2005

Software Development Process Change Management: Implementation of ASDM

George Mangalaraj
University of Texas at Arlington, mangalaraj@uta.edu

Sridhar P. Nerur
University of Texas at Arlington, snerur@uta.edu

Software Development Process Change Management: Implementation of ASDM

George A. Mangalaraj
The University of Texas at Arlington
mangalaraj@uta.edu

Sridhar P. Nerur
The University of Texas at Arlington
snerur@uta.edu

ABSTRACT

Agile software development methodologies have been receiving a lot of attention in recent times. Although doubts have been cast on the efficacy of these methods for very large projects, some of the techniques and practices they advocate are very appealing and are being seriously considered by many organizations. It is our contention that many of these practices are antithetical to the orthodoxy of prevailing software approaches. In particular, nontrivial reconfigurations of organizational form, management practices, and workflows have to be undergone to successfully integrate agile principles into existing software development practices. This paper draws on the organizational change management literature to argue that the nature of change involved is resonant of the efforts to introduce Business Process Reengineering (BPR) in organizations. The magnitude of the change as well as the implications of migrating to agile methodologies is also presented.

Keywords

Agile Software Development, Change management, Software process innovation, Extreme programming.

INTRODUCTION

Agile software development methodologies (ASDM) are a new breed of software development methodologies that are gaining prominence. These methodologies are strikingly different from the traditional software development methodologies and they radically alter the way software is developed. Adoption of software development process innovation is one of the least researched areas (Mustonen-Ollila and Lyytinen 2003) and the existing studies mostly deal with factors influencing adoption / assimilation of innovations. One factor that is often overlooked is the organizational change management when new software development process innovations occur. This study attempts to fill this void by considering the implications of change brought about by newer software development methodologies. This paper analyzes the managerial challenges associated with adoption of ASDMs. Towards this end, it relies on past researches in business process reengineering and organizational change management.

The outline of this paper is as follows. In the first section, we present a brief overview of various software development methodologies. The subsequent section analyzes the change phenomenon involved in the adoption of ASDM. The third section develops a model for change management based on research models developed for Business Process Reengineering. We conclude with a discussion of future research directions.

SOFTWARE DEVELOPMENT METHODOLOGIES

Software development projects are notorious for their failure rates. The high failure rates in software development projects has been a major source of concern (Larman 2003; The Standish Group 1995).. Increasing failure rates in the software development projects have called for newer developmental methodologies. This need has spawned a plethora of new developmental methodologies that are collectively called Agile Software Development Methodologies (ASDM). Companies are increasingly moving toward these new methodologies and many consultants are harping on the virtues of these new methodologies. Practitioners' increased interest in the agile approach to software development has been highlighted (Charette 2003).

Software development methodologies are created to formalize software development processes. There are many software development methodologies in vogue. Underlying the development methodology is a process model that specifies the order in which development proceeds. Agile software development is a subset of iterative / evolutionary delivery model (Larman 2003). The various process models are presented in table 1.

Agile Software Development Methodology

Agile software development methodologies address many problems associated with traditional software development. Agile methodology is an iterative and incremental development approach. These methods are less planning oriented and they emphasize customer participation during software development projects. These methodologies also emphasize less documentation and rely on collocation between team members rather than documentation as a means of communication (Highsmith 2003). Collocated project teams, pair programming, reflection workshops, refactoring, and test driven development are some of the salient aspects of these methodologies. ASDMs family comprises many methodologies such as Extreme Programming, Scrum, Adaptive Software Development (ASD), Dynamic Systems Development Methodology (DSDM), Feature Driven Development, etc. (Highsmith 2002).

