A NEW PARADIGM FOR ADDING SECURITY INTO IS DEVELOPMENT METHODS

MIKKO SIPONEN[a] AND RICHARD BASKERVILLE[b]

[a] *University of Oulu, Department of Information Processing Science, P. 0. BOX 3000, 90014 Oulu Oulu, Finland, Mikko. T.siponen@oulu.fi*

[B]*Georgia State University, 35 Broad Street, Atlanta, Georgia 30031, baskerville@gsu.edu*

Abstract: Information system (IS) development methods pay little attention to security aspects. Consequently, several alternative approaches for designing and managing secure information systems (SIS) have been proposed. However, many of these approaches have shortcomings. These approaches lack fully comprehensive modeling schemes in terms of security, i.e. no single method covers all modeling needs. Rarely can these approaches be integrated into existing IS development methods. Also, these approaches do not facilitate the autonomy of developers. This paper describes a framework that helps us understand the fundamental barriers preventing the alternative SIS design approaches from more effectively addressing these shortcomings. This framework is illustrated with an example of a framework-based solution: meta-notation for adding security into IS development methods. Future research questions and implications for research and practice are presented.

Key words: IS security

1. INTRODUCTION

IS and computer science literatures are rife with the importance of security considerations for IS organizations (e.g. Anderson, 1999;

Backhouse & Dhillon, 2001; Straub & Welke, 1998). The exploding use of the Internet by different organizations and the general public has led to 'security' becoming a recognised public buzzword. As a result, a huge number of technical solutions exist in the area of computer and communication security. However, technical solutions do not provide much help from the viewpoint of IS and Management Information Systems (MIS); for example, such solutions do not help to design or manage secure IS (Baskerville, 1989; Sandhu & Thomas, 1994; Baskerville, 1993; Backhouse & Dhillon, 2001). Furthermore, the methods for developing IS do not give much help with respect to security issues (Baskerville, 1993; Dhillon & Backhouse, 2001). Various security design approaches (ranging from simple checklists to more advanced approaches modified from IS and software development methods) have been expounded by practitioners and researchers. Table 1 summarizes the existing approaches/paradigms for design and managing secure IS.

Table 1. The different approaches for information security management IS security design.

The different paradigms	Examples/advocates	Key ideas/arguments
Mainstream Approaches		
Checklists/standard	BS 7799 (1993); GASSP (1999); ITSEC (1990); OECD (1996); Wood et al. (1987)	List the ideal protection means that organizations should implement
Risk Management	Wong (1977); Custance (1996); Bennett & Kailay (1992); Freeman et al. (1997); Spruit & Samwel (1999) Jung et al. (1999)	Calculate occurrence of risks and the justification of controls
Formal development	Anderson (1993); Barnes (1998)	Formalization is key for achieving security.
Self-reflected security management cookbooks	Parker (1981; 1998); Perry (1985); Schwaitzer (1984); Warman (1993).	Each author presents his/her own principles for security management. The use of risk management and checklists-orientation are common to these approaches
Integrative Approaches		
Information/Data Base modeling approaches	Smith (1989); Ellmer et al. (1995); Pernul et al. (1998)	Modeling notations for expressing security constraints based on structural and object-oriented paradigms
Responsibility approaches	Dhillon & Backhouse (1996); McDermott & Fox (1999)	Identification of workers' role responsibilities is key for designing security in organizations

Business process	Herrmann & Pernul (1999), Röhm et al. (1998); Röhm and Pernul (1998; 1999)	Add security constraints in the business processes
The security-modified IS development approaches	Baskerville (1998); Booysen & Eloff (1995); Hitchings (1995, 1996); James (1996)	These security approaches are influenced by different IS development methods

For a closer look at these approaches, readers are referred to (Baskerville, 1993; Dhillon & Backhouse, 2001; Siponen, 2001).

Barriers to Mainstream Approaches

The mainstream methods include checklists/standards, risk management, formal development and self-reflected cookbooks. Checklists/standards (Eloff & Solms, 2000; Fitzgerald, 1995; Janczewski, 2000; Solms, 1997, 1998, 1999) - have at least three fundamental roadblocks that are problematic.

First and foremost, the driven IS security development paradigm of checklists/standards methods have been based on "what can be done" by means of available solutions (Baskerville, 1993). In the case of checklists, this means that the unique needs and problems of organizations are overlooked (e.g. Baskerville, 1988; 1993). The needs of organizations are replaced by ideal protection techniques based on the intuitions of security gurus', and organizations are required to follow these as "a gift from the Heaven". Furthermore, when practitioners applying Checklists/standards are confronted with any management decision-making questions, they have to play it by ear. For these reasons, the real developmental or management support of these methods remains very weak.

