that the members pool-in their expertise [38]. For example, developing a customized software application requires deep knowledge about the domain and user needs as well as knowledge about software development methodologies and technologies. Software teams serve as forums for integrating such specialized slivers of knowledge that lie distributed within the team (as in different team members possessing unique skills and knowledge) [76]. Prior studies have reported that software teams combine individual stocks of know-how, skills, and abilities into team-level expertise, and use this expertise to improve software outcomes [23, 79].

Although it is critical, knowledge integration is a challenging activity [83], especially in an outsourced software development setting, where complex information flows carried out between the vendor development team and various external entities can upset the team's knowledge integration efforts. For example, teams have to regularly share project updates with the top management, and receive performance-related feedback in return. Also, to maintain quality compliance, teams have to share information with and receive feedback from the quality-control department [31]. Similarly, client personnel may seek information from the team via the project manager and send their feedback. Feedback from these external entities has the clout to influence team's operations, so teams have to ensure that only appropriate information is shared with them. Sometimes, teams may also have to protect their members from negative external feedback. At times, the team may also be asked by peer teams to share their internal resources, such as expertise, which could influence their own resource equilibrium.

All these external information exchanges can impact the team's ability to integrate knowledge effectively. If left unmanaged, such exchanges may overwhelm the team's information-processing capabilities, which can diminish the team's knowledge integration abilities. Even worse, unsupervised information exchanges with external entities may lead to undue external pressure on a team's structure and political maneuvering of the team's operations, both of which could disrupt the team's knowledge integration [78]. Therefore, vendor teams have to devise

“boundary buffering” activities, such as sentry and guard, to manage information flows with external entities [82].

Teams performing sentry activities actively manage their access to knowledge inputs from external sources. In doing so, sentry activities allow the team to integrate various internal and external knowledge inputs without undue external disturbances. Sentry activities also act as both facilitators and filters of information inflow from external entities. While the facilitator aspect of sentry activities helps the team decide what information is needed from external sources, the filter aspect helps the team closely monitor incoming information [64, 82]. Accurate information inflows facilitate their integration into project-level knowledge, while protecting the team’s knowledge integration efforts from “unwanted information” [2].

Guard activities prevent unauthorized access to the team’s information and resources. When external entities request information or resources from the vendor team, the team performs guard activities to manage the outflow of internal information [2, 82]. For example, teams have to regularly provide project-status updates to the vendor management. Guard processes can help teams decide exactly how to respond to these queries by releasing just the right kind of information. This could influence the nature of management’s interference in the team’s affairs. Releasing just enough information could keep managerial interference at bay, thereby allowing the team to perform its knowledge integration activities without external disturbance. [31].

In additiona, another team may seek a critical knowledge resource, which, if released unknowingly, may incur a cost to the core vendor team. It has been observed that such resource requests, whether official or unofficial, are common in knowledge-intensive work settings [2]. In the absence of guard activities, the team may unknowingly lend the resource, thus affecting the availability and usage of that resource for the team’s own knowledge integration efforts. Teams performing guard activities have procedures in place, to first decide the legitimacy of such resource requests, and to carefully negotiate them to minimize the impact of lending critical resources on the team’s knowledge integration. Such teams achieve a careful balance between maintaining good relations with external entities, while simultaneously ensuring the availability of their key internal resources for their knowledge integration activities [64, 82].

In sum, effective knowledge integration in vendor teams seems to rely on both sentry and guard activities. To the best of our knowledge, the existing literature is relatively silent on the importance of these activities and their impact on knowledge integration. This represents a critical gap in the literature that we seek to address in this study. We show that vendor development teams actively engage in both sentry and guard activities to manage information flows and that these activities have a critical and complex influence on the team’s knowledge integration. When performed in tandem and at appropriate levels, these activities fortify the team by regulating its importation of useful external information, while protecting its critical functions from undue external disturbances and restricting external access to its information and knowledge resources.

Furthermore, we enhance our investigation by examining proposed effects under conditions of project uncertainty. Uncertainty is broadly defined as the absence of complete information about the organizational phenomenon being studied [4]. Zmud aptly summarized the role of

uncertainty in software development by explaining that "most difficulties can be traced to the uncertainty that pervades software development. Software development is an information-intensive activity, and decision points are continually reached where the decision maker possesses inadequate information" [84, p. 46].

A variety of uncertainties affecting software development projects have been examined in the IS literature. These include uncertainties regarding project requirements, technologies, and personnel [59, 84]. Because addressing all the sources of uncertainty is a complex endeavor, our study focuses on requirements uncertainty because of its central importance to software development [59]. Indeed, prior research has shown that software operation errors resulting from incorrect requirements can cost up to a hundred times more to fix than if they were corrected during system development [9].

As Barki, Rivard, and Talbot [7] have noted, the greater the uncertainty in a software project, the greater the amount of information decision makers must process during project execution. Uncertainty can be even more pervasive in the outsourcing context because vendor teams need to process information from sources scattered across geographical, cultural, and knowledge domains [17]. Thus, a vendor development team's plans to regulate the inflow and outflow of information and knowledge resources can be frustrated by high information-processing requirements of an uncertain project [66]. In other words, project uncertainty may interact significantly with sentry and guard activities to influence the effectiveness of knowledge integration in vendor development teams. This is one of the first studies in the IS literature to empirically examine the impact of uncertainty on knowledge integration.

Sentry Processes and Knowledge Integration

Teams use sentry processes to actively monitor and control the inflow of information from external entities, deciding what type of information or resource to acquire. These coordination activities are therefore designed to help teams decide what type of inputs to access, from whom to accept inputs, and how much of external inputs to absorb [2, 64].

In outsourced software development, the vendor development team has to rely on several external sources for key information inflows. These information inflows help the vendor team to facilitate, among other things, the development of a collective mind, among key project stakeholders. Yet, the unregulated inflow of external information, even from stakeholders, can pose risks such as overengineering and missed critical deadlines, because unmitigated access to information can impose additional coordination costs on the team.

Sentry processes are aimed at improving the software team's ability to successfully access useful knowledge while minimizing unproductive knowledge access. Thus, teams performing sentry activities have necessary routines to manage the inflow of external information [68].

Teams with little or no sentry activities lack appropriate procedures to manage information inflow from external sources. This affects the team's internal environment. For example, the team may allow diverse inputs from multiple external sources, which can lead to differences of opinion within the team, thereby creating dysfunctional conflicts [36]. These conflicts typically

reduce internal cohesiveness, inhibiting open discussions and hurting the integration of team's specialized knowledge resources [35]. In addition, the absence of sentry activities may render the team's productive core unprotected from unrestricted inflow of information from divergent external sources [82]. This may bring in too many perspectives on critical project issues [41]. For example, one of the project leaders remarked, "our biggest mistake has been to keep accepting new requirements. We are now facing a bad case of requirements creep, and we don't even have the resources to fulfill the new requirements. Honestly, I don't foresee an easy way out of this mess." This creates an environment of confusion in the team and hurts the efficiency with which team members share their ideas and knowledge with each other [51]. Thus, knowledge integration is likely to deteriorate in vendor development teams with little or no sentry activities to regulate the inflow of information.

Sentry processes are also required to selectively source critical external inputs, insofar as software teams rarely have all the knowledge required to execute a project [71]. This involves connecting with authorized external sources and facilitating the transfer of information inputs from these sources. For example, one of the project leaders we interviewed remarked: "information inflow, both formal as well as informal, is actively monitored at various project stages … teams practice access control wherein access to external information is controlled in terms of its appropriateness." By minimizing unproductive transfer (e.g., unwanted or inaccurate information), sentry processes prevent the team's information-processing capabilities from overloading [2, 41]. Team members are able to use these capabilities to combine their individual knowledge resources to develop shared project concepts and jointly solve project-related problems.

However, sentry activities are good only up to a point for vendor development teams [6]. Teams performing sentry processes excessively may monitor external stakeholders too closely and exercise very tight controls on the inflow of information. In doing so, teams face multiple risks—first, overly vigilant teams may alienate helpful external sources by restricting their contacts with these sources, for example, limiting their touchpoints with key users as well as with other teams. This curtails the inflow of critical knowledge inputs from external sources, thereby hurting the team's knowledge integration activities. Furthermore, by limiting their contacts with the external environment, teams run the risk of losing out on valuable feedback from helpful external entities, making them smug and complacent and hurting their motivation to try innovative approaches to improving project outcomes. This lack of motivation to innovate prevents the team members from performing critical project activities such as sharing their knowledge and ideas with each other.

Taken together, the above-mentioned arguments suggest that very high levels of sentry activities will diminish the team's ability to effectively integrate its internal knowledge resources for successful software development. These arguments are summarized in Table 1. Therefore, we predict that the influence of sentry activities on knowledge integration will follow an inverted-U shape—with optimum levels of knowledge integration more likely to occur at medium levels of sentry activities. In other words, there exists an inflection point beyond which increasing the level of sentry activities becomes counterproductive to the vendor team. Thus,

Hypothesis 1: There is an inverted U-shaped relationship between a vendor development team's sentry processes and its knowledge integration.

Table 1. Summary of Arguments for Curvilinear Relationships

	Low level	**Medium level**	**High level**
Sentry processes	• Team allows diverse external inputs, causing internal conflicts • Team allows too many perspectives on critical project issues, causing confusion	• Team minimizes unproductive information transfer from external sources, preventing its information processing capabilities from overloading	• Team overprotects members from valuable external feedback, making them smug and complacent • This hurts members' ability to actively perform critical project-related activities
Guard processes	• Team is more open to lending its resources to other teams • Team resources gain new skill sets and expertise, thus enriching the team's knowledge base	• Team lacks clarity about the extent of guard activities, leading to ineffective use of the team's internal resources	• Team closes ranks to focus on key processes • Forms supportive environment for knowledge integration

Guard Processes and Knowledge Integration

Guard activities are directed at actively controlling the outflow of internal information and resources by preventing their unauthorized or unwarranted access by external entities. Guard activities are therefore designed to help vendor teams decide what internal information and resources (e.g., a core team member's programming expertise) they might share externally, and with whom [2, 64].

