Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2009 Proceedings

Americas Conference on Information Systems (AMCIS)

2009

Building Security in during Information Systems Development

Vijay Raghavan
Northern Kentucky University, raghavan@nku.edu

Xiaoni Zhang
Northern Kentucky University, zhangx@nku.edu

Building Security in during Information Systems Development

Vijay V. Raghavan
Northern Kentucky University
raghavan@nku.edu

Xiaoni Zhang
Northern Kentucky University
Zhangx@nku.edu

ABSTRACT

There are many facets of managing security in information systems. Although there are prior studies that focus on how to build secure code from an architectural standpoint, an often overlooked aspect of security is the relationship between the systems development policies and procedures and the security of the systems developed. We focus on this relationship and draw from a general software quality model to provide a foundation for testing this relationship. This study discusses ideas that follow from prior research and develops a survey instrument for exploring the relationship between policies and procedures during systems development life cycle and the security quality of the system developed.

Keywords

Systems Development, Security, Systems Development Life Cycle.

INTRODUCTION

According to the Economist ("A Hidden Menace," 2004, p. 61) security accounts for about six to eight percent of the Information Technology (IT) budget in developed countries. Amidst this increased attention to security-related issues, the emphasis on software development security has often taken a back seat to other pressing considerations such as delivering a product within budget and within the promised time frame.

Security refers to the ability of a system to protect information and system resources with respect to confidentiality and integrity. Many industrial groups and companies have developed security tools to enforce security features during application development process. To an extent, it can be argued, that the awareness of developers to security considerations along with an organization's focus on delivering a secure product determines security quality of applications. There is evidence that bad coding practices create severe security problems (Vijayan, 2001). Being proactive during application development lifecycle is often considered the best approach to address security (see for example Howard & Lipner 2006).

Security must be designed and built into applications development life cycle (Coffee, 2004). Many of the security guidelines of regulations and standards can be bypassed. Unsophisticated software development techniques, a lack of security-focused quality assurance, and scarce security training for software developers, software architects, and project managers are often the unwitting culprits (Jones & Rastogi, 2004). It has been pointed out that one of the guiding principles of security management is to ensure that people understand their responsibility as well as their individual roles in establishing a secure information system (Dhillon & Backhouse, 2000). Developers are an important link in software security. And yet, it is very likely that some developers may not fully comprehend as to how the software they're building or maintaining could be exploited in future (Van Wyk & Steven, 2006). In essence, security problems are people problems. Organizations often prefer to allocate resources to other areas of software development as the marginal returns on security investments are not easily quantifiable. A major reason for this mistaken emphasis might be that a secure application does not display its virtues, as for example an elegant user-interface might.

Improving the capability of the system development process has emerged as an important strategy for addressing recurring problems in software development, such as poor quality, high development costs, and long delivery lead times (Ravichandran & Rai, 2003). The proliferation of web applications has introduced many security holes. Web application code is the major reason to make a web site vulnerable (Scott and Sharp 2002). Despite increased security interest in industries, research on application development security is sparse.

Therefore, developing an enhanced understanding of secured code or applications provides timely information to further our knowledge of software security. This study explores related issues in application development. It uses survey research to elicit organizational security practices during application development and to understand current state of security strategies in application development.

We draw from Ravichandran and Rai's (2003) model of general *software quality* and apply it to the particular domain of *security quality*. The main difference between the two is that the software quality is multi-dimensional while security quality is but one dimension of an overall software quality. This adoption is justified for two reasons: (1) the current study focuses not on security for the general information systems as a whole but addresses it from the limited perspective of security of the software developed, and (2) there is general support for transporting many of the constructs used in the software quality model to the security context as has been highlighted through out this paper. A contribution of this study is the inclusion of many items from the numerous checklist approaches within a formal model of security quality, thus combining streams of security research that has remained disparate thus far.

PRIOR STUDIES

There is a large body of literature on application development in general; however, few works focus on how software development procedures and policies can affect application security. In spite of this lack of extensive research in this area, there are some important findings available from previous research that provide the motivation for this study. As an example, previous research results show that the deployment of methodologies by IS developers is primarily associated with a hierarchical culture that is oriented toward security, order, and routinization (Iivari & Huisman, 2007).

