MIS Quarterly

RESEARCH ARTICLE

# SOFTWARE PROCESS DIVERSITY: CONCEPTUALIZATION, MEASUREMENT, AND ANALYSIS OF IMPACT ON PROJECT PERFORMANCE[1]

**Narayan Ramasubbu**
Joseph M. Katz Graduate School of Business, University of Pittsburgh, 354 Mervis Hall,
Pittsburgh, PA 15260 U.S.A. {narayanr@pitt.edu}

**Anandhi Bharadwaj**
Goizueta Business School, Emory University, 1300 Clifton Road NE,
Atlanta, GA 30322 U.S.A. {abharad@emory.edu}

**Giri Kumar Tayi**
School of Business, State University of New York at Albany, 1400 Washington Avenue,
Albany, NY 12222 U.S.A. {g.tayi@albany.edu}

*This article investigates software process diversity, defined as the project condition arising out of the simultaneous use of multiple software development process frameworks within a single project. Software process diversity is conceptualized as the response of a project team to such contingencies as requirements volatility, design and technological novelty, customer involvement, and the level of organizational process compliance enforced on the project. Moreover, we conceptualize that the degree of fit (or match) between a project's software process diversity and the level of process compliance enforced on the project impacts overall project performance. This conceptualization was empirically tested by utilizing data collected from 410 large commercial software projects of a multinational firm. The results show that higher levels of requirements volatility, design and technological novelty, and customer involvement increased software process diversity within a project. However, software process diversity decreased relative to increases in the level of process compliance enforced on the project. A higher degree of fit between the process diversity and process compliance of a project, rather than the effects of those variables independently, was found to be significantly associated with a higher level of project performance, as measured in terms of project productivity and software quality. These results indicate that increasing software process diversity in response to project-level contingencies improves project performance only when there is a concomitant increase in organizational process compliance efforts. The implications of these results for research are discussed and prescriptive guidelines derived to manage the fit between process diversity and process compliance for improving software project performance.*

**Keywords**: Software process diversity, process compliance, plan-based processes, agile processes, software engineering, productivity, quality, fit as matching

[1]Arun Rai was the accepting senior editor for this paper. Likoebe Maruping served as the associate editor.

## Introduction

Pursuing firm-wide process standardization has traditionally been prescribed as a "best practice" for software organizations aiming to address the challenges of developing high-quality software in a cost-effective way (Harter et al. 2000; Humphrey 1989; Ramasubbu et al. 2008; Van der Pijl et al. 1997). Until recently, such standardization efforts have been typically implemented through a firm-wide adoption of a single plan-based or agile normative software process framework[2] (Krishnan and Kellner 1999; Lycett et al. 2003; Ramasubbu 2014). Accordingly, a majority of prior research on software development processes implicitly treats project-level processes as being uniformly derived from a single normative process framework (Agarwal and Chari 2007; Jiang et al. 2004; Krishnan et al. 2000). However, project-level processes in software development can be divisible, and project teams can customize their processes by selecting and combining elements from multiple plan-based and agile normative frameworks (Fitzgerald et al. 2006; Napier et al. 2008). Practitioner reports from the field also suggest that firms are increasingly adopting multiple process frameworks for their projects and embracing *software process diversity* (Anderson 2005; Bella *et al* 2008). Software process diversity refers to the condition of project-level processes being composed of elements drawn from *multiple* normative software process frameworks that could differ in their underlying attributes and philosophies (Deck 2002; Lindvall and Rus 2000).

There is growing recognition that software process diversity could aid teams in overcoming the limitations imposed by standardized processes based on a single normative process framework that advocates strict adherence to either plan-based processes or agile processes (Magdaleno et al. 2012; Napier et al. 2008; Vinekar et al. 2006). Past research studies have highlighted that the combination of plan-based and agile processes within a single project, and the resulting increase in software process diversity, could help software teams to better adapt to changing user requirements and design specifications (Harris et al. 2009; Ramesh et al. 2012). Capabilities gained from embracing process diversity could also help teams address the conflicting demands in a project such as the need to be efficient versus the need to be flexible (Subramanyam et al. 2012). By increasing process diversity, software teams that traditionally use plan-based frameworks could infuse flexibility into their context by adopting certain components of agile process frameworks such as accepting requirement changes and by submitting to more frequent testing (Boehm 2003; Ramasubbu and Balan 2009). Similarly, by embracing process diversity, agile software teams could incorporate some elements of structure and formal communication associated with plan-based process approaches to enhance efficiency and overall predictability of outcomes (Ramesh et al. 2012).

While examining the impacts of software process diversity on the performance of custom software development projects, it is important to recognize the presence of another countervailing force in the form of the organizational *process compliance* enforced in those projects. Stringent compliance to established process standards is considered to be an important firm capability in the custom software development industry (Ethiraj et al. 2005; Ramasubbu 2014). Software firms are known to voluntarily seek compliance verification and certification from third-party standards organizations in order to gain the reputation of "high process maturity" firms (Gopal and Gao 2009; Van der Pijl et al. 1997). These firms allocate significant resources to monitor project teams for their compliance to organizational process standards (McGarry and Decker 2002; Ramasubbu et al. 2008; Staples and Niazi 2008). Enforcing such organizational process compliance on project teams can be expected to influence both the levels of process diversity pursued by individual project teams and their subsequent performance. However, there is a dearth of research examining the impact of software process diversity on the performance of projects executed in an environment of stringent process compliance mandates. We address this gap through this study.

We raise the following research questions: *What factors contribute to an increase in software process diversity in real world projects that are also actively monitored for compliance to mandated process standards? What is the joint effect of software process diversity and process compliance on project performance?* To answer these research questions, we proceeded as follows. First, we conceptualized the software process diversity construct and developed its empirical formulization. Then, by theorizing software process diversity as a risk response mechanism to project-level contingencies, we developed a research model of the antecedents and performance implications of software process diversity. Subsequently, we partnered with a leading multinational software firm in order to conduct field observations of software projects and data collection. We tested and verified our conceptualization by utilizing archival data from 410 software projects completed at the research site. Finally, based on the results from the analysis and on follow-up discussions with

[2]Normative software process frameworks refer to a wide array of models (or methodologies) of software development published by standards organizations in the software industry, such as the Software Engineering Institute, Scrum.org,, and the International Organization for Standardization (ISO). These frameworks, such as the capability maturity model (CMM) and Scrum, provide templates and guides for firms to model their software development processes. Firms often seek certifications from the industry standards bodies to signal to the market that their software processes are compliant with the prescriptions of the adopted normative frameworks.

project managers from the field, we developed a set of prescriptive guidelines for effectively balancing process diversity and process compliance in software projects.

## A Conceptualization of Software Process Diversity

When software teams adopt and fuse multiple normative process frameworks within a single project, they increase the overall software process diversity of the project. While there have been anecdotal and case descriptions of software process diversity in prior research (e.g., Boehm 2003; Jakobsen and Jhonson 2008; Jakobsen and Sutherland 2009), a rigorous analysis of the concept of software process diversity, its relationship with organizational process compliance, and their joint impact on project performance is absent in the literature.

To address this gap, we draw on the broader management literature, which has focused on various forms of organizational diversity, such as demographic differences and differences in values, functional skills, and personality types. Harrison and Klein (2007) reviewed the literature on organizational diversity and provided guidelines for its conceptualization and measurement. They recommended conceptualizing diversity as an amalgamation of the differences in the beliefs and values of team members (diversity as *separation*); the differences in the knowledge, experience, and perspectives of team members (diversity as *variety*); and the differences in the influences and resources held by team members (diversity as *disparity*). To build on this, we theoretically define *software process diversity* as *the composition of differences in separation, variety, and disparity of key process areas implemented in a software project*.

A *key process area* (KPA) is a cluster of related activities (or tasks) in a project that, when performed collectively, achieves a set of goals for successful software development (Ramasubbu et al. 2008). Normative software process frameworks prescribed by industry-based standards organizations clearly specify the number and scope of KPAs that they support. For example, the CMMI framework for software development consists of a total of 22 KPAs covering the entire lifespan of a development project.[3] When a software organization adopts a normative process framework such as the CMMI, a *process template* covering the KPAs of the framework is instituted. This process template provides a predefined collection of artifacts that can be used to organize the workflow spanning the KPAs throughout a project's life cycle. In organizations that allow project teams to simultaneously use multiple normative frameworks, process templates corresponding to the different frameworks are made available to the software project team. Software teams typically activate specific components of each of those templates for use in their projects (Fitzgerald et al. 2006; Ramasubbu et al. 2008). Thus, in this scenario, multiple process templates collectively implement the entire set of KPAs of a project. Since the use of process templates by a software team is logged and traceable, longitudinal observations of the membership of a project's KPAs to the different process templates provide an empirical basis to reliably measure software process diversity and the underlying dimensions of separation, variety, and disparity of the KPAs implemented in a project.

The *separation* dimension of software process diversity measures the composition of differences among KPAs in a software project belonging to process templates as derived from a plan-based versus an agile normative process framework. This is analogous to assessing diversity in a project team by acknowledging the differences in knowledge bases, perspectives, and deeply held values and beliefs among the team members (Kilduff et al. 1995; Homan et al. 2007). Plan-based and agile process frameworks vary in their overall values, beliefs, and practice attitudes (Boehm 2002; Boehm and Turner 2003). For example, while plan-based frameworks such as the CMMI favor structured planning processes and comprehensive documentation for traceability, agile frameworks such as extreme programming tend to emphasize "individuals and interactions over tools, working software over documentation, customer collaboration over contract negotiation, and responding to change over following a plan" (Beck et al. 2001). Measuring the composition of KPAs belonging to the plan-based and agile process templates in a project captures the extent to which a project's KPAs are diverse in their process values and beliefs.

The *variety* dimension of software process diversity measures the composition of differences in the *spread* of KPAs across the different process templates used in a project. This accounts for differences among the varieties of process frameworks used in a project even when they are not separated in their overall philosophy (e.g., plan versus agile). For example, a software project might utilize templates from two plan-based frameworks such as the CMMI and rational unified process (RUP), and allocate to each 50 percent of the total project KPAs. Another project in the firm might utilize the same two process templates, but allocate 25 percent of its KPAs to the CMMI process template and 75 percent of its KPAs to the RUP template. The separation diversity score for

[3]For more details on the 22 KPAs of the CMMI framework for development, refer to the technical report, "CMMI for Development, Version 1.3," CMU/SEI-2010-TR-033, published by the Software Engineering Institute.

these two projects would be zero, as both CMMI and RUP impose a plan-based development paradigm and there is no separation of values in this case. However, the two projects still differ in the extent to which they allocate KPAs to the two varieties of process templates (CMMI and RUP). Thus, there is a need to capture differences among software projects with respect to the number of process templates that they use and the way they allocate KPAs to those different varieties of process templates. The variety dimension of software process diversity accomplishes this by capturing the spread of the KPAs among the different varieties of process templates used in a project.