Process Models	Description	Reference
Waterfall Model	The earliest development methodology which relies on step by sequential process.	(Royce 1970)
Spiral Model	This model relies on cyclical repetitive processes to avoid risk /uncertainty.	(Boehm 1988)
Recursive/Parallel Model	Modular development of applications is done independently and then combined.	(Berard 1990)
Evolutionary Delivery Model	This model relies on small cycles combined with incremental release of software.	(Gilb 1988)

Table 1. Software Development Process Models

Implementing agile software development methodologies require many new managerial and technological innovations to be adopted by the organizations. Success of the adoption of ASDM greatly depends on how well the organizations are able to cope with the dynamics involved in the change. Changing software development processes are taken for granted and this area needs closer attention due to the problems associated with any change process. Next section draws parallels between the change processes involved in software development with that of business process reengineering.

AGILE SOFTWARE DEVELOPMENT METHODOLOGIES VS. BUSINESS PROCESS REENGINEERING

Business process reengineering (BPR) had its origin in the 1990s. BPR initiatives were strategy driven and they are aimed at radical change of the work processes. Success of BPR initiatives has been often highlighted in both the academic and practitioner press (Grover 1999). Many companies adopted BPR practices in order to reduce cost / cycle time and be competitive in the market place.

Dimensions based on past research	Business Process Reengineering	ASD implementation
Assumptions questioned (Grover 1999)	Fundamental	Planning, process optimization etc.
Focus of change (Grover 1999)	Radical changes over broad core entities.	Many of the activities performed in software development are targeted.
Orientation(Grover 1999)	Business processes	Software development processes.
Role of IT(Grover 1999)	Very important	New tools for software development
Improvement of goals(Grover 1999)	Significant	Significant improvement in productivity, quality.
Frequency(Grover 1999)	Usually one time	Initial change is a one time affair.
Emphasis(Pruijt 1998)	Integrating processes across functional boundaries and using generalists.	Reliance on generalists who can perform interchangeable roles in a team.

Dimensions based on past research	Business Process Reengineering	ASD implementation
Approach (Pruijt 1998)	Top-down	Top-down
Role of consultants (Pruijt 1998)	Consultants played a major role in spreading BPR implementation and it has been equated management fad.	Agile methodologies were primarily developed by consultants.

Table 2. Comparison – ASD and Business Process Reengineering

In the past there have been many studies on software processes innovations. The following table 3 classifies them according to their underlying theme of the study. Adoption of software process innovations has been studied at both the individual and organizational levels. These studies have brought in different theories to provide an explanation for the adoption. Outcomes of the adoption of a software processes innovation have also been studied. There have also been studies that explored a single antecedent condition for the software process innovation to occur.

Theme	Articles
Organizational Adoption	(Fichman 2001; Leung 2001; Sultan and Chan 2000)
Individual Adoption	(Hardgrave, Davis and Riemenschneider 2003)
Adoption (single antecedent condition)	Organizational Learning (Fichman and Kemerer 1997) Organizational Culture (Ngwenyama and Nielsen 2003) Leadership (Sharma and Rai 2003)
Outcome	(Janz and Wetherbe 1997; Yang 1999)

Table 3. Past studies in the adoption of Software Process Innovations

Studies pertaining to organizational change management during the adoption of software process innovation are very limited. Orlikowski (1993), used a grounded theory approach to study the change phenomenon in organizational adoption of CASE tools that relied on the empirical data from two case studies. This paper utilizes the research in business process reengineering / organizational change management to formulate a framework for software development process change management.

THEORETICAL FRAMEWORK OF SOFTWARE DEVELOPMENT PROCESS CHANGE MANAGEMENT

Software process models guide the participants in a software development project to achieve the desired results in terms of delivery time, cost, and defect rate. In this paper, based on Kettinger and Grover's (1995) definition of Business Process Change Management, we define Software Development Process Change management (SDPC) as a strategy driven organizational initiative to improve and redesign software development process to achieve desired performance level.

Changing software development process is contingent upon many antecedents. Success of such process change initiatives is dependent on the changes in other organizational factors. The following diagram describes a model for SDPC management derived from the works of Kettinger and Grover (1995). Suitable alterations were done to the original model in order to incorporate the unique requirements of the software development arena, particularly in the context of ASDM.