Secondly, checklists/standards, risk management and formal development, being stand-alone/separated methods, are confronted with the problem of developmental duality, meaning that security and IS developments are separate activities having conflicting requirements (Baskerville, 1994).

Thirdly, these techniques (checklists and formal method development) are necessarily mechanistic and functionalistic and therefore conflict with the modern view of the social nature of organizations (Dhillon & Backhouse, 2001). In the case of checklists, this is perhaps due to the fact that the mainstream academic research on security is focused on technical matters, and since checklists simply attempt to reflect state-of-the-art research, the help checklists offer remains very mechanistic. Formal methods, owing to the engineering viewpoint of development, focus only on technical aspects of development. Also several books on security management exist, presenting less systematic cookbook approaches, based on the author's own personal experiences and speculations. It is typical for these books that the authors are highly self-reflective, i.e. they do not bother to take into account

what others have done in the field nor have they carried out empirical studies (hence, the label self-reflected security management cookbooks).

Blocked by these barriers, these approaches have had only limited success for designing and managing secure IS (Dhillon & Backhouse, 2001).

Barriers to Integrative Approaches

To avoid the shortcomings of these mainstream methods (checklists, risk management, formalization and cookbooks), more integrative approaches have been developed that integrate security design more closely with the social organization, the essential information system design, or the organizational goals. These approaches do not seem to have received wide attention. In Table 1, these approaches are classified in terms of different paradigms: information modeling, responsibility, business process, the security-modified IS development approaches, according to (Siponen, 2001). Even though these advanced approaches improved security management/design considerably (e.g. paying attention to organizations' security requirements), they engender their own set of barriers.

Firstly, these approaches lack a comprehensive modeling support in terms of security (Siponen, 2001). In other words, three levels of modeling/abstraction for an IS is widely recognized: organizational, conceptual and technical level (Iivari & Koskela, 1987; Iivari, 1989; Lyytinen, 1987). The different IS security approaches cover the different levels of IS, but no single method provide a comprehensive modeling support (e.g. for all three levels: organizational, conceptual and technical level).

Secondly, the existing approaches apart from Baskerville (1988; 1989), Booysen & Eloff (1995); James (1996) and McDermott & Fox (1999), are difficult – even impossible - to integrate into IS or software development process. This results in the problem of developmental duality (cf. Baskerville, 1992). Developmental duality is a fundamental conflict between the functionality (designed into the basic information system by business systems analysts) with the security (designed and added by security systems analysts).

Thirdly, the existing approaches restrict the autonomy of developers to use the approaches they prefer. Developers are increasingly recognized for their practice of using a "toolkit" of methods and method fragments, and selecting these according to the situation (Kumar & Welke, 1992). They may choose universal modeling for one project, and extreme programming for the next, according to the problem setting. If developers want to address security aspects into IS development using the existing security methods, they are not only forced to abandon their existing IS development methods, but also their practice of autonomously selecting the methods they desire.

Fourthly, IS methods are known to be emergent (Truex et al., 1999). New methods spring up every now and then, and methods are never really executed in practice exactly the same way (Truex et al., 2000). It is difficult to predict this emergence, and to allow for a universal security method that will match every development method and its permutations. Our goal should be to integrate security into each (existing, forthcoming and unpredictable) IS development method. However given the trend of current security methods, security approaches would, at best, always come a few steps behind IS development methods. The more rapidly the development evolution, the farther behind are the security approaches.

A New Paradigm: Overcoming the Barriers

From the problem framework above, it becomes clear that a meta-level viewpoint could provide one solution. Rather than present yet another new method with its own novel security features, we propose that security approaches must rise a level of abstraction above the barriers. Moving a level away from methodology takes us into the realm of meta-methodology, which will help developers use and modify our existing methods as needed.

Meta-methodology is not a well-explored area. Much of the work to date has focussed on method engineering (Brinkkemper, et al 1996; Kumar & Welke, 1992), and formalisms to support computer-aided method engineering (Odell, 1996). Such computer-based meta-methods essentially provide the means for rapidly (almost instantly) developing computer-aided systems analysis and software engineering (CASA/CASE) tools to match development needs and settings. While the security imperative for such meta-methods has been declared (Baskerville, 1996), there is little formal work in security meta-methodology.