The *disparity* dimension of software process diversity measures the composition of differences in the *resource allocation* to the KPAs belonging to the different varieties of process templates utilized in a project. Measuring disparity in resource allocation is important, because it captures the extent to which the diverse elements in a process were actually utilized in project activities. Resource allocation in software projects has been typically measured as the percentage of total project effort allocated to a particular activity (Krishnan et al. 2000; Ramasubbu et al. 2008). Accordingly, the disparity dimension of process diversity accounts for resource allocation differences by considering the differences in the project effort allocated to the different KPAs and the process templates with which they are associated.

In summary, rather than assuming that the KPAs of a project are homogenously distributed with respect to the underlying normative process framework(s), the construct of software process diversity allows us to account for the separation, variety, and disparity of software processes within a project. Next, we develop a research model of the antecedents and consequences of software process diversity, and formulate our key hypotheses.

## Relating Software Process Diversity, Process Compliance, and Project Performance

### Research Model

Keeping in mind the dual forces of software process diversity and process compliance in high process maturity environments, we began our model formulation by building on prior research findings of "controlled-flexible" process designs (Harris et al. 2009). Controlled-flexible process designs aim for both process improvisation and the establishment of disciplined management controls to enforce compliance. Using case studies, Harris et al. (2009) established evidence for the superior performance of controlled-flexible process designs when compared with pure plan-based designs. Those case studies revealed cross-project variations in the way controlled-flexible process designs were enacted by different project teams under the influence of technological and market uncertainties. Addressing how teams could adapt to such uncertainties, Vidgen and Wang (2009) studied software process design through the lens of complex adaptive systems and discussed the coevolution of software processes along with the business environment. In the software engineering literature, project-level contingencies such as requirements volatility, degree of customer involvement, and code size have been reported as influencing specific process design choices made by a software team (Boehm and Turner 2003; Magdaleno et al. 2012; Maruping et al. 2009). Finally, prior research has also established that high process maturity organizations, such as those assessed at CMM level-5 process maturity, enforce stringent process compliance mechanisms to ensure minimal deviance from prescribed standards (Ethiraj et al. 2005; Ramasubbu et al. 2008). Process compliance efforts tend to reduce process variations within a project, and they have been associated with the benefits of minimizing schedule deviation and lowering the number of defects in the delivered software (Harter et al. 2000; Harter et al. 2012; Krishnan and Kellner 1999).

The above research findings motivated our model development as follows: (1) depending on specific project-level contingencies, software teams make choices regarding the underlying normative process frameworks for their projects, which could result in increased levels of software process diversity within the projects; and (2) process diversity and process compliance efforts expended on a project jointly impact project performance outcomes. We build on these to develop specific hypotheses on the antecedents of within-project software process diversity and on the joint influence of organizational process compliance and process diversity on project performance.

As noted earlier, software process variations within a project reflect the choices made by the software team in response to specific contingencies encountered during the project life cycle. Prior research on software project risk has examined how specific project-level contingencies (or project risk factors) contribute to overall project risk and impact performance outcomes (Barki et al. 1993; Keil et al. 2000). Much of this work characterizes project risks as stemming from various types of uncertainties related to project requirements, end user involvement, underlying technological and design complexity, project team composition, and the overall organizational environment (Wallace et al. 2004). We draw on the risk

framework proposed by Wallace et al. (2004) to develop specific hypotheses that relate project risk factors to software process diversity. In doing so, we characterize the extent of process diversity observed in a project as the collective response of the project team to the perceived project risks through a key mechanism they control, namely, process design. However, as noted earlier, a countervailing choice exercised by managers in the form of process compliance enforcements attempts to rein in the within-project process diversity. Software process diversity is, therefore, the result of both *project-specific risk factors* and the extent of organizational *process compliance* enforced on the project.

In contrast to prior studies that have examined project risks in relation to overall project performance, we focus on the intermediate outcome of software process diversity, which in our conceptualization is a key conduit through which project risk factors impact project performance. Accordingly, among the six dimensions of project risk identified by Wallace et al.,[4] we develop hypotheses pertaining to requirements risk, design and technology risk, and end-user (customer) involvement risk. We account for the remaining risk factors in our empirical models through appropriate control variables, such as software project size, team size, and personnel experience.

The final component of our conceptualization is the joint impact of process diversity and process compliance on project performance. In hypothesizing this effect, we consider two project performance dimensions widely examined in the information systems development literature: productivity and quality (Harter et al. 2000; Krishnan et al. 2000; Ramasubbu et al. 2008). We utilize the "fit as matching" perspective (Venkatraman 1989) to characterize the fit (or match) between process diversity and process compliance and consider its impact on project performance. Conceptually, project performance improves with a good fit. That is, when a higher (lower) level of process diversity is matched with a higher (lower) level of investment in process compliance, there is an improvement in productivity and quality. Table 1 presents a summary of the key hypothesized constructs and Figure 1 shows our overall research model.

### Requirements Volatility

Requirements volatility refers to the extent to which the functional and nonfunctional requirements for a software system under development change during the project's life span (Agarwal and Chari 2007; Barry et al. 2006). Requirement changes and subsequent alterations of software code have been reported to create ripple effects of rework in a project (Harter et al. 2000; Mookerjee and Chiang 2002). Rework might result due to legitimate changes in a client's needs, but can also stem from ambiguous, inadequate, or incorrect articulation of requirements in the first place (Wallace et al. 2004).

Plan-based and agile processes vary in their approaches to handling requirements volatility, the underlying causes, and the subsequent ripple effects of rework. Plan-based approaches primarily work toward minimizing requirements volatility through a combination of formal contracts that evoke client commitments and partitioning mechanisms that freeze requirements over certain stages of the life cycle of a project. In contrast, agile processes tend to respond to requirements volatility through improvisation, rapid adaptation, and incremental delivery of software (Boehm 2002; Maruping et al. 2009). The rapid adaptation and social approach inherent in the agile process framework, however, have been noted as unscalable due to the inefficiencies of frequent informal meetings (Begel and Nagappan 2007; Dyba and Dingsoyr 2008). Thus, in the presence of requirements volatility, there are tradeoffs to be made when any single framework is used exclusively in a project.

One way to remove the constraints of such tradeoffs, then, is to adopt both plan-based and agile process frameworks within a project. Using multiple process frameworks would help project teams to better handle fluidity in requirements, as they could cordon off parts of the project with different levels of volatility and then tackle them through different process approaches. For example, teams could partition a project into different phases (or iterations) of high and low volatility in order to use different process frameworks for these phases (MacCormack and Verganti 2003). Teams could also partition the software into different modules according to the requirements volatility levels and then suitably apply different varieties of processes to them (MacCormack et al. 2001).

Using different varieties of process templates from both plan-based and agile frameworks creates *separation* in the overall philosophies toward software development within a project. Also, it has been observed that requirements volatility may not occur homogeneously throughout a project's life cycle, and it does not affect all parts of a system's development in a uniform way (Banker and Slaughter 2000; Barry et al. 2006). The heterogeneous patterns of requirements volatility, in turn, cause a disparity in effort expenditure on the different project phases, the parts of partitioned systems developed during those phases, and their associated process templates. Thus, an

[4]The six dimensions are organizational environment risk, user risk, requirements risk, project complexity risk, planning and control risk, and team risk.

**Table 1. Key Constructs in Conceptualization**

| Model Component | Construct | Description | Correspondence to Wallace et al. (2004) Risk Dimension |
|---|---|---|---|
| Antecedents of software process diversity | Requirements volatility | Uncertainty surrounding project requirements and frequent change in requirements | Requirements risk |
| | Design and technological novelty | The level of newness and unfamiliarity with the system design and the implementation technology | Project complexity risk |
| | Customer involvement | The level of involvement from key stakeholders of the client firm | User risk |
| | Organizational process compliance | The extent of effort spent on meeting process standards mandated by the firm | Planning and control risk |
| Performance implications of software process diversity | Fit (misfit) between process diversity and process compliance | The degree of congruence (deviance) between the extent of process diversity and efforts spent on process compliance activities in a project | NA |
| | Productivity | Overall efficiency in delivering the project (output/input) | Project performance |
| | Quality | Overall quality of delivered software (1/defects delivered) | Project performance |

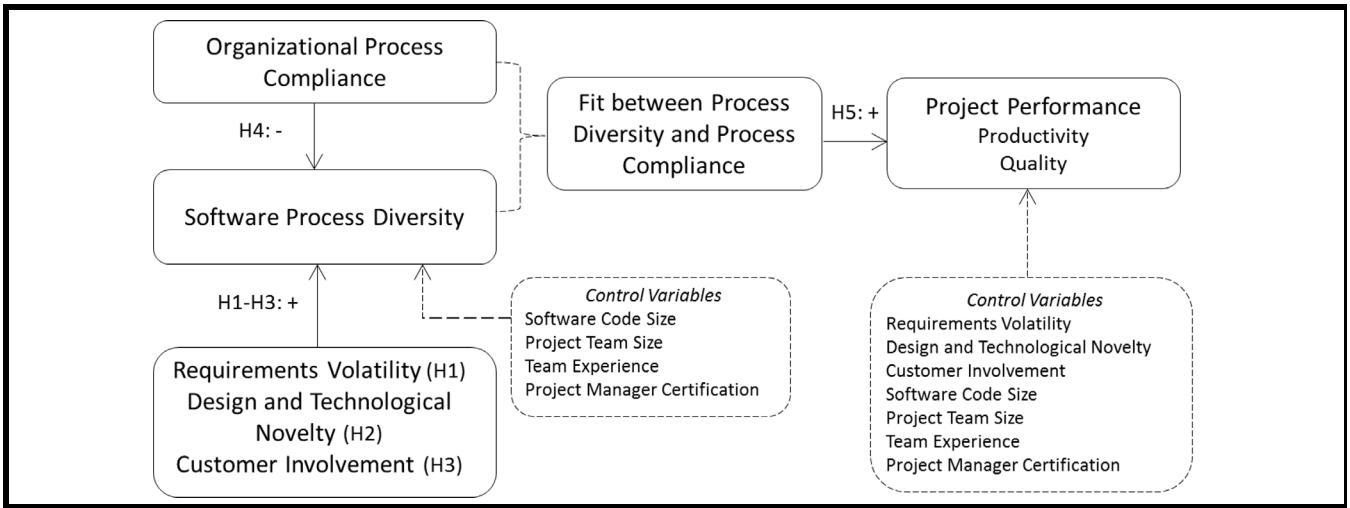


**Figure 1. Research Model**

increase in the requirements volatility of a project tends to increase the variety, separation, and disparity of processes within the project, which leads to an overall increase in software process diversity. Therefore, our first hypothesis is

*H1. A higher level of requirements volatility in a project is associated with a higher level of software process diversity.*

### *Design and Technology Novelty*

Project teams facing new software design and technologies face a steep learning curve during the development process (Boh et al. 2007; Kemerer 1992). Initially, such teams might employ agile processes with rapid iterations to gain early feedback and accelerate team learning (MacCormack et al. 2001). However, over time, as the new technological features

become more familiar and tasks get predictable, the focus can shift to a plan-based approach that facilitates enhanced efficiency and benefits due to the scale economies of longer iterations (Banker and Slaughter 1997). Furthermore, the use of novel design and technologies in a software project could be a significant contributor to project complexity, resulting in higher levels of project management risk (Wallace et al. 2004). Normative process frameworks that prescribe an exclusive plan-based or agile paradigm advocate different approaches for mitigating such risks. Plan-based frameworks often prescribe explicit risk management goals and practices. For example, in the CMMI framework, risk management is a level-3 KPA that prescribes specific steps for preparation, identification, analysis, and mitigation of risks (Chrissis et al. 2011). In contrast, agile process frameworks tend not to formally prescribe explicit templates for risk management, leaving it open for resolution by project teams (Anderson 2005; Moran 2014).