The Kettinger and Grover (1995) model has been used in studying many major change phenomenon such as, self-managed work teams in software development (Janz et al. 1997), and lean manufacturing (Motwani 2003).

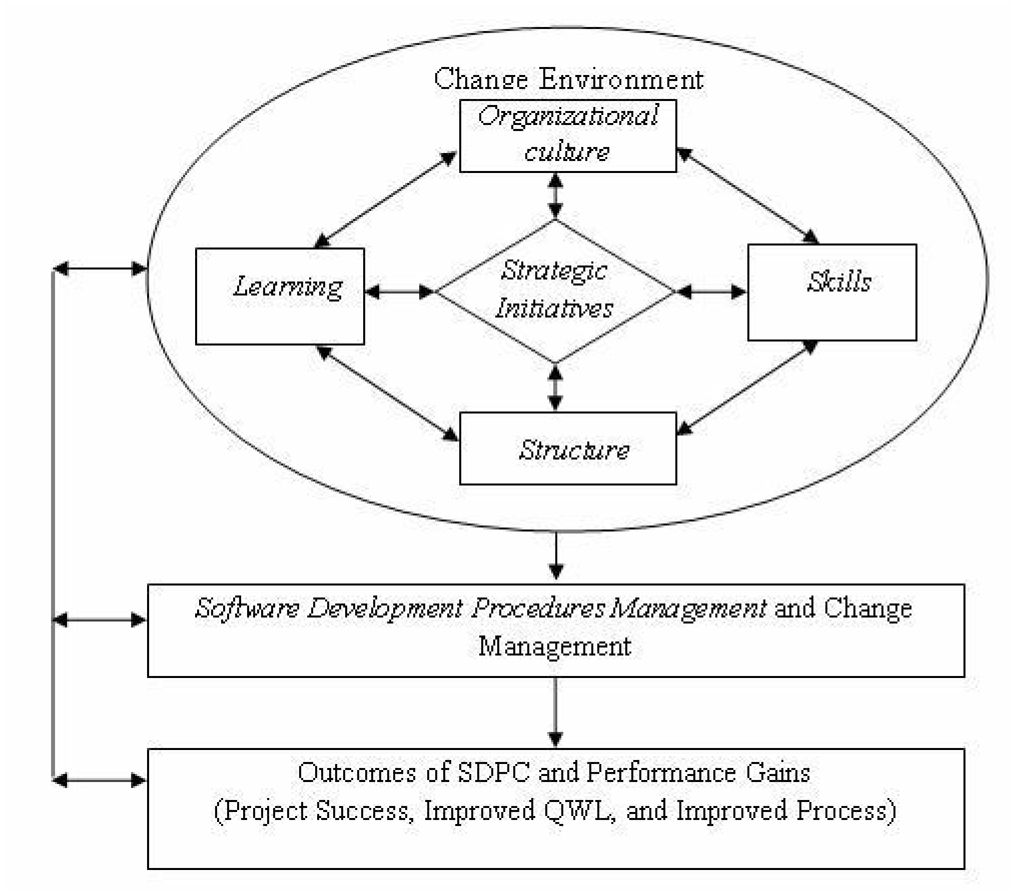
Change Environment
Organizational culture
Learning
Strategic Initiatives
Skills
Structure
Software Development Procedures Management and Change Management
Outcomes of SDPC and Performance Gains (Project Success, Improved QWL, and Improved Process)

Figure 1. Theoretical Framework of Software Development Process Change Management

Adapted from Kettinger and Grover (1995)

Note: Factors represented by italics are analyzed in this study

In order to make the Kettinger and Grover (1995) model more appropriate for studying SW process change management, we altered some of the original components. Two of the components in the original model, namely Relationship Balancing and Knowledge Capability were replaced by factors that were pertinent in SW development. In the original model Relationship balancing primarily pertains to the relationship with the partners in a company's value chain. In the present study's context, it is irrelevant. This study replaces the concept of relationship balancing with organization structure, which is more inwardly focused. Likewise knowledge capability is replaced by the skills of the people involved in the software development project. This due to the fact that changes in software development process requires both technical as well managerial skills to be learned by the employees. The factor representing "IT Leveragability" in the original model was assumed to be the common denominator in software development projects and hence it was omitted.

