December 2003

Colonising the Field) Who,s Playing with Web-based Information Systems Development?

Chris Barry
Department of Accountancy & Finance, National University of Ireland

Jeremy Brown
Department of Accountancy & Finance, National University of Ireland

16th Bled eCommerce Conference

eTransformation

Bled, Slovenia, June 9 - 11, 2003

Colonising the Field – Who's Playing with Web-based Information Systems Development?

Chris Barry, Jeremy Brown

Department of Accountancy & Finance, National University of Ireland, Ireland
Chris.Barry@nuigalway.ie, Jeremy.Brown@nuigalway.ie

Abstract

Two main schools - information systems (IS) and software engineering (SE) - occupy the domain of systems and software development, in both practice and research. Surprisingly, while there would appear to be many common activities, the academic fields have traditionally had limited overlap or shared experience. The information systems school has largely focused on in-house systems, concentrating on the socio-technical approach toward systems development while software engineering attempts to apply engineering principles and formal methods to the production of software systems. However the fields collide where new, Web-based systems share both in-house usage and external commercial software characteristics. While it might be expected that practitioners would be informed by innovative development methods, research indicates that practitioners are not making use of new multimedia and web development method and techniques. The crossover between the fields of IS and SE resurrects up some old problems and new questions. This paper traces the roots of IS and SE; briefly contrasts education and research of each; and examines the differences and common areas of the fields. From a study of how each field is characterised an IS body of knowledge (ISBOK) is identified. How Web-based Information Systems relate to each field is discussed and from the analysis a simple classification framework is constructed, weighing a systems life cycle against quality. The paper concludes with a call for greater cross-fertilization between the fields. Finally the authors suggest important subjects that IS researchers should be studying and others that should be of interest to both SE and IS researchers.

1. Introduction

The fields of information systems (IS) and software engineering (SE) occupy the domain of systems and software development, in the areas of practice and research. In a period of extensive multidisciplinary, cooperative research in several fields of study, it is perhaps anomalous that these academic schools have traditionally had little overlap or shared experience. The IS school has largely focused on in-house systems development where the computer system is just one part of an information system that also includes people, software, methods and organisational procedures (Avison et al. 2001). On the other hand, SE attempts to apply engineering principles and more formal methods to the production of software systems (Pressman 2000a), (Sommerville 2001). The distinctions are clear to those who study and work in each respective field. These schools are brought together in the world of Web-based Information Systems development – IS Departments now need to develop Websites that demand the robustness and reliability of software written with SE methods and techniques. Such Web-based systems share characteristics of both in-house IS usage and external commercial software. Any assumption that traditional IS and SE methods and techniques are being used in developing Web-based Information Systems is not supported by research. Nonetheless, each academic field is laying strong claims to the area of Web-based systems development. To make sense of this, the roots of each field are explored and both the similarities and dissimilarities are discussed. The nature of Web-based Information Systems is discussed and how they relate to each field is examined. The authors question whether now is the time to unify IS and SE development theory and practice or, conversely, whether they should remain separate, partially overlapping, domains with their own distinct Weltanschauung.

2. Origins of IS

The expansion of IS in the academic world quickly followed the growth of data processing departments within business organisations. Early successes with computer-based applications such as billing and sales order processing swiftly generated interest within the business community. The natural home for the design, management and administration of such applications was within the accounting function. Organisational structures changed with the creation of data processing departments, typically reporting to a financial controller. In the beginning programmers came from scientific and engineering backgrounds, but management needed people with more rounded commercial and technical skills - third level academic programmes soon began to fill this void.

3. Origins of SE

Initially programming took second place to the construction of hardware systems. Systematic methods for the development of computer programmes did not exist – programming was undisciplined and more likely to yield good software by trial and error or through the application of intellectual brute force. It was a young field in which gifted individuals fashioned a mystique about the creative process. However by the late 1960s a "software crisis" had emerged - mainframe applications had grown to unmanageable proportions not the least because programming practices were undisciplined and immature. In response, new languages using structured programming concepts were developed to improve software quality and maintainability.

Many techniques and methods used in SE today were developed during the era of structured programming in the early 1970s. Since then contemporary models such as Boehm's spiral model (Boehm 1988), have been developed. Boehm's model acknowledged the importance of iteration, feedback and risk assessment. There followed a period of fundamental change as object-oriented (OO) methods were widely adopted in SE. Metrics and more comprehensive testing techniques were used extensively to improve standards.

