Inf Syst Front (2010) 12:219–234
DOI 10.1007/s10796-008-9093-4

A socio-technical approach to improving the systems development process

Ravi Patnayakuni · Cynthia P. Ruppel

Published online: 21 May 2008

Abstract Research on improving the systems development processes has primarily focused on mechanisms such as tools, software development methodologies, knowledge sharing and process capabilities. This research has yielded considerable insights into improving the systems development process, but the large majority of information systems development projects still continue to be over budget, late, and ineffective in meeting user needs. Together with the advent of software development moving offshore, or consisting of offshore team members, a more holistic approach is appropriate. Approached from a socio-technical perspective the software development process is viewed as a process embedded in a social and a technical subsystem. Drawing upon socio-technical work design principles, this paper suggests how capabilities of the development process can be improved. Data collected from a survey of software development practices in organizations indicates that organizations at different levels of process capabilities differ in work system characteristics as well as process performance. For example, the use of multi-skilled teams was found to be significantly related to the systems development process maturity level as well as significantly related to all the performance measures studied. This paper provides empirical support for the socio-technical approach and provides a theoretical foundation for designing software process initiatives in organizations.

Keywords Socio-technical systems · Work systems · Systems development · CMMI · Teams

R. Patnayakuni (✉)
Department of Economics and Information Systems,
University of Alabama in Huntsville,
301 Sparkman Drive,
Huntsville, AL 35899, USA
e-mail: r.patnayakuni@uah.edu

C. P. Ruppel
Huizenga School of Business and Entrepreneurship,
Nova Southeastern University,
3301 College Ave.,
Davie, FL 33314, USA
e-mail: ruppel@nova.edu

1 Introduction

In a rapidly changing business and technological environment the ability to develop and deploy new systems is an important capability that can differentiate organizations. As a result, a variety of systems acquisition strategies have been developed by organizations including using application service providers, acquiring Commercial Off-The-Shelf Software (COTS), outsourcing and off-shore development projects. For instance, the Software Engineering Institute (SEI) states that about 40% percent of top rated (CMM level 5) software companies are located in India (King 2005). However, of the $2.5 trillion spent on information technology during 1997–2001, nearly $1 trillion was spent on underperforming IS projects (Benko and McFarlan 2003). A significant proportion of these projects eventually fail, costing US firms more than $78 billion each year (Levinson 2001). These trends only underscore the importance of managing the systems development process whether it is in house or is distributed across organizations and/or locations.

Systems development from a software engineering perspective (Pfleeger 2001), in association with structured methods (Martin 1986) and more recently, object oriented approaches with Unified Modeling Language (UML) 2.0 from Object Management Group, are perhaps the most significant influence on contemporary systems development

(Rose 2002). Efforts to improve systems development performance have focused on the introduction of new methodologies, development technologies and process improvement techniques (Lyytinen and Robey 1999; Patnayakuni and Rai 2002). While many organizations have adopted a variety of development technologies and methodologies, there is little evidence that they lead to performance gains (Fichman and Kemerer 1997; Schmidt et al. 2001). Lyytinen and Robey (1999) argue that despite advances in tools and methods, it is the failure of organizations to learn from prior systems development projects that results in poor performance. Others argue that Information Systems Departments (ISDs) in organizations do not change their software development process in order to take advantage of new methods and tools (Rai and Howard 1994; Fichman and Moses 1999). Similarly, normative models, such as the Capability Maturity Model (CMM)[1] developed by the Software Engineering Institute, have been recommended as frameworks to structure and direct software process improvement efforts. While CMM has repositioned the focus of performance improvement initiatives towards development practices, the use of the model has had mixed results in the field (Pfleeger 1996; Diaz and Sligo 1997; Hollenbach et al. 1997; Harter et al. 2000; Staples et al. 2007).

Meanwhile, the socio-technical systems approach has been used successfully to design manufacturing and service organization processes for the past three decades (Shani et al. 1992; Mumford 2003). This approach focuses on work design and views an organizational work system as an open system interacting with its environment that consists of two separate but interdependent subsystems: the social and the technical (Pasmore 1998). The fundamental premise of social-technical systems is that organizations that 'jointly optimize' the two interdependent subsystems are more likely to obtain positive outcomes. An integrated approach to designing organizational work systems requires complementary changes to be made to both subsystems. This paper presents the socio-technical approach to work design of the software development process by, (1) presenting the principles for work design and (2) its application to this process. Normative changes in the software development process and the organizational context at different levels of process capability are identified. To facilitate the understanding of different levels of process capability and allow comparisons across organizations, this paper uses the descriptions and labels of the CMM framework.

We begin by examining the predominant approaches to improving the software development process discussed in literature. The general principles of socio-technical design as an alternate approach to improving the software development are then introduced. An examination of the software development process from a socio-technical perspective at different levels of process capability as well as the development of a conceptual model to examine its impact on process performance follows. The next section presents descriptive analysis of data on socio-technical work design, CMM and process performance obtained from an empirical study of systems development practices in organizations. The last section discusses the research and managerial implications of the findings from this study.

2 Improving the software development process: No silver bullets

Since Fred Brooks (1987) pointed out the inherent difficulties in developing software, researchers and practitioners have been exploring different approaches to improve the development process. Much of the focus has been from an engineering perspective; an attempt to bring the discipline of engineering to the software development process. Initiatives to improve the software development process have generally been directed at development methodologies, tools or practices (Lyytinen and Robey 1999; Avgerou 2001; Rose 2002) and more recently using knowledge management approaches (Patnayakuni et al. 2006, 2007)

2.1 Using development methodologies

A dominant theme in the software engineering literature suggests that developing and disseminating better development methods will produce better applications. Organizations approach their systems development function from a variety of methodological approaches, and often subscribe to no particular methodology or to several different methodologies (Ivari et al. 2001). The limited empirical evidence available suggests that few organizations actually use any development methodology consistently. In a recent interview Rumbaugh, one of the developers of UML, commented, "We hope that more and more people will do modeling—that they won't sit down and build software without designing it first. As far as modeling in general, I doubt that the majority of developers still do it. I think a lot of them still sit down and write code" (Anonymous 2002). One of the major criticisms against development method-

[1] CMM, or Capability Maturity Model has been replaced by the Software Engineering Institute by Capability Maturity Model Integration (CMMI). However, on their website they state: "Many of the skills used in applying the Software CMM are useful in implementing a CMMI-based process improvement program, since many of the best practices, issues, and improvement approaches are essentially the same." Since our study primarily uses CMM as an indicator of process capabilities, the results are applicable across both models proposed by SEI.

ologies is that for a variety of reasons they appear to have not been able to organize and guide the software development process (Ivari et al. 2001). Others have argued that methodologies are useful primarily for guiding beginners rather than for use by seasoned developers (Unhelkar and Mamdapur 1995; Mathiessen 1998). UML and OPEN represent two efforts to unify the growing number of methodologies for object oriented development. Although most software engineers would agree that development methodologies can provide the platform for a more disciplined development process, limited usage and a vast diversity of methodologies suggests that the use and application of development methodologies has not taken root in practice or strongly impacted the development process.

2.2 Implementing tools for systems development

Another approach to improving systems delivery performance has been the deployment of tools to support a variety of development activities in the software development process. These range from tools that support or automate a specific software development task such as diagramming tools for drawing flow charts, to those that can support the entire process of software development (Rai and Howard 1994; Purvis et al. 2001). Organizations are likely to have one or more development tools as a part of their development environment, or more commonly, a portfolio of such tools. Use of development tools is expected to (1) increase development efficiency, (2) raise reliability of developed systems, and (3) ensure conformity to user requirements (Tate et al. 1992).