Higher levels of design and technological novelty present a demanding requirement on software teams, who are tasked with accommodating quick iterations to both facilitate effective learning and implement a planned approach to manage risks. Teams facing such challenging requirements resort to combining the attributes of multiple varieties of process frameworks, as it helps them to develop ambidextrous capabilities (Magdaleno et al. 2012; Napier et al. 2008; Ramesh et al. 2012). In such a scenario, the variety of process frameworks chosen by a project team would include a number of agile-oriented process components, because rapid iterations facilitated by agile processes aid efficient learning and effective client feedback during early project stages (MacCormack et al. 2001). Inclusion of agile-oriented process components in high process maturity environments, where the use of structured and plan-based approaches is the organizational norm, naturally leads to a *separation* of values and beliefs within a project.

Furthermore, learning curve effects resulting from higher levels of design and technological novelty cause variance in the effort expended on project tasks over the life cycle of the project (Kemerer 1992). As teams learn and become familiar with new designs and technologies, the effort required to complete project tasks tends to decrease, although the patterns of such a decrease may not be uniform during the different project stages (Boh et al. 2007). This variance in effort expenditure corresponds with a variance associated with the use of different varieties of process templates during the early and late project stages. That is, there is *disparity* in the allocation of resources to the different varieties of process templates utilized during the various project life cycle stages. Thus, an increase in design and in technological novelty increases the variety, separation, and disparity of processes within a project and therefore, our second hypothesis is

*H2. A higher level of design and technological novelty in a project is associated with a higher level of software process diversity.*

### Customer Involvement

There are divergent findings in prior research about the level of customer (end user) involvement in a project and its influence on a project's processes and performance outcomes. Wallace et al. (2004) categorize the lack of user involvement in a project as a significant project risk and note that such risks could contribute to project failures. Exploring customer-developer links in software projects, Keil and Carmel (1995) show that projects with a higher degree of participation from customers tend to perform better. Such project success has been attributed to the ability of software teams to match their processes to customer needs through appropriate changes to development processes (MacCormack and Verganti 2003). On the one hand, when there is a high degree of customer participation in a project, team members might be expected to more actively respond to evolving customer requirements (Maruping et al. 2009; Matook and Maruping 2014; Subramanyam et al. 2010). On the other hand, a higher degree of customer participation might also necessitate the need for more formal communication plans, especially when there are organizational boundaries between the customer and the vendor teams (Pikkarainen et al. 2008; Ramesh et al. 2012).

To respond to such dual needs in managing customer involvement in a project, software teams could potentially draw appropriate process components from multiple normative frameworks for their projects as customer engagement approaches in plan-based and agile approaches vary significantly. While the plan-based approaches opt for formal and punctuated interactions, the agile approaches prefer continuous, informal, and frequent interactions (Matook and Maruping 2014). Furthermore, as highlighted in prior research, a higher degree of customer involvement positively influences a project team's orientation toward selecting more agile process components (Matook and Maruping 2014; Ramasubbu and Balan 2009). Naturally, in the context of custom software development executed by high process maturity firms, this leads to a situation in which a software team must grapple with *separate* value systems enshrined in the different *varieties* of process templates used in the project.

Moreover, the degree of client involvement in a project may not be uniform throughout the different phases of a software project (Subramanyam et al. 2010). There are also variations in the expectations of different end users of a customer firm with respect to the formalism required in project-level communications and the corresponding coordination mech-

anisms (Andres and Zmud 2002; Nidumolu 1995). These variations result in overall heterogeneity in effort expenditures across formal (documents, contracts, etc.) and informal (mutual adjustments, informal interaction, etc.) coordination schemes. In turn, these would cause a corresponding *disparity* in the effort allocation to the different KPAs belonging to the plan-based and agile process templates that respectively facilitate formal and informal coordination. Thus, we expect the degree of customer involvement to positively influence the variety, separation, and disparity of the processes of a project team. Therefore, our third hypothesis is

*H3. A higher level of customer involvement in a software project is associated with a higher level of software process diversity.*

### Process Compliance

Process compliance mechanisms are procedures enacted to ensure the fidelity of the software design processes to the selected process frameworks (such as the adherence of the team to the normative prescriptions of the CMMI framework). Process compliance is typically enforced by quality control groups, who are charged with the responsibility of auditing and evaluating the adherence to quality metrics including the normative prescriptions for process design and documentation (Ogasawara et al. 2006; Ramasubbu 2014). In high process maturity environments, project teams are expected to report all process tailoring activities, and the teams are monitored and evaluated through internal audits and related quality assurance activities. Thus, process compliance efforts contribute to an indirect selection mechanism wherein a deliberate investment is made to monitor the project-level processes and assess their fit with the overall project and firm context. Compliance efforts act as a stabilizing force that tends to limit process diversity by blocking process variations that are deemed unfit, unnecessary, or noncompliant with firm standards, as evaluated by the independent quality assurance personnel.

Apart from pruning project-level process variations, higher levels of process compliance enforcement can also provide learning opportunities for reconciling disparate values among project members (Ramasubbu et al. 2008). This can result in project teams utilizing more cohesive process components that are not too distinct in their underlying value attributes. Also, with a higher level of process compliance, process components that do not receive significant levels of resource commitments from the project team could be identified as potentially less useful variations that can be eliminated without significantly impacting project outcomes. Thus, we expect process compliance to act as a countervailing force to software process diversity by reducing the variety, separation, and disparity of processes in a software project. Hence, we hypothesize

*H4: A higher level of process compliance effort in a software project is associated with a lower level of software process diversity.*

### Fit Between Process Diversity and Process Compliance

The discussion up to this point has focused on examining the project-level contingencies that increase software process diversity and the role of process compliance in reining in process diversity. In this section, we examine the fit (or match) between process diversity and process compliance, and examine their joint effects on overall project performance. Following well-established protocols in prior literature for the measurement of software project performance (e.g., Harter et al 2000; Krishnan et al 2000; Ramasubbu et al. 2008), we conceptualize project performance along the two dimensions of the *development productivity* of a project team and the *conformance quality* in the software code delivered by the project team.

Development productivity refers to the efficiency of a project team in developing and delivering software code by using resources available to the team (Krishnan et al. 2000). When lower levels of process diversity in a project are matched with a high degree of process compliance investments, the result is suboptimal use of organizational resources. The excessive compliance efforts in the above scenario could have otherwise been used for core project tasks such as software development and verification. Similarly, when a project team is subjected to a lower level of investments in organizational process compliance, but operates using a higher level of process diversity, there is a mismatch and a negative impact on productivity. This is because insufficient process compliance efforts often lead to the propagation of suboptimal or bad practices that cause rework during the late stages of a project (Stamelos 2010). The costs of such rework are difficult to constrain because of complex dependencies that cause ripple effects through all parts of the software system that is under development (Mookerjee and Chiang 2002). Such additional expenditures due to rework ripples dent the overall development productivity of a project. Similarly, when higher levels of process diversity are met with lower levels of process compliance efforts, there are insufficient resources available to detect software errors committed during development (i.e., lower levels of conformance quality). Such undetected development errors are likely to be reported by customers after the delivery of software code; for example, during acceptance

testing. Rectifying software errors after delivery is challenging and leads to an overall increase in project rework and cost of quality (Harter et al. 2000).

In contrast, a good fit between process diversity and process compliance enables a project team to optimally balance the increased flexibility resulting from the use of multiple process frameworks and the rigor needed to detect and rectify software development errors before delivery. Such an optimal balance has been reported to engender process-based learning in software development that helps teams to realize improvements in productivity and reduce the number of defects delivered to customers (Ramasubbu et al. 2008; Subramanyam et al. 2012). A good fit between process diversity and process compliance also leads to the development of coevolutionary change mechanisms that help project teams adapt to uncertain business situations in an efficient way (Vidgen and Wang 2009; Volberda and Lewin 2003). Thus, a better fit between process diversity and process compliance endows project teams with capabilities that help them cope with project contingencies in an efficient way and reduce defects (or improve quality) through improved learning and effective quality control mechanisms. Therefore, our final set of hypotheses is

*H5a:* *A higher degree of fit between process diversity and process compliance in a software project is associated with a higher level of project productivity.*

*H5b:* *A higher degree of fit between process diversity and process compliance in a software project is associated with a higher level of software quality.*

## Analysis and Results

### Data Collection

To collect data for this study, we collaborated with a leading multinational software development firm that was a recipient of the IEEE Software Process Achievement Award (IEEE SPA 2010) and had an industry-wide reputation as a leader in process innovation. The firm operated in 55 countries with over 100,000 employees and about $8 billion in annual revenues at the time of our data collection. All the development centers from which we collected data for this study were assessed as operating at CMM level-5 process maturity by an independent auditing agency. The high process maturity environment at the firm was attractive for our study, as it would facilitate our collection of reliable and fine-grained data on project-level software processes.