4. Definitions of IS

Authoritative definitions of IS are difficult because of the breadth of the field and the fact that it falls outside the domain of the physical sciences where precise meaning is more realizable. In the normal course, a definition of IS is an inclusive term that describes a range of systems used at various levels of an organisation. One example states:

> *"An information system is an arrangement of people, data, processes, information presentation, and information technology that interact to support and improve day-to-day operations in a business as well as support the problem-solving and decision-making needs of management".*
>
> (Whitten et al. 2001)

Other authors prefer to use a descriptive explanation rather than a necessarily broad definition:

> *"... at least four classes of information systems: transaction processing systems, management information systems, decision support systems (for individuals, groups and executives), expert systems. In addition many organisations recognize scientific (or technical) computing and office automation systems".*
>
> (Hoffer et al. 2002)

What is clearly evident from the definition and description above is that they are broad and inclusive. Between them, they illustrate how new types of information systems can be keenly incorporated into the welcoming ministry of IS over time.

5. Definitions of SE

To those who work exclusively in the field of SE it is seen as a distinct field dealing only with the production of computer software and taking its cue from a broader philosophy of engineering. There are numerous definitions of SE all with a common theme of related fields contributing toward the production of a software system:

> *"The application of science and mathematics by which the capabilities of computer equipment are made useful to man via computer programs, procedures, and associated documentation".*
>
> (Boehm 1981)

> *"That form of engineering that applies the principles of computer science and mathematics to achieving cost-effective solutions to software problems".*
>
> (Humphrey 1993)

There is an obvious preoccupation with the metaphor of engineering and the use of hard science in problem solving. This is evident in both definitions and their chronology maintains the historical connection between the reference disciplines and SE.

6. IS Education and Research

Early calls for quality management information began a debate (Ackoff 1967), (Rappaport 1968) about the nature of information provision and decision-making. Models that differentiated the information characteristics at various levels of an organisation (Anthony 1965) led to classifications of different types of information systems and improved understandings (Mason 1969), (Gorry & Scott Morton 1971), (Keen & Scott Morton 1978), (Sprague 1980), (Rockart & Treacy 1982), (Watson et al. 1991). Many new journals began publishing and major conferences were established. The field of IS was emerging as a strong academic and professional discipline. Its applied nature and management-focus clearly distinguished it from computer science.

IS academics soon saw programming as a relatively minor part of a much bigger picture in which it was but one of many activities. It was a step in a life cycle, essential but subservient to systems analysis, systems planning and managerial decision-making. The essential focus was on a socio-technical "system" rather than "software". This holistic perspective fostered by IS academics, placed the emphasis on the use of information in an organisational context. Unlike SE, minor errors in business systems were ceded a higher level of forgiveness because they could normally be repaired in situ.

In a comprehensive analysis of the issues and challenges facing the IS academic field, Lynn Markus believes the IS field needs to change. The new mission for IS should be to focus on "the electronic integration of socio-economic activity" (Lynn Markus 1999). This all-inclusive redefinition of the role of IS would "unite the technical and behavioural segments of our field, would work for current and potential customer groups, and would work for both existing and emerging technologies for the foreseeable future". Such a new mission would require a major revision of the IS curricula and research agenda. The appeal in doing this would need to be tempered with the past failings of the discipline in forgoing the establishment of sound theoretical foundations. Indeed, the lack of a cumulative tradition has been cited as a key reason why there are few barriers to entry into the field (Fitzgerald & Adam 1996).

7. SE Education and Research

Like IS, SE education evolved closely with that of industry, initially there was very little available in the area of software education - the main focus was on hardware systems. As educational programmes were introduced it was widely accepted that the academic domain had its roots firmly within Computer Science, Engineering and Mathematics. There is a broadly agreed framework and understanding of what SE entails within the SE academic and professional communities. By and large the boundaries of the field are well known and research takes place within those borders. There are codes of ethics suggested by both the ACM and IEEE for software engineering and indeed many argue for the discipline to be formally recognised and licensed as an engineering profession. Parnas has strongly argued that while software engineering differs from traditional engineering disciplines in particular ways it is an engineering discipline (Parnas 1997). This view however is not universally shared, Wang contends that, unlike traditional engineering, SE does not have a framework "of the immutable laws of nature" (Wang 2002).