Published research on development tool implementation is primarily based on experiences of practitioners in organizations (Wynekoop 1993). Despite claims of potential positive impacts of development tools, there is little empirical evidence to support this notion. While some studies have reported productivity gains from use of development technologies (Banker and Kauffman 1991; Finlay and Mitchell 1994) others have found the expected gains to be elusive (Card et al. 1987; Keil et al. 2000; Purvis et al. 2001; Ravichandran and Rai 2002). Some empirical studies suggest that development technologies have only a marginal effect on quality (Card et al. 1987; Ravichandran and Rai 2002). It can be argued that one of the underlying reasons for the lack of use and impact of development technologies is the notion that IT by itself cannot improve performance outcomes; a belief widely held today by researchers and practitioners alike (Markus and Benjamin 1997; Avgerou 2001). When implementing development technologies, organizations must account for the underlying business processes, specify complementary organizational changes (Avgerou 2001; Aladwani 2002; Iversen and Mathiassen 2003) and knowledge management practices (Patnayakuni et al. 2006, 2007). This belief has resulted in an increased focus on software development practices to improve the effectiveness of the development process.

2.3 Focus on development practices and standardized processes

In a move to establish standards for software engineering practices and methods, in 1984 the US Department of Defense formed the Software Engineering Institute at Carnegie Mellon University. The project that attempted to characterize the capabilities of software development organizations resulted in the development of the capability maturity model (CMM); a framework for software process improvement. Based on the principles of total quality management, it argues that the application of the principles which has proven to be effective in both engineering and manufacturing would be equally effective in the case of software development (Humphrey and Curtis 1991). CMM proposes a five level model in which higher levels are indicative of greater process maturity and hence capability. It establishes the evolutionary stages that an organization needs to pass through to establish a culture of software engineering excellence (Humphrey and Curtis 1991). Each level is considered as the foundation for building effective practices for the next level.

CMM has raised awareness concerning the importance of improving the software development process as well as allowing comparisons between organizations when outsourcing all, or portions of, the development process. Adoption of CMM is seen as a growing phenomenon evidenced by the increasing number of assessments both within US and globally outside the US (Herbsleb and Zubrow 1997). However, the use of the model has had mixed results in practice. Organizations find that the time and cost of implementing CMM for software process improvements often exceeds their expectations. Some organizations implementing CMM-based software process improvement initiatives have realized gains in development cycle time and programmer productivity (Diaz and Sligo 1997; Hollenbach et al. 1997; Agrawal and Chari 2007). A study of 30 software products and their development processes in a major IT firm showed that higher process maturity levels, although associated with higher product quality, require increased development effort (Harter et al. 2000). Other reports indicate that several organizations face substantial difficulties in adhering to the sequence of maturity levels and accompanying process changes (Card et al. 1987; Pfleeger 1996), and the lack of theory informing these stages and their sequence also raises questions about the implementation of CMM (Ravichandran and Rai 2002).

One criticism of the framework has been that while it defines the process normatively, it provides little guidance on how to achieve actual process improvement (Herbsleb and Zubrow 1997). Trienekens et al. (2007) found that the three most important improvement drivers in software process improvement among CMM level three groups were (1) commitment of engineering management, (2) commitment of development staff and (3) sense of urgency, all of which are related to the social systems. Similarly, Niazi et al. (2005) state that, "The importance of SPI (Software Process Improvement) implementation demands that it be recognized as a complex process in its own right and that organizations should determine their SPI implementation maturity through an organized set of activities". Thus, the application of the socio-technical approach to systems development processes provides one such theoretical perspective to supplement the normative maturity levels presented by CMM and to guide overall software development process improvement.

3 Socio-technical systems approach to work design

The socio-technical approach has its roots in work done by Eric Trist and a group of social scientists who formed the Tavistock Insititute of Human Relations in London following the Second World War (Mumford 1995). They established the foundation for socio-technical systems theory and design. With a strong predilection for scientific theory, the principles were developed from a series of experiments conducted by the Tavistock group. Based on the premise that organizations are open systems that consist of a social and a technical system, organizations need to recognize the need to optimize and bring together social and technical systems in organizing work. The work is probably amongst the earliest to recognize the importance of self-managing groups in organizations. Over the last three decades this approach has been used as an organizational design tool for examining and changing the work place environment, in particular manufacturing and production work environments (Shani et al. 1992; Mumford 1995; Alter 2001).

The joint optimization of both the social and the technical systems in an organization is central to the socio-technical systems approach to designing work. Work design here refers to the organization of tasks in transforming inputs to outputs and the technical and social subsystems refer to the organizational context in which the transformation process is embedded. Work design, based on socio-technical systems design principles, has been shown to result in increased productivity through better utilization of human resources and capital equipment, as well as the improved quality of work life. Mumford (1995) draws parallels between the socio-technical approach and business process reengineering and suggests that many of the ideas put forth by current reengineering practice can be evidenced in the early work done at Tavistock. She states, "It is not unusual to read in management texts that multiskilled teams are a Japanese invention or that a process approach is new and American. The great strength of the Tavistock approach is that it combines good practice with good theory and it always has the dual objective of using technology and people as effectively as possible" (Mumford 1995, p. 207).

The socio-technical approach to design offers a structured approach for assessing and redesigning work systems. It is based on broad design principles (Cherns 1987; Mumford 1995), that can provide integrated guidelines for work design. The principles, with brief descriptions, are listed in Table 1 and are adapted from Shani et al. (1992) and Mumford (1995).

4 Applying the socio-technical approach to systems development

The systems development process in an organization can be conceived as a work system (Alter 1999) in which

Table 1 Socio-technical principles of work design

Principle	Description
Variance control	Error should be detected and corrected as close to their origin as possible and preferably by the same group of employees
Boundary location	Organizational boundaries should be eliminated where there are task interdependencies that require close coordination and information sharing. Boundaries between different employee-task configurations are identified by looking for discontinuities of time, place and product development
Deliberation legitimization	Decision-making should be explicit, and the decision making process should be designed and structured to maximize information availability and knowledge utilization
Redundant functions	Introducing variety in the work organization through multi-functional teams and multi-skilled individuals
Compatibility of design process	The process of design of the technical and social system should be compatible to the needs of multiple users. Individual and social attributes of work system participants should be taken into consideration

developers build information systems using organizational resources, which include human resources, in terms of such things as skill and knowledge, as well as technological resources such as development tools and the IT infrastructure. The application of socio-technical design principles would then be evident in the characteristics of the technical, social and work system of the systems development organization. In addition it would be manifested in improved systems development capabilities and in improved performance outcomes.

4.1 The technical subsystem

The technical subsystem of a work system consists of the tools, techniques, devices, artifacts, methods, configurations, procedures and knowledge used by participants to acquire inputs, and transform them into outputs (Pasmore 1988). In systems development, the portfolio of development tools and methodologies used to build systems would represent the technical subsystem.

Integration in the development platform can be characterized by tool and object integration (Mi and Scacchi 1992). Tool integration would provide seamless integration of the development platform across spatially and temporally distributed software development tasks. Such integration would ensure that development tools can interface and share information with each other across different stages of the development process. Object integration is the consistent view of development artifacts in the development process (Mi and Scacchi 1992). It ensures that the semantic content of development objects is not subject to attrition during the systems development lifecycle. The scope of technology support for development determines the extent to which the different stages of the development life cycle are supported. Accordingly the technical subsystem in the organization is characterized by the level of integration and the scope of development tools. The measurement items are listed in Appendix 1 along with scale properties.

From a socio-technical work design perspective, an integrated systems development platform that spans across the entire development life cycle enables an accessible and consistent view of development thus reducing the need for correcting errors and supporting variance control. It also enables automatic collection of data that facilitate the measurement and identification of sources of variance in the development process thereby providing the necessary information to development teams for making decisions (deliberation legitimization). By supporting multiple stages of development, it can bridge discontinuities of time and space in the development life cycle (boundary location). The characteristics of the technical system support the principle of redundant functions, where information is created once and then is accessible to all members of the development team any time and any place. The importance of making knowledge explicit to all members of the team has recently begun to be linked to the CMM process via knowledge management concerns (Dayan and Evans 2006).