Vendor development teams have to balance their time and resource constraints with external requests for information and resources. When faced with requests for information and resources from external entities, teams need to decide whether sharing information and resources will benefit the team or not. In order to optimize project outcomes, vendor teams routinely have to make smart choices about such issues.

Given that the guard activities are focused on controlling and limiting the access of external entities to the team's resources, teams may actually benefit from performing low-level guard activities. All teams have finite resources, and at some point in time, they may need to borrow resources from other teams [44, 64]. For example, one of the project leaders we interviewed remarked, "we all face resource crunch and sometimes another team has just what we need." In such routine situations, teams that perform low-level guard activities are more open to sharing their resources, such as a field expert, with other teams. As a direct benefit, working on different projects offers exposure to the team's experts and propels them to learn something new, such as a new skill set or the ability to see things from a different perspective [2]. As one project leader noted, "Other teams borrow our people, and we have noticed that when they come back, they have typically learnt something new. It's like we gained two resources by lending one!" This enriches a team's knowledge base because the new learning can be utilized to gain novel insights or to find innovative solutions to project-related problems. Thus, low-level guard activities can support the team's resource enrichment and knowledge integration [2].

Less strongly guarded teams schedule their project work with an understanding that a particular resource may or may not be available to them at a certain time. For example, one project leader explained, “Lending them [other teams] one of my people is not easy for me. I have to plan my timeline accordingly; especially if I have a scarce resource onboard that I know others will ask for.” As a consequence, despite a resource-sharing mindset, less highly guarded teams are still able to use their internal resources efficiently by planning in advance [82].

Interestingly, high-level guard activities could also facilitate team’s knowledge integration, albeit for very different reasons. Highly guarded teams are extremely vigilant in monitoring external requests for information, scrupulously determining the legitimacy of those requests, and tightly controlling the use of their resources by fulfilling a very limited number of requests [2, 64]. Ancona and Caldwell [2] observed that teams performing high-level guard activities often closed ranks to focus on their work without interruptions. Even in outsourced software development, it is common for vendor teams to display a very high level of guardedness for various reasons. For example, one respondent observed: “we are working on an intellectual property building project, and we control our internal information very strictly.” Such behaviors may also be helpful to teams facing impending deadlines, requiring them to focus on the project and not to release information that might hurt the project schedule or progress. For example, vendor teams typically perceive the management as driven by outcomes, budgets, and schedules, and to protect their own interests, the team may be very guarded in their updates to management [35]. As one respondent explained, “We share just enough information with the management to keep them satisfied. Sometimes, we tell them to back off, otherwise the project deliverable will suffer. That (reasoning) always does the job.” Closing their ranks to external interference not only helps teams to focus on their key processes but also creates a supportive environment within the team [2]. Members of such teams tend to form close working bonds, which facilitates their knowledge integration in many ways. First, close bonds remove the suspicion of opportunistic behavior among team members, so they can freely share knowledge that is critical to the project [72]. Second, a supportive environment motivates team members to share a variety of information without fear of ridicule, thereby enriching the team’s knowledge base. Finally, close working bonds promote harmony among team members and make them more amenable to using each other’s knowledge inputs for developing innovative project concepts [78].

Teams performing moderate levels of guard processes lose the benefits accrued by teams performing a high level of guard activities and by teams performing a low level of guard activities. Unlike teams performing high- and low-level guard activities, moderately guarded teams lack clarity about the extent to which they want to perform guard activities. For example, the team may sometimes fulfill management’s project status requests, and sometimes not. Or, the team may sometimes share their resources with peer teams and at other times avoid sharing. This behavior may lead them to be stereotyped by other entities as inconsistent, at best, and as noncooperative, at worst. More importantly, inconsistencies in guard activities hurt a team’s knowledge processes because the resources are not efficiently used. For example, the team may need a particular resource for its own project, but the resource may be on loan to another team at that time. At other times, the team may refuse to lend a resource that may be sitting idle, thereby forgoing the learning that could have enriched the team’s knowledge base [2]. As a project leader recounted her experience as a team member, “As a team member, I learnt what not to do as a project leader [PL]. My project didn’t need me and another PL would ask for me, but my PL

would not let me go, just in case I was needed there. It was so frustrating to lose all those growth opportunities." Such inconsistencies and their resulting dissatisfaction among the team members can also spoil the team's internal environment, which hinders its efforts to effectively combine team members' individual expertise and to share their ideas and knowledge with each other [69].

Based on these arguments, which are summarized in Table 1, we expect that a team's knowledge integration will decline as the team increases its guard activities, but there is an inflection point beyond which guard activities will start to be beneficial to the team, leading to a U-shaped curvilinear relationship.

> **Hypothesis 2:** There is a U-shaped curvilinear relationship between a vendor development team's guard processes and its knowledge integration.

Interaction Effects of Project Uncertainty and Sentry Processes

As noted earlier, project uncertainty plagues many software projects. Vendor teams working on projects marred with high uncertainty have to constantly seek critical information. The demand for external information inputs may be especially high during the requirements analysis phase the team has to refine the project scope and specifications [59]. Under these conditions, we expect the role of sentry processes to be even more salient for effective knowledge integration. This is because sentry processes may help teams working on high-uncertainty projects from overloading their information-processing capacity by preventing unwanted information inflows from the external environment [2]. This ability of sentry processes to serve as both facilitators and selective filters for accessing the most relevant and critical information inputs becomes even more important in uncertain projects. Sentry processes can also help negate high uncertainty by managing the within-team distribution of external information, for example, by ensuring that key inputs reach the team members who confront uncertainty the most [84]. Thus, compared to vendor teams working on low-uncertainty projects, those struggling with high-uncertainty projects can benefit more by using sentry processes as effective facilitators and filters of external information access and its internal distribution.

Sentry processes can also protect teams in high-uncertainty work settings from unwanted external influences. Compared to teams working in low-uncertainty settings, such teams obtain more frequent information, know-how, and feedback from multiple external entities, some of which may have an undue influence on a team's operations. For example, teams facing higher uncertainties may receive more stringent management feedback, which can sometimes demotivate the team [9]. Teams using sentry processes buffer themselves from such external influences by strictly monitoring what information reaches their members [2]. Furthermore, in high-uncertainty situations, teams spend valuable time and energy in gathering critical information, which leaves their members less time to allocate to its knowledge integration activities [20]. Sentry processes can be especially useful to improving a team's knowledge integration in such situations, by tightly managing the team's information gathering activities. Comparably, in low-uncertainty situations, sentry processes may be less helpful because the relative stability of project requirements may render the careful selection and filtering of new information inflow from external sources less relevant.

In summary, vendor development teams working on high-uncertainty projects can improve their knowledge integration in multiple ways by performing sentry activities. They can use sentry activities as both facilitators and selective filters for accessing the most relevant and critical information inputs. Sentry activities can also improve knowledge integration in uncertain projects by ensuring that key information inputs reach the team members who confront uncertainty the most. In addition, sentry processes can buffer the team from external disturbances, which leaves the team more time and energy to integrate information and face the challenges of high uncertainty. With the above arguments in mind, we propose the following hypothesis:

> **Hypothesis 3:** Project uncertainty will moderate the relationship between sentry processes and knowledge integration in such a way that the same level of sentry activity corresponds to a higher level of knowledge integration when project uncertainty is high.

Interaction Effects of Project Uncertainty and Guard Processes

Teams use guard activities to classify and respond to external demands for their knowledge resources [62]. Software teams facing high uncertainty lack critical information and resources that are necessary for a project, and to compensate, they regularly share information and resources with external sources. Teams performing guard activities may face problems doing this because high uncertainty can interact with the guard activities to make the teams less predisposed to sharing their information and resources. In other words, the guardedness of teams facing high uncertainty might prevent them from fulfilling even legitimate external requests for information and resources. This can come back to hurt the team's ability to source critical resources and knowledge inputs from external sources. For example, if a team facing high uncertainty receives a request from another team to borrow a technical expert, it has to decide whether the expert can be spared or not. The team may not need the expert at that moment, but given high uncertainty, the project requirements could change in the future, and then the team would need that expert. Guarded teams have typically been observed to be less open to sharing their information and resources [2], so teams performing guard processes may decide to decline the external request, thereby diminishing their chances of receiving critical knowledge inputs from the other team in future. Thus, teams performing guard activities under conditions of high uncertainty will hurt their knowledge integration.

External demands for information are also likely to be more frequent in uncertain projects, and performing guard activities very often to manage the heavy influx of demands can be detrimental to knowledge integration in high-uncertainty projects. For example, requirements uncertainty causes a great deal of flux in project scope, leading to more demands than usual for status updates from the client and vendor management. Teams performing guard activities have to perform them more frequently to prevent releasing "too much" information, which takes time away from their knowledge integration efforts. Furthermore, performing guard activities in high-uncertainty projects to monitor the outflow of information to the client hinders open communication with a critical source of external knowledge inputs. This is harmful to a team's knowledge integration, more so in high-uncertainty projects, which need critical knowledge inputs more than low-uncertainty projects. Therefore, we propose:

Hypothesis 4: Project uncertainty will moderate the relationship between guard processes and knowledge integration in such a way that the same level of guard activity corresponds to a lower level of knowledge integration when project uncertainty is high.

Methodology

Measures

To measure knowledge integration, items were developed based on previous studies [60, 74, 77]. A high score on the knowledge integration measure indicates that the team vigorously combined and aggregated the members' expertise, know-how, and ideas to jointly achieve project outcomes.

To measure sentry and guard processes, items were modified from prior studies to suit the context of software development projects (see [2, 3]). A high score on the items for sentry processes indicates that the team actively monitored and controlled the inflow of external information and explicitly decided what type of information or resource to acquire externally [2, 41]. The scale for guard processes included three items to measure the outflow of resources to external entities both within and outside the firm. A high score on the items for guard processes meant that the team actively controlled the outflow of internal information and resources to external entities and keenly prevented their unauthorized and unwarranted external access.