We first engaged with the firm to assess the suitability of its business context, processes, and overall environment for our study. We conducted 27 focus group meetings and 28 structured interviews that involved 109 managers and software project personnel. These discussions and interviews involved participants who described their team's development environment, process design rationale, and lessons learned from their experiences. Through these interactions, we were able to confirm the presence of within-project process design variations involving both plan-based and agile-process approaches at the firm. Practitioners also highlighted the influential role played by the firm's software engineering process group (SEPG), which was an autonomous organizational unit that conducted rigorous internal audits on project teams to enforce compliance to the firm's process standards. Those operational characteristics of the firm made it an ideal setting for testing the research model of this study, and we proceeded to the second stage of our data collection.

In the second stage, we collected archival data from 410 commercial software projects completed by the firm in a recent two-year period. The archival data were collated from multiple databases at the firm, including those maintained by the SEPG, individual project managers, and the human resources division of the firm. The data set utilized in the study was independently audited multiple times by the firm's external auditors for quality certification purposes, including the annual CMMI level-5 compliance checks and the nomination for the IEEE SPI award. Thus, we were able to utilize a unique and highly reliable data set for our analysis.

### Variables in Dataset

#### Process Diversity

As noted earlier, we consider a KPA as the fundamental unit of a software process design. At our research site, a majority of the software development centers used the 22 KPAs prescribed for CMMI level-5[5] development operations. Other normative process frameworks to which project teams subscribed during the time period of this study were IBM's Rational Unified Process (RUP), Agile RUP, Scrum, and Extreme Programming (XP). The baseline process templates for all of these five normative process frameworks were standardized and controlled by the firm's centralized SEPG. Individual projects were allowed to implement all of or a subset of the five process templates for their specific use.

[5]CMMI is an integrated process improvement framework developed by the Software Engineering Institute at Carnegie Mellon University. Level-5 of the CMMI is the highest maturity level, indicating a quantitatively controlled and well-optimized process.

Therefore, software teams had the flexibility to both choose a set of relevant process templates for their project and to design the spread of KPAs across the process templates, which resulted in project-specific customizations of plan-based and agile frameworks. To be able to monitor and control such customized processes through uniform metrics, the firm's SEPG teams mapped the KPAs of the normative process frameworks under a seamless and uniform process compliance framework.[6] The project-level process customizations and the spread of KPAs to the different process templates formed the basis for the measurement of the process diversity construct. For each of the KPAs implemented in a project, we observed the membership of the KPA to the corresponding process template and traced its use throughout the project's life cycle. Then, we calculated the separation, variety, and disparity process diversity scores for each project.

The separation dimension was derived using a standard deviation-based measurement scheme. By assigning an agile score for each KPA of a project (1 if the KPA belongs to an agile process template, 0 otherwise), the separation dimension of process diversity is derived as

$$Separation = \sqrt{\sum_{i=1}^{s} \frac{(agilescore_i - agilescore_{mean})^2}{s}}$$

where a project uses $s$ KPAs belonging to process templates derived from either a plan-based normative framework (in which case the agile score = 0) or an agile-based process framework (agile score = 1). For example, consider a scenario in which three different projects each use 22 KPAs for software development; project 1 uses only CMMI; project 2 implements 12 KPAs using CMMI and 10 KPAs using RUP; and project 3 uses CMMI for 12 KPAs and SCRUM for 10 KPAs. The separation diversity score for both project 1 and project 2 would be zero, as there is no separation among the KPAs in these projects (they all belong to plan-based process frameworks, either CMMI or RUP). In contrast, for project 3, the separation score is 0.498,[7] indicating the presence of a plan-based (CMMI) and agile (Scrum) process separation among the KPAs of the project.

The variety dimension of diversity has been traditionally measured using the Blau's index (Gibbs and Martin 1962; Harrison and Klein 2007), which we adapted to the context of software process diversity as follows:

$$Variety = 1 - \sum_{i=0}^{v} p_i^2$$

where, $p_i$ would be the proportion of the $s$ KPAs that belong to the $v$ variety of process templates. Consider the previous example scenario in which three different projects each use 22 KPAs for software development: project 1 uses only CMMI; project 2 implements 12 KPAs using CMMI and 10 KPAs using RUP; and project 3 uses CMMI for 12 KPAs and Scrum for 10 KPAs. As explained above, the separation diversity scores for projects 1, 2, and 3 would be 0, 0, and 0.498, respectively, which captures the separation of KPAs into plan-based and agile categories. On the other hand, the variety diversity scores for projects1, 2, and 3 would be 0, 0.496, and 0.496, respectively,[8] which captures the different patterns of the spread of KPAs across different process templates, but unlike the separation diversity score, it does not explicitly capture the separation of plan-based KPAs and agile KPAs.

Disparity dimension of diversity has been typically measured using a coefficient of variation (Harrison and Klein 2007), which we adapted to the context of this study as follows:

$$Disparity = \frac{\sqrt{\sum_{i=1}^{v} \frac{(effort_i - effort_{mean})^2}{v}}}{effort_{mean}}$$

where $effort_i$ is the percentage of total project effort spent on KPAs belonging to a $v$ variety of process templates. Expanding the previous example scenario where each of the three different projects used 22 KPAs for software development, let us say project 1 uses only CMMI and allocates 100 percent of its effort to it; project 2 implements 12 KPAs using CMMI and 10 KPAs using RUP, and allocates 65 percent of project effort to CMMI KPAs and the remaining 35 percent to RUP KPAs; and project 3 uses CMMI for 12 KPAs and Scrum for 10 KPAs, and allocates 70 percent of project effort to the CMMI KPAs and the remaining 30 percent to the RUP KPAs. As discussed before, the separation diversity scores for projects 1–3 are 0, 0, and 0.498, respectively; the corresponding variety diversity scores are 0, 0.496, and 0.496. The disparity scores for projects 1–3 are 0, 0.424, and 0.566, respectively,[9] which capture the differences in the effort allocation between the different varieties of process frameworks used in the projects.

[6] A generalized example of such a mapping of CMM KPAs and Extreme Programming framework can be found in Paulk (2001).

[7] The mean agile score for project 3 is 0.455 and the overall separation score is the standard deviation of the agile scores (10 counts of 1 and 12 counts of 0, yielding a standard deviation of 0.498).

[8] Project 1 variety score = (1 – 1) = 0); project 2 variety score = $1 - (12/22)^2 - (10/22)^2 = 0.496$; project 3 variety score = $1 - (12/22)^2 - (10/22)^2 = 0.496$.

[9] Project 1 disparity score = $\sqrt{((100 - 100)^2/1)/100} = 0$; project 2 disparity score = $\sqrt{(((65 - 50)^2+(35 - 50)^2)/2)/50} = 0.424$; project 3 disparity score = $\sqrt{(((70 - 50)^2 + (30 - 50)^2)/2)/50} = 0.566$.

### Process Compliance

Process compliance efforts in a project involve the monitoring, assessment, and auditing of process engineering activities in a project. The process compliance personnel at our research site reported to both the project manager of a software team and the compliance supervisor at the SEPG. Although compliance personnel had sufficient autonomy within a software project team, executing process compliance activities required them to collaborate with individual developers. The extent of resources allocated to process compliance varied across the projects and we observed a corresponding variance in the extent to which the process compliance personnel were able to systematically monitor and assess process variants across projects. We had access to the task-level accounting and work log of all SEPG personnel participating in a software project, and we were able to identify and trace the overall process compliance effort expended on a project. Similar to prior research in which investments in quality management processes were measured as a percentage of total project effort (Ramasubbu et al. 2008), we derived the process compliance variable as the percentage of total project effort spent on process compliance activities.

### Fit Between Process Diversity and Process Compliance

To derive the fit variable, we follow the "fit as matching" perspective developed in prior literature (see Tang and Rai 2014; Venkatraman 1989). The fit as matching technique enables us to derive the match between process diversity and process compliance without reference to an external criterion variable (such as project performance). The criterion-independent matching perspective is particularly apt for our study because process diversity and process compliance efforts are primarily influenced by different and independent actors in a project who have different goals and incentives. While process diversity is driven by project-level contingencies faced by software teams, process compliance efforts are driven by standards mandated at the organizational level that are enforced by central SEPG agents.

We utilized the deviation score and residual score approaches to measure the match between process diversity and process compliance (see Venkatraman 1989). In the deviation score approach, misfit (or mismatch) is derived as the absolute difference between the standardized scores of process diversity and process compliance variables. Inversing the sign of the misfit variable obtained from the deviation score approach reflects fit, and it is used for easier interpretation of the regression coefficients. In the residual score approach, we derive the fit variable using the residuals from the regression of process compliance on process diversity. To verify the robustness of results, we also used the residuals from the regression of process diversity on process compliance to derive the fit variable as recommended in the literature (see Dewar and Werbel 1979).

### Project Performance

The dependent construct in our model, project performance, is measured using two variables: software productivity and quality. To derive the productivity variable, we first measured project size using function points, which is a programming language-independent measure of software functionality (Kemerer 1993). Then, we calculated productivity as an efficiency measure by treating the total function points produced by a software team as the output of a project and the total project effort as the input. Following methods applied in prior research, we utilized the log-transformed productivity variable for the regression estimations (Harter et al. 2000; Krishnan et al. 2000; Ramasubbu et al. 2008). We derived the second performance measure, software quality, as the inverse of the total number of unique software defects reported by customers (Subramanyam and Krishnan 2003).

### Requirements Volatility

This variable measures the extent to which requirements changes from customers induced rework in a software project. We calculated requirements volatility as the percentage of effort spent on rework due to change in customer requirements, and we held the planned development effort before the change occurred as the baseline.

### Design and Technology Novelty

The extent to which the software design and technology involved in a project was new to the project team was self-reported by project managers and programmers using a questionnaire adapted from Takeishi (2002). We sought at least two survey responses from each project team: one from a manager and the other from a technical team member. The survey asked participants to rate the extent to which their project involved "design newness" and their familiarity with the technology used to implement the designs. The responses were measured using a 0 to 100 point scale, ranging from old design (0%), some modifications to old design (less than 30%), to radically different from prior designs (100%). Similarly, technological familiarity was measured using scales ranging from being completely familiar to old technologies used (0%) to dealing with a completely new and unfamiliar

technology (100%). The responses to the design newness and technology familiarity items were highly correlated (0.94), and we took an average of those item scores to calculate the value of design and technology novelty in a project.

### Customer Involvement

We calculated the extent of customer involvement in a project as the percentage of total project effort spent by a software team in engaging with end users. We systematically parsed the daily time logs submitted by software project members that were utilized for billing and cost accounting purposes of the firm in order to identify the time allocation of individual team members for client-facing activities. Then, we aggregated the total time spent on these client-facing activities at the project level. Next, we identified and accounted for the presence of clients in team meetings, peer reviews, and quality inspection meetings that were not specifically reported as client-facing activities in the time logs by matching the meeting participant logs with the list of project personnel that we obtained from project managers.