There also remains a debate today as to the relevance of the subjects that are being taught within SE courses (Andrews 1999), (Lethbridge 2000). Whether or not SE is a mature field is still being questioned. While some say that there is a recognised body of knowledge that defines SE, others insist that it is still an immature discipline (Wasserman 1996), (Jackson 1998), (Pour et al. 2000). Indeed recent research points to a disjoint between research and the state of practice in SE (Glass et al. 2002). In an extensive study of six leading SE research journals, academic were discovered to be choosing a narrow range of research methods. Glass et al found that conceptual analysis was heavily used for technical aspects of the field but that case study and field research were seldom chosen where richer and perhaps more valuable insights might be found. They particularly pointed to the slowness of technology transfer and highlight the lack of research that might explain why this is happening.

8. Differences between the Fields

That one field, either IS or SE is an interloper on the stage of systems development is not being contended or argued in this paper. The distinctions between the fields have long been there, reflected in literature, practice, academic forums and professional associations. IS schools have focussed on in-house systems development, looking at the socio-technical system that is made up of people as well as machines and software. The IS Department in an organisation is a service function, delivering computing resources and systems to a user organisation made up of internal groups and individuals.

SE differs in that it normally has a product rather than a service focus. The software is often commercial in nature where a low tolerance for errors is self-evident. The developer has a real contractual obligation to ensure the systems are extremely stable and reliable. The user is normally the client, outside of the organisation commissioned to develop the software. Generally the functional requirements specification for software engineers is much tighter than an equivalent design specification or systems proposal in the IS development world.

9. How are IS and SE Projects Characterised?

Typical IS Projects include the integration and installation of "off the shelf" software, in-house bespoke business systems, outsourced bespoke systems and systems maintenance. They are: made up of software, hardware and people; open systems; have more control over the end user environment; subject to change on a regular basis; developed with languages like Basic, Cobol or 4GLs and typically use relational databases. In some contrast, typical SE Projects include real-time software, scientific and engineering applications, system software and embedded software. They are: made up of software; more closed in nature; have less control over the end user environment; less likely to be changed on a regular basis and developed with languages like C and C++.

From the characterisation of IS and SE systems above, it is clear that while both involve the production of computer programmes, the projects are often (but not always) dissimilar. Also, the emphasis and explicit importance of programming is in marked contrast. The explanation lies in the narrowness of the SE domain where there is a near exclusive focus on software. On the other hand programming is considered just one stage in a larger systems development process, that is itself just one aspect, of the IS field. This can be further illustrated by looking at the key knowledge areas and the related disciplines of each field as shown below in Tables 1 to 4.

Table 1: *ISBOK Knowledge Areas of the IS Field*

• IS Development • Database design and management • Technology management • IT/IS strategy • "Hard" and "Soft" approaches to systems development	• Knowledge management • End user computing (EUC) • Specialised decision support applications (DSS or EIS) • E-commerce

Table 2: *ISBOK Related Disciplines of the IS Field*

• Organisational theory • Communications • Managerial decision-making • Computer science	• Human computer interaction • Software engineering • Management science and operations research

The IS knowledge areas and the related disciplines of IS (we call the Information Systems Body of Knowledge - ISBOK) identified by the authors demonstrate the extensive range of topics that stretch across the domain. Its reference disciplines range from Organisational Theory to Computer Science. The breadth of the field is reflected in both the IS curricula and the research subjects sought for IS journals.

The knowledge areas and the related disciplines identified by the Software Engineering Body of Knowledge (SWEBOK) project (Bourque et al. 1999), (SWEBOK 2001) are shown below. It is revealing to note that the related disciplines do not include IS. One wonders why this should be the case - many SE projects are commissioned by MIS or IS Departments; SE (or programming) is typically an integral part of any IS development project and many SE project need to be integrated with other, larger IS applications architectures.