Security

Secure applications capture user input accurately, perform business functions correctly, and resist application breaches. In practice, some applications do not enforce data validation; some do not function as expected; and others cannot secure data. Insufficient security features in application development may cause tremendously losses for companies. Software developers sometime form illusive trust assumptions and because of such assumptions software development efforts often do not address potential security consequences adequately (Viega et al., 2001). According to Computer Weekly (2004), common faults in coding such as lack of validation for user inputs, untested and inappropriate file calls leave security holes, and consequently, application vulnerabilities occur. Some security flaws are built into systems from the earliest stages of development due to insufficient awareness of security problems. During the development life cycle, it is quite often that security procedures and audits are ignored. Application developers focus more on functionality rather than the security of applications.

Prior research on security has provided useful results. For example, Nabi (2005) examines ecommerce security and suggests strategies for secure business application logic: good design and engineering, secured configuration, defensive programming and secured wrappers for server-side software. Adams and Blandford (2005) advocate the understanding of *communities of practice* to enforce security and privacy issues within organizations. Among practitioners, Wang and Wang (2003) discuss security and quality issues related to software development and identify security risks and discuss the impact of security risks on quality factors.

Training

Due to the rapid development of technologies and increasing dynamics of the business environment, improving security requires constant training and vigilance of developers (Fisher, 2003). Training is a learning process which can help developers to be sensitized to the potential impact of security problems on organizations at large. Educating developers on the need for good coding practices can result in good coding habits that improve security. Training has been shown to improve general software quality (Subramanian, Jiang & Klein, 2007; Parzinger & Nath, 2000). A recent study found that the use of hypermedia, multimedia and hypertext as training materials increased the information security awareness among the three awareness levels in an online training environment (Shaw et. al., 2009). Thus, it can be argued that *training* included in the software quality model can be equally applicable in the software security context as well. Microsoft develops security-related on line training materials for its developers through a formalized process. In its experience, although more people attend on-line training face-to-face training allows greater opportunity to answer security related questions by developers (Howard & Lipner, 2006).

Management Support

Strategic leadership theory suggests that the CEOs' decisions and behavior are likely to explain organizational outcomes (Boeker, 1992; Allen & Panian, 1982). Prior works emphasize that top management leadership is an important and critical factor in quality improvement (Deming 1986; Schoenberger 1984). Top management's commitment to security can be showed in different forms: vision, mission and value. In addition, transformational leadership theory addresses the importance of leaders' rapport with followers and the need to instill their values into followers. This theory posits that leaders have the ability to transform their followers and encourage them to accomplish desired tasks. It is important for leaders to

motivate followers to perform in excess of expectations (Yukl & Van Fleet, 1992; Bass, 1991). Other studies also found that top management's order and mandates lead to lead to improved quality performance (Anderson et al. 1995; Flynn et al., 1995). Senior management play several roles: visionary, transformational leadership has four dimensions: charisma, inspiration, intellectual stimulation, and individualized consideration. Through these four dimensions, leaders create profound impact on their followers (Yammarino et al., 1997). If the top level management sponsors and values the ideas of security, developers would consider security is the expectations of them and make efforts to actively implement security practices.

Our study transplants the leadership theory to security practices in application development. This theory suggests that by making followers more aware of the importance and value of security, the participants in the system development process will be more responsive to the potential vulnerability and risks associated with application development. If management desires and demand clean and secure code and also provide incentives for application developers, security will be greatly improved.

Security Policy

The protection of information systems is a critical problem faced by organizations. The application of a security policy is of utmost importance for managing the security of information systems. There are many factors affecting implementation of a successful security policy in an organization. Karyda et al. (2005) explore the processes of formulating, implementing and adopting a security policy in two different organizations and propose a theoretical framework based on the theory of contextualism. Each organization has its own characteristics and security policy is context specific. They highlight the dynamic nature of the application of security policies and bring forth contextual factors that affect their successful adoption.

Development Process Control

Application development goes through analysis, design, development and maintenance stages. Effective process control leads to better software quality and process efficiency (Ravichandran & Rai, 2000). Organizations rely on their employees to follow the rules they establish. Especially in security implementation such rule adherence is critical to the security quality of the software developed. Research has shown that the deployment of methodologies by IS developers is primarily associated with a hierarchical culture that is oriented toward security, order, and routinization (Iivari & Huisman, 2007). There also have been significant studies dealing with the ability of the organizations to regulate employee conduct in general (Tyler & Blader 2005). Security should be built from ground up and emphasize throughout application development life cycle. A life cycle process emphasizing security assurance at each phrase is necessary to improve the overall security of applications (Gilliam et al., 2003). The extent to which the established security standards are being followed in the actual development of information system is an important contributor to the overall security quality. While there are numerous process control checklists that are widely available, as a recent computerworld article points out (Hayes, 2009), these lists alone will not solve the problems of security.