### Control Variables

As mentioned before, we used software code size measured using function points to derive the productivity variable. Similar to methods used in prior research, we used log-transformed code size in the regression analysis as a control variable (Harter et al. 2000; Krishnan et al. 2000). We utilize the full-time equivalent count of personnel involved in a project as the team-size control variable to account for the extent of the coordination and administration needs of the project. Since the cumulative experience of a project team could potentially influence the overall project performance (Boh et al. 2007), we included the average professional work experience of the project team (in years) in the regression models. Finally, some project managers at our research site had obtained professional certifications on specific normative process frameworks (e.g., certified Scrum Master). To account for this difference among the managers, we included a categorical control variable in the models, coded as 1 if a project manager possessed any professional certifications and as 0 otherwise.

## Estimation and Results

The functional forms of the various relationships embedded in the research model (Figure 1) are shown in equations 1–3. Equation 1 shows that process diversity within a project is a function of project-specific conditions and process compliance. Equations 2 and 3 indicate that, after controlling for other project variables, the project performance outcomes, productivity, and quality, are a function of fit between process diversity and process compliance. To establish the robustness and validity of the results using the fit as matching perspective, original process diversity and process compliance variables are also included in equations 2 and 3 apart from the match variable derived from these individual components. We also included the absolute level of agile process KPAs used in a project in equations 2 and 3 to control for an alternate explanation that process agility might drive project performance.[10]

$$\text{Process diversity} = \beta_0 + \beta_1*(\text{process compliance}) + \beta_2*(\text{requirements volatility}) + \beta_3*(\text{design and technology novelty}) + \beta_4*(\text{customer involvement}) + \beta_5*(\text{team size}) + \beta_6*(\text{team experience}) + \beta_7*(\text{code size}) + \beta_8*(\text{project manager certification}) + \varepsilon \quad (1)$$

$$\text{Productivity} = \alpha_0 + \alpha_1*(\text{fit between process diversity and process compliance}) + \alpha_2*(\text{process diversity}) + \alpha_3*(\text{process compliance}) + \alpha_4*(\text{agile KPA score}) + \alpha_5*(\text{requirements volatility}) + \alpha_6*(\text{design and technology novelty}) + \alpha_7*(\text{customer involvement}) + \alpha_8*(\text{team size}) + \alpha_9*(\text{team experience}) + \alpha_{10}*(\text{code size}) + \alpha_{11}*(\text{project manager certification}) + \mu \quad (2)$$

$$\text{Quality} = \Upsilon_0 + \Upsilon_1*(\text{fit between process diversity and process compliance}) + \Upsilon_2*(\text{process diversity}) + \Upsilon_3*(\text{process compliance}) + \Upsilon_4*(\text{agile KPA score}) + \Upsilon_5*(\text{requirements volatility}) + \Upsilon_6*(\text{design and technology novelty}) + \Upsilon_7*(\text{customer involvement}) + \Upsilon_8*(\text{team size}) + \Upsilon_9*(\text{team experience}) + \Upsilon_{10}*(\text{code size}) + \Upsilon_{11}*(\text{project manager certification}) + \zeta \quad (3)$$

Before estimating the regression equations, we examined the empirical relationship between the three process diversity scores, namely, separation, variety, and disparity scores. Pairwise correlation between the three scores (shown in Table 2) revealed a high degree of correlation (at $p < 0.01$) between them.

One reason for such a high correlation between the different diversity scores could be because all of the projects at our research site during our observation period existed in a CMMI level-5 organizational environment, and they moved toward higher levels of process diversity from that baseline. So project-level variations that caused process variety also simultaneously caused a separation of KPAs into plan-based and

[10]We thank an anonymous reviewer for this suggestion. We measured the agile KPA score control variable as the percentage of total KPAs of a project that were implemented using agile process frameworks.

**Table 2. Correlation Between Variety, Separation, and Disparity Scores**

| Diversity Score | | 1 | 2 | 3 |
|---|---|---|---|---|
| Separation | 1 | 1.00 | | |
| Variety | 2 | 0.79 | 1.00 | |
| Disparity | 3 | 0.78 | 0.80 | 1.00 |

agile frameworks in a project. Further, the effort allocation across the different varieties of process templates followed the same pattern as the proportion of KPAs allocated to the different varieties of process templates in the project. Therefore, we are not able to sufficiently tease out the differences between the separation, variety, and disparity dimensions of software process diversity using our empirical data. As a result, to avoid multicollinearity issues, we only infer our results using each of the separation, variety, and disparity scores individually and mutually exclusively in the regressions. Results were similar across the models where the separation, variety, and disparity scores were separately used as the process diversity variable. Also, the process diversity variable calculated as a linear combination (additive, average, etc.) of the separation, variety, and disparity scores yielded similar results.[11]

The summary statistics and correlations between the variables utilized in the regression models are presented in Tables 3 and 4, respectively. We note that the fit variables derived using the deviation score and the residual score approaches are highly correlated with each other (0.83) as expected. However, the fit variables have relatively lower levels of correlation with the individual components that are used to derive them (process diversity and process compliance), which indicates a good level of discriminant validity for them.

Given that our model specification contained three linear equations (equations 1–3) with potential contemporaneous cross-equation error correlations, we use seemingly unrelated regression (SUR) to estimate the model (Zellner 1962). We tested for endogeneity in the system of equations using the Durbin–Wu–Hausman test (Davidson and MacKinnon 1993, pp. 237-242).

In all cases, the Durbin–Wu–Hausman test indicated that the null hypotheses (variables are exogenous) could not be rejected, which indicates that the SUR regression results are consistent.[12] As an additional robustness check, we also performed a two-stage least squares instrumental variable regression to test the relationship among software process diversity, productivity, and quality.[13] The two-stage least squares regressions yielded results consistent with SUR, lending support for the use of the SUR results for hypotheses testing.

Our results pertaining to the antecedents of process diversity, and which were utilized to verify hypotheses 1–4, are presented in Table 5. In Table 6, we present results pertaining to two project performance variables, productivity and quality, which we estimated using two different ways of measuring fit between process diversity and process compliance. We see that all models are statistically significant at the $p < 1\%$ level and the adjusted R-squared values are indicative of good explanatory power. Post-regression diagnostics used to detect outliers from Cook's distance metric and leverage plots did not reveal any problems. The highest variance inflation factor among all the models was 2.6, and the highest condition index was 16.23, which confirms that multicollinearity issues are of no concern (Belsley et al. 2004, p. 105).

Referring to Table 5, we see that requirements volatility, design novelty, and customer involvement all had a positive and statistically significant effect on process diversity. Confirming hypothesis 1, the results indicate that a percentage increase in requirements volatility leads to about a 13 percentage-point increase in process diversity ($\beta = 0.129$; $p < 0.001$). In a validation of hypothesis 2, we see that a unit increase in the design and technology novelty score increases process diversity by about 31 percentage points ($\beta = 0.309$; $p < 0.001$). As predicted by hypothesis 3, a percentage increase in customer involvement in the project increased process diversity in the project by about 6 percentage points ($\beta = 0.056$; $p < 0.001$). Hypothesis 4 posited that an increase in

[11]We thank the review team for suggesting the different ways of deriving a process diversity variable to be used in the regression models.

[12]Durbin–Wu–Hausman test statistics for the productivity and quality equations were, respectively, $F(1, 399) = 0.63$, $p > F = 0.427$; $F(1, 399) = 1.02$; $p > F = 0.312$.

[13]We utilized instrumental variables used in prior studies—namely, defect density of in-process defects and the ratio of upfront project investments—in order to uniquely identify the two-stage least square regression models (Harter et al. 2000; Krishnan et al. 2000).

**Table 3. Summary Statistics (*N* = 410)**

| Variable | Mean | Std. Dev. | Min | Max |
|---|---|---|---|---|
| Productivity | 0.16 | 0.26 | 0.01 | 3.37 |
| Quality | 0.02 | 0.03 | $0.71 \times 10^{-3}$ | 0.28 |
| Process diversity† | 0.28 | 0.26 | 0.00 | 0.91 |
| Process compliance | 16.48 | 11.17 | 1.01 | 42.35 |
| Fit between process diversity and process compliance (deviation score) | -1.12 | 0.87 | -3.12 | -0.01 |
| Fit between process diversity and process compliance (residual score) | 0.03 | 1.17 | -2.15 | 2.53 |
| Agile KPAs score | 27.14 | 24.67 | 2.27 | 41.54 |
| Requirements volatility | 15.58 | 22.27 | 0.00 | 275.05 |
| Design and technology novelty | 35.00 | 48.00 | 0.00 | 100.00 |
| Customer involvement | 16.75 | 11.31 | 0.00 | 100.00 |
| Team size | 10.86 | 10.48 | 2.00 | 116.00 |
| Team experience | 3.81 | 2.26 | 0.19 | 21.86 |
| Code size | 1,467.34 | 2,645.18 | 25.94 | 32,767.00 |
| Project manager certification | 0.45 | 0.50 | 0.00 | 1.00 |

Notes: †Disparity process diversity score used in regression models; results are similar for the use of separation and variety process diversity scores.

**Table 4. Correlations (*N* = 410)**

| Variable | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Productivity | 1 | 1.00 | | | | | | | | | | | | | |
| Quality | 2 | 0.18 | 1.00 | | | | | | | | | | | | |
| Process diversity† | 3 | 0.16 | 0.13 | 1.00 | | | | | | | | | | | |
| Process compliance | 4 | -0.11 | -0.10 | -0.33 | 1.00 | | | | | | | | | | |
| Fit (deviation) | 5 | 0.09 | 0.08 | -0.32 | 0.41 | 1.00 | | | | | | | | | |
| Fit (residual) | 6 | 0.15 | 0.10 | -0.07 | 0.05 | 0.83 | 1.00 | | | | | | | | |
| Agile KPA score | 7 | -0.25 | 0.16 | 0.38 | -0.18 | -0.31 | -0.25 | 1.00 | | | | | | | |
| Requirements volatility | 8 | -0.03 | 0.04 | 0.24 | -0.11 | -0.27 | -0.22 | 0.37 | 1.00 | | | | | | |
| Design and technology novelty | 9 | -0.10 | -0.13 | 0.54 | -0.27 | -0.54 | -0.51 | 0.25 | 0.04 | 1.00 | | | | | |
| Customer involvement | 10 | -0.02 | -0.10 | 0.55 | -0.12 | -0.41 | -0.49 | 0.52 | 0.40 | 0.02 | 1.00 | | | | |
| Team size | 11 | -0.06 | -0.31 | -0.11 | -0.07 | -0.01 | -0.15 | -0.23 | 0.01 | 0.18 | -0.18 | 1.00 | | | |
| Team experience | 12 | -0.02 | 0.04 | -0.01 | 0.03 | 0.06 | 0.01 | -0.07 | 0.13 | 0.19 | 0.17 | 0.03 | 1.00 | | |
| Code size | 13 | 0.47 | -0.17 | 0.15 | -0.10 | -0.07 | -0.02 | 0.21 | 0.01 | 0.17 | -0.04 | 0.29 | 0.01 | 1.00 | |
| Project manager certification | 14 | 0.03 | 0.05 | 0.21 | -0.12 | 0.06 | 0.03 | 0.19 | 0.07 | 0.23 | 0.04 | 0.04 | 0.27 | 0.03 | 1.00 |

Notes: All pairwise correlations > 0.1 or $\leq$ 0.1 are significant at $p < 0.05$. †Disparity process diversity score used in regression models. Results are similar for the use of separation and variety process diversity scores.