Table 3: *SEWBOK Knowledge Areas of the SE Field*

• Software engineering process • Software engineering evolution & maintenance • Software configuration management • Software construction • Software design	• Software engineering infrastructure • Software engineering management • Software quality analysis • Software requirements analysis • Software testing

Table 4: *SEWBOK Related Disciplines of the SE Field*

❑ Computer sciences and human factors ❑ Computer engineering ❑ Computer science ❑ Systems engineering	❑ Mathematics ❑ Project management ❑ Management and management science

10. Shared Subjects in IS and SE

Demonstrative differences between the two academic communities in preceding sections appear to make a compelling case for field "dissimilarity". However such distinctions can seem less obvious when shared subjects are identified (see Figure 1). Indeed to many non-IS academics (and the outside world generally) the dissimilarity may, as Lynn Markus puts it, be "completely incomprehensible" (Lynn Markus 1999). Despite visible areas of shared interests such as development methods and techniques, requirements determination and project management there exists quite distinct perspectives. For example Boehm's widely cited spiral model (Boehm 1988), the Component Assembly Model (Nierstrasz et al. 1992) and the Concurrent Development Model (Davis & Sitaram 1994) rarely appear in IS analysis and design texts. Where cross-fertilization does occur it typically takes place from SE to IS. If criticisms that IS do not embrace more disciplined software development methods may be justified, SE appears completely cut-off from the world of IS development.

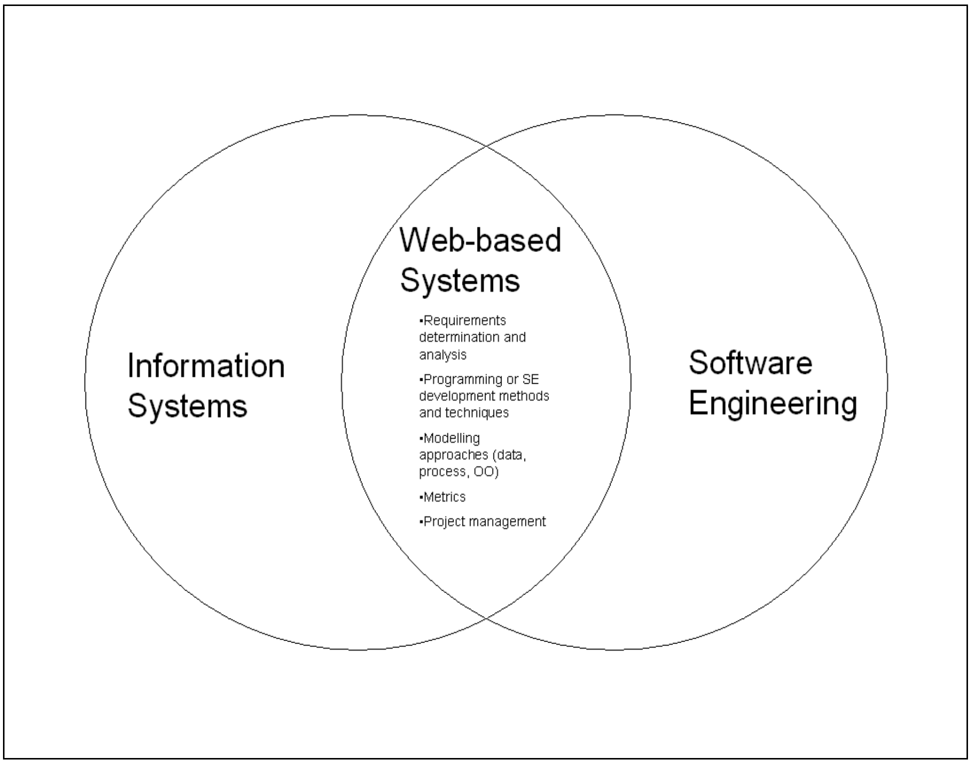

Figure 1: *Diagrammatic Representation of Overlap between IS & SE*

11. What are Web-based Systems?

Web-based systems projects have characteristics of both conventional in-house development and commercial software, aimed at an external audience. The coming together of these two fields presents great challenges to both of them. There are now numerous terms that have been used to describe the relatively new phenomenon of what we have called "Web-based Systems Development" (see Table 5 below).