Attitude

Consumer behavior literature suggests that attitude affects behavior (Ajzen and Fishbein). This attitudes behavior model has been adopted to IT field and studied extensively in Technology Acceptance Model (David 1996; Agarwal and Prasad, 1999). It is commonly believed that for effective security, users have to make a conscious decision to comply with the organization's security policies and adopt computer security behavior (Ng, Kankanhalli & Xu 2009). Previous research has shown that how certain attitudinal dimensions such as morality can influence certain computer-related behavior (Gattiker & Kelly, 1999). The premise for including attitude in our model is that during application development, developers may have different attitude and value towards security procedures and practices. In the context of adoption of software process innovations it has been shown that how perceptions of productivity and quality benefits can explain how developers perceive the usefulness of software process innovations which in turn explain some variance SPI use. The SPIs must be perceived as useful to a developer for it to be adopted during the software development process (Green, Hevner & Webb Collins, 2005). The present study adds these notions of developer perceptions of security measures to the security quality model. We measure the attitude directly by using a scale that is widely recognized and adopted.

Perceived Security

Perceived security is being measured by a newly developed multi-item measure that includes the need for revisions due to security testing, vulnerability to exploitation, the extent to which the final product is expected to comply with security standards and the overall confidence in the security of the system. Measuring and analyzing customer or user satisfaction is the ultimate validation of general software quality (Kan, 2003). Hence it was deemed appropriate to measure how well the

ultimate users or customers are satisfied with the security quality of the system. Software is often developed by an outsourced firm for the use of a customer organization. We can then expect the developing organization to have knowledge of any complaints that customers may have about the security aspects of the system.

RESEARCH MODEL

Our research model is based on Ravichandran and Rai's (2000) general model of software quality performance. Their model identifies critical organizational levers that IS managers can use in their efforts to improve software quality performance as well as employee specific factors that contribute to software development quality. Our model is presented in Figure 1 and incorporates elements of leadership, structure, process and outcome with constructs specific to the software security implementation.

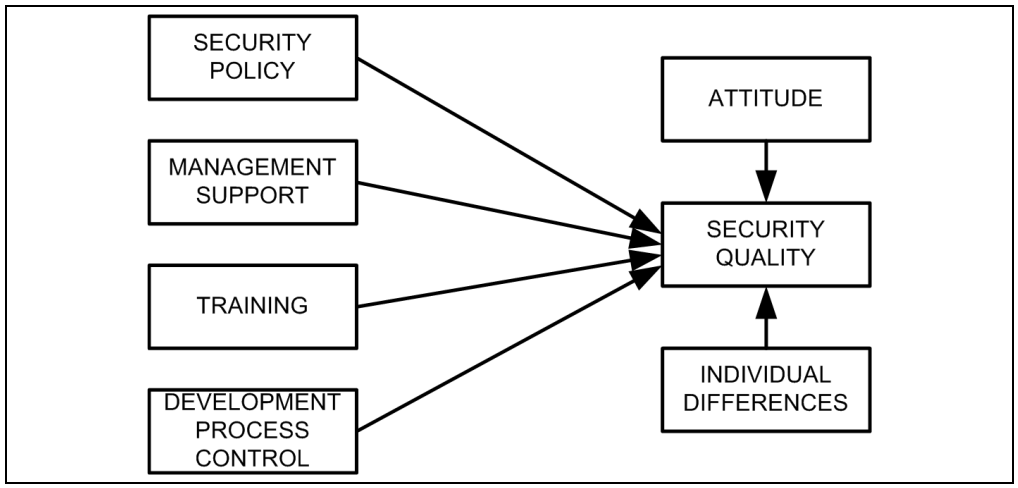

Figure 1. A Model of Implementing Software Security During Systems Development

HYPOTHESES

A greater rigor in the specification of security policies will result in higher perceived security quality of the system.

A greater management support to security implementation during the software development process will result in a higher the perceived security quality of the system.

A greater emphasis on the security-related training of the development personnel will result in a higher the perceived security quality of the system.

A greater rigor in the control during the development process will result in a higher perceived security quality of the system.

A positive attitude to enforcing security standards will result in greater perceived security quality of the system.