Following Nidumolu [59], project uncertainty was measured in terms of the *instability, diversity,* and *analyzability* of project requirements. Requirements instability represents the extent to which requirements changed over the course of the project. Requirements diversity connotes the extent of differences among clients about the requirements of the project. Finally, requirements analyzability represents "the extent to which the process of converting user needs to a set of requirements specifications can be reduced to mechanical steps or objective procedures" [59, p. 80]. A high score on uncertainty items meant that project requirements fluctuated quite a bit and the team had to exert a strong effort to reconcile the requirements.

To account for the possible effects of extraneous variables, we included six control variables. These variables are vendor firm (to control for firm-specific effects), team size (measured as the average number of members in the team), project duration (total time that it took to complete the project), project leader's experience (number of years of industry experience), knowledge heterogeneity of the project team, and relational capital within the project team. Knowledge heterogeneity represents the diversity of team members' technical and functional background and their expertise and skills [73]. Items were adapted from Campion, Medsker, and Higgs [12] to measure knowledge heterogeneity. Relational capital represents the extent of mutual trust, closeness of relationships, and reciprocity among the team members. Items to measure relational capital were adapted from Tiwana and McLean [77].

We refined the perceptual measures and assessed their validity in accordance with the recommendations of prior studies [28, 54]. We began with conceptual validation of the items by IS faculty, and four IS doctoral students who were researching various issues pertaining to software development projects. Each set was mixed up and given to an IS doctoral student.

These students were also provided names and definitions of the constructs, and were asked to sort the items by assigning them to various construct categories or an "other" (no fit) category. This process helped to identify items that were ambiguously worded or did not conceptually fit with other items of the same construct. The four sorters correctly assigned 94.4 percent of items to the intended constructs. The interrater reliability was 0.98. Based on this exercise, and on the feedback of four IS faculty members, nine items were dropped. These included two items each for knowledge integration, guard processes, sentry processes, and relational capital, and one item for requirements uncertainty.

To ensure that the resulting items were meaningful in the software development context, they were then subjected to a pretest. The items were placed in a questionnaire format and administered through the World Wide Web to fifty project leaders of vendor development teams in a Capable Maturity Model (CMM) Level-5 firm not included in the final data collection [63]. Thirty-six usable responses were received, of which 85.8 percent were male respondents and 14.2 percent were female. Respondents had an average 7.3 years of experience in the software industry. To test construct validity, exploratory factor analysis was conducted on the pretest data using the principal components analysis (PCA) extraction method with Varimax rotation. The analysis reported clean loadings (in excess of .70) of all items on their respective factors. Cronbach's alpha coefficients, calculated to assess scale reliabilities, were all in excess of .78. We report the final list of items and the sources from which they were adapted in Appendix Table A1.

Sample and Data Collection

To test the research model, data were collected over four months including the end of 2005 and the beginning of 2006. The questionnaires were administered through the World Wide Web to project leaders of vendor development teams. Only project leaders who had prior experience with teams on developing a single-application system were selected. Following the key informants methodology, data were collected from project leaders in sixteen mid- to large-sized software services firms in India. Project leaders were chosen as key informants because they possessed the most comprehensive knowledge of their team processes [72]. Thus, their responses could be used to represent aggregated information about the team activities being examined.

Links to the questionnaires were forwarded to the chief knowledge officer in sixteen CMM Level-5 certified software firms in India, who sent the link to 225 project leaders in their companies. CMM Level-5 certification ensured consistency in the software development processes of the responding firms. The project leaders were asked to respond to the survey based on a recently completed project that required developing a single application system. A total of 139 complete and usable responses were received (response rate of 61.8 percent). Project leaders had average industry experience of 8.9 years, with minimum experience of 1 year and maximum of 19 years. Average project duration was 12.1 months, with a minimum duration of 1 month and a maximum duration of 72 months. Average team size was about 17 members, with a minimum of two members per team and a maximum of 250 members per team. Comparisons of the sample demographics with population figures from an industry standard research report [57] yielded no significant differences, suggesting that nonresponse bias was unlikely to be an issue [5].

Additional Robustness Checks

We conducted several tests to assess the threat of common methods bias because data for both the dependent and the independent variables were collected from the same source and using the same method. Harman's one-factor test was conducted by entering all independent and dependent variables in an exploratory factor analysis [65]. If the data had a common method bias problem, a single factor would account for a large percentage of variance in the resulting factors. However, a single factor did not emerge, and the most covariance explained by one factor is 32.6 percent, indicating that common method bias was not a likely contaminant of our results. Next, we conducted the marker-variable (MV) test [49], which uses a theoretically unrelated "marker" variable to adjust the correlations among the model's principal constructs. Using the project leader's experience as the dummy marker variable (since it was not significantly correlated with any of the main constructs), we decided to partial out its effect from other correlations to assess the extent of common method variance. The partial correlations between knowledge integration and the independent variables remained more or less unchanged, suggesting that common method bias was not driving the observed relationships among the study variables.

Finally, prior studies have suggested that common method bias is unlikely to distort interaction effects [11]. It is unlikely that respondents would manipulate their responses on the respective scale items to produce interaction effects. Furthermore, because our model includes squared terms to examine curvilinear relationships, it would be extremely difficult for respondents to manipulate their responses to produce the desired effects [16].

Statistical Methods and Results

Partial least squares (PLS) technique was used to validate the measurement model and to test the hypotheses.[4] PLS is a second-generation structural equation modeling technique that uses a correlational, principal component-based approach to estimation [51]. It is a favorable technique for causal-predictive analysis in situations characterized by early stages of theory development [45]. Because this study is an early attempt to examine the theoretical underpinnings of teams' knowledge integration, PLS is an appropriate technique.

PLS is also preferred to regression in situations involving moderator analysis [15, 75]. Regression typically uses interaction terms to conduct moderator analysis. Moderated regression involving multiple item measures holds two key assumptions. The first assumption, called "equal item reliability," implies an equal contribution of all items toward estimating the interaction effect. The second assumption, referred to as "unchanging scale reliability," presumes no change in the reliability of the summated multiple item scale when it is applied in the theoretical model. But, as Chin, Marcolin, and Newsted profess, "Unfortunately, by virtue of the summation process, we have no opportunity to assess the validity of these two assumptions" [15, p. 190]. Thus, moderator's measurement error should be considered both in the initial reliability assessment and in the subsequent analysis of the theoretical model. The latent variable modeling approach within PLS allows the subsequent assessment of this error, thereby providing more accurate estimates of the interaction effects [15]. Goodhue, Lewis, and Thompson [30] raised concerns about the "product indicators" approach proposed by Chin, Marcolin, and Newsted to estimate interaction effects. Following their recommendation, moderation and quadratic terms

were computed using the "product of the sums" approach. For example, *SP* * *PU* was obtained as follows:

$$SP * PU = (SP_1 + SP_2 + SP_3) * (PU_1 + PU_2 + PU_3)$$

Estimation of the Measurement Model

The measurement model was assessed by examining the internal consistency, convergent validity, and discriminant validity of the constructs. As presented in Table 2,all item loadings on respective constructs were statistically significant, Cronbach's alpha scores were well above the recommended cutoff of 0.70 confirming the reliability of the scales, and composite reliabilities (ρ_c), which avoid the assumption of equal weighting of items, were all above .80 (see Table 2) [24, 27]. Convergent validity was assessed by calculating average variance extracted (AVE). All AVE values were higher than the recommended value of 0.5 [24].

We then assessed the discriminant validity of the measures in two ways. First, the measurement items displayed higher loadings on their "assigned factor" than on any other factor (Table 3) [27, p. 93]. Second, the square root values of AVEs are greater than the interconstruct correlations (shown in Table 2) [24, 75]. A more conservative estimate of discriminant validity is to compare the AVE values themselves to the correlations (square root values of AVEs are higher than the AVE values). This comparison also supported the discriminant validity of the constructs.

Estimation of the Structural Model

To deal with possible multicollinearity between the squared and the interaction terms, we mean-centered the independent variables before running the PLS models [75]. We also tested for multicollinearity by calculating variance inflation factors (VIFs) [34]. A VIF value in excess of 10 typically indicates that multicollinearity may be influencing estimates [37]. As shown in Table 2, the VIFs of all predictor variables were below 2, indicating that multicollinearity is not a problem.

In PLS, paths are interpreted as standardized regression weights. Thus, assessing the structural model involved estimating the magnitude, sign, and statistical significance of path coefficients in the model. We assessed three separate structural models (M1, M2, and M3). M1 included only the control variables, whereas M2 estimated the linear effects of sentry processes, guard processes, and requirements uncertainty and the quadratic effects of sentry and guard processes. M3 extended M2 to include the moderation effects of project uncertainty.