Table 5: *Common Synonyms for Web-based Systems Development*

Common Synonyms
• Web-based Information Systems – WIS (Isakowitz et al. 1998) • Web Site Engineering (Powell et al. 1998) • Web Information Systems – WIS (Koufaris et al. 1999) • Web Engineering - WebE (Murugesan et al. 1999) • WebApps (Pressman 2000b)

Conceptually, some are calling it a new "Web Application Paradigm" (Enguix & Davis 1999), or is it, as Murugesan et al consider it, a new quasi-engineering field called "Web

Engineering" (Deshpande and Hansen. 2001)? Others are more sanguine and doubtful. In their essence, how new or unique are Web-based Information Systems (or whatever one might like to call them)? Looking at established IS models and frameworks it is clear than that there is little new in any theoretical sense. For example, Gorry and Scott Morton's framework easily accommodates Web-based applications (Gorry & Scott Morton 1971), Web-based Information Systems comfortably span Mason's continuum of information systems (Mason 1969) and contingency models from Davis (Davis 1982) stand up well to scrutiny. A more detailed analysis of how the IS literature has been able to absorb Web-based Information Systems into the family of information systems without too much difficulty can be seen in Barry (Barry 2000).

Nonetheless it is clear there are many differences to traditional IS and SE development projects. Powell et al suggests that Web-based systems are: "…a mixture between print publishing and software development, between marketing and computing, between internal communications and external relations, and between art and technology" (Powell et al. 1998). Such a view suggests an obvious role for Software Engineers but there are numerous other indicated roles such as Graphic Artists, Business & Systems Analysts, Video & Audio Producers, Storyboarders, Technical & Document Writers and HCI specialists. If existing methods are inappropriate then we need new methods and techniques that cater for the variety of roles within Website development teams. While it might be expected that practitioners would be informed by new and innovative development methods (Isakowitz et al. 1995), (Gellersen et al. 1997) , (Schwabe & de Almeida Pontes 1998), research indicates that practitioners are not making use of new multimedia and Web development methods, tools and techniques (Fratenali 1998), (Barry & Lang 2001).

In looking at the clear need to improve the quality of Web-based Information Systems (as opposed to traditional IS) an interesting relationship with a system's life cycle emerges. A simple classification framework (shown in Figure 2) can be constructed, weighing a systems life cycle against systems quality. The demand for high quality Web-based applications, developed in "Web-time", that have a relatively short life cycle (illustrated in the lower left quadrant) contrast with commercial in-house IS that traditionally have had an acceptable, lower level of reliability but a longer development period and life cycle (illustrated in the upper right quadrant). It can also be argued that SE applications are generally of high quality and have a long life cycle while end-user developed applications are typically of low quality with a short life cycle.

		Life Cycle	
		Short	**Long**
Quality	**Low**	End-user developed Applications	In-house non-critical Business Information Systems
	High	Web-based Information Systems Applications	Software Engineering Applications In-house mission-critical Business Information Systems

Figure 2: *Classification Framework for of IS and SE Applications - Quality versus Lifespan*

From this analysis, the authors suggest that there is an evident life cycle for Web-based Information Systems and it appears to be very short. Most organisations have been through at least two and some as many as four Web-based versions of the same information systems in almost as many years. This continuous evolution presents huge challenges in both business, technical and development terms.

12. Conclusions

This paper has traced the beginnings of both IS and SE and illustrated how they quickly diverged into two different fields with just a little overlap. Separate academic programmes reflected the degree of specialisation that grew over time. Now, Web-based Information Systems have brought about, if not a convergence, then at least a milieu in which both fields are struggling to regain their feet. Central to these conclusions is the contention that Web-based Information Systems development needs to draw water from the wells of both IS and SE theory and practice a view also expressed by Avison and Wilson (Avison and Wilson 2001). The authors believe they have demonstrated that the fields are separate disciplines and that each has a different, legitimate, raison d'être. Nonetheless both fields need to do more to embrace concepts and ideas from each other, and to collaborate on common frontiers. For in-house Web-based development, lessons from the more disciplined SE approach to software development need to be learned. However the wholesale adoption of SE methods is not being suggested since there may well be a need for changes in attitude to traditional SE practices (Carstensen & Vogelsang 2001). Likewise, SE needs to look at its relationship to its environment. As noted earlier, the SWEBOK guide does not include IS as a "related discipline". Cross-fertilization from other areas and the courage, as Matsubara and Ebert (Matsubara & Ebert 2000) put it, to "go beyond traditional boundaries" must be embraced by both disciplines. Obviously, other fields outlined earlier have roles to play in improving the practice of Web-based systems development. The multidisciplinary nature of Web team composition inevitably leads to cross-cultural and indeed philosophical differences. There is a need to reconcile the language and the development environment of various developers on Web-based projects.