MEASURES

Our measures intend to capture how companies develop and enforce secure coding practices, self-assess code during development, implement security checks into the quality assurance cycle and consider security during change control. We adopted prior measures for the constructs in our model. Security policy, management support, training, perceived security quality is adopted from Ravichandran's work (2000). A seven point Likert scale is used to measure the constructs. Security index is measured with the checklist of security audits. The list of questions related to security practice is extracted from trade journals and sample list of professional IT audits.

Experience is adapted from Igbaria et al.'s work (1995). Igbaria et al. asked respondents to indicate the extent of their experience in using five different generic types of computer software. Because our research questions are focused on security issues, and our respondents are professional application developers, our measures intend to capture the number of years of programming and application development and the number of years of security education.

To measure software development process control, we used a checklist of questions developed based on an IT auditing list. Our check list of questions taps into the following dimensions: analysis, design, development, implementation and maintenance. In addition, we included selected questions from professional IT auditing checklist for assessing security practice. This is measured as a formative construct since it contains items that describe rather than define the construct (Petter, Straub & Rai, 2007).

We have developed a new measure of perceived security quality from the software development professionals' perspective. Although this can evoke an immediate concern of a measurement bias, a careful examination of the context will mitigate these concerns. First, we recognize that software development is often a group effort. An individual developer is less likely to see herself as the sole responsible person for providing a secure system. Hence an expectation of unbiased responses is not unreasonable. Second, the alternative of measuring this from the end users is not appropriate as the end users as a group is unlikely to be aware all security holes in the product delivered; the security concerns more readily comes to the attention of development team.

METHODOLOGY

Data will be collected from developers and project managers in the Midwest area. The participating organizations are from different industries with a wide ranging degree of security needs. A mailing list has been compiled and the survey instrument is currently being reviewed by a review board. Since we are using previously established measures in our instrument, there is less of a need for extensive pilot testing.

CONCLUSION

If the hypotheses of our study are confirmed as expected, we can begin to address concerns of lack of security in the software products at many levels. Organizations can focus on executive leadership, project structure and process areas of improving security by fostering a secure application development culture. If the need of training is validated, universities can develop more appropriate curriculum measures to improve security awareness as there is a general consensus that training on security awareness should start from the college level. Currently, there is a gap between the emphasis on teaching programming security and its need in the workplace. However, universities are becoming increasingly involved in developing security-related courses. To some extent, security can be embedded into every course in information system (Goodwin, 2003). Undergraduate curriculum should cover such security contents as: analyzing the security of code, model threats and vulnerabilities, fix programs, and the differences between secure and insecure programming languages. By doing so, after students join the labor force, they are better trained with security concepts, procedures, coding practices.

From the vendors and systems developers' perspective, they should deliver good, secure, and clean computer applications to their customers. From application users' perspective, there is need to learn to protect them from bad code and put code-quality requirements into their software contracts (Rapoza, 2005).

SYSTEMS-DEVELOPMENT SECURITY INSTRUMENT

IS Management support for security	Strongly Agree						Strongly Disagree	
1. IS chief executive assumes responsibility for security.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
2. IS chief executive is evaluated for security performance.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
3. IS chief executive supports security quality improvement process.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
Security policy and goals								
4. IS management has clear security quality and goals.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]

5.	Quality goals within IS are very specific.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
6.	Security importance is attached to quality in relation to cost and schedule objectives.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
Commitment to security skill development (Training)									
7.	Regular training in security management tools and techniques are given to IS professionals.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
8.	Team building and group dynamic training are given to IS personnel.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
9.	Business skill training is given to IS personnel.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
10.	Resources are made available for training IS personnel.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
Development process control									
11.	Security Performance standards have been established for design.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
12.	Security Performance standards have established for programming.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
13.	Security Performance standards have established for testing.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
14.	Security code review is performed regularly. (Fisher 2003)	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
Attitude									
15.	Enforcing security during development is good.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
16.	Enforcing security during development is useful.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
17.	Enforcing security during development is beneficial to the company.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
18.	Enforcing security during development is beneficial to me.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
19.	Enforcing security during development is valuable.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
Perceived Security Quality									
20.	The software produced/maintained is susceptible for security exploitation.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]
21.	The software produced/maintained will pass most security standards.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[NA]

22. The system will need revisions if it was subjected to a strict security testing. [1] [2] [3] [4] [5] [6] [7] [NA]
23. As a development professional or manager involved in this project, I have full confidence in the security of this system. [1] [2] [3] [4] [5] [6] [7] [NA]
24. I am aware that users/customers are generally satisfied with the SECURITY aspects of the system. [1] [2] [3] [4] [5] [6] [7] [NA]
25. Users perceived that the system meet the intended security requirements.
26. Users are satisfied with the overall security of the system.