Table 2. Descriptive Statistics, Reliabilities, Validities, and Correlations

	Construct	**Mean(SD)**	**Cronbach's alpha**	**Composite reliability (ρ_c)**	**Item loadings (*t*-statistic)**	**AVE**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**VIF**
1	Project duration (months)	12.07				NA	NA									1.07
		(9.92)														
2	Team size	16.92				NA	0.22**	NA								1.10
		(24.21)														
3	Experience (years)	8.92				NA	0.10	0.08	NA							1.05
		(3.15)														
4	Knowledge heterogeneity	4.99	0.86	0.88	.78 (21.49)	0.71	0.09	0.09	–0.08	**0.84**						1.17
		(1.20)			.89 (42.63) .84 (24.24)											
5	Relational capital	5.10	0.78	0.86	.88 (40.07)	0.66	0.03	0.08	0.03	0.19*	**0.81**					1.19
		(1.10)			.81 (19.95) .74 (14.78)											
6	Project uncertainty	4.32	0.72	0.82	.81 (38.80)	0.61	0.05	0.14	–0.17	0.08	–0.14	**0.78**				1.09
		(1.51)			.84 (53.55) .70 (24.36)											
7	Sentry processes	4.72	0.84	0.88	.88 (87.51)	0.70	0.02	0.07	–0.14	0.33**	0.40**	–0.27**	**0.84**			1.43
		(1.14)			.88 (72.28)											
8	Guard processes	4.04 (1.26)	0.73	0.88	.78 (15.68) .96 (47.12) .81 (21.48)	0.76	–0.02	0.02	–0.05	0.25**	0.16	–0.07	0.25**	**0.88**		1.12
9	Knowledge integration	5.52 (1.10)	0.92	0.94	.76 (61.67) .84 (84.92) .94 (118.57)	0.80	0.09	–0.03	–0.10	0.25**	0.58**	–0.33**	0.53**	0.14	**0.89**	—

Notes: **Correlation is significant at the 0.01 level (two-tailed); *correlation is significant at the 0.05 level (two-tailed). Diagonal elements show the square-root of average variance extracted (AVE) for each construct. Bold diagonal elements show the square-root of average variance extracted (AVE) for each construct.

Table 3. Item Loadings and Cross-Loadings

	KI	**KH**	**RC**	**PU**	**SP**	**GP**
KI_1	**0.7690**	0.2196	0.4935	–0.3253	0.4180	0.1551
KI_2	**0.8412**	0.3007	0.5309	–0.3310	0.4709	0.1202
KI_3	**0.9416**	0.1843	0.6471	–0.3322	0.4843	0.1522
KI_4	**0.9478**	0.1841	0.6018	–0.3801	0.5356	0.1112
KH_1	0.2037	**0.8418**	0.1624	–0.0696	0.2540	0.1059
KH_2	0.2159	**0.8991**	0.1791	–0.0363	0.3389	0.1952
KH_3	0.1382	**0.7837**	0.1275	–0.0219	0.2655	0.2767
RC_1	0.4399	0.1695	**0.8129**	–0.0587	0.3272	0.2507
RC_2	0.3507	0.1718	**0.7481**	–0.0666	0.2134	0.0786
RC_3	0.7033	0.1406	**0.8803**	–0.3177	0.4207	0.1358
PU_1	–0.3938	–0.1390	–0.2546	**0.8449**	–0.3168	–0.1678
PU_2	–0.2649	–0.0161	–0.1681	**0.8108**	–0.1944	–0.0412
PU_3	–0.1812	0.1288	–0.0211	**0.7059**	–0.1657	–0.0042
SP_1	0.4932	0.3401	0.3501	–0.2769	**0.8858**	0.2345
SP_2	0.4772	0.2853	0.3364	–0.2860	**0.8828**	0.2751
GP_1	0.0632	0.2413	0.0530	–0.0688	0.2744	**0.7816**
GP_2	0.1568	0.1787	0.2202	–0.1200	0.2476	**0.9679**
GP_3	0.3907	0.2231	0.3663	–0.1975	0.1875	**0.8125**

Notes: KI = knowledge integration; KH = knowledge heterogeneity; RC = relational capital; PU = project uncertainty; SP = sentry processes; GP = guard processes.

The models (M1–M3) are as follows:[5]

Knowledge Integration =

$$\textbf{Step1}(\textbf{M1}): \alpha_1 + \beta_1 \text{Duration} + \beta_2 \text{TeamSize} + \beta_3 \text{Experience} + \beta_4 \text{KnowledgeHeterogeneity} + \beta_5 \text{RelationalTrust}$$

$$\textbf{Step2}\ (\textbf{M2}): M1 + \beta_6 \text{SentryProcess} + \beta_7 \text{GuardProcess} + \beta_8 \text{ProjectUncertainty} + \beta_9 \text{SentryProcess}^2 + \beta_{10} \text{GuardProcess}^2$$

$$\textbf{Step3}(\textbf{M3}): M2 + \beta_{11} \text{Project Uncertainty}^2 + \beta_{12}(\text{ProjectUncertainty} * \text{SentryProcess}) + \beta_{13}(\text{ProjectUncertainty} * \text{GuardProcess}) + \beta_{14}(\text{ProjectUncertainty} * \text{SentryProcess}^2) + \beta_{15}(\text{ProjectUncertainty} * \text{GuardProcess}^2)$$

A bootstrapping method using 1,000 resamples was used to determine the statistical significance of the parameter estimates. Results indicate that the controls only model (M1) explained 46 percent of variance in knowledge integration. Adding main effects to the model (M2 in Figure 1) increased the variance explained to 58 percent, while the full model (M3 in Figure 2) explained 61 percent variance.

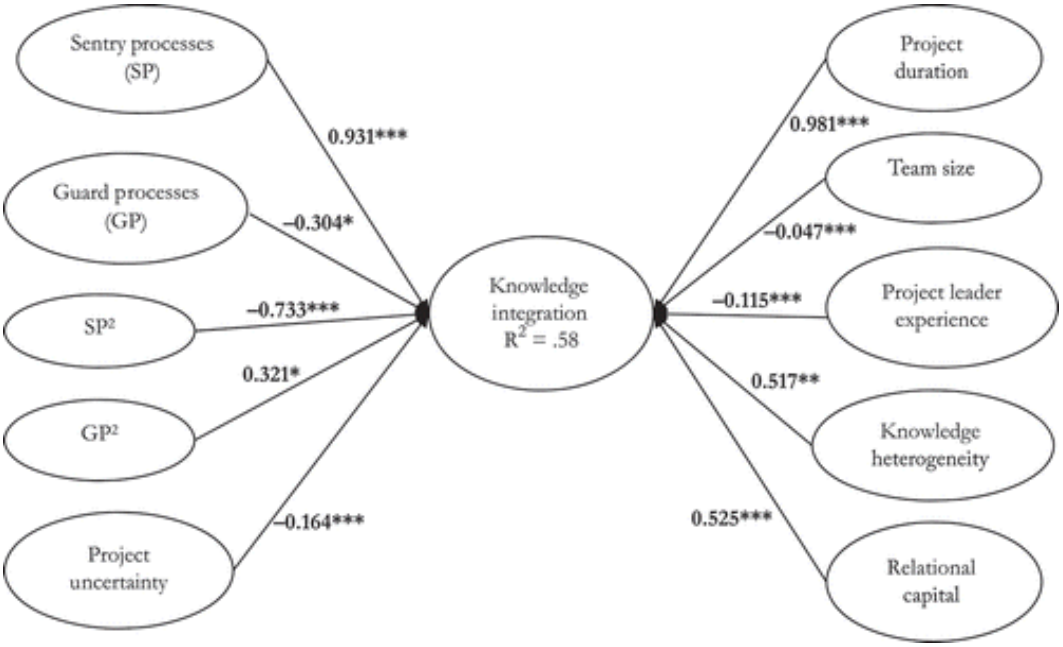

Figure 1. Main Effects Model (M2)
*$p < 0.05$; **$p < 0.01$; ***$p < 0.001$.

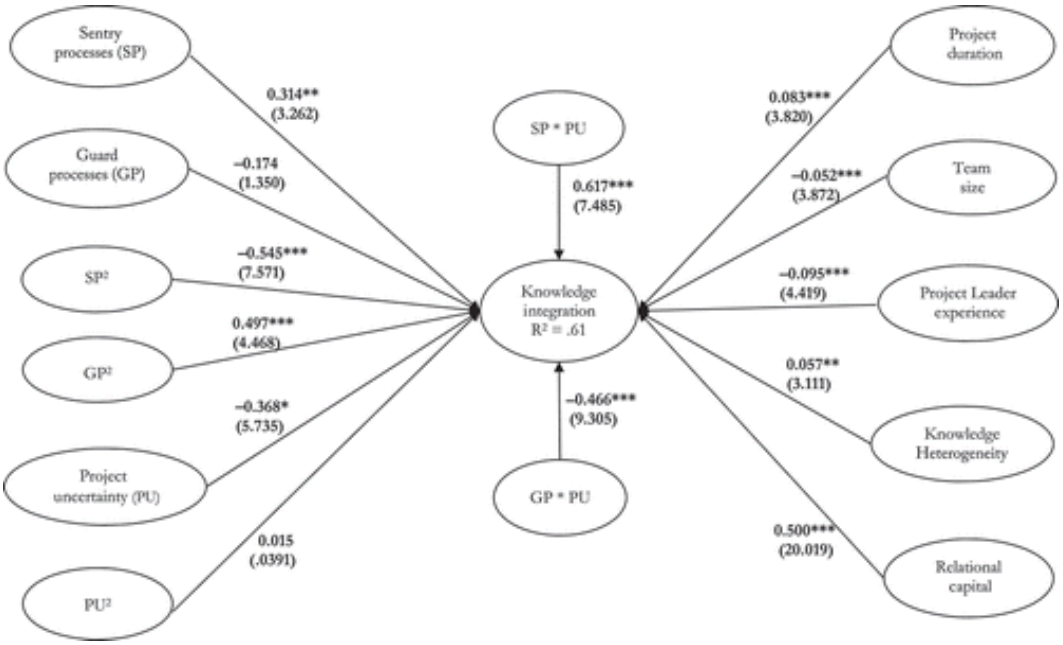

Figure 2. Full Model (M3)
*$p < 0.05$; **$p < 0.01$; ***$p < 0.001$; numbers in parentheses are respective *t*-values.

Among the control variables, project duration, knowledge heterogeneity, and relational capital were positively related to knowledge integration ($p < 0.001$). The effect of project duration is understandable because in projects that last longer, team members may have a better idea of each other's expertise and skill sets, which might help their knowledge integration efforts. Similarly, teams with more knowledge heterogeneity hold diverse sets of expertise and skills, and do have an opportunity to integrate these resources for better project outcomes [10]. It is also understandable that a high level of relational capital in software teams, as indicated by strong mutual trust, close working relationships, and a culture of give-and-take among team members, may improve a team's knowledge integration [77]. Project team size had a negative impact on knowledge integration ($p < 0.001$), suggesting that larger teams had more difficulty in effectively integrating knowledge. Project leader's experience also had a negative impact on knowledge integration ($p < 0.001$), suggesting that the more experienced project leaders were less open to combining individual knowledge, expertise, and ideas.