Developing a full security meta-methodology is beyond the scope of a single paper. However, to illustrate the feasibility and the need for the solution that proceeds from this framework, a method fragment will suffice. This paper will describe a meta-notation, a key feature of most methods and meta-methods. The remainder of this study is organized as follows. The second section discusses the background of the meta-notation. In the third section, the meta-notation is explicated along with an exemplification. The fourth section is a discussion of the issues raised, and it is followed by a conclusion where the contributions of the paper are summarized.

2. PATTERNS IN IS DEVELOPMENT AND SECURITY METHOD NOTATION

Moving to a meta-method level of abstraction yields a perspective of information systems design methods that are in a constant state of emergence and change. No systems problem setting is an exact repeat of a former setting, and no method can actually be applied exactly the same way every time. This thesis is well known in practice (Jaaksi, 1998), and has been shown empirically and logically in academe (Truex et al., 2000). Neither however, are systems problem settings and development approaches total chaos and relativism. Otherwise, practical and empirical experience would have no value. Instead developers recognize regularities or patterns in the way problem settings arise and methods emerge. A comprehensive meta-methodology must apprehend a sufficient range of these patterns in order to be useful.

SUBJECTS AND OBJECTS

In seeking patterns in systems development methods, notation systems quickly rise in their apparent regularity across methods and problem settings. Traditionally IS development methods help developers to understand and model reality with respect to the system-to-be-built. In order to do such analysis and modeling, developers need to agree on how to describe things. Traditionally things are classified as 'entities' (e.g. structural methods) or 'objects' (e.g. object-oriented methods) in the IS literature. Additionally, some approaches uses 'Actors' (e.g. UML type of use case) or 'stickfigures' (e.g. Checkland's rich picture) at the organizational level (e.g. requirements analysis stage) as a unit of analysis. In our schema, the 'entities' or 'objects' are classified as security subjects and objects. As a result, these notational dimensions can, in principle, be added into any of the many notational patterns that use entities or objects as units of analysis. For example, both the structural and object-oriented forms of information engineering methods (cf. Lyytinen, 1987) are good candidates for the addition of such security notation.

The terms security subjects and objects are commonly used in database security literature to describe computer access control policies and models (e.g. Castano et al. 1995; Foley, 1991; McLean, 1990; Sandhu, 1993; Summers, 1997). Common security classification schemes are in turn influenced by these computer access control policies and models from the field of database security.

CONSTRAINTS

In the security literature, there is wide agreement on three security requirements (Parker, 1998): confidentiality or non-disclosure prevent/detect improper disclosure of information), integrity (information should not be modified by unauthorized personnel), availability (information should only be available to authorized personnel). In addition to these three requirements, the need for "non-repudiation" has been recently recognized. Non-repudiation is important for enabling financial transactions and data interchange key to electronic commerce. It is needed to make valid large-scale contracts between vendors and customers. Non-repudiation means that a contracting party cannot subsequently deny their actions. These four security requirements are called security constraints. While some computer security researchers, particularly cryptographers, distinguish other security requirements (see e.g. Menezes et al., 1997), the four described here are sufficient for our purposes to capture most of the common patterns found in security and systems development. Additional requirements can certainly be added to the meta-notation on an ad-hoc basis as required.

3. META-NOTATION SEMANTICS

The meta-notation includes five dimensions: security subjects, security objects, security constraints, security classifications, and security policy. The security classification (e.g., "secret," "top secret" etc.) can be regarded as more optional than the other dimensions (see below). Security subjects denote the different security relevant entities, i.e. entities that have a relevant security connection to the assets of the organization (security objects). Security subjects may include employees of the organization, business partners and third-parties. Actors in use cases and stickfigures in rich pictures, given that they have a security relevant connection to the assets of the organizations, are security subjects. The term security objects refers to the assets of the organization that are of relevance in terms of information security. Such assets (security objects) may range from physical things such as paper to electronic entities such as files. Security constraints include confidentiality, integrity, availability and non-repudiation. In order to define the kind of access (e.g. read, write, etc.) to objects that subjects need, a security classification of the security objects and security subjects may be relevant. If this is the case, subjects are classified according to their security sensitivity/need. Herein security policy dimension defines other constraints that may apply to security subjects (see the example below).