Development Process Control (from Checklist)

27. Secure coding practices are enforced during application development. [1] [2] [3] [4] [5] [6] [7] [NA]
28. The developers self-assess code during development for security compliance. [1] [2] [3] [4] [5] [6] [7] [NA]
29. I believe that our development organization implement security checks into the quality assurance cycle. [1] [2] [3] [4] [5] [6] [7] [NA]
30. I believe that our development organization considers security when changes are made to the code (change control). [1] [2] [3] [4] [5] [6] [7] [NA]
31. I believe that our organization has created security-related development standards. [1] [2] [3] [4] [5] [6] [7] [NA]
32. I believe that user-input is validated in the code related to this project. [1] [2] [3] [4] [5] [6] [7] [NA]
33. I believe that authentication credentials in this system are encrypted. [1] [2] [3] [4] [5] [6] [7] [NA]
34. We perform peer review of code. [1] [2] [3] [4] [5] [6] [7] [NA]
35. We allow developers to manage the change control process. [1] [2] [3] [4] [5] [6] [7] [NA]
36. We keep developers in the dark about security practices. [1] [2] [3] [4] [5] [6] [7] [NA]
37. We conduct vulnerability testing. [1] [2] [3] [4] [5] [6] [7] [NA]
38. We use security tools to enforcing security for application development [1] [2] [3] [4] [5] [6] [7] [NA]
39. Age:
40. Gender:
41. Years of programming experience:
42. Type of projects: ☐ in-house ☐ outsourcing
43. Size of the projects:
44. Application type: Web-based? ☐ yes ☐ no

REFERENCES

1. A hidden menace (2004 June 5th), Economist, 371, 8378, 61.
2. Adams, A. and Blandford, A. (2005) Bridging the gap between organizational and user perspectives of security in the clinical domain, *International Journal of Human-Computer Studies*, 63, 2, 175-202.
3. Agarwal, R. and Prasad, J. (1999) Are individual differences germane to the acceptance of new information technologies?, *Decision Sciences*, 30, 2, 361-391.
4. Allen, M. P. and Panian, S. (1982) Power, performance and succession in the large corporation, *Administrative Science Quarterly*, 27, 538-547.
5. Anderson, J. C. and Gerbing, D. W. (1992) Assumptions and comparative strengths of the two-step approach, *Sociological Methods & Research*, 20, 3, 321.
6. Anderson, J. C., Rungtusanthanam, M., Schroeder, R. and Devaraj, S. (1995) A path analytic model of a theory of quality management underlying the Deming management method: Preliminary empirical findings, *Decision Sciences*, 26, 5, 637-658.
7. Barki, H. and Hartwick, J. (2001) Interpersonal conflict and its management in information system development, *MIS Quarterly*, 25, 2, 195-228.
8. Bass, B. M. (1985) Leadership and performance beyond expectations, Free Press, New York.
9. Boeker, W. (1992) Power and managerial dismissal: Scapegoating at the top, *Administrative Science Quarterly*, 27, 538-547.
10. Coffee, P. (2004) Security is a moving target, *eWeek,* 21, 50, D1.
11. Coffee, P. (2005) Developers' growing challenge, *eWeek,* 22, 20, D1-D8.
12. Davis, F.D. 1993. User acceptance of information technology: system characteristics, user perceptions and behavioral impacts. *International Journal of Man-Machine Studies*, 38,3, 475-487.
13. Deming, E. (1986) Out of the crisis, MIT Center for Advanced Engineering, Cambridge.
14. Dhillon, G., & Backhouse, J. (2000). Information system security management in the new millennium. *Communications of the ACM, 43*, 7, 125-128.
15. de Paula, R., Ding, X., Dourish, P., Nies, K., Pillet, B., Redmiles, D. F., Ren, J., Rode, J. A. and Silva Filho, R. (2005) In the eye of the beholder: A visualization-based approach to information system security, *International Journal of Human-Computer Studies,* 63, 1/2, 5-24.
16. Fisher, D. (2003) New security rules to raise Windows, *eWeek,* 20, 1, 1.
17. Flynn, B., Schroeder, R. G. and Sakakibara, S. (1995) The impact of quality management practices on performance and competitive advantage, *Decision Sciences*, 26, 5, 659-692.
18. Gattiker, U. E., & Kelley, H. (1999). Morality and computers: Attitudes and differences in moral judgments. *Information Systems Research*, 10, 3, 233.
19. Gilliam, D. P., Wolfe, T. L., Sherif, J. S., & Bishop, M. (2003). Software security checklist for the software life cycle. Paper presented at the *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on,* 243-248.
20. Goodwin, B. (2003) Secure coding course, *Computer Weekly*, 3.
21. Green, G. C., Hevner, A. R., & Webb Collins, R. (2005). The impacts of quality and productivity perceptions on the use of software process improvement innovations. *Information and Software Technology, 47*, 8, 543-553.
22. Hayes, F. (2009) More than a list, Computerworld, 1/19/2009, 43, 3, p40-40, 1p
23. Huang, Y. W., Tsai, C. H., Lin, T. P., Huang, S. K., et al. (2005) A testing framework for web application security assessment, *Computer Networks*, 48, 5, 739.