**Table 5. Estimation Results: Antecedents of Process Diversity†**

| Independent Variables | | Effect on Process Diversity | |
|---|---|---|---|
| Process compliance | 1 | -0.087** | (0.035) |
| Requirements volatility | 2 | 0.129*** | (0.038) |
| Design and technology novelty | 3 | 0.309*** | (0.091) |
| Customer involvement | 4 | 0.056*** | (0.015) |
| Team size | 5 | -0.024*** | (0.006) |
| Team experience | 6 | -0.146*** | (0.039) |
| Code size | 7 | 0.087*** | (0.033) |
| Project manager certification | 8 | 0.293*** | (0.086) |
| Intercept | 9 | 0.352 | (0.281) |
| Sample size | | 410 | |
| Adjusted $R^2$ | | 0.713 | |
| $\chi^2$ | | 1112.20*** | |

Notes: Standard error in parentheses. ***Significant at < 1%; **Significant at < 5%; *Significant at < 10%. †Disparity process diversity score used in regression model; results do not vary across the use of disparity separation, and variety process diversity scores as well as their linear combinations.

**Table 6. Estimation Results: Performance Implications**

| Independent Variables | | Fit Measured as Deviation Score | | Fit Measured as Residual Score† | |
|---|---|---|---|---|---|
| | | (1) Effect on Productivity | (2) Effect on Quality | (3) Effect on Productivity | (4) Effect on Quality |
| Fit between process diversity and process compliance | 1 | 1.152*** (0.340) | 0.312*** (0.088) | 1.14*** (0.322) | 0.282*** (0.098) |
| Process diversity‡ | 2 | -0.559 (0.365) | -0.015 (0.010) | -0.309 (0.287) | -0.007 (0.005) |
| Process compliance | 3 | -0.547 (0.384) | -0.017 (0.011) | -0.208 (0.152) | -0.014 (0.010) |
| Agile KPA Score | 4 | -0.229 (0.202) | -0.038 (0.050) | -0.149 (0.139) | -0.032 (0.043) |
| Requirements volatility | 5 | -0.296** (0.119) | -0.086 (0.062) | -0.215** (0.087) | -0.054 (0.050) |
| Design and technology novelty | 6 | -0.257*** (0.087) | 0.006 (0.007) | -0.292*** (0.081) | 0.003 (0.004) |
| Customer involvement | 7 | -0.007** (0.003) | -0.001*** ($0.28 \times 10^{-3}$) | -0.006** (0.003) | -0.001*** ($0.24 \times 10^{-3}$) |
| Team size | 8 | -0.038*** (0.011) | -0.005*** (0.001) | -0.037*** (0.011) | -0.006*** (0.002) |
| Team experience | 9 | 0.013 (0.014) | 0.002*** (0.001) | 0.006 (0.012) | 0.001** ($0.44 \times 10^{-3}$) |
| Code size | 10 | 0.614*** (0.181) | -0.075*** (0.022) | 0.616*** (0.182) | -0.067*** (0.020) |
| Project manager certification | 11 | -0.021 (0.035) | -0.001 (0.002) | -0.037 (0.055) | -0.005 (0.009) |
| Intercept | 12 | -5.404*** (1.389) | 0.114*** (0.029) | -6.033*** (1.551) | 0.174*** (0.045) |
| Sample size | | 410 | 410 | 410 | 410 |
| Adjusted $R^2$ | | 0.605 | 0.251 | 0.582 | 0.243 |
| $\chi^2$ | | 613.34*** | 136.96*** | 743.55*** | 129.78*** |

Notes: Standard error in parenthesis. ***Significant at < 1%; **Significant at < 5%; *Significant at <1 0%. †Residual scores calculated from regression of process diversity on process compliance and vice versa yield consistent results. ‡Disparity process diversity score was used in regression model; results do not vary across the use of disparity, separation, and variety process diversity scores as well as their linear combinations.

process compliance efforts in a project would be negatively associated with process diversity. Our results support this prediction and show that a unit increase in process compliance efforts decreases process diversity within a project by about 9 percentage points ($\beta$= -0.087; $p < 0.05$). Results for the other control variables show that larger, more experienced teams have lower levels of process diversity. In contrast, projects with a bigger code base and projects led by managers who were professionally certified in any of the five normative process frameworks adopted by the research site were associated with increased levels of software process diversity.

Recall that hypotheses H5a and H5b predicted that a better fit between process diversity and process compliance efforts yields higher productivity and quality respectively. To verify this prediction, we refer to the results in columns 1–4 in Table 6. Coefficients presented in columns 1 and 2 pertain to the regression specifications using the fit variable derived as a deviation score. Columns 3 and 4 present the regression coefficients when an alternate residual score measure is used to derive the fit variable. Overall, the regression results are consistent irrespective of the way the fit between process diversity and process compliance is measured. The results confirm our hypotheses that a better fit between process diversity and process compliance efforts is associated with higher levels of project performance. We see that a unit increase in the fit between process diversity and process compliance improves productivity by 1.15 units ($\alpha = 1.152$; $p < 0.001$; Table 6, column 1) and increases software quality by 0.31 units ($\Upsilon = 0.312$; $p < 0.001$; Table 6, column 2). The individual, direct effects of both process diversity and process compliance on productivity and quality are not statistically significant. An important insight from this empirical result is that the fit between process diversity and process compliance, and not those mechanisms acting independently, has an impact on project performance. Therefore, in order to benefit from process diversity, concomitant investments in organizational process compliance activities are necessary.

Results for other control variables show that requirements volatility has a significant negative effect on productivity, but does not significantly affect quality. Based on this result, we infer that project teams in our data set encountered productivity losses for maintaining higher-level software quality when they faced client-driven requirements changes. We also find that bigger software code base and larger team size have a negative effect on software quality. Finally, we find that a higher degree of customer involvement is negatively associated with project performance. This implies that the project teams we observed worked better when presented with well-specified and codified customer contracts that facilitated minimal intervention from customers. Although a higher level of customer involvement in a project is commonly expected to be positively associated with project success, prior research findings have shed light on some of the mixed effects of high degree of customer involvement, such as the possibility of a conflict-ridden, lengthy, and less-effective development process (Heinbokel et al. 1996; Subramanyam et al. 2010). It is possible that the presence of process diversity in a high process maturity environment exacerbates such negative effects of a higher degree of client involvement. Also, clients could have increased their involvement if they perceived their projects as not performing well. Beyond these speculations, we do not have a sufficient basis to tease out the causal direction of the effects of customer involvement. There is a need for further investigation of the effects of customer involvement on project performance in the presence of process diversity in high process maturity environments.

### Robustness Test for Group Size Effects

Diversity variables operationalized to measure group-level separation, variety, and disparity scores have been reported to suffer from biases when there is a large variation of group sizes in the sample (Biemann and Kearney 2010). Recall that we treat a KPA as the fundamental unit of a software process, and group size is determined by the membership of KPAs to process templates. To rule out group-size-induced biases in our results, we specifically accounted for group size (number of KPAs), while deriving the variance and standard deviation scores as recommended in the literature (see Biemann and Kearney 2010). Regression estimates using the original and the bias-corrected process diversity scores were similar, indicating that group size-induced biases are not of a concern in this study.[14]

## Discussion

### Implications for Research

This study takes an important step forward in rigorously conceptualizing, measuring, and analyzing software process diversity. We have laid the groundwork for theorizing software process diversity as an outcome of choices made by a project team in response to specific contingencies faced during the life cycle of the project. By drawing on prior work on organizational and demographic diversity (see Harrison and Klein 2007), we conceptualized how software process diversity can be understood not only in terms of the traditional

[14]We thank the senior editor and the associate editor for recommending this robustness check.

separation of plan versus agile dichotomies (i.e., the separation dimension), but also along the variety and disparity dimensions. Going beyond the surface level classification of plan versus agile, a consideration of the additional process variety and disparity dimensions might suggest that there is greater heterogeneity, such as when project teams adopt and fuse different frameworks from within the agile or within the plan-based family of process frameworks. We believe that the study has laid a good foundation to "move beyond entrenched disagreements about planning versus agility" (Austin and Devin 2009, p. 476), and has created a theoretically well-informed and empirically verifiable framework for studying software process diversity. We have established a rigorous case for usefully combining the disparate control and flexibility-focused process frameworks that could facilitate a new generation of software process innovations.

Our finding regarding the importance of a good fit between process diversity and process compliance has important implications. It suggests that, as process diversity proliferates within a project, adequate and appropriate process compliance mechanisms are needed to mitigate hazards due to poor or ill-conceived process variants. This result calls for expanding the current theoretical conceptualizations of software project risk management to accommodate process variations within a project as a potential and distinct source of risk that spans both the social and technical subsystems of a project (Iversen et al. 2004; Wallace et al. 2004).

The need to achieve a good fit between software process diversity and process compliance has at least two additional implications for software project management. First, it is important to understand how adequate autonomy and appropriate coordination mechanisms could be instituted between project-level personnel and centrally organized (at firm-level) compliance personnel without exacerbating the tradeoffs between efficiency and flexibility (Ramasubbu 2014; Subramanyam et al. 2012). Second, prior conceptualizations of project control (e.g., Kirsch et al. 2002; Nidumolu and Subramani 2003) should be complemented with new mechanisms that explicitly take into consideration the effects of process diversity. We only studied the overall effort expended on process compliance, leaving open the need for further investigations of how specific portfolios of controls could be designed and implemented to regulate the fit between process diversity and process compliance.

### Implications for Practice

An important implication for practice stemming from this research is the need to judiciously manage the alignment between process diversity and process compliance in software projects. Based on our discussions with practitioners at our research site, we provide a few implementable guidelines for software development organizations. First, development of meta-routines or problem-solving procedures that are independent of specific normative process frameworks would provide firms with an organization-wide platform for improving the efficiency of compliance mechanisms without compromising overall flexibility (Adler et al. 1999). Such meta-routines need to be enabled through appropriate infrastructural and organizational support mechanisms such as automation of process template audits and adequate separation of concerns between organizational-level SEPG personnel and project-level development personnel (Ramasubbu 2014).