In the proposed SDPC management model, management strategy drives the need for change and this in turn influences other elements in the change environment. Management's vision in improving the success of software development process depends on the organizational culture, structure of the organization, work procedures, skills of the employees and their learning capability. This change environment, in turn, influences the change management practices adopted by the organization. Management may hire an external consultant or develop strategies on its own to impart the necessary skills, create procedures, organize the work functions and create a receptive culture.

Better managed change in the software development practices towards ASDM could improve productivity of the developers, increase the success rate of software development projects, and enhance the quality of the delivered product. Moreover the quality of the work life of the people involved in such projects can be improved.

Organizational change environment will facilitate the implementation of newer software development processes. These newer practices and the change management practice will lead to better software development outcomes on various dimensions.

Having presented an adapted theoretical framework for SDPC management framework, it would be germane to study its implications for practice. The next section discusses the managerial implications of the change management by using a framework that was proposed by Adler and Shenhar (1990) to analyze the organizational change phenomenon.

IMPLICATIONS FOR THE PRACTICE: SOFTWARE PROCESS CHANGE MANAGEMENT

As the previous section explained, an organization undertaking the change in software development process should pay attention to various organizational components such as (a) skills, (b) procedures, (c) structure, (d) strategy, and (e) culture. Depending on the magnitude of change, the above mentioned organizational components will have impact at varying levels. Adler and Shenhar (1990) studied technology and process change in various fields such as defense industries, and computer aided design/computer aided manufacturing and arrived at a hierarchy of these organizational assets. Their model is quite similar to the SDPC management presented earlier in this paper. The figure 2 depicts the framework proposed by Adler and Shenhar for analyzing the organizational change management (Adler et al. 1990). Two key characteristics determine the time taken to change the organizational technology / processes: level of learning required and magnitude of technology / process change.

Magnitude of technology / process change varies according to the existing conditions in the organizations. For example, consider two organizations that are interested in adopting Feature Driven Development (FDD) an ASDM that relies on object-oriented design and development. The first organization uses object-oriented design and development and the second organization uses structured software development methodology. An organization using object-oriented programming may have lesser trouble in migrating to FDD and the amount of learning and time spent and migration would be comparatively limited. For the organization utilizing structured programming languages / methodologies the employees may have to learn new programming as well as design techniques and hence the migration to FDD will be more learning intensive as well as time consuming. Likewise, an organization that has adopted team based software development approach may find it easier to adopt ASDM than an organization that relies on functional specialization.

Level of organizational learning required also determines the time to affect change in an organization that transitions to ASDM. As in the previous example, learning object-oriented programming language and design methods are at the skill level and it is easier to accomplish within a short span of time. Change in the organizational culture level is much more difficult to come by as it requires more fundamental change in the way the organization functions. Importance of learning has been highlighted elsewhere in the literature, Levine (2001), states that learning is key in any technology as well as process change management.

Skills

Adopting ASDM requires retooling of the employees' skill sets. According to Adler and Shenhar (1990), skills involve both technical as well as technological management skills. In the context of software development, technical skills pertain to the adoption of new analysis and design techniques, programming tools and languages, etc. ASDM has introduced many new innovative concepts which fall in the realms of technological management skills and some examples of them are: refactoring, reflection workshops, pair-programming, and continuous integration. Emphasis on test driven development in ASDM has given rise to new testing frameworks such as JUnits that help in automating the entire testing process. These skills are not usually needed in the traditional software development methodologies and any move to adopt ASDM requires them.