In the main effects Model M2 (see Figure 1), we added the sentry processes (SP), guard processes (GP), project uncertainty (PU), SP^2, and GP^2 terms, and found that SP was positively related to knowledge integration ($p < 0.001$), whereas SP^2 was negatively related to knowledge integration ($p < 0.001$), thereby supporting the inverted U-shaped relationship between sentry activities and knowledge integration (H1). In addition, GP was negatively related to knowledge integration ($p < 0.05$) and GP^2 was positively related to knowledge integration ($p < 0.05$), which provided support for the U-shaped association between guard activities and knowledge integration (H2). The main effect of PU was found to be overall negative and significant ($p < 0.001$). The R^2 value of 0.58 for Model M2 showed that the addition of SP, GP, PU, SP^2, and GP^2 accounted for an additional 12 percent of the variance in knowledge integration when compared to the controls only model. To gain further insights into the relationships modeled in this study, we plotted the effects of sentry and guard process on knowledge integration. As

depicted in Figure 3,[6] the curvilinearity of the graphs corresponds to the predictions in the hypothesis. Also, we conducted the test to confirm the presence of inverted U-shaped and U-shaped relationships.[7] The results confirmed that our curvilinear relationships fulfilled the requirements of a U-shaped and an inverted U-shaped relationship. For the U-shaped relationship between guard processes and knowledge integration, we have one extreme point (8.249), and the slope starts with a negative value (–0.7714) and changes to positive at the end of the interval (0.7100). For the inverted U-shaped relationship between sentry processes and knowledge integration, we have one extreme point (16.707), and the slope starts with a positive value (1.344) and changes to negative at the end of the interval (–0.4209).

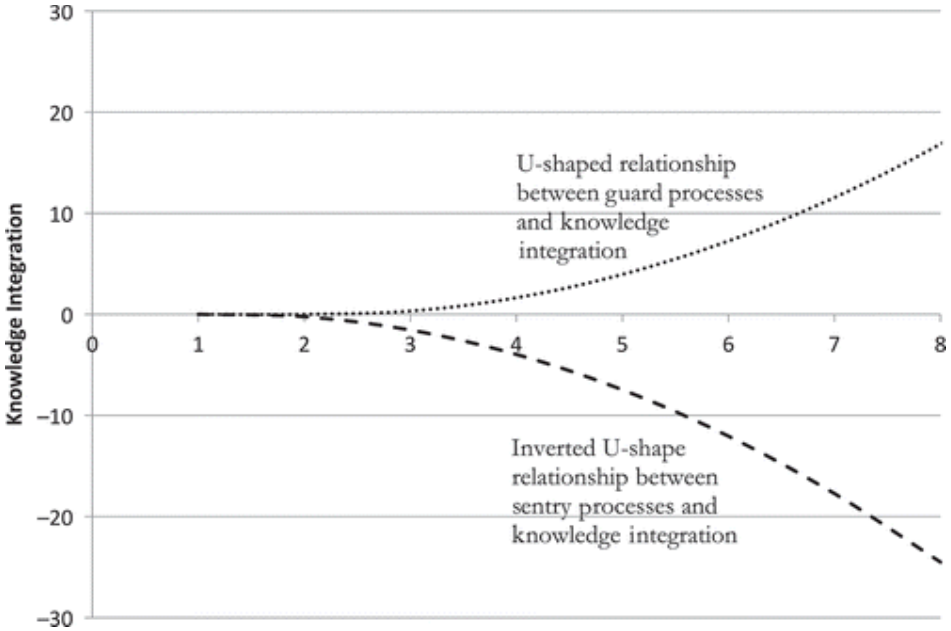

Figure 3. Curvilinear Relationships of Sentry & Guard Activities with Knowledge Integration

Finally, in the full model (M3 in Figure 2), we tested for the interaction effects of project uncertainty with sentry and guard activities. The interaction term of project uncertainty and sentry processes (SP * PU) was positively related to knowledge integration ($p < 0.001$), and the interaction term of project uncertainty and guard processes (GP * PU) was negatively related to knowledge integration ($p < 0.001$). As predicted, project uncertainty moderated the relationship between sentry processes and knowledge integration such that teams performing sentry activities under high uncertainty had higher knowledge integration compared to teams performing sentry activities under low uncertainty (H3). Project uncertainty also moderated the influence of guard activities on knowledge integration, such that teams performing guard activities under high uncertainty conditions had lower knowledge integration compared to teams performing guard activities under low uncertainty (H4).

The model R^2 for the full model (which included all the controls, main effects, second-order effects, and interactions) was 0.61, which is higher than the R^2 for M2. We also calculated effect size (f^2) for the full model. Effect size can be calculated as:

$$\frac{R^2_{\text{included}} - R^2_{\text{excluded}}}{1 - R^2_{\text{included}}}$$

where, $R^2_{included}$ and $R^2_{excluded}$ are the squared Rs for the dependent latent variable when the interaction term is included and omitted, respectively, in the main-effects model. Values of 0.02, 0.15, and 0.35 are recommended as small, moderate, and large effects, respectively [18]. The f^2 value for the full model was .08, which does not necessarily connote an unimportant effect. As Chin, Marcolin, and Newsted explain, "Even a small interaction can be significant under extreme moderating conditions, if the resulting beta changes are meaningful, then it is important to take these conditions into account" [15, p. 211].

Q^2 is another measure of predictive relevance of the PLS (structural) model [43]. It is calculated using a blindfolding procedure that excludes a part of the data for a particular block of indicators during parameter estimations, and then tries to estimate the omitted part using the estimated parameter [15]. $Q^2 > 0$ means that the model has predictive relevance, whereas $Q^2 < 0$ suggests a lack of it. Q^2 values of 0.31, 0.38, and 0.41 were obtained for the controls-only model, main-effects model, and full model respectively, which suggests that all the models have predictive relevance. To account for multiple projects per firm, we performed two robustness checks. First, we tested for any significant firm-level effects using hierarchical linear modeling (HLM). As a first step, a one-way analysis of variance was performed to confirm that the variability in the outcome variable, that is, knowledge integration, by a level-2 group, that is, a firm, is significantly different from zero. This tests whether there are any differences at the firm level on the knowledge integration, and confirms whether HLM is necessary [81]. The chi-square test statistic was not significant ($\chi^2 = 4.225$, $p = 0.237$), indicating that there is no variance in our outcome variable by firm-level groupings, and that there is no statistical justification for running HLM.

Second, we conducted a fixed-effects regression with clustered heteroskedasticity-corrected standard errors [80]. By using a fixed-effects model with firm and clustered standard errors, we ensured that no firm-specific effects impacted the results. After mean-centering the independent variables [1], we first tested a model with all the control variables, the first-order terms of the main-effect variables, namely, sentry and guard processes, and the quadratic terms of these two variables in order to test our prediction that sentry and guard processes would exhibit curvilinear relationships with knowledge integration. Next, we tested the moderating effect of project uncertainty by adding the interaction terms.

The results of regression analysis, as shown in Table 4, were the same as those of PLS analysis, indicating that no firm-specific effects were impacting our results. Then, we followed Aiken and West's [1] procedures to plot graphs to further investigate the curvilinear relationships between sentry and guard processes and knowledge integration under varied levels of project uncertainty. Specifically, we examined the relationship at low uncertainty (two standard deviations below the mean), medium (project uncertainty at the mean value), and high uncertainty (two standard deviations above the mean). The interaction plots (see Figures 4a and 4b)[8] lend support to the predictions made in H3 and H4. The curve representing the relationship between sentry processes and knowledge integration shifts upward under conditions of high uncertainty but remains an inverted-U shape in low uncertainty conditions—— indicating that teams performing sentry processes under high-uncertainty conditions will improve their knowledge integration more than teams performing sentry processes under low-uncertainty conditions. Interestingly, the

curve representing the relationship between guard processes and knowledge integration shifts downward under conditions of high uncertainty, indicating that teams performing guard activities under conditions of high uncertainty will have lower levels of knowledge integration than teams performing guard activities under low-uncertainty conditions. Overall, the figures show that sentry and guard processes become much more critical to knowledge integration when dealing with high-uncertainty projects.

Table 4. Regression Results

Variable	Model 1	Model 2	Model 3
Control variables	0.01**	0.007*	0.00
Duration	–0.003**	–0.003*	–0.002*
Team size	0.02	0.012	0.019
Project leader experience	0.11**	0.07	0.13**
Knowledge heterogeneity	0.45***	0.36***	0.37***
Relational trust			
Main effects		1.31***	0.69**
Sentry processes		–0.58**	–0.06***
Guard processes			
Sentry processes^2 (H1)		–0.12***	–0.56*
Guard processes^2 (H2)		0.06*	0.06**
Project uncertainty			–0.25***
Interaction effects			
Sentry processes X Project uncertainty (H3)			0.21**
Guard processes X Project uncertainty (H4)			–0.11**
N	139	139	139
R^2	0.40	0.51	0.62
Change in R^2		0.11***	0.11***