Second, we noticed that teams used scope and temporal partitioning for creating effective boundaries between the applications of different varieties of processes in their projects. The possibility of spatial partitioning—that is, the use of separate specialized units that could independently handle the distinct aspects of plan-based and agile processes—was also raised during our discussions with project managers. The use of such scope, temporal, and spatial partitioning of tasks within a unit has been known to aid organizational ambidexterity (Puranam et al. 2006). In such scenarios, the partitioning and switching events of a project need to be used to trigger the necessary process compliance investments. For example, whenever a software team performs process partitioning or switching, an automatic audit by SEPG could be triggered, thereby facilitating a systematic way to match the levels of process diversity and process compliance in a software project. Furthermore, process partitioning and switching milestones could be used for enrichment and learning activities such as peer-reviews, root-cause analyses, and formal training that improve software project performance (Ramasubbu et al. 2008). Thus, we recommend that SEPG managers develop infrastructural capabilities that help implement mechanisms to track and trace project-level process partitioning and switching events.

Finally, since our results indicated that customer involvement plays an important role in the way project teams initiate process variations, we recommend SEPG personnel use cues from customer communication at the organizational level (e.g., customer satisfaction surveys) to triangulate and verify the actual need to approve or block process variations. Such triangulation of customer feedback from multiple communication channels (organizational and project levels) could help process compliance personnel to identify and rectify the existence of contradictory customer policies at different levels, and, thereby, avoid unnecessary or risky process variations.

## Limitations and Further Research

There are some limitations of this study that future research could address. First, we could not fully distinguish the unique effects of separation, variety, and disparity dimensions of software process diversity in this research. The three diversity scores were highly correlated in our data set, and we suspect that idiosyncratic characteristics of the development environment at our research site could be a reason for this. Future research could shed light on the factors that lead to distinct effects of the separation, variety, and disparity process diversity dimensions through data gathered from a broader set of firms. Second, since we observed only custom (bespoke) software development projects, we should be cautious in generalizing our results across all types of software development projects (maintenance, reengineering, product development, etc.). The research model and empirical analysis utilized in this study can be replicated in other project settings and future research could embark on the necessary comparative analyses. Third, we did not observe the long-term impacts of process diversity and only studied its impact on immediate project performance outcomes. Further research is needed to ascertain the long-term impacts of software process diversity on the learning curves and the capability development of project teams. Fourth, we studied only software projects using standard process components drawn from well-established normative process frameworks. Not all software teams utilize standard process frameworks, and we need to be cautious in extrapolating the results reported in the paper to teams that use a variety of homegrown processes. Finally, the distinct coexistence of several process designs in a software production ecosystem for longer periods of time, and the way those diverse process designs coevolve and adapt to each other, warrants further examination. We believe these are fruitful lines of inquiry for future research on software process diversity.

## Acknowledgments

We thank Senior Editor Arun Rai, the associate editor, three anonymous reviewers, Sandra A. Slaughter, and seminar participants at the University of Pittsburgh, Temple University, Georgia Institute of Technology, Indiana University, University of Waterloo, Naval Postgraduate School, Southern Methodist University, Nanyang Technological University, and Microsoft Research for helpful comments. We thank Gregory M. Latshaw for excellent editing. Errors and omissions remain our own.

## References

Adler, P. S., Goldoftas, B., and Levine, D. I. 1999. "Flexibility Versus Efficiency: A Case Study of Model Changeovers in the Toyota Production System," *Organization Science* (10:1), pp. 43-68.

Agarwal, M., and Chari, K. 2007. "Software Effort, Quality, and Cycle Time: A Study of CMM Level 5 Projects," *IEEE Transactions on Software Engineering* (33:3), pp. 145-156.

Anderson, D. J. 2005. "Stretching Agile to Fit CMMI Level 3—The Story of Creating MSF for CMMI Process Improvement at Microsoft Corporation," in *Proceedings of 2005 Agile Development Conference*, Washington, DC, July 24-29, pp. 193-201.

Andres, H. P., and Zmud, R. W. 2002. "A Contingency Approach to Software Project Coordination," *Journal of Management Information Systems* (18:3), pp. 41-70.

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

Banker, R. D., and Slaughter, S. A. 1997. "A Field Study of Scale Economies in Software Maintenance," *Management Science* (43:12), pp. 1709-1725.

Banker, R. D., and Slaughter, S. A. 2000. "The Moderating Effects of Structure on Volatility and Complexity in Software Enhancement," *Information Systems Research* (11:3), pp. 219-240.

Barki, H., Rivard, S., Talbot, J. 1993. "Toward an Assessment of Software Development Risk," *Journal of Management Information Systems* (10:2), pp. 203-225.

Barry, E. J., Kemerer, C. F., and Slaughter, S. A. 2006. "Environmental Volatility, Development Decisions, and Software Volatility: A Longitudinal Analysis," *Management Science* (52:3), pp. 448-464.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., and Thomas D. 2001. "Manifesto for Agile Software Development" (http://agilemanifesto.org/; accessed April 6, 2014).

Bella, F., Hormann, K., and Vanamali, B. 2008. "From CMMI to SPICE—Experiences on How to Survive a SPICE Assessment Having Already Implemented CMMI," in *Proceedings of the 9th International Conference on Product-Focused Software Process Improvement*, Monte Porzio Catone, Italy, June 23-25, pp. 133-142.

Belsley. D. A., Kuh, E., and Welsch, R. E. 2004. *Regression Diagnostics: Identifying Influential Data and Source of Collinearity*, Hoboken, NJ: John Wiley & Sons.

Begel, A., and Nagappan, N. 2007. "Usage and Perceptions of Agile Software Development in an Industrial Context: An Exploratory Study," in *Proceedings of the First International Symposium on Empirical Software Engineering and Measurement*, Madrid, Spain, September 20-21.

Biemann, T., and Kearney, E. 2010. "Size Does Matter: How Varying Group Sizes in a Sample Affect the Most Common Measures of Group Diversity," *Organizational Research Methods* (13:3), pp. 582-599.

Boehm, B. 2002. "Get Ready for Agile Methods, with Care," *IEEE Computer* (35:1), pp. 64-69.

Boehm, B. 2003. "Using Risk to Balance Agile and Plan-Drive Methods," *IEEE Computer* (36:6), pp. 57-66.

Boehm, B., and Turner, R. 2003. *Balancing Agility and Discipline: A Guide for the Perplexed*, Reading, MA: Addison-Wesley.

Boh, W. F., Slaughter, S. A., and Espinosa, J. A. 2007. "Learning from Experience in Software Development: A Multilevel Analysis," *Management Science* (53:8), pp. 1315-1331.

Chrissis, M. B., Konrad, M., and Shrum, S. 2011. *CMMI for Development: Guidelines for Process Integration and Product Improvement*, Reading, MA: Addison-Wesley.

Davidson, R., and MacKinnon, J. G. 1993. *Estimation and Inference in Econometrics*, New York: Oxford University Press.

Deck, M. 2002. "Managing Process Diversity While Improving Your Practices," *IEEE Software* (18:3), pp. 21-27.

Dewar, R., and Werbel, J. 1979. "Universalistic and Contingency Predictions of Employee Satisfaction and Conflict," *Administrative Science Quarterly* (24:3), pp. 426-448.

Dyba, T., and Dingsoyr, T. 2008. "Empirical Studies of Agile Software Development: A Systematic Review," *Information and Software Technology* (50:9-10), pp. 833-859.

Ethiraj, S., Kale, P., Krishnan, M. S., Singh, J. V. 2005. "Where Do Capabilities Come From and How Do They Matter? A Study in the Software Services Industry," *Strategic Management Journal* (26:1), pp. 24-25.

Fitzgerald, B., Hartnett, G., and Conboy, K. 2006. "Customising Agile Methods to Software Practices at Intel Shannon," *European Journal of Information Systems* (15:2), pp. 197-210.

Gibbs, P. J., and Martin, W. T. 1962. "Urbanization, Technology, and the Division of Labor: International Patterns," *American Sociological Review* (27:5), pp. 667-677.

Gopal, A., and Gao, G. 2009. "Certification in the Indian Offshore IT Services Industry," *Manufacturing and Service Operations Management* (11:3), pp. 471-492.

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

Harrison, D. A., and Klein, K. J. 2007. "What's the Difference? Diversity Constructs as Separation, Variety, or Disparity in Organizations," *Academy of Management Review* (32:4), pp. 1199-1228.

Harter, D., Kemerer, C., and Slaughter, S. A. 2012. "Does Software Process Improvement Reduce the Severity of Defects? A Longitudinal Field Study," *IEEE Transactions on Software Engineering* (38:4), pp. 810-827.

Harter, D., Krishnan, M. S., and Slaughter, S. A. 2000. "Effects of Process Maturity on Quality, Cycletime, and Effort in Software Product Development," *Management Science* (46:4), pp. 451-466.

Heinbokel, T., Sonnentag, S., Frese, M., Stolte, W., and Brodbeck, F. C. 1996. "Don't Underestimate the Problems of User Centeredness in Software Development Projects—There Are Many!," *Behaviour and Information Technology* (15:4), pp. 226-236.

Homan, A. C., van Knippenberg, D., Van Kleef, G. A., and De Dreu, C. K. W. 2007. "Bridging Faultlines by Valuing Diversity: Diversity Beliefs, Information Elaboration, and Performance in Diverse Work Groups," *Journal of Applied Psychology* (92:5), pp. 1189-1199.

Humphrey, W. S. 1989. *Managing the Software Process*, Reading, MA: Addison-Wesley

IEEE SPA. 2010. IEEE Computer Society Software Process Achievement Award (http://www.computer.org/web/awards/humphrey-spa; accessed October 17, 2010).

Iversen, J. H., Mathiassen, L., and Nielsen, P. A. 2004. "Managing Risk in Software Process Improvement: An Action Research Approach," *MIS Quarterly* (28:3), pp. 395-433.

Jakobsen, C. R., and Johnson, K. A., 2008. "Mature Agile with a Twist of CMMI," in *Proceedings of the 2008 Agile Conference*, Toronto, August 4-8, pp. 212-217.

Jakobsen, C. R., and Sutherland, J. 2009. "Scrum and CMMI Going from Good to Great," in *Proceedings of the 2009 Agile Conference*, Chicago, August 24-28, pp. 333-337.