3.1 An example of the use of meta-notation

In the following example, the meta-notation (five security dimensions: security subjects, security objects, security constraints, security classifications, and security policy) is applied into a use case notation. Most objectsriented methods employ 'use cases' in requirements analysis phase to collect and analyze requirements (Jaaksi, 1998). Use case demonstrates what a user should be able to do with the system; use cases are graphical or textual presentation of the usage of system (Jaaksi, 1998). "when a user uses the system, she or he will perform a behaviorally related sequence of transactions in a dialogue with the system. We call such a special sequence as use case" (Jacobson et al., 1992). Figure 2 illustrates a traditional textual use case description (the template is modified from Jaaksi, 1998). The use case is 'booking' (Figure 2: Booking). Presume a 'use case' where a clerk works in a travel agency (Figure 2: actor). The booking clerk books journeys for different customers (Figure 2: functional summary). Such booking occurs several times a day (Figure 2: frequency) and booking or query to database should take less than 30 seconds (Figure 2: usability requirements). In order to do such 'booking', the booking clerk needs to access to the booking database/file, and the customer file.

Use case: **Booking.**
Version: **1.0**
Functional Summary: **A booking clerk books journeys for customers.**
Frequency: **several times a day**
Usability requirements: **Any database query and booking must be able to carry out in less than 30 seconds.**
Actor: **a clerk.**
Preconditions: Preconditions: **Booking and customer databases exist.**
Exceptions: **If information on certain journey is not available an appropriate error message is produced.**

Figure 2. A traditional use case.

Now presume that security considerations are wanted to address in systems development (including this use case). Figure 3 describes the use case presented in figure 2 enriched by five security dimensions (meta-notation).

Use case: Booking.
Version: 1.0
Functional Summary: A booking clerk books journeys for customers.
Frequency: several times a day
Usability requirements: Any database query and booking must be able to carry out in less than 30 seconds.
Actor/security subject: A clerk
Security classification: confidential
Security objects and access types to security objects: object 11: customer file (the clerk must be able to read, write and delete the customer information); object 15: booking database (the clerk must be able to read, write and delete the customer information on the database)
Security policy/Specific security restrictions: the clerk is only allowed to access security objects classified as confidential with the booking department.
Preconditions: Booking and customer databases exist. The identity of the booking clerk/security subject has been validated.
Exceptions: If information on certain journey is not available an appropriate error message is produced.

Figure 3. A security enriched use case.

In figure 3, five security dimensions (security subjects, security classification, access to objects, security constraint, security policy) are added to the use case. In the use case 'booking', the security subject is the booking clerk (Figure 3: Actor/security subject). In this example, the clerk's security classification is confidential (Figure 3: security classifications: classification of security subjects). Presume that the relevant security objects with respect to this use case are objects 11 (the customer file) and 15 (the booking database) – figure 3: Security objects. The types of access to these objects are read, write/delete/update (Figure 3: The types of access to the security objects). This is because in this example/use case, the booking clerk must be able to read and update (write, delete) to these objects. Presume that is specified in a security policy that the booking clerks can only access to objects labeled as confidential within the booking department. This security policy requirement is expressed in the security enriched use case (Figure 3: Security policy/Specific security restrictions).

4. DISCUSSION AND IMPLICATIONS

The existing SIS design approaches lack a comprehensive modeling support. The different IS security approaches cover the different levels of IS, but no single method provide a comprehensive modeling support. Yet, many of the existing approaches can not be integrated into IS or software development methods. This is problem as termed as the problem of developmental duality (cf. Baskerville, 1992). Moreover, the existing approaches restrict the autonomy of developers to use the approaches they prefer. Finally, IS development methods are known to be emergent and evolving (Truex et al. 1999): novel methods arise every now and then, and are modified by practitioners to fit different situations (Jaaksi, 1998). However, it is difficult to put forth one universal security method that will match every existing, forthcoming and unpredictable IS development methods and they permutations. It is argued a meta-level viewpoint is one solution to address the aforementioned concerns.

With respect to the problem concerning a lack of comprehensive modeling support by existing single methods, this paper proposes a meta-notation which provides modeling support for different levels of IS. The same goes for the lack of methodological support for security aspects of web-IS.