Apart from the changes in the technical skills required most of the ASDM emphasis on communication / collaboration to a great extent. These interpersonal skills greatly influence the functioning of the ASDM project. Learning news skills is relatively easier than changing the rest of the dimensions in the model. Few weeks of training can change the skill base of the employees depending on the magnitude of the change sought.

Procedures

Procedures adopted in ASDM differ from the traditional software development methodologies. Agile methods do not emphasize the need for extensive documentation and instead they emphasize on working software. Likewise, extensive planning is also not emphasized in ASDM. Procedures involved in gathering and prioritizing requirements, testing and integrating code, and inspecting code differ from the traditional software development methodologies.

Procedures also pertain to the planning and control of the projects (Adler et al. 1990). Various ASDMs use different approaches to do the planning and controlling of software development effort. Time-frame and the content of the plan vary according to the method used. For instance, ASD calls for the emergent plans and whereas DSDM calls for deliberate planning. These differences also come in the way in changing the work practices.

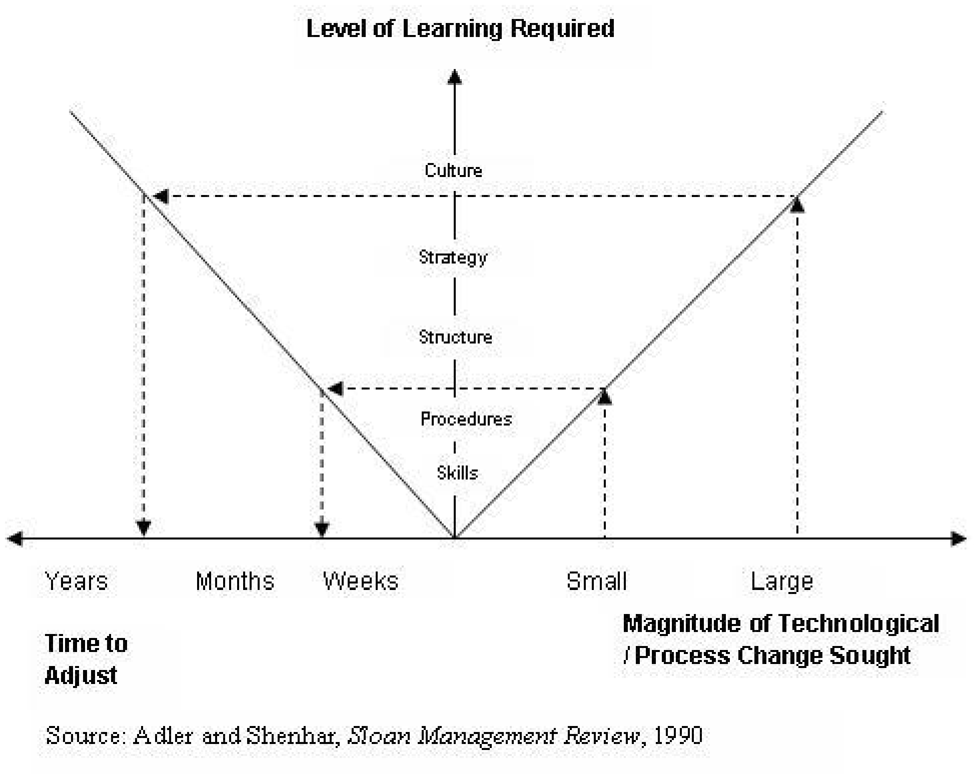
Level of Learning Required
Culture
Strategy
Structure
Procedures
Skills
Years
Months
Weeks
Small
Large
Time to Adjust
Magnitude of Technological / Process Change Sought
Source: Adler and Shenhar, Sloan Management Review, 1990

Figure 2. Framework for Software Development Process Change Management

Structure

Structure is an important aspect that needs attention during the transition to ASDM. According to Adler and Shenhar, the key dimensions in the design of organizational structures are (i) need to keep people together, and (ii) the need to collaborate across different functions.