Jiang, J. J., Klein, G., Hwang, H-G., Huang, J., Hung, S-Y. 2004. "An Exploration of the Relationship Between Software Development Process Maturity and Project Performance," *Information and Management* (41: 3), pp. 279-288.

Keil, M., and Carmel, E. 1995. "Customer-Developer Links in Software Development," *Communications of the ACM* (38:5), pp. 33-44.

Keil, M., Wallace, L., Turk, D., Dixon-Randall, G., and Nulden, U. 2000. "An Investigation of Risk Perception and Risk Propensity on the Decision to Continue a Software Development Project," *Journal of Systems and Software* (53:2), pp. 145-157.

Kemerer, C. F. 1992. "How the Learning Curve Affects CASE Tool Adoption," *IEEE Software* (9:3), pp. 23-28.

Kemerer, C. F. 1993. "Reliability of Function Points Measurement: A Field Experiment," *Communications of the ACM* (36:2), pp. 85-97.

Kilduff, M., Angelmar, R., and Mehra, A. 1995. "Top Management Team Diversity and Firm Performance: Examining the Role of Cognitions," *Organization Science* (11:1), pp. 21-34.

Kirsch, L. J., Sambamurthy, V., Ko, D-G., and Purvis, R. L. 2002. "Controlling Information Systems Development Projects: The View from the Client," *Management Science* (48:4), pp. 484-498.

Krishnan, M. S., and Kellner, M. I. 1999. "Measuring Process Consistency: Implications for Reducing Software Defects," *IEEE Transactions on Software Engineering* (25:6), pp. 800-815.

Krishnan, M. S., Kriebel., C. H., Kekre, S., and Mukhopadhyay, T. 2000. "An Empirical Analysis of Productivity and Quality in Software Products," *Management Science* (46:6), pp. 745-759.

Lindvall, M., and Rus, I. 2000. "Process Diversity in Software Development," *IEEE Software* (7:4), pp. 14-18.

Lycett, M., Macredie, R. D., Patel, C., and Paul, R. J. 2003. "Migrating Agile Methods to Standardized Development Practice," *IEEE Computer* (36:6), pp.79-85.

MacCormack, A., and Verganti, R. 2003. "Managing the Sources of Uncertainty: Matching Process and Context in Software Development," *Journal of Product Innovation Management* (20:3) pp. 217-232.

MacCormack, A., Verganti, R., and Iansiti, M. 2001. "Developing Products on 'Internet Time': The Anatomy of a Flexible Development Process," *Management Science* (47:1), pp. 133-150.

Magdaleno, A. M., Werner, C. M. L., and Araujo, R. M. 2012. "Reconciling Software Development Models: A Quasi-Systematic Review," *Journal of Systems and Software* (85:2), pp. 351-369.

Maruping, L. M., Venkatesh, V., and Agarwal, A. 2009. "A Control Theory Perspective on Agile Methodology Use and Changing User Requirements," *Information Systems Research* (20:3), pp. 377-399.

Matook, S., and Maruping, L. M. 2014. "A Competency Model for Customer Representatives in Agile Software Development Projects," *MIS Quarterly Executive* (13:2), pp. 77-95.

McGarry, F., and Decker, B. 2002. "Attaining Level 5 in CMM Process Maturity," *IEEE Software* (19:6), pp. 87-96.

Mookerjee, V., and Chiang, I. R. 2002. "A Dynamic Coordination Policy for Software System Construction," *IEEE Transactions on Software Engineering* (28:7), pp. 684-694.

Moran, A. 2014. "Agile Risk Management," Chapter 3 in *Agile Risk Management*, pp. 33-60.

Napier, N. P., Mathiassen, L., and Robey. D. 2008. "From Dichotomy to Ambidexterity: Transcending Traditions in Software Management," in *Proceedings of the 14th Americas Conference on Information Systems*, Toronto, Canada, August 14-17.

Nidumolu, S. R. 1995. "The Effect of Coordination and Uncertainty on Software Project Performance: Residual Performance Risk as an Intervening Variable," *Information Systems Research* (6:3), pp. 191-219.

Nidumolu, S. R., and Subramani M. R. 2003. "The Matrix of Control: Combining Process and Structure Approaches to Managing Software Development," *Journal of Management Information Systems* (20:3), pp. 159-196.

Ogasawara, H., Ishikawa, T., and Moriya, T. 2006. "Practical Approach to Development of SPI Activities in a Large Organization: Toshiba's SPI History since 2000," in *Proceedings of the 28th International Conference on Software Engineering*, Shanghai, China, May 20-28, pp. 595-599.

Paulk, M. C. 2001. "Extreme Programming from a CMM Perspective," *IEEE Software* (18:6), pp. 19-26.

Pikkarainen, M., Haikara, J., Salo, O., Abrahamasson, P., and Still, J. 2008. "The Impact of Agile Practices on Communication in Software Development," *Empirical Software Engineering* (13:3), pp. 303-337.

Puranam, P., Singh, H., and Zollo, M. 2006. "Organizing for Innovation: Managing the Coordination–Autonomy Dilemma in Technology Acquisitions," *Academy of Management Journal* (49:2), pp. 263-280.

Ramesh, B., Mohan, K., and Cao, L. 2012. "Ambidexterity in Agile Distributed Development: An Empirical Investigation," *Information Systems Research* (23:2), pp. 323-339.

Ramasubbu, N. 2014. "Governing Software Process Improvements in Globally Distributed Product Development," *IEEE Transactions on Software Engineering* (40:3), pp. 235-250.

Ramasubbu, N., and Balan, R. K. 2009. "The Impact of Process Choice in High Maturity Environments: An Empirical Analysis," in *Proceedings of the 31st International Conference on Software Engineering*, IEEE Computer Society, Washington, DC, pp. 529-539.

Ramasubbu, N., Mithas, S., Krishnan, M. S., and Kemerer, C. F. 2008. "Work Dispersion, Process-Based Learning, and Offshore Software Development Performance," *MIS Quarterly* (32:2), pp. 437-458.

Stamelos, I. 2010. "Software Project Management Anti-Patterns," *Journal of Systems and Software* (83:1), pp. 52-59.

Staples, M., and Niazi, M. 2008. "Systematic Review of Organizational Motivations for Adopting CMM-Based SPI," *Information and Software Technology* (50:7-8), pp. 605-620.

Subramanyam, R., and Krishnan, M. S. 2003. "Empirical Analysis of CK Metrics for Object-Oriented Design Complexity: Implications for Software Defects," *IEEE Transactions on Software Engineering*, (29:4), pp. 297-310.

Subramanyam, R., Ramasubbu, N., and Krishnan, M. S. 2012. "In Search of Efficient Flexibility: Effects of Software Component Granularity on Development Effort, Defects, and Customization Effort," *Information Systems Research* (23:3), pp. 787-803.

Subramanyam, R., Weisstein, F. L., and Krishnan, M. S. 2010. "User Participation in Software Development Projects," *Communications of the ACM* (53:3), pp. 137-141.

Takeishi, A. 2002. "Knowledge Partitioning in the Interfirm Division of Labor: The Case of Automotive Product Development," *Organization Science* (13:3), pp. 321-338.

Tang, X., and Rai, A. 2014. "How Should Process Capabilities be Combined to Leverage Supplier Relationships Competitively?," *European Journal of Operational Research* (239:1), pp. 119-129.

Van der Pijl, G. J., Swinkles, G. J. P., and Verrijdt, J. G. 1997. "ISO 9000 Versus CMM: Standardization and Certification of IS Development," *Information & Management* (32:6), pp. 267-274.

Venkatraman, N. 1989. "The Concept of Fit in Strategy Research: Toward Verbal and Statistical Correspondence," *The Academy of Management Review* (14:3), pp. 423-444.

Vidgen, R., and Wang, X. 2009. "Coevolving Systems and the Organization of Agile Software Development," *Information Systems Research* (20:3), pp. 355-376.

Vinekar, V., Slinkman, C. W., and Nerur, S. 2006. "Can Agile and Traditional Systems Development Approaches Coexist? An Ambidextrous View," *Information Systems Management* (23:3), pp. 31-42.

Volberda, H. W., and Lewin, A. Y. 2003. "Co-evolutionary Dynamics Within and Between Firms: From Evolution to Co-evolution," *Journal of Management Studies* (40:8), pp. 2111-2136.

Wallace, L., Keil, M., and Rai, A. 2004. "How Software Project Risk Affects Project Performance: An Investigation of the Dimensions of Risk and an Exploratory Model," *Decision Sciences* (35:2), pp. 289-321.

Zellner, A. 1962. "An Efficient Method of Estimating Seemingly Unrelated Regressions and Tests for Aggregation Bias," *Journal of American Statistics Association* (57), pp. 348-368.

## About the Authors

**Narayan Ramasubbu** is an assistant professor of Business Administration at the Joseph M. Katz Graduate School of Business. He received a Bachelor of Engineering degree from Bharathiar University, India, and a Ph.D. degree from the University of Michigan, Ann Arbor, MI. Prior to his academic career, he was a senior software developer and product manager at SAP AG and CGI Inc. His current research interests include software process and product design with a focus on globally distributed product development and service delivery; the design, implementation, and governance of enterprise information systems; and software-drive innovation. He currently serves as an associate editor for *Management Science* and *Information Technology and Management*.

**Anandhi Bharadwaj** is the Goizueta Term Professor of Information Systems at the Goizueta Business School of Emory University. Her research interests include the business value and impact of information technology, digital business strategies, health IT systems, and technology outsourcing. She currently serves as a department editor for *Management Science* and as a senior editor for *Information Systems Research*. In the past she has served on the editorial boards for *MIS Quarterly* and *Journal of the Association for Information Systems*. Her research has been presented in leading journals such as *Information Systems Research, Management Science, MIS Quarterly, Organization Science, Production & Operations Management*, and *IEEE Transactions on Engineering Management*, and at numerous conferences worldwide.

**Giri Kumar Tayi** is a professor of Management Science and Information Systems at the State University of New York at Albany. His research and teaching interests are interdisciplinary and span the fields of Information Systems, Operations Management, and Operations Research. He serves or has served on the editorial boards of academic journals such as *Information Systems Research, IEEE Intelligent Systems, IEEE Computing Now, Decision Sciences, ACM Journal of Data and Information Quality, Information Technology Management, Information Systems Frontiers, Information and Management,* and *International Journal of Shipping and Transport Logistics*. His papers have appeared in *Operations Research, Information Systems Research, Management Science, MIS Quarterly, IEEE Transactions, Networks, Naval Research Logistics, EJOR, Journal of Combinatorial Optimization, INFORMS Journal of Computing, Journal of Computer Security, Government Information Quarterly,* and *Communications of the ACM.*