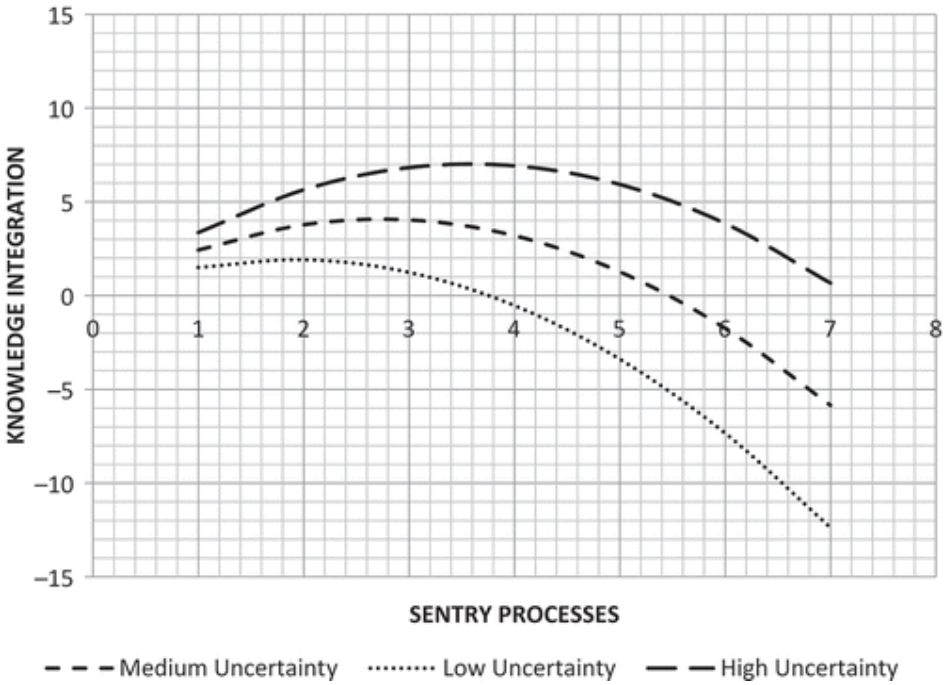

Figure 4a. Moderating Effects of Project Uncertainty on Sentry Processes and Knowledge Integration

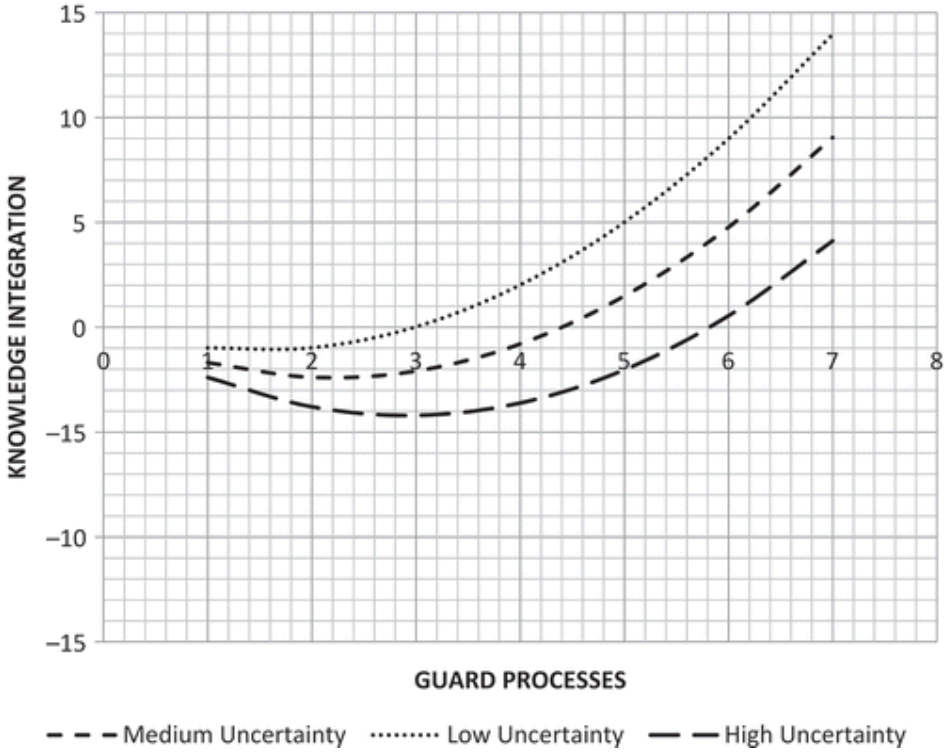

Figure 4b. Moderating Effects of Project Uncertainty on Guard Processes and Knowledge Integration

Discussion

Building on the literature on team-level processes, we examined the relationships between sentry and guard activities and knowledge integration in software development teams. Furthermore, given the salience of uncertainty in software development, we examined the extent to which the relationships observed here were moderated by the level of uncertainty faced by the software development team. Our results demonstrate complex, curvilinear relationships between sentry and guard processes and a team's knowledge integration and a significant influence of project uncertainty on these team processes. These results not only provide impetus for future research but also have sound implications for practice as noted below.

Implications for Research

This paper makes two important theoretical contributions. First, it provides a theoretical explanation for how and why sentry and guard activities, which have previously been identified as important team-level processes, impact knowledge integration in software teams. Knowledge integration has been described in prior IS literature [cf. 53, 76] as a critical antecedent of overall software team performance. Therefore, there is significant interest in examining the underlying team level factors that impact a team's ability to successfully integrate knowledge. Our work extends the findings of Ancona and Caldwell [2], who first observed sentry and guard to be critical team-level activities in new product development teams. They conjectured that there were associations between these activity sets and team performance, but the nature of the

associations was not clear. Moreover, in the context of information systems development, it was not understood whether these activity sets were: (a) prevalent among software teams, or (b) had any influence on the team's knowledge integration ability. Our study confirmed the existence of complex curvilinear associations among sentry, guard, and knowledge integration activities.

The second contribution of this study is its examination of how the patterns of relationships among team-level activity sets (sentry and guard) and knowledge integration are impacted by the level of uncertainty that the project faces. It is a well-known fact that when compared to other projects, software projects typically encounter higher levels of uncertainty and it is not surprising that a great deal of work in the IS literature has examined the direct impact of uncertainty on software project performance [cf. 59]. However, the nuances of *how* uncertainty interacts with other team-level activities to influence performance parameters remain a mystery. Our findings help to explain how activities aimed at regulating the inflow and outflow of information to and from software teams influence project teams facing greater flux in their requirements.

The current study has certain limitations that provide opportunities for future research. First, while we examine the extent to which teams engage in sentry and guard activities, we do not investigate how these activities are carried out. In other words, the roles that team leaders and team members play as gatekeepers is not something we study or measure. Hence the study is unable to shed any light on how these important activities are coordinated within the team. This limitation can be an important basis for future studies aimed at studying roles and behaviors that affect the extent to which teams engage in sentry and guard activities. In particular, the studying the role of the team leader vis-à-vis other team members would shed light on how the powers associated with performing sentry and guard activities are negotiated within teams. A related offshoot of this topic could entail identifying the specific skills and personality types of people who are more comfortable performing sentry and guard activities.

Second, prior research points to varying levels of internal and external information and resource sharing across the different project stages, which suggests that teams might judiciously vary the extent to which they engage in sentry and guard activities during the different stages in the project life cycle [35, 71]. However, our study treated these effects as invariant over the entire project and only measured them at the end of the project. It would be interesting to extend this study to see how the effects seen here vary across the different stages of software development projects. For example, in the analysis stage, more external access might be preferred, whereas in the development stage, teams may actively seek to shut off certain external channels of communication in order to protect against the threat of scope creep [71]. Another interesting area of inquiry would be the relationship between sentry and guard activities. For example, future research can examine whether different levels of these activities interact with one another to affect project outcomes.[9]

A third limitation stems from the fact that all the data for the study come only from the responses of the team leaders and not from any of the team members. Although we were careful to guard against the well-known biases of single-respondent studies, additional concerns that team leaders may not fully reflect the underlying "true behaviors" of their teams persist. Triangulating the data obtained from team leaders with those obtained from team members should be an important consideration in future studies. An additional caveat in interpreting the results of this study is that

the data are obtained from software teams from Indian vendor firms working on outsourced projects. It would be worthwhile to determine whether these results hold for other types of projects—internal (nonoutsourced) projects as well as projects outsourced to vendors in other countries. Similarly, it would be interesting to examine whether the results hold for projects that are deemed "important" or are "highly visible" because they involve developing a "revolutionary" application.

Finally, in this study we propose that a high level of guard processes will facilitate a team's knowledge integration. One reason to support this proposition is that a high level of guard processes helps teams to close ranks on external entities such as client and vendor management, so the team can focus on key processes such as knowledge integration. Interestingly, this might create principal–agent information asymmetry where the agents (software teams) have some vested interests in guarding the information from the principals (clients and management).[10] Therefore, although a high level of guard activities might aid in team-level knowledge integration efforts, potentially adverse consequences could appear elsewhere, such as diminished trust between the team and external entities. Thus, future studies could examine whether sentry and guard activities affect these other project-level outcomes.

Implications for Practice

Our study provides several novel and useful insights to managers in charge of software development. Software project team leaders seeking to enhance knowledge integration for improved team performance would do well to pay attention to how well their teams manage the activity sets of sentry and guard, insofar as they influence the team's ability to integrate knowledge successfully. For knowledge-intensive projects, team leaders are known to assemble software teams with greater knowledge heterogeneity and relational capital in order to have better access to different types of knowledge. But even after controlling for these well-known factors, we find that the manner in which software teams share and protect the information resources they have impacts the knowledge integration ability of the team. Since software teams operate under conditions of resource dependence, team leaders need to ensure that their teams have not only the requisite technical skills but also the ability to import and transfer information and resources. This also implies that organizations develop holistic performance appraisal policies that assess individuals for both intergroup and within-group activities.

Software teams sometimes perform sentry processes excessively, mostly to protect themselves from political agendas of external entities, but team leaders need to realize that excessive sentry processes put tremendous pressure on the people performing these processes. Individuals performing sentry activities typically have high levels of stress created by the high personal cost of protecting the team from external pressure [2]. This could impede their contribution to the team's knowledge integration efforts. In addition, software teams typically depend on each other for exchanging critical information and resources, and teams performing excessive sentry activities may run the risk of being branded as apathetic to external help, which could dry up their pool of helpful external sources.