In the context of this study the limited capability of the technical subsystem will be characterized by the stand-alone use of tools that tend to have a local impact on individual tasks rather than having an impact on downstream tasks or any linkages to upstream tasks, thus limiting knowledge sharing, particularly in non co-located teams. A low level of integration will likely require the developer to undertake substantial rework from loss of information across the development lifecycle.

At higher levels of integration and support for different stages of the development process the capability and sophistication of software and hardware are also likely to increase. A development infrastructure marked by a high level of tool integration that provides the platform for physical, logical and semantic integration of development objects across the systems development process, and will be reflected in improved organizational capability to develop quality systems.

4.2 The social subsystem

Traditionally the social subsystem of an organization comprises of the individuals who work in the organization and the sum total of their individual and social attributes (Shani et al. 1992). Rather than regarding it as an aggregate of individual attributes the social subsystem of an organization may be viewed as the context in which the organizational work system operates and is characterized by its attributes as a whole rather than as an aggregate of individual attributes.

The social system is viewed here as one that possesses the capacity to change so that socio-technical work design principles can be employed. The capacity to change and adapt has been associated with organic forms of organization. Both structural and process-oriented approaches to organizational change are based on the assumption that organic forms of organization are desirable (Zanzi 1987). Organic forms of organization have been found to interact positively with modern technological systems such as flexible manufacturing (Parthasarthy and Prakash 1993). Mechanistic organizations are considered to resemble the traditional, bureaucratic model while organic organizations represent more flexible, process-oriented, open type internal arrangements (Burns and Stalker 1961; Bahrami 1992). The organic versus mechanistic continuum represents the traditional emphasis in organizational classification on structural dimensions where researchers are concerned with the differentiation of tasks and positions, rules and procedures, and the prescription of authority (Greenwood

1993). An important aspect of flexibility in an organizational context is the ability to precipitate intentional changes, continuously respond to unanticipated changes, and to adjust to the unexpected consequences of predictable changes (Bahrami 1992). At the core of such organizations is diffusion of control in the organizational work context that enables employees to respond to changes. Thus diffusion of control is one of the dimensions used to characterize the social subsystem in this study.

Communication channels in an organization are often considered as one of the primary structural features of the organization in realizing effective implementation outcomes (Fidler and Johnson 1984). Social traditions of the organization can encourage employees inclination to talk and engage in "water-cooler talk," whenever co-workers find themselves proximate and marking time (Sarbaugh-Thompson and Feldman 1998). Informal channels are considered to be more flexible, as they can activate more senses, and are more attuned to specific problems of employees. They are also able to carry more information through a variety of codes resulting in "richness" of communication (Daft and Lengel 1986). The presence of informal communication in the organization is likely to result in rapid dissemination of information giving the organization the necessary flexibility to cope with change and uncertainty. It also increases the diffusion of the knowledge base of the organization and facilitates collaboration. In addition it encompasses relationships within groups and between groups including; lateral relationships (with peers) and vertical relationships (with supervisors and management); formal and informal relationships; and political relationships that reflect the distribution of power, culture and tradition. The extent of informal communication is the second attribute used to characterize the social system in this study.

The presence of these two characteristics enable the socio-technical work design principles of boundary location by improving communication across and within organizational units, and deliberation legitimization by providing autonomy to employees and bringing to bear their knowledge and experience in performing organizational work. It will also reflect the principle of compatibility whereby employees are empowered to cultivate open communication in the organization. Thus, it is proposed that social systems at lower levels of process capability are likely to be characterized by centralized control and lack of informal communication. Higher levels of process capability are likely to be associated with diffusion of control and high levels of informal communication.

4.3 Work design

A basic premise of the socio-technical approach is that work systems should be designed and implemented to account for the social and task requirements of different stakeholders. It should provide opportunities for participation, knowledge sharing, and growth. A work environment that recognizes the social, growth, task-related and motivational needs of employees is more likely to result in an effective software development processes. Formal work practices in organizations may be oriented towards efficiency where there is minimal redundancy of tasks, de-emphasis of collaboration, and a focus on hierarchical control (Melcher et al. 1990). An orientation towards socio-technical design in contrast would be associated with knowledge generation and transfer, institutionalization of control structures and work processes that enable collaboration and cross-fertilization of individual employee knowledge. We focus on control and work structures that are enacted on a day-to-day basis and hence form 'practice' in the organization (Brown and Duguid 2001; Orlikowski 2002). Work design that emphasizes combining employees with diverse skills, viewpoints, ideas and values are likely to result in better process capabilities. Formalization, use of teams, and multi-skilling are considered to be the characteristics of systems development work in the organization.

Formalization of tasks, reporting relationships and lines of authority is likely to make work structures inflexible and difficult to change. Traditionally organizations have emphasized specialization in jobs based on division of labor that are managed with controls and hierarchies to coordinate tasks. From a socio-technical perspective, especially in the knowledge intensive context of systems development, formalization will likely diminish process capability. Lack of formalization will facilitate redundancy and result in knowledge sharing in the work system by deploying individuals with multiple skills and by combining them into work teams.

The use of teams provides forums for rapid exchange of information with possible detection of errors at an early stage in the development process and enables employees to govern themselves. Working in teams sets the stage for individual knowledge to spiral up to groups and eventually the organization (Okhuysen and Eisenhardt 2002). It enables employees to create shared understandings that facilitate the organizational integration of individual knowledge. Such teams have been shown to be 'task dominant' i.e. teams that are highly focused and committed to the end product (Souder 1987). In knowledge intensive processes, the work of teams can become central to organizational success.

Socio-technical work design also suggests that boundaries between different employee-task configurations should be examined by looking for discontinuities of time, place and product development and ensuring that they are managed effectively. It ensures that employees have the necessary information to hand over tasks smoothly to the next stage of transformation. An organization that eliminates functional distinctions and allows for open and free

flow of information across the work system is more likely to be more productive. The importance of knowledge sharing in implementing CMM is being studied (Dayan and Evans 2006) Work systems designed in this fashion do not require elaborate and redundant controls; hence management hierarchies can be flattened as teams members absorb greater responsibility for coordination and decision making.

As organizations improve their systems delivery capabilities, software development tasks are likely to be increasingly complex, overlapping, and require multiple skills. When different phases of software development are linked together, less specialization with increased flexibility and versatility in the application of multiple skills will be enabled by the use of teams. With integrated and optimized software processes, employment arrangements need to be more flexible as high levels of integration are conducive to broader, less specific job assignments. Project teams operating autonomously facilitate the use of flexible employment arrangements. Similarly, increasing process integration emphasizes knowledge sharing and is enabled by a lack of formalization, use of teams and multi-skilling of developers resulting in higher process capabilities. Table 2 shows the expected profile of technical, social and work design characteristics at different levels of process capability.

4.4 Performance

Work systems that reflect the application of socio-technical principles should result in improved process capabilities which in turn should be reflected in improved performance of the development process. Satisfaction of users with the systems developed (customer satisfaction), on time and on budget delivery of systems (process performance), and quality of systems developed (product performance) are useful dimensions along which systems development performance can be assessed (Finlay and Mitchell 1994; Ravichandran and Rai 2002).

Performance may be assessed by the extent to which customers of the ISD are satisfied with the systems delivered by it. Given trends in computing, such as decentralization and outsourcing, organizations today have substantial discretion in the use and purchase of IS services. Therefore improvement in systems delivery capability should be associated with higher levels of user satisfaction among organizational customers of the development process. Process performance should then be assessed in terms of improvement in productivity and cycle time reduction for systems delivery, and product performance in terms of improvement in the quality of systems delivered.