ASDM addresses these issues in its own way. ASDM utilizes self-managed work teams that fosters employees involvement and participation in all the aspects related to work organization. There have been calls for the structure to emerge based on the requirements of the project and other situational conditions (Highsmith 2000). Self-managed work teams are claimed to be more effective to conventional teams / groups (Yeatts and Hyten 1998). Hence the change in the development team / group structure of existing software development groups becomes a key issue in the adoption of ASDM.

Strategy

Strategy of agile software development can be gleaned from the Agile Manifesto. Organizations need to change their strategies to suit these new requirements. ASDM stresses on customer orientation far beyond the traditional software development methodologies and goes to the extent of allowing customer representative in the development team. Moreover, strategies such as frequent product deliveries need more accommodation on the part of both the partners. Organizations can use the agile manifesto to redefine their organization strategies and it gives some kind of direction to the adopting organizations.

Strategies are generally enforced top-down and in an ASDM environment, participation of the all the stakeholders becomes increasingly important. Technology management strategies are also crucial in any change efforts (Adler et al. 1990). In a software development environment new tools and techniques are evolving and they have great deal of impact on the assimilation of the newer ASDMs. For instance strategies for acquiring and using automated test tools are important in many of the ASDMs. Likewise, strategies for addressing the issues related to bugs and refactoring may require technical strategies.

Culture

Though last in the Adler & Shenhar framework, an organization culture is the most difficult to change. Changing organization culture also requires the greatest amount of time. Every organization has its own unique culture. Organizational culture is deeply ingrained in an organization and they evolve over a period of time. Some of the prescriptions of ASDM may not be consistent with the prevailing organization culture and this requires effort in changing it. According to Schein, organizational culture consists of assumptions, values, and artifacts (Schein 1984).

Traditional software development methodology's fundamental assumption is that software development is an engineering field and we can plan and resolve any uncertainities in these projects. ASDM questions this fundamental assumption and it calls for adaptation. ASDM values customer participation in the software development effort whereas in traditional software development methodologies customers are outsiders to the project team. Likewise, ASDM emphasis on maintaining open communication lines in the project team. Some of the ASDM principles such as pair programming, reflection workshops require a culture where mutual trust and commitment are nurtured. The artifacts that are produced during the development of software are treated differently. In traditional software development methodologies, programming code is not easily discarded whereas, ASDM calls for discarding the code if it found to be not good. In the software development arena, (Ngwenyama et al. 2003) studied the issues related to organizational culture in the implementation of software process innovations (CMM).

CONCLUSION AND FUTURE RESEARCH DIRECTIONS

Agile software development methodologies seem to be a universal panacea for the problems plaguing the current software development environment. But caution is advised in the adoption of it without due regard to the managerial aspects involved in the change. As illustrated in this paper, management should emphasize on the various facets involved in an organizational change before they adopt ASDM. Toward this end we believe this paper has filled in a crucial void in the literature in the managerial aspects involved in the transition to ASDM.

This study has borrowed concepts from the BPR and organizational change management literature to analyze the impact of the change in software development work practices. Empirical validation of the SDPC framework proposed will greatly help organizations that transition from software development methodology to another methodology. In general, changing software development processes is not a well researched upon area in the IS research and this advent of ASDMs have ushered in new opportunities to study this phenomenon.