Software team leaders should also calibrate their guard activities. A low level of guard activities enriches team's knowledge base by exposing team members to new knowledge. Thus, keeping

guard processes at a low–key level might be better for the team's knowledge integration, unless situations, such as impending project deadlines, warrant a high level of guard activities. In such situations team leaders need to judiciously decide what information to share and what sharing mechanisms to follow (e.g., planned releases). Another situation could be a client's demand for secrecy because of the classified nature of the application being developed. A high level of guard activities is also justified if the team needs to protect its unique or rare information and resources [2].

Our findings regarding the impact of uncertainty have key implications for project leaders. The project leaders not only should know what degree of sentry and guard activities they need to improve project outcomes, but also need to be aware that requirements uncertainty could shift those relationships. Project leaders facing high uncertainty should therefore calibrate the levels of sentry and guard activities appropriately.

Conclusion

To improve their project outcomes, software teams must integrate their knowledge resources effectively. We examined the impact of sentry and guard activities, which teams use to manage the complex inflow and outflow of information and resources, on knowledge integration. Furthermore, given the presence of uncertainty in most software projects, we examined how these associations shift under conditions of uncertainty. Results highlight complex curvilinear associations between sentry and guard activities, and knowledge integration. Enhancing our understanding of knowledge integration, the results also show that the knowledge integration of a team battling high uncertainty will benefit more by the performance of sentry activities. On the other hand, performing guard activities under high uncertainty conditions will lead to lower level of knowledge integration. These results should be analyzed in light of some limitations. We do not examine how teams perform sentry and guard activities, nor do we examine whether software teams vary the degree of sentry and guard activities during different project stages. We also do not examine any unintended effects of guard activities, such as diminished trust levels between the team and other project stakeholders. Despite these limitations, the results have some key implications. We recommend that team leaders should intentionally staff their teams with people capable of performing both inter- and intragroup activities. Team leaders will also need to carefully calibrate both sentry and guard activities to optimize their project outcomes, more so if the project is facing high uncertainty.

Notes

1. New product development teams have been described as being similar in goals and structure to information systems development (ISD) teams. For more details on how social activities of ISD teams are comparable to those of new product development, please see [55, 71].

2. Please note that we are referring to the processes that software teams use to manage their formal and informal interactions with external entities.

3. From here onward, team members' knowledge integration will be referred to as knowledge integration.

4. SmartPLS 2.0 was used to conduct the PLS analysis.

5. When discussing Models M1–M3 the abbreviations SP (sentry process), GP (guard process), KI (knowledge integration), and PU (project uncertainty) are used.

6. To generate Figure 3 we plotted the equation in Model M3, but taking only the coefficients of SP and SP^2 and GP and GP^2 respectively, and keeping uncertainty at its mean value.

7. The test is proposed in J.T. Lind and H. Mehlum, With or without U? The appropriate test for a U-shaped relationship," *Oxford Bulletin of Economics and Statistics, 72,* 1 (2010), 109–118.

8. To generate Figures 4a and 4b, we plotted the equation in Model M3, but taking only the coefficients of SP and SP^2 and GP and GP^2, respectively, and keeping uncertainty at its mean value and two standard deviations below the mean (for low uncertainty) and two standard deviations above the mean (for high uncertainty).

9. We would like to thank one of the reviewers for highlighting this implication.

10. We would like to thank one of the reviewers for highlighting this possibility.

References

1. Aiken, L.S., and West, S.G. *Multiple Regression: Testing and Interpreting Interactions*. Newbury Park, CA: Sage, 1991.
2. Ancona, D.G., and Caldwell, D.F. Beyond task and maintenance: Defining external functions in groups. *Group and Organizational Studies*, 13 (1988), 468–494.
3. Ancona, D.G., and Caldwell, D.F. Bridging the boundary: External activity and performance in organizational teams. *Administrative Science Quarterly*, 37 (1992), 634–665.
4. Argote, L. Input uncertainty and organizational coordination in hospital emergency units. *Administrative Science Quarterly*, 27 (1982), 143–174.
5. Armstrong, J.S. and Overton, T.S. Estimating nonresponse bias in mail surveys. *Journal of Marketing Research*, 14 (1977), 396–402.
6. Awazu, Y. Knowledge management in distributed environments: Roles of informal network players. In System Sciences, In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences*, Hawaii: IEEE, (2004, January). (pp. 6)
7. Barki, H.; Rivard, S.; and Talbot, J. An integrative contingency model of software project management. *Journal of Management Information Systems*, 17, 4 (2001), 37–69.
8. Bartel, C. Social comparisons in boundary spanning work: Effects of community outreach on members' organizational identity and location. *Administrative Science Quarterly*, 46 (2001) 379–413.
9. Boehm, B.W. *Software Risk Management*. Washington, DC: IEEE Computer Society Press, 1989.
10. Bittner, E.A.C., and Leimeister, J.M. Creating shared understanding in heterogeneous work groups: Why it matters and how to achieve it. *Journal of Management Information Systems*, 31, 1 (2014), 111–143.
11. Brockner, J. When trust matters: The moderating effect of outcome favorability. *Administrative Science Quarterly*, 42 (1997), 558–583.
12. Campion, M.A.; Medsker, G.J.; and Higgs, A.C. Relations between work group characteristics and effectiveness: Implications for designing effective work groups. *Personnel Psychology*, 46, 4 (1993), 823–850.

13. Cha, H.S.; Pingry, D.E.; and Thatcher, M.E. A learning model of information technology outsourcing: Normative implications. *Journal of Management Information Systems*, 26, 2 (2009), 147–176.
14. Chin, W.W.; Marcolin, B.L.; and Newsted, P. R. A partial least squares latent variable modeling approach for measuring interaction effects: Results from a Monte Carlo simulation study and an electronic-mail emotion/adoption study. *Information Systems Research*, 14, 2 (2003), 189–217.
15. Chou, C., and Yang, K. The interaction effect of strategic orientations on new product performance in the high-tech industry: A nonlinear model. *Technological Forecasting and Social Change*, 78 (2011), 63–74.
16. Choudhury, V., and Sabherwal, R. Portfolios of control in outsourced software development projects. *Information Systems Research*, 14, 3 (2003), 291–314.
17. Cohen, J. *Statistical Power Analysis for the Behavioral Sciences*. Hillsdale, NJ: Lawrence Erlbaum, 1988.
18. Cummings, J.N. Work groups, structural diversity, and knowledge sharing in a global organization. *Management Science*, *50*, 3 (2004), 352–364.
19. Dutta, A., and Roy, R. Offshore outsourcing: A dynamic causal model of counteracting forces. *Journal of Management Information Systems*, 22, 2 (2005), 15–35.
20. Eisenhardt, K., and Bourgeois, L.J. Politics of strategic decision making in high-velocity environments: Toward a midrange theory. *Academy of Management Journal*, 31 (1988), 737–770.
21. Espinosa, J.A.; Cummings, J.N.; Wilson, J.M.; and Pearce, B.M. Team boundary issues across multiple global firms. *Journal of Management Information Systems*, 19, 4 (2003), 157–190.
22. Espinosa, J.A.; Slaughter, S.A.; Kraut, R.E.; and Herbsleb, J.D. Team knowledge and coordination in geographically distributed software development. *Journal of Management Information Systems*, 24, 1 (2007), 135–169.
23. Faraj, S., and Sproull, L. Coordinating expertise in software development teams. *Management Science*, 46 (2000), 1554–1568.
24. Fornell, C., and Larker, D.F. Evaluating structural equations models with unobserved variables and measurement error. *Journal of Marketing Research*, 18 (1981), 39–50.
25. Galeghar, J., and Kraut, R.E. Computer-mediated communication for intellectual teamwork: An experiment in group writing. *Information Systems Research*, 5, 2 (1994), 110–138.
26. Gefen, D.; Benbasat, I.; and Pavlov, P. A research agenda for trust in online environments. *Journal of Management Information Systems*, *24*, 4 (2008), 275–286.
27. Gefen, D., and Straub, D. A practical guide to factorial validity using PLS-GRAPH: Tutorial and annotated example. *Communications of the AIS*, 16 (2005), 91–109.
28. Gerbing, D.W., and Anderson, J.C. An updated paradigm for scale development incorporating unidimensionality and its assessment. *Journal of Marketing Research*, 25 (1988), 186–192.
29. Gladstein, D. Groups in context: A model of task group effectiveness. *Administrative Science Quarterly*, 29 (1984), 499–517.
30. Goodhue, D.; Lewis, W.; and Thompson, R. Statistical power in analyzing interaction effects: Questioning the advantage of PLS with product indicators. *Information Systems Research*, 18, 2 (2007), 211–227.