5 Empirical study

From a socio-technical perspective, development process capabilities will depend on an organizations ability to bond the technical system, quite often consisting of advanced information technologies such as Computer Aided Software Engineering (CASE) tools, with the social system for achieving an optimal fit between the organization and its environment (Pasmore 1998). According to this approach, bringing the two systems together occurs at the work design level. Work design includes many organizational design

Table 2 A Socio-technical system based comparative examination of systems development process

Process capability	Level 1 ad hoc	Level 2 repeatable	Level 3 defined	Level 4 managed	Level 5 optimized
Technical system					
Scope of development tools	Pockets of automation	Adjacent task integration	Adjacent stage integration	Multiple stage integration	Integrated
Level of integration	Low	Moderate	Moderate/high	High	High
Social system					
Diffusion of control	Managers makes decisions based on authority	Some programmer participation is invited	Moderate programmer/ developer participation	High levels of participation	Participative decision making is the norm
Communication	Most communication is formal	Low levels of informal communication within developers	Informal communication crosses organizational and rank boundaries	High	Informal communication is encouraged and is the norm
Work design					
Use of teams	Mostly individual task design	Mostly individual task design	Semi-autonomous work groups	Semi-autonomous work groups	Autonomous work groups
Formalization	Rigid/mechanistic	Mechanistic	Semi-organic	Organic	Organic/networked
Specialization	High specialization with routine tasks	Overlapping Specialization	Some multiple skills	Multiple skill requirements	Multiple and anticipated skills

elements such as skill requirements, formalization and authority, and use of autonomous work teams as a structural unit of work. As organizations design their systems development process based on socio-technical approaches, the work system will possess improved process capabilities resulting in higher levels of customer satisfaction, product quality and process performance. The overall framework for this investigation is presented in Fig. 1.

5.1 Data collection and analysis

Data was collected by a survey questionnaire as part of a study examining systems development practices. After an initial pilot test with senior IS managers and developers in five organizations the survey was mailed to systems development managers listed in the Directory of Top Computer Executives published by Applied Computer Research, Phoenix, Arizona. Surveys were sent to 708 organizations in the manufacturing and service sector whose primary business was not software development, implementation or maintenance. A total of 123 responses were received after two mailings, giving us a response rate of 18.14%. While the response rate is modest, it is acceptable (Pinsonneault and Kraemer 1993) and is also quite close to the minimum of 20% recommended by some researchers (Yu and Cooper 1983; Grover et al. 1996). Of the returned surveys, 110 responses were considered complete and usable. The sample was tested for non-response bias and no significant differences were found among organizations responded and those that did not. We further tested for differences among the respondents grouped by their self-assessed levels of process capability. One-way ANOVA indicated that there were no differences across the groups in terms of organization size, budget of the department, and size of the Information Systems Department (ISD) as measured by the number of full time employees. This is consistent with results of Herbsleb et al. (1997) who found no differences in process capability across organizational size.

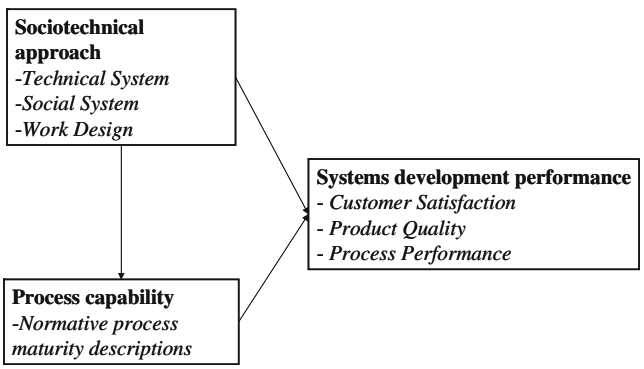

Fig. 1 The research model

The average annual revenue of organizations, an indicator of organization size, in the sample was 1.4 billion dollars. About 60% of the organizations had annual revenue less than one billion dollars. The size of the ISD of organizations ranged from two to 450 full time employees. The average number of employees in the ISD was 59.89. Fifty percent of the organizations in the sample employed thirty or less full time employees in their IS unit. The distribution of number of employees, number of employees involved in development work, and the number of managers in the information systems department's is shown in Table 3. The average IS budget of the organizations that responded was 9.34 million dollars.

Respondents were asked questions relating to the elements of technical subsystem, social subsystem, work design characteristics, self-assessed development process capabilities and performance of the development process in terms of customer satisfaction, product quality, and process efficiency. Respondents were asked to indicate their agreement/disagreement with each statement on seven-point Likert type scale (strongly agree–strongly disagree). The individual survey items are listed in Appendix 1 together with the analysis of scale properties. Process capability assessment was based on the descriptions and labels of the CMM framework which provide desirable states of process capability. Respondents were asked to place their organizations systems development capability at any one of the levels. The descriptions used for the survey items are provided in Appendix 2.

The content validity of the measurement instrument was established through careful attention to the process of instrument development. The process included interviews with systems development managers followed by a pilot study with two doctoral students, six IS researchers and four IS executives from different organizations who reviewed the instrument for both content coverage and clarity of the questions. Reliability of the scales was assessed using Cronbach's alpha. Factor analysis was used to examine the dimensionality and construct validity of the variables. If the factor analysis revealed more than one underlying dimension for a construct, each factor was further examined. Reliability and validity are reported in Appendix 1.

The analytical approach presented here blends descriptive and inferential analysis by depicting changes in the work system characteristics with process capability using graphs, mean scores for each work system characteristics as well as performance across different levels of process capability to provide insights at a finer level of granularity. In order to conduct the analysis, a simple average of all the items belonging to construct was calculated, then all the scores were then standardized to a 0–1 scale. Responses were grouped by the level of process capability indicated in the

Table 3 Composition of IS departments

Number of employees	Frequency (percent)	Number of employees in systems development	Frequency (percent)	Number of managers	Frequency (percent)
0–10	18 (16.5%)	0–5	25 (22.9%)	1	12 (11.0%)
11–20	19 (17.4%)	6–10	19 (17.5%)	2	11 (10.1%)
21–50	37 (34.0%)	11–25	32 (29.3%)	3–5	52 (47.7%)
51–100	15 (13.1%)	26–50	14 (12.9%)	6–10	21 (19.3%)
101–200	12 (11.0%)	51–100	11 (10.1%)	11–50	12 (11.0%)
>200	8 (6.3%)	>100	8 (6.3%)	>50	1(0.9%)
Total	109		109		109
Mean	59.890		29.138		6.706
Standard deviation	79.326		39.172		10.133

survey. Since the total number of respondents who indicated their systems development capabilities as corresponding to levels 4 and 5 descriptions of CMM were 8 and 12 respectively, respondents from the two groups were combined to create a single group for the purpose of this analysis. As a result the number of respondents in each CMM level was: level 1–26, level 2–43, level 3–21, and level 4 and 5–20. Work system characteristics, process capabilities and performance across different levels of process capability were compared using one-way ANOVA. In addition contrasts of two adjacent levels were performed to see if the increase/decrease at subsequent levels of process capability were statistically significant. For example, the significance of the difference across levels 2 and 3 in their mean scores of level of integration in the technical subsystem was tested in the analysis. Finally, regression models were run to explore the association of the work system elements with each of the three performance variables (customer satisfaction, process efficiency and product quality).

5.2 Results

The results generally support the idea that a socio-technical approach to work design is associated with higher levels of process capability and in turn development performance. The differences in these elements are marked between organizations at level 1 vs. level 2 and level 2 vs. level 3 of process capability. The descriptive analysis is presented using graphs and a table showing the results of contrasts performed using one-way ANOVA.