24. Howard, M., & Lipner, S. (2006). *The security development lifecycle : SDL, a process for developing demonstrably more secure software*. Redmond, Wash: Microsoft Press.
25. Igbaria, M., Guimaraes, T. and Davis, G. (1995) Testing the determinants of microcomputer usage via a structural equation model, *Journal of Management Information Systems*, 11, 4, 87-114.
26. Iivari, J., & Huisman, M. (2007). The relationship between organizational culture and the deployment of systems development methodologies. MIS Quarterly, 31, 1, 35-58.
27. Jones, R. L. and Rastogi, A. (2004) Secure coding: Building security into the software development life cycle, *Information Security Journal: A Global Perspective*, 13, 5, 29-39.
28. Kan, S. H. (2003). *Metrics and models in software quality engineering* (2nd ed.). Boston: Addison-Wesley.
29. Karyda, M., Kiountouzis, E., Kokolakis, S. (2005) Information systems security policies: A contextual perspective, *Computers & Security,* 24, 3, 246.
30. King, S. (2004) Bridging the gap between security and developers, Computer Weekly, 32.
31. Kumar, R. L., Park, S., & Subramaniam, C. (2008). Understanding the value of countermeasure portfolios in information systems security. *Journal of Management Information Systems*, 25, 2, 241-279.
32. Iivari, J., & Huisman, M. (2007). The relationship between organizational culture and the deployment of systems development methodologies. MIS Quarterly, 31(1), 35-58.
33. Nabi, F. (2005) Secure business application logic for e-commerce systems, *Computers & Security*, 24, 3, 208.
34. Ng, B., Kankanhalli, A., & Xu, Y. (2009). Studying users' computer security behavior: A health belief perspective. *Decision Support Systems*, 46, 4, 815-825.
35. Petter, S., Straub, D., & Rai, A. (2007). Specifying formative constructs in information systems research. *MIS Quarterly, 31*, 4, 623-656.
36. Rapoza, J. (2005) Say 'no' to bad code, *eWeek*, 22, 2, 41-41.
37. Ravichandran, T. and Rai, A. (2000) Quality management in system development: An organizational system perspective, *MIS Quarterly*, 24, 3, 381-415.
38. Ravichandran, T. and Rai, A. (2003) Structural analysis of the impact of knowledge creation and knowledge embedding on software process capability, *IEEE Transactions on Engineering Management*, 50, 3, 270-284.
39. Shaw, R. S., Chen, C. C., Harris, A. L., & Huang, H. (2009). The impact of information richness on information security awareness training effectiveness. *Computers & Education, 52*, 1, 92-100.
40. Subramanian, G. H., Jiang, J. J., & Klein, G. (2007). Software quality and IS project performance improvements from software development process maturity and IS implementation strategies. *Journal of Systems & Software, 80*, 4, 616-627.
41. Tyler,T.R., & Blader,S.L. (2005). Can businesses effectively regulate employee conduct? the antecedents of rule following in work settings. *Academy of Management Journal, 48*, 6, 1143-1158.
42. van Wyk, K. R., & Steven, J. (2006). Essential factors for successful software security awareness training. *Security & Privacy, IEEE, 4*, 5, 80-83.
43. Wang, H., & Chen Wang. (2003). Taxonomy of security considerations and software quality. *Communications of the ACM, 46*, 6, 75-78.