31. Gopal, A., and Gosain, S. The Role of organizational controls and boundary spanning in software development outsourcing: Implications for project performance. *Information Systems Research*, 21, 4 (2010), 960–982.
32. Grant, R.M. Toward a knowledge-based theory of the firm. *Strategic Management Journal*, 17 (1996), 109–122.
33. Grant, R.M. Prospering in dynamically-competitive environments: Organizational capability as knowledge integration. *Organization* Science, 7, 4 (1996), 375–387.
34. Greene, W.H. *Econometric Analysis*. 6th ed. Upper Saddle River, NJ: Prentice Hall, 2003.
35. Guinan, P.J.; Cooprider, J.G.; and Faraj, S. Enabling software development team performance during requirements definition: A behavioral versus technical approach. *Information Systems Research*, 9, 2 (1998), 101–125.
36. Haas, M.R. Knowledge gathering, team capabilities, and project performance in challenging work environments. *Management Science*, *52*, 8 (2006), 1170–1184.
37. Hair, J.F.; Anderson, R.E.; Tatham, R.L., and Black, W.C. *Multivariate Data Analysis*. Englewood Cliffs, NJ: Prentice Hall, 1998.
38. Hansen, M.T. The search-transfer problem: The role of weak ties in sharing knowledge across organization subunits. *Administrative Science Quarterly*, 44 (1999), 82–111.
39. Harter, D.E.; Krishnan, M.S.; and Slaughter, S.A. Effects of process maturity on quality, cycle time, and effort in software product development. *Management Science*, 46, 4 (2000), 451–466.
40. Jain, R.P.; Simon, J.C.; and Poston, R.S. Mitigating vendor silence in offshore outsourcing: An empirical investigation. *Journal of Management Information Systems*, 27, 4 (2011), 261–297
41. Jemison, D.B. The importance of boundary spanning roles in strategic decision making. *Journal of Management Studies*, 21, 2 (1984), 131–152.
42. Jiang, J.J.; Chang, J.Y.T.; Chen, H.; Wang, E.T.G.; and Klein, G. Achieving IT program goals with integrative conflict management. *Journal of Management Information Systems*, 31, 1 (2014), 79–106.
43. Jöreskog, K.G., and Wold, H. The ML and PLS techniques for modeling with latent variables: Historical and competitive aspects. In K.G. Jöreskog and H. Wold (eds.), *Systems Under Indirect Observation: Causality, Structure, Prediction*. Amsterdam: North-Holland, 1982, pp. 263–270.
44. Joshi, A. 2006. The influence of organizational demography on the external networking behavior of teams. *Academy of Management Review, 31*, 3 (2006), 583–595.
45. Kankanhalli, A.; Tan, B.C.Y.; and Wei, K. Contributing knowledge to electronic knowledge repositories: An empirical investigation. *MIS Quarterly*, 29, 1 (2005), 113–143.
46. Kim, K.K., and Umanath, N.S. Structure and perceived effectiveness of software development subunits: A task contingency analysis. *Journal of Management Information* Systems, 9, 3 (1992–1993), 157–181.
47. Kirsch, L. Portfolios of control modes and IS project management. *Information Systems Research*, 8, 3 (1997), 215–240.
48. Levina, N., and Ross, J. From the vendor's perspective: Exploring the value proposition in IT outsourcing. *MIS Quarterly*, 27, 3 (2003), 331–364.
49. Lindell, M., and Whitney, D. Accounting for common method variance in cross-sectional research designs. *Journal of Applied Psychology*, *86*, 1 (2001), 114–121.

50. Majchrzak, A.; Beath, C.; Lim, R.; and Chin, W.W. Managing client dialogues during information systems design to facilitate client learning. *MIS Quarterly*, 29, 4 (2005), 653–672.
51. Mastrogiacomo, S.; Missonier, S.; and Bonazzi, R. Talk before it's too late: Reconsidering the role of conversation in information systems project management. *Journal of Management Information Systems*, 31, 1 (2014), 47–77.
52. Mathiassen, L., and Pourkomeylian, P. Managing knowledge in a software organization. *Journal of Knowledge Management*, *7*, 2 (2003), 63–80.
53. Mitchell, V.L. Knowledge integration and information technology project performance. *MIS Quarterly*, 30, 4 (2006), 919–939.
54. Moore, G.C., and Benbasat, I. Development of an instrument to measure the perceptions of adopting an information technology innovation. *Information Systems Research*, 2, 3 (1991), 192–222.
55. Nambisan, S. Information systems as a reference discipline for new product development. *MIS Quarterly*, 27 (2003), 1–18.
56. Nambisan, S., and Wilemon, D. Software development and new product development: Potentials for cross-domain knowledge sharing. *IEEE Transactions on Engineering Management*, 47, 2 (2000), 211–220.
57. NASSCOM-Hewitt Total Rewards Study. 2005. www.nasscom.in (accessed on April 30, 2008).
58. Nelson, K.M., and Cooprider, J.G. The contribution of shared knowledge to IS group performance. *MIS Quarterly*, 20, 4 (1996), 409–432.
59. Nidumolu, S. A comparison of the structural contingency and risk-based perspectives on coordination in software-development projects. *Journal of Management Information Systems*, 13, 2 (1996), 77–113.
60. Norman, P.M. Knowledge acquisition, knowledge loss, and satisfaction in high technology alliances. *Journal of Business Research*, *57*, 6 (2004), 610–619.
61. Okhuysen, G., and Eisenhardt, K. Integrating knowledge in groups: How formal interventions enable flexibility. *Organization Science*, 13, 4 (2002), 370–386.
62. Patnayakuni, R.; Rai, A.; and Tiwana, A. Systems development process improvement: A knowledge integration perspective. *IEEE Transactions on Engineering Management*, 54, 2 (2007), 286–300.
63. Paulk, M.; Curtis, B.; Chrissis, M.; and Weber, C. Capability Maturity Model, Version 1.1. *IEEE Software*, 10, 4 (July 1993), 18–27.
64. Pawlowski, S.D., and Robey, D. Bridging user organizations: Knowledge brokering and the work of information technology professionals. *MIS Quarterly*, 28, 4 (2004), 645–672.
65. Podsakoff, P.; MacKenzie, S.; Lee, J.; and Posakoff, N. Common method biases in behavioral research: A critical review of the literature and recommended remedies. *Journal of Applied Psychology*, 88, 5 (2003), 879–903.
66. Pressman, R. *Software Engineering: A Practitioner's Approach*. 9th ed. New York: McGraw-Hill, 2009.
67. Reich, B.H., and Benbasat, L.I. Factors that influence the social dimension of alignment between business and information technology objectives. *MIS Quarterly*, 24, 1 (2000), 81–113.

68. Richter, A.W.; West, M.A.; Dick, R.; and Dawson, J.F. Boundary spanners' identification, intergroup contact, and effective intergroup relations. *Academy of Management Journal*, *48*, 6 (2006), 1252–1269.
69. Rulke, D., and Galaskiewicz, J. Distribution of knowledge, group network structure, and group performance. *Management Science*, *46*, 5 (2000), 612–625.
70. Sabherwal, R. The evolution of coordination in outsourced software development projects: A comparison of client and vendor perspectives. *Information and Organization*, 13, 3 (2003), 153–202.
71. Sawyer, S.; Guinan, P.J.; and Cooprider, J. Social interactions of information systems development teams: A performance perspective. *Information Systems Journal*, 20 (2010), 81–107.
72. Siponen, M. and Vance, A. Neutralization: New insights into the problem of employee information systems security policy violations. *MIS Quarterly*, 34, 3 (2010), 487–502.
73. Smith, K.G.; Collins, C.J.; and Clark, K.D. Existing knowledge, knowledge creation capability, and the rate of new product introduction in high-technology firms. *Academy of Management Journal*, *48*, 2 (2005), 346–357.
74. Templeton, G.F.; Lewis, B.R.; and Snyder, C.A. Development of a measure for the organizational learning construct. *Journal of Management Information Systems*, *19*, 2 (2002), 175–218.
75. Titah, R. and Barki, H. Nonlinearities between attitude and subjective norms in information technology acceptance: A negative synergy? *MIS Quarterly*, 33, 4 (2009), 827–844.
76. Tiwana, A.; Bharadwaj, A.; and Sambamurthy, V. The antecedents of information systems development capability in firms: A knowledge integration perspective. In *Proceedings of the Twenty-Fourth International Conference on Information Systems*. AIS: Seattle, Washington, (2003), 21.
77. Tiwana, A., and McLean, E. Expertise integration and creativity in information systems development. *Journal of Management Information Systems*, *22*, 1 (2005), 13–43.
78. Tushman, M.L., and Katz, R. External communication and project performance: An investigation into the role of gatekeepers. *Management Science*, 26, 11 (1980), 1071–1085.
79. Walz, D.B.; Elam, J.J.; and Curtis, B. Inside a software design team: Knowledge acquisition, sharing, and integration. *Communications of the ACM*, 36, 10 (1993), 63–77.
80. White, W. A Heteroskedasticity-consistent covariance matrix estimator and a direct test for heteroskedasticity." *Econometrica*, 48, 4 (1980), 817–838.
81. Woltman, H.; Feldstain, A.; MacKay, J.C.; and Rocchi, M. An introduction to hierarchical linear modeling. *Tutorials in Quantitative Methods for Psychology*, *8*, 1 (2012), 52–69.
82. Yan, A., and Louis, M.R. The migration of organizational functions to the work unit level: Buffering, spanning, and bringing up boundaries. *Human Relations*, 52, 1 (1999), 25–47.
83. Zellmer-Bruhn, M. Interruptive events and team knowledge acquisition. *Management Science*, 49, 4 (2003), 514–528.
84. Zmud, R.W. Management of large software development efforts. *MIS Quarterly*, 4, 2 (1980), 45–55.

Appendix

Table A1. Scales and Items

Construct	Scale items (7-point scale: 1 = strongly disagree and 7 = strongly agree)	Adapted from
Knowledge integration	1. Team members combined their individual expertise to jointly solve project-related problems. 2. Team members combined their individual perspectives to develop shared project concepts. 3. Team members often gained new insights by sharing their ideas with each other. 4. Team members improved their task efficiency by sharing their knowledge with each other.	[64, 74, 83, 77]
Sentry processes	1. The team actively monitored information coming from external sources such as other teams, individuals, and departments. 2. The team actively decided what type of information to acquire from external sources such as other teams, individuals, and departments.	[2, 41]
Guard processes	1. The team avoided releasing internal information to others in the company. 2. The team actively controlled the use of its internal resources by others in the company. 3. The team shared its internal resources only in response to legitimate external requests.	[2]
Requirements uncertainty	1. Compared to other projects, requirements for this project fluctuated quite a bit. 2. Compared to other projects, a lot of effort was spent in reconciling the requirements for this project. 3. Compared to other projects, established procedures and practices could not be relied upon to generate requirement specifications.	[59]
Relational trust	1. The team was characterized by close personal relationships among members at multiple levels. 2. The team was characterized by high reciprocal behavior among members at multiple levels. 3. The team was characterized by mutual trust among members at multiple levels.	[77]
Knowledge heterogeneity	1. Members of this team vary widely in their areas of expertise. 2. Members of this team have a variety of different background and experiences. 3. Members of this team have wide-ranging skills and abilities.	[12]