The technical system in which the systems development process is embedded, is characterized by the scope of the development environment and level of integration. The proportion of organizations with a development platform that supports the full life cycle of systems development increases from 23% to 81% at level 3 and 60% of organizations at level 4 and 5. There is a corresponding decline in the use of single phase tools at higher levels of process capability (see Fig. 2). Thus organizations at higher levels of process capability appear to provide a more comprehensive technological platform for systems development.

A similar pattern is observed with the level of integration. The mean scores for level of integration are plotted as a line graph in Fig. 3 and the results of the contrasts are given in Table 4. Although it would appear from the graph that there is a drop in level of complexity from level 3 to subsequent higher levels, the difference is not statistically significant.

The social subsystem was profiled in terms of diffusion of control and extent of informal communication. A similar trend is observed for the two work system elements in the social subsystem. Organizations appear to create and support informal communication and participative decision making at higher levels of process capability. Organizations at higher levels of process capability created an environment where employees constantly exchange information thereby building a knowledge base that enables improved systems delivery capability. Analysis of variance indicates that although organizations at different levels of process capability are significantly different, the individual contrast across levels 1 and 2 is significant although contrasts across levels 2 and 3, and level 3 to levels 4 and 5 are not significant. This indicates that while the overall trend is significant the differences from one level to the next are not significant (Fig. 4).

Work design of the systems development process in organizations is characterized in terms of formalization,

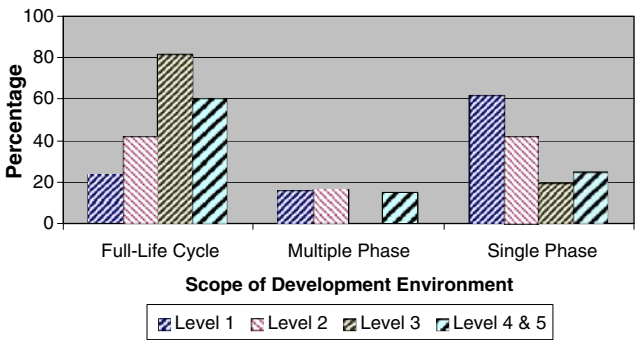

Fig. 2 Development environment across levels of process capability

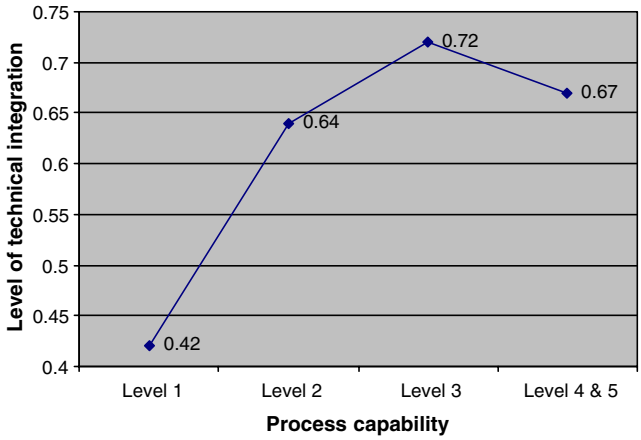

Fig. 3 Technical integration at different levels of process capability

multi-skilling and use of teams. The use of teams at level 1 tends to be minimal (mean score 0.37) and is more predominant at level 3 and above (mean score, 0.69). A graph showing each individual work design element is shown in Fig. 5. Although the difference in use of teams, and multi-skilling across all levels and from level 1 to level 2 as well as level 2 to level 3 was significant, the level of formalization did not differ across levels of process capability. Results of the individual paired contrasts from one level to the next are shown in Table 4.

Next, analysis was conducted to examine whether higher levels of capability have any relation to performance outcomes assessed in terms of customer satisfaction, quality of the developed product and performance of the systems development process. One-way ANOVA was conducted where differences in performance along these dimensions across groups at different levels were analyzed. As can be seen in Fig. 6, performance along each of these dimensions improves with process capability and the overall difference among groups is significant on all three performance indicators. The difference in performance is significant from level 1 to 2 and from level 2 to 3 only in the case of process performance. Overall the analysis indicates that higher levels of process capability based on the CMM reference model do lead to improved systems development process outcomes.

Finally a multiple regression analysis was conducted with elements of the social system, technical system and work design as independent variables and each of the systems development performance measures as the dependent variable (Table 5).

The results indicate that development process performance is significantly impacted by work system elements (each model is significant). Informal communication and use of teams seem to have the greatest impact on multiple measures of development performance. Further, many work systems elements are crucial in attaining higher levels of process capability. Although the analysis does not directly test the 'joint optimization' of subsystems it does demonstrate the importance of work design elements. In addition it highlights the influence of social system characteristics on development process performance.

6 Discussion

The process of developing systems and improving the outcomes in organizations has historically been primarily viewed from an engineering perspective. This could also be

Table 4 Results of contrasts across levels of process capability

Process Capability	Level 1	Level 2	Level 3	Level 4 & 5
Technical System				
Level of integration		$p = .000$	$p = .121$	$p = .446$
Social System				
Diffusion of control		$p = .005$	$p = .243$	$p = .736$
Informal communication		$p = .025$	$p = .716$	$p = .213$
Work Design				
Use of teams		$p = .001$	$p = .019$	$p = .383$
Formalization		$p = .478$	$p = .342$	$p = .947$
Multi-skilling		$p = .002$	$p = .011$	$p = .581$

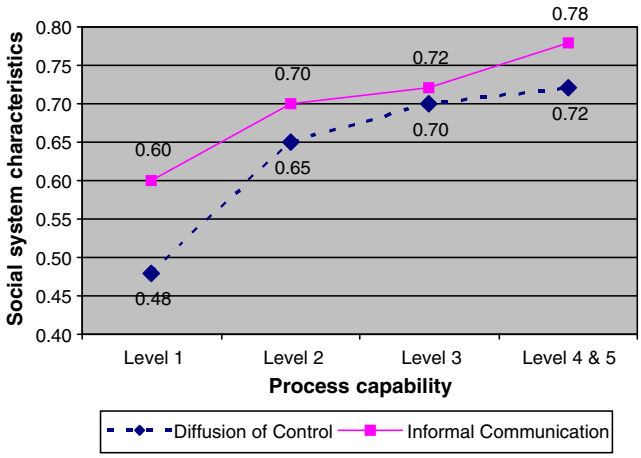

Fig. 4 Social system characteristics at different levels of process capability

Fig. 6 Performance outcomes at different levels of process capability

characterized as a technology deterministic perspective (Markus and Robey 1988) where the focus has been on creating technologies to support systems development or the introduction of methodologies and techniques to systematically develop systems with the discipline of engineering. With increasing dependence on information technology, developing information systems for business use has become a critical organizational capability. The technology implementation literature has emphasized the need for making concomitant organizational changes while implementing new technologies and methods (Leonard-Barton 1991; Shani et al. 1992; Alter 2001).

To improve the systems delivery capabilities of organizations it is necessary to recognize that the development process is embedded in the social and technical environment of the organization. Approaching organizational processes as a work system (Alter 2001) takes into account the constituents of an organizational process as well as the context in which it is embedded. The socio-technical approach conceptualizes organizational work systems as

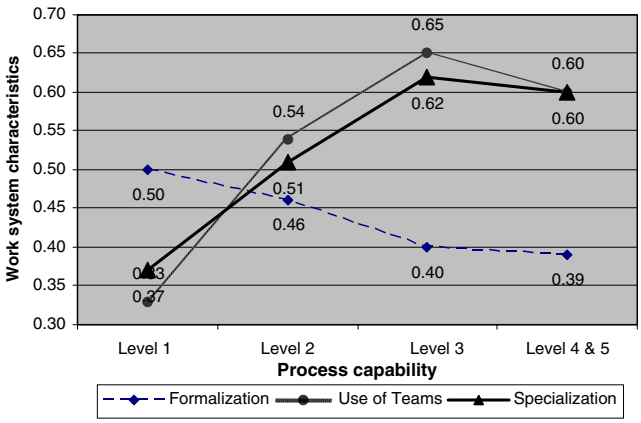

Fig. 5 Work design characteristics at different levels of process capability

open systems interacting with an environment that consists of two separate but interdependent subsystems, the social and the technical subsystem as well as provides a theoretical foundation for improving the systems development process. A primary focus of this approach has been work design which deals with how organizational tasks are configured, performed, and measured. It takes into account not only the organizational process but also the organizational practices that support the process.