The solution avoids the problem of developmental duality. This meta-notation can be added to the any existing notation for modeling IS or software. Given that one wants to avoid the problem of developmental duality, it is impossible to provide security specific notation (excluding the meta-notation that can be applied to different, if not all, existing modeling notations). Security specific notation would inevitably lead to the problem of developmental duality. Such security specific notation cannot be used to model normal IS development, as security development and normal IS or software development would be carried out using different methods. However, any introduction of a new method, which includes both normal and security development, would restrict the autonomy of developers to use the approaches they prefer (they would have to use this particular approach and in all likelihood abandon their “old” methods/practices). This proposed solution facilitates the developers autonomy: developers can use the methods they prefer as a basis of development.

IMPLICATIONS FOR RESEARCH

This paper has suggested a new paradigm for addressing security aspects in IS development. This new paradigm proposed that, in order to integrate security smoothly into IS development processes, we should move a level away from methodology - such as into the domain of meta-methodology.

Yet, by adapting such a meta-view, the proposed approach has shown a new direction to pay attention to developers' autonomy. The possibility to maximize the use of developer's preferred IS development methods may avoid such problems as the ever increasing cost of system development (e.g. Necco et al, 1987) and lack of developer's satisfaction with respect to the methods used (Mahmood, 1987). However, future research is needed to increase our understanding of the usability and implications of this approach. Empirical research is needed to explore the applicability of this approach to practice. Yet, future research should study can this approach, by facilitating developers' autonomy, improve the developers' motivation with respect to methods.

IMPLICATIONS FOR PRACTICE

As for practitioners, the proposed meta-notation (five dimensions) ensures that security issues are addressed properly and easily in IS and software development. Yet, this proposal satisfies the requirements of autonomy better than any existing methods for designing secure IS. Practitioners can continue to use their favored IS development methods as a basis for the development/management of secure IS.

5. CONCLUSIONS

Information system development methods do not address security aspects seriously enough. As a result, several SIS design and management approaches, from checklists to more advanced endeavors reflecting IS development, are put forth. However, these approaches have a few weaknesses. The different SIS design approaches cover the different levels of IS, but lack the comprehensiveness needed. Most of the approaches for designing SIS are difficult to integrate into IS development methods. Moreover, existing SIS design approaches do not assist the autonomy of developers. This paper considered a framework describing barriers preventing the alternative SIS design approaches from more effectively addressing these lapses. Based on this framework, a meta-notation, for adding security into IS development methods, was presented. Finally, implications for research and practice were presented.

ACKNOWLEDGEMENTS

This research was supported by a research grant by University of Oulu (Finland), and also financed by OWLA-project (http://www.hytec.oulu.fi/).

This paper has been written while Mikko Siponen has been with the Department of Computer Information Systems, J. Mack Robinson College of Business, Georgia State University, Georgia, USA.

6. REFERENCES

Anderson, R., (1999), How to Cheat at the Lottery (or, Massively Parallel Requirements Engineering), Annual Computer Security Applications Conference (ACSAC99).

Baskerville, R., (1988), Designing Information Systems Security. John Wiley Information System Series.

Baskerville, R., (1989), Logical Controls Specification: An approach to information system security", In H. Klein & K. Kumar (eds.) systems development for human progress. Amsterdam: North-Holland.

Baskerville, R., (1993), Information Systems Security Design Methods: Implications for information Systems Development. ACM Computing Surveys 25, (4) December, pp. 375-414.

Baskerville, R. (1996). Structural Artifacts in Method Engineering: The Security Imperative. In S. Brinkkemper & K. Lyytinen & R. Welke (Eds.), Method Engineering (pp. 8-28). London: Chapman & Hall.

Booysen, H.A.S., & Eloff, J.H.P., (1995), A Methodology for the development of secure Application Systems. In proceeding of the 1lth IFIP TC11 international conference on information security, IFIP/SEC'95.

Brinkkemper, S., Lyytinen, K., & Welke, R. (Eds.). (1996). Method Engineering. London: Chapman & Hall.

Castano, S., Fugini, M., Martell, G., & Samarati, P., (1995), Database Security. Addison-Wesley.

Code of Practice for Information Security Management, (1993), Department of Trade and Industry. DISC PD003. British Standard Institution, London, UK.

Dhillon, G. and Backhouse, J., (2001), Current directions in IS security research: toward socio-organizational perspectives. Information Systems Journal. Vol 11, No 2.

Ellmer, E., Pemul, G., Kappel, G., (1995), Object-Oriented Modeling of Security Semantics. In: Proceedings of the 1lth Annual Computer Society Applications Conference (ACSAC'95). IEEE Computer Society Press.

Foley, S.N., (1991), A Taxonomy for Information Flow Policies and Models. Proceedings of the 1991 IEEE Computer Security Symposium on Research in Security and Privacy.