REFERENCES

1. Adler, P.S., and Shenhar, A. (1990) Adapting Your Technological Base - the Organizational Challenge, *Sloan Management Review*, 32, 1, 25-37
2. Berard, E.V. (1990) Understanding the Recursive/Parallel Life-Cycle, *Hotline of Object-Oriented Technology*, 10-13
3. Boehm, B. (1988) A Spiral Model of Software Development and Enhancement, *IEEE Computer*, 21, 5, 61-72
4. Charette, R. (2003) The Decision is In: Agile versus Heavy Methodologies, 2003, 28th Nov, Cutter Consortium Free Research Articles (2:19) http://www.cutter.com/freestuff/epmu0119.html
5. Fichman, R.G. (2001) The role of aggregation in the measurement of it-related organizational innovation, *MIS Quarterly*, 25, 4, 427
6. Fichman, R.G., and Kemerer, C.F. (1997) The assimilation of software process innovations: An organizational learning perspective., *Management Science*, 43, 10, 1345
7. Gilb, T. (1988) Principles of Software Engineering Management, Addison-Wesley, Boston, MA
8. Grover, V. (1999) From business reengineering to business process change management: A longitudinal study of trends and practices, *IEEE Transactions on Engineering Management*, 46, 1, 36-46
9. Hardgrave, B.C., Davis, F.D., and Riemenschneider, C.K. (2003) Investigating Determinants of Software Developers' Intentions to Follow Methodologies., *Journal of Management Information Systems*, 20, 1, 123
10. Highsmith, J. (2002) Agile Software Development Ecosystems, Addison Wesley, Boston, MA
11. Highsmith, J. "Agile Project Management: Principles and Tools," Cutter Consortium, Arlington, MA.

12. Highsmith, J.A. (2000) Adaptive software development : a collaborative approach to managing complex systems, Dorset House Pub., New York
13. Janz, B.D., and Wetherbe, J.C. (1997) Reengineering the systems development process: The link between autonomous teams and business process outcomes, *Journal of Management Information Systems*, 14, 1, 41
14. Kettinger, W.J., and Grover, V. (1995) Special Section: Toward a theory of business process change management, *Journal of Management Information Systems*, 12, 1, 9-30
15. Larman, C. (2003) Agile & iterative development: A manager's guide, Addison-Wesley Pub Co, Boston, MA
16. Leung, H. (2001) Organizational factors for successful management of software development, *Journal of Computer Information Systems*, 42, 2, 26
17. Levine, L. (2001) Integrating knowledge and processes in a learning organization, *Information Systems Management*, 21-33
18. Motwani, J. (2003) A business process change framework for examining lean manufacturing: a case study., *Industrial Management & Data Systems*, 103, 5, 339
19. Mustonen-Ollila, E., and Lyytinen, K. (2003) Why organizations adopt information system process innovations: a longitudinal study using Diffusion of Innovation theory, *Information Systems Journal*, 13, 3, 275-298
20. Ngwenyama, O., and Nielsen, P.A. (2003) Competing values in software process improvement: An assumption analysis of CMM from an organizational culture perspective, *IEEE Transactions on Engineering Management*, 50, 1, 100-112
21. Orlikowski, W.J. (1993) Case Tools as Organizational-Change - Investigating Incremental and Radical Changes in Systems-Development, *MIS Quarterly*, 17, 3, 309-340
22. Pruijt (1998) Multiple personalities: the case of business process reengineering, *Journal of Organizational Change Management*, 11, 3, 260-268
23. Royce, W.W. "Managing the Development of Large Software Systems," IEEE WESCON, 1970.
24. Schein, E.H. (1984) Coming to a New Awareness of Organizational Culture, *Sloan Management Review*, 25, 2, 3-16
25. Sharma, S., and Rai, A. (2003) An assessment of the relationship between ISD leadership characteristics and IS innovation adoption in organizations., *Information & Management*, 40, 5, 391
26. Sultan, F., and Chan, L. (2000) The Adoption of New Technology: The Case of Object-Oriented Computing in Software Companies., *IEEE Transactions on Engineering Management*, 47, 1, 106
27. The Standish Group "Chaos Report," The Standish Group, West Yarmouth, MA.
28. Yang, H.-L. (1999) Adoption and implementation of CASE tools in Taiwan., *Information & Management*, 35, 2, 89
29. Yeatts, D.E., and Hyten, C. (1998) High-performing self-managed work teams: A comparison of theory to practice, Sage Publications Inc., Thousand Oaks, CA