This approach provides a set of principles for work design such as variance control, boundary location, and redundant functions that are remarkably similar to those proposed by business process redesign advocates such as gap analysis, examination of inputs and outputs and elimination of non-value added tasks (Hammer 1990). Significant parallels between the two approaches (Mumford 1995) to design processes, coupled with the significant impact of work design on direct measures of systems

Table 5 Regression models of systems development performance

Work system components	Customer satisfaction	Product quality	Process performance
R square (*F* value)	0.42 (10.61***)	0.40 (9.69***)	0.1 (10.13***)
Constant	2.13	1.25	−0.12
Technical system			
Scope of development tools	−0.42	0.29	−0.11
Level of integration	3.00***	1.61	1.37
Social system			
Diffusion of control	−0.02	0.84	1.82*
Informal communication	3.06***	1.27	2.88***
Work design			
Formalization	0.97	−0.21	0.21
Use of teams	2.35**	2.67***	1.65*
Specialization	−0.76	0.64	−0.23

*$p<0.10$
**$p<0.05$
***$p<0.01$

development performance, further reinforces the usefulness and applicability of the socio-technical approach.

The design principles, when applied to systems development work, show the association of organizational infrastructure and practices with improved process capability and subsequently organizational performance. The interdependence of changes in the two subsystems and work design is emphasized in the progression towards higher levels of capability. This indicates the importance of joint optimization of different aspects of organizational work. Without a complex and sophisticated development infrastructure it will be difficult for the process to develop syntactical and semantic integration across different stages of systems development. Patnayakuni et al. (2007) illustrate the importance of development infrastructure in managing the integration of development knowledge in the systems development process. Similarly without appropriate technology in place it would probably be difficult to install mechanisms for automatic data collection and use it for effective management of the process. Team work and multi-skilled employees's work can be facilitated with technology, however, unless reinforced with appropriate delegation and team-based performance measurement, it is unlikely to bring about process improvements. Thus approaching organizational work systems from a socio-technical perspective needs to take into account a holistic view of the context in which such systems exist.

The descriptive and inferential analysis of data collected concerning the development process in organizations supports the proposition that improved process capabilities are associated with the technical, social and work design characteristics suggested by the socio-technical approach. The correspondence between organizational work design characteristics and improved process capability are validated by the improved performance outcomes for organizations operating at higher levels of self assessed process capability. The differences were significant at initial levels and were not statistically different at higher levels. This explains the findings in other research on CMM, organizations face considerable difficulty in moving from level 1 to level 2 and organizations find it very difficult to move beyond level 3 (Herbsleb and Zubrow 1997). Organizations face the most difficulty in initiating change from existing systems and practices at lower levels of process capability. At higher levels of capability, organizations would have developed the technical and social infrastructure and work design practices to support more flexible yet effective processes.

The elements of socio-technical work design seemed to indicate an overall impact on systems development performance with the greatest impact on the social system principle of diffusion of control and the work design principle of the use of teams. This further validates the importance of incorporating social system considerations in systems development projects. In addition, the lack of impact of several aspects of formal work design and the actual tool portfolio indicate that while the design of task and technology is a critical element in achieving higher levels of process capability, in and of itself, it may not directly impact systems development performance. An interesting issue suggested by the analysis is that of balancing the dynamic of formalization and flexibility. While the use of teams appears to be consistently associated with process capability, there is almost no variation in the extent of formalization across different levels of process capability. This finding warrants further study.

Development work in organizations with lower capabilities is typically focused around functional specializations. This represents a mechanistic structure where specialized work tasks are managed under hierarchical control. Reward systems tend to be based on individual performance, which is quantitatively assessed, and information exchange is along formal channels that parallel the vertical relationships in the organizations. With improved process capabilities there will be a need for organic structure/organic overlays that offers greater flexibility and require less hierarchical control. Work design will change from isolated development to teams working in an integrated development environment. Organizations may wish to modify their reward systems to reinforce team-based functioning in the organization by relying on group-based and team-based rather than individual rewards. Similarly they should have metrics at a group level rather than at an individual level only. Furthermore, developers should increasingly interact with one another as well as with users and clients. While higher levels of process capabilities may often result in increased formalization of the software development process in the organization; as organizations work towards improving their systems development processes they will have to balance the tension between increased formalization and the need for participation and involvement of developers. Organizations can balance the demands of increasing formalization by making the control systems more self-regulated by the work groups with use of teams and diffusion of control. This approach would be conducive to maintaining developer motivation in what may otherwise be perceived as an increasingly rigid development environment.

Self-regulated groups that work in close coordination are facilitated with the automated, fully networked transfer of information as characterized by increased integration in the technical support environment. A contradiction often evident with such integration is that it promotes an organizational structure that possesses elements that extend beyond those associated with organic structures (Shani et al. 1992). This is evident in the need to be multi-skilled yet with a blend of specialized skills, maintaining formal control yet having autonomous groups, being in a mode of continuous improvement yet being stable, increased interaction with the environment yet maintaining boundaries,

and so on. These are likely to be the characteristics of a new flexible organization (Bahrami 1992) that by means of structural overlays of team organization over a formal structure can cope with a rapidly changing environment. Thus improved process capabilities will typically be associated not only with improved productiveness but also with improved capabilities to learn, adapt and change.

These results also provide guidelines for those designing arrangements for software development with outsourcing firms. Organizations need to examine both the social and technical systems of those firms and to carefully design the work system between the organization and the outsourcing firm incorporating the principles discussed here. This study indicates that failure to do so will impact development performance.

6.1 Limitations

This paper explores the applicability of the socio-technical approach to work systems specifically applied to the process of developing systems. This study is exploratory in nature and a research model is developed to guide the investigation. A more rigorous test of the research model with an analysis of construct level relationships should be undertaken in future research. More dimensions and additional indicators of the technical subsystem, the social subsystem and work design elements need to be developed. A limitation of the study is the use of CMM process maturity descriptions as a normative reference scale based on self-assessment, although it represents a practical industry standard. The analysis suggests that differences in process capabilities beyond level 3 are marginal at best. This would suggest that either other work system elements not captured here may change, that different aspects of performance are impacted or that better measures of process capabilities need to be developed. Perhaps in the future, as more organizations achieve higher levels of capabilities, this research should be revisited. Developing more comprehensive measures and testing their psychometric properties is an important next step in this research. Also, this study is subject to inherent limitations of a research design where data is collected by means of a survey.

7 Concluding remarks

Effective systems development is key to developing responsive strategies in the new digital economy. Software engineering typically takes a very objectivist, technology deterministic approach to systems development (Bansler and Bodker 1993; Booch 1999). In contrast the socio-technical approach is subjective in nature (Ivari et al. 2001) where not only is process efficacy an objective it also emphasizes quality of work life issues. The usefulness of this approach is validated by the practice of process reengineering and process simplification in industry, as well as in research where it is suggested that work systems should be the central and focal point of analysis for studying information systems (Alter 1999, 2001).

The socio-technical approach provides an integrated view of the organizational work context in which work processes and information technology can be examined. The principles for designing work suggest how information technology and work design can complement each other to provide positive outcomes. In the context of the systems development process, the usefulness of the socio-technical approach is illustrated with initial empirical support for the premise that work design characteristics influence process capabilities and performance. This perspective provides opportunities for research in other areas, most importantly for studying post-adoption processes subsequent to implementing new information technology in organizations.