Hirschheim, R., Klein, H. K., & Lyytinen, K., (1995), Information Systems Development and Data Modelling: Conceptual and Philosophical Foundations. Cambridge University Press, UK.

Hitchings, J., (1995), Achieving an Integrated Design: The Way forward for Information Security. Proceedings of the IFIP TC11 eleventh international conference on information security, IFIP/SEC'95.

Hitchings, J., (1996), A Practical solution to the complex human issues of information security design. Proceedings of the 12th IFIP TC11 international conference on information security, IFIP/SEC'96.

Iivari, J., (1989), Levels of abstraction as a Conceptual Framework for an Information Systems. In E. D. Falkenberg and P. Lindgreen (eds): Information System Concepts: An In-depth Analysis. North-Holland, Amsterdam.

Iivari, J & Koskela, E., (1987), The PIOCO model for IS design, MIS Quarterly, Vol. 11, No. 3, pp. 401-419.

Jaaksi, A., (1998), Our Cases with Use Cases. Joumal of Object-Oriented Programming, vol. 10, no. 9, pp. 58-65.

Jacobson, I., Christerson, P. Jonsson, P., Övergaard, G., (1992), A Use Case Driven Approach. Addison-Wesley Publishing Company.

James, H.L., (1996), Managing information systems security: a soft approach. Proceedings of the Information Systems Conference of New Zealand. IEEE Society Press.

Kumar, K. & Welke, R.J., (1992), Methodology engineering: A Proposal for situation-specific Methodology construction. In W.W. Cotterman & J.A. Senn (eds): Challenges and Strategies for research in systems development, pp. 257-269.

Lyytinen, K., (1987), Two Views on Information Modeling. Information & Management, Vol. 12, pp. 9-19.

Lyytinen, K., (1991), A Taxonomic Perspective of Information Systems Development: Theoretical Constructs and Recommendations. In R.J. Boland & R.A. Hirscheim (ed): Critical Issues in Information Systems Research, John Wiley & Sons Ltd.

McDermott, J. & Fox, C., "Using abuse case models for security requirements", Proceedings of the 15th Annual Computer Security Applications Conference (ACSAC). IEEE Computer Society Press (1999).

McLean, J., (1990), The specification and modelling of computer security. IEEE Computer. January, vol. 23, issue 1, pp. 9-16.

Menezes, A.J., van Oorschot, P.C. and Vanstone, S.C., (1999), Handbook of Applied Cryptography. CRC Press, USA.

Odell, J. J. (1996). A primer to method engineering. In S. Brinkkkemper & K. Lyytinen & R. Welke (Eds.), Method Engineering: Principles of method construction and tool support (pp. 1-7). London: Chapman & Hall.

Parker, D. B., (1998), Fighting Computer Crime - A New Framework for Protecting Information. Wiley Computer Publishing. USA.

Pemul, G., Tjoa A. M., & Winiwarter, W., (1998), Modelling Data Secrecy and Integrity. Data & Knowledge Engineering. Vol. 26, pp. 291-308. North Holland.

Röhm, A.W., Pernul, G. & Henmann, G., (1998), Modelling secure and fair electronic commerce. Proceedings of the 14th Annual Computer Security Applications Conference, 1998.

Röhm, A.W., Pernul, G., (1999), COPS: a model and infrastructure for secure and fair electronic markets. Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences (HICSS-32).

Sandhu, R.S., (1993), Lattice-based access controls. IEEE Computer. Vol. 26, no. 11, November, pp. 9-19.

Siponen, M.T., (2001), An analysis of the recent IS security development approaches: descriptive and prescriptive implications. In: G. Dhillon (eds:) Information Security Management - Global Challenges in the Next Millennium, Idea Group (2001).

Summers, R. C., (1997), Secure Computing: Treats and Safeguards. McGraw-Hill.

Straub, D.W. & Welke, R.J., (1998), Coping with Systems Risk: Security Planning Models for Management Decision Making. MIS Quarterly, Vol. 22, No. 4, p. 441-464.

Truex, D. P., Baskerville, R., & Klein, H. K. (1999). Growing Systems in an Emergent Organization. Communications of The ACM, 42(8), 117-123.

Truex, D., Baskerville, R., & Travis, J. (2000). Amethodical Systems Development: The Deferred Meaning of Systems Development Methods. Accounting, Management and Information Technology, 10, 53-79.