Appendix 1

Unless otherwise specified, respondents were required to indicate their agreement or disagreement on a seven-point Likert-type scale (strongly disagree, disagree, slightly disagree, neutral, agree slightly, agree, strongly agree).

Table 6 Analysis of reliability and unidimensionality

Items	Factor loading	Variance explained (%)	Standardized α
Technical system			
Development environment			
Scope of development tools (single item measure)			
Our portfolio of development tools is mainly			
□ Full-life cycle			
□ Front-end analysis and design			
□ GUI client development			
□ Back end construction			
□ Other (please specify)			
Level of integration			
Data generated during a particular task/phase of systems development is easily accessed by related tasks/phases	0.84	62.7	0.88
Modifications made to development information (such	0.82		

Table 6 (continued)

Items	Factor loading	Variance explained (%)	Standardized α
as logical models, requirements data etc.) in a particular task/phase are communicated to related tasks/phases			
Same formats and operating conventions are used by development tools across different task/phases	0.72		
Development information is easily portable from one development task/phase to other tasks/phases	0.87		
Logical models remain consistent across different development tasks/phases	0.82		
No semantic information (such as entity definitions) is lost in moving from one task/phase of development to another	0.67		
Social system			
Diffusion of control			
Participative decision making is broadly used in these development projects	0.81	62.4	0.70
Decision making authority rests with managers as opposed to development staff*	0.73		
Joint decision making by managers and analysts/programmers is the norm in our ISD	0.82		
Informal communication			
There is extensive informal communication among IS employees at the same level	0.69	63.7	0.80
There is extensive informal communication among IS employees at different levels	0.81		
There is extensive informal communication between IS employees and employees at the same level in other departments	0.84		
There is extensive informal communication between IS employees and employees at different levels in other departments	0.83		
Work design			
Formalization			
Tasks in projects have been formalized and structured as routine	0.80	63.4	0.71
Lines of authority in projects are well defined	0.76		
There are clearly specified job descriptions for individuals	0.82		
Use of teams			
Systems development is team based and problem focused	0.86	66.81	0.75
All projects are managed by autonomous teams	0.73		
Project team performance is evaluated rather than individual performance	0.85		
Multi-skilling			
We frequently rotate development staff among various positions	0.72	46.44	0.62
Job roles in development projects are overlapping rather than distinct	0.75		
We follow a 'one person, one task' approach for systems development*	0.63		
Analysts/programmers have specialized skills	0.62		
Performance			
Customer satisfaction			
Users are satisfied with developed systems	0.93	87.3	0.85
Users are satisfied with the overall quality of developed systems	0.93		
Product quality			
Systems that have been developed have high reliability	0.87	75.3	0.67
Fixing bugs and other rework account for a significant proportion of our development effort	0.87		
Process performance			
Projects finish within budgets	0.89	57.4	0.72
Projects finish on schedule	0.88		
Productivity of our development staff is high compared to other IS organizations in similar environments	0.78		
Backlog of development work is high compared to other IS organizations in similar environments (dropped)	0.33		

Appendix 2

Process capability descriptions

Level 1: Ad hoc, without formalized procedures, cost estimates and project plans.

Level 2: Stable and repeatable process based on accumulated experience of individuals, some project controls and metrics, but no process framework used.

Level 3: A defined process that is consistently implemented across projects. Sufficient data is collected to analyze process efficiency

Level 4: A managed process with comprehensive and defined process measurements. Systematic record of process performance measures is maintained.

Level 5: In a continuous improvement mode for optimizing the process.

References

Aladwani, A. M. (2002). An empirical examination of the role of social integration in systems development projects. *Information Systems Journal*, *12*, 339–353.

Alter, S. (1999). A general, yet useful theory of information systems. *Communications of the AIS*, *1*, 1–70.

Alter, S. (2001). Which life cycle—Work, information, or software? *Communications of the AIS*, *7*, 1–54.

Agrawal, M., & Chari, K. (2007). Software effort, quality and cycle time: A study of CMM Level 5 Projects. *IEEE Transactions on Software Engineering*, *33*, 145–156.

Anonymous (2002). Q&A: James Rumbaugh, a modeling champion. *Application Development Trends*, *6*, 2.

Avgerou, C. (2001). The significance of context in information systems and organizational change. *Information Systems Journal*, *11*, 43–63.

Bahrami, H. (1992). The emerging flexible organization: Perspectives from Silicon Valley. *California Management Review*, *34*, 33–52.

Banker, R. D., & Kauffman, R. J. (1991). Reuse and productivity in integrated computer-aided software engineering: An empirical study. *MIS Quarterly*, *15*, 375–401.

Bansler, J. P., & Bodker, K. (1993). A reappraisal of structured analysis: Design in organizational context. *ACM Transactions on Information Systems*, *11*, 165–193.

Benko, C., & McFarlan, W. (2003). *Connecting the dots: Aligning projects with objectives in unpredictable times*. Boston: Harvard Business School Press.

Booch, G. (1999). UML in action. *Communications of the ACM*, *42*, 26–28.

Brooks, F. P. (1987). No silver bullet: Essence and accidents of software engineering. *IEEE Computer*, *20*, 10–19.

Brown, J. S., & Duguid, P. (2001). Knowledge and organizations: A social-practice perspective. *Organization Science*, *12*, 198–213.

Burns, T., & Stalker, G. M. (1961). *The management of innovation*. London, UK: Tavistock Publications.

Card, D. N., Mcgarry, F. E., & Page, G. T. (1987). Evaluating software engineering technologies. *IEEE Transactions on Software Engineering*, *13*, 845–851.

Cherns, A. (1987). The principles of sociotechnical design. *Human Relations*, *40*, 45–51.

Daft, R. L., & Lengel, R. H. (1986). Organizational information requirements, media richness and structural design. *Management Science*, *32*, 554–571.

Dayan, R., & Evans, S. (2006). KM your way to CMMI. *Journal of Knowledge Management*, *10*, 69–80.

Diaz, M., & Sligo, J. (1997). How software process improvement helped Motorola. *IEEE Software*, *14*, 75–81.

Fichman, R. G., & Kemerer, C. F. (1997). The assimilation of software process innovations: An organizational learning perspective. *Management Science*, *43*, 1345–1363.

Fichman, R. G., & Moses, S. A. (1999). An incremental process for software implementation,. *Sloan Management Review*, *40*, 39–52.

Fidler, L. A., & Johnson, J. D. (1984). Communication and innovation implementation. *Academy of Management Review*, *9*, 704–711.

Finlay, P. N., & Mitchell, A. C. (1994). Perceptions of benefits from the introduction of case: An empirical study. *MIS Quarterly*, *18*, 353–370.

Greenwood, R. H., & Hinnings, C. R. (1993). Understanding strategic change: The contribution of archetypes. *Academy of Management Journal*, *36*, 1052.

Grover, V., Jeong, S. R., & Segars, A. H. (1996). Information system effectiveness: The construct space and patterns of application. *Information and Management*, *31*, 177–191.

Hammer, M. (1990). Reengineering work: Don't automate, obliterate. *Harvard Business Review*, *68*, 104–102.

Harter, D. E., Krishnan, M. S., & Slaughter, S. A. (2000). Effects of process maturity on quality, cycle time and effort in software product development. *Management Science*, *46*, 451–466.

Herbsleb, J., & Zubrow, D. (1997). Software quality and the capability maturity model. *Communications of the ACM*, *40*, 30–40.

Herbsleb, J., Zubrow, D., Goldenson, D., Hayes, W. & Paulk, M. C. (1997). Software quality and the capability maturity model. *CACM*, *4*, 31–40.

Hollenbach, C., Young, R., Pflugrad, A., & Smith, D. (1997). Combining quality and software process improvement. *Communications of the ACM*, *30*, 41–45.

Humphrey, W. S., & Curtis, B. (1991). Comments on 'a critical look'. *IEEE Software*, *8*, 42–46.

Ivari, J., Hirscheim, R., & Klien, H. K. (2001). A dynamic framework for classifying information systems development methodologies and approaches. *Journal of Management Information Systems*, *17*, 179–218.

Iversen, J., & Mathiassen, L. (2003). Cultivation and engineering of a software metrics program. *Information Systems Journal*, *13*, 3–19.

Keil, M., Mann, J., & Rai, A. (2000). Why software projects escalate: An empirical analysis and test of four theoretical models. *MIS Quarterly*, *24*, 631–664.

King, W. R. (2005). Outsourcing becomes more complex. *Information Systems Management*, *22*, 89–90.

Leonard-Barton, D. (1991). The role of process innovation and adaptation in attaining strategic technology capability. *International Journal of Technology Management*, *6*, 303–320.

Levinson, M. (2001). Let's stop wasting $78 billion a year. CIO, 78–83.

Lyytinen, K., & Robey, D. (1999). Learning failure in information systems development. *Information Systems Journal*, *9*, 85–101.

Markus, M. L., & Benjamin, R. I. (1997). Magic bullet theory in IT-enabled transformation. *Sloan Management Review*, *38*, 55–68.

Markus, M. L., & Robey, D. (1988). Information technology and organizational change: Causal structure in theory and research. *Management Science*, *15*, 583–598.

Martin, J. (1986). *Information engineering*. Camforth: Savant.

Mathiessen, L. (1998). Reflective systems development. *Scandinavian Journal of Information Science*, *10*, 67–117.

Melcher, A., Acar, W., Dumont, P., & Khouja, M. (1990). Standard-maintaining and continuous-improvement systems: Experiences and comparisons. *Interfaces*, *20*, 24–41.

Mi, P., & Scacchi, W. (1992). Process integration in case environments. *IEEE Software*, *9*, 45.

Mumford, E. (1995). Creative chaos or constructive change: Business process re-engineering versus socio-technical design. In G. Burke & J. Peppard (Eds.), *Examining business process re-engineering* (pp. 192–216). London, UK: Kogan Page.

Mumford, E. (2003). *Redesigning human systems*. Hershey, PA: Idea Group.

Niazi, M., Wilson, D., & Zowghi, D. (2005). A Maturity model for the implementation of software process improvement: an empirical study. *The Journal of Systems and Software*, *74*, 155.

Okhuysen, G. A., & Eisenhardt, K. M. (2002). Integrating knowledge in groups: How formal interventions enable flexibility. *Organization Science*, *13*, 370–386.

Orlikowski, W. J. (2002). Knowing in practice: Enacting a collective capability in distributed organizing. *Organization Science*, *13*, 249.

Parthasarthy, R. S., & Prakash, S. (1993). Relating strategy and structure to flexible automation: A test of fit and performance implications. *Strategic Management Journal*, *14*, 529.

Pasmore, W. A. (1998). *Designing effective organizations: The sociotechnical system perspective*. New York, NY: Wiley.

Patnayakuni, R., & Rai, A. (2002). Development infrastructure capabilities and process maturity. *Communications of the ACM*, *45*, 201–210.

Patnayakuni, R., Rai, A., & Tiwana, A. (2007). Systems development process improvement: A knowledge integration perspective. *IEEE Transactions on Engineering Management*, *54*(2), 286–300.

Patnayakuni, R., Ruppel, C., & Rai, A. (2006). Managing the complementarity of knowledge integration and process formalization for systems development performance. *Journal of Association of Information Systems*, *7*(8), 545–567.

Pfleeger, S. L. (1996). Realities and rewards of software process improvement. *IEEE Software*, *13*, 99–101.

Pfleeger, S. L. (2001). *Software engineering: Theory and practice*. Upper Saddle River, NJ: Prentice Hall.

Pinsonneault, A., & Kraemer, K. L. (1993). Survey research methodology in management information systems: An assessment. *Journal of Management Information Systems*, *10*, 75–105.

Purvis, R. L., Sambamurthy, V., & Zmud, R. W. (2001). The assimilation of knowledge platforms in organizations: An empirical investigation. *Organization Science*, *12*, 117–135.

Rai, A., & Howard, G. (1994). Propagating CASE usage for software development: An empirical investigation of key organizational correlates. *Omega*, *22*, 133–147.

Ravichandran, T., & Rai, A. (2002). Quality management in systems development: An organizational systems perspective. *MIS Quarterly*, *24*, 381–415.

Rose, J. (2002). Interaction, transformation and informations systems development—An extended application of soft systems methodology. *Information Technology & People*, *15*, 242–258.

Sarbaugh-Thompson, M., & Feldman, M. S. (1998). Electronic mail and organizational science: Does saying "hi" really matter? *Organization Science*, *9*, 685–698.

Schmidt, R., Lyytinen, K., Keil, M., & Cule, P. (2001). Identifying software project risks: An intenational Delphi study. *Journal of Management Information Systems*, *17*, 5–36.

Shani, A. B., Grant, R. M., Krishnan, R., & Thompson, E. (1992). Advanced manufacturing systems and organizational choice: Sociotechnical systems approach. *California Management Review*, *34*, 91–111.

Souder, W. (1987). *Managing new product innovations*. Lexington, MA: Lexington Books.

Staples, M., Niazi, M., Ross, J., & Abrahams, A. (2007). An exploratory study of why organizations do not adopt CMMI. *The Journal of Systems and Software*, *80*, 883.

Tate, G., Verner, J., & Jeffery, R. (1992). Case: A testbed for modeling, measurement, and management. *Communications of the ACM*, *35*, 65–72.

Trienekens, J., Kusters, R., vanGenuchten, M., & Aerts, H. (2007). Targets, drivers and metrics in software process improvement: Results of a survey in a multinational organization. *Software Quality Journal*, *15*, 135–153.

Unhelkar, B., & Mamdapur, G. (1995). Practical aspects of using a methodology: A road map approach. *Report on Object Analysis and Design*, *2*, 34–36.

Wynekoop, J. L. (1993). Strategies for implementation research: Combining research methods. International Conference on Information Systems, Dallas, TX, pp 185–193.

Yu, J., & Cooper, H. (1983). A quantitative review of research design effects on response rates to questionnaires. *Journal of Marketing Research*, *20*, 36–44.

Zanzi, A. (1987). How organic is your organization? *Journal of Management Studies*, *24*, 125.

Ravi Patnayakuni is an Associate Professor of information systems in the Department of Economics and Information Systems at the University of Alabama in Huntsville. His research focuses on, digital supply chains, IT business value, IT implementation, systems development and knowledge management. Ravi received his D.B.A from Southern Illinois University at Carbondale and has held positions at Temple University and The University of Melbourne, Australia prior to his current appointment. His research has been published in *Journal of Management Information Systems, MIS Quarterly, Communications of the ACM, Omega, Communications of the AIS, Journal of the AIS, IEEE Transactions on Engineering Management and Information Systems Journal*.

Cynthia P. Ruppel is an Associate Professor of information system in the H Wayne Huizenga College of Business and Entrepreneurship at Nova Southeastern University. She received her PhD. in MIS from Kent State University. Her research interests include the use of telecommunications to conduct business such as in telecommuting, ecommerce and supply chains,virtual teams as well as the impacts of innovation adoption and diffusion in organizations. Her work has been published in *IEEE Transactions on Professional Communications, Database for Advances in Information Systems, Journal of the AIS, Information Resource Management Journal, Business Process Management Journal, e-Service Journal*, and other journals.