The short life cycle and high-quality needs of Web-based Information Systems, identified in the last section, illustrates how they differ from other types of systems. The business, technical and development challenges presented by the continuous and rapid evolution of Web-based systems demands further research from both the IS and SE communities. IS researchers need to study the current practice of Web-based development; to look at developer experiences; to determine the factors that aid or hinder successful outcomes; to reconcile abstract and implementation level modelling and to resolve Web-based systems integration problems. Both SE and IS researchers need to continue to develop usable methods and techniques; to develop faster development life-cycles; to construct mechanisms that facilitate continuous development and maintenance; and to develop improved development environments that fully join together modelling, programming languages, tools and people.

References

Ackoff, R. (1967). Management Misinformation Systems. Management Science, 11(4), 147-156.

Andrews, D. (1999). Software engineering education in the 21st century. Information & Software Technology, 41, 933-936.

Anthony, R. (1965). Planning and Control Systems: A Framework for Analysis. Harvard University Graduate School of Business Administration, Boston.

Avison, D., D. Fitzgerald and P. Powell (2001). Reflections on Information Systems Practice, Education and Research. Information Systems Journal, 11(1), 3-22.

Avison, D. and D. Wilson (2001). A viewpoint on software engineering and information systems: what we can learn from the construction industry? Information & Software Technology 43(13): 795-799.

Barry, C. (2000). Issues and Perspectives on Web-based Information Systems Development. In Third International Asia-Pacific Web Conference, p.163-171, International Academic Publishers, Beijing, Xi'an China.

Barry, C. and M. Lang (2001). A Survey of Multimedia and Web Development Techniques and Methodology Usage. IEEE Multimedia, 8(3), 52-60.

Boehm, B. (1988). A Spiral Model for Software Development and Enhancement. Computer, 21(5), 61-72.

Boehm, B. (1981). Software Engineering Economics. Prentice Hall, New Jersey.

Bourque, P., R. Dupuis, A. Abran, J. Moore, L. Tripp and S. Wolff (1999). The Guide to the Software Engineering Body of Knowledge. IEEE Software, 16(6), 35-44.

Carstensen, P. and L. Vogelsang (2001). Design of Web-based Information Systems - New Challenges for Systems Development? In 9th European Conference on Information Systems, p.536-547, ECIS, Bled, Slovenia.

Davis, A. and P. Sitaram (1994). A concurrent process model of software development. ACM SIGSOFT Software Engineering Notes, 19(2), 38-51.

Davis, G. B. (1982). Strategies for Information Requirements Determination. IBM Systems Journal, 21(1), 4-30.

Deshpande, Y. and S. Hansen (2001). Web Engineering: Creating a Discipline among Disciplines. IEEE Multimedia, 8(2), 82-87.

Enguix, C. and J. Davis (1999). Filling the Gap: New Models for Systematic Page-based Web Application Development and Maintenance. In International Web Engineering Workshop, WWW8, Toronto, Canada.

Fitzgerald, B. and F. Adam (1996). The Future of IS: Expansion or Extinction? In Proceedings of First Conference of the UK Academy for Information Systems, p.1-15, Cranfield University.

Fratenali, P. (1998). Web development tools: a survey. Computer Networks and ISDN Systems, 30, 631-633.

Gellersen, H., R. Wicke and M. Gaedke (1997). WebComposition: An Object-Oriented Support System for the Web Engineering Lifecycle. In Proceedings of the Sixth

International WWW Conference, Computer Networks and ISDN Systems., p.1,429-1,437, Vol.29, Santa, Lara, CA.

Glass, R., I. Vessey and V. Ramesh (2002). Research in software engineering: an analysis of the literature. Information and Software Technology, 44, 491-506.

Gorry, G. and M. Scott Morton (1971). A Framework for Management Information Systems. Sloane Management Review, Fall, 55-70.

Hoffer, J., J. George and J. Valacich (2002). Systems Analysis and Design. Prentice Hall, New Jersey.

Humphrey, W. (1993). Software Engineering. An Encyclopedia of Computer Science (Ralston, A. Ed.). MacMillan, London.

Isakowitz, T., E. Stohr and P. Balasubramanian (1995). RMM: A Methodology for Structured Hypermedia Design. Communications of the ACM, 38(8), 34-44.

Isakowitz, T., M. Bieber and F. Vitali (1998). Web Information Systems. Communications of the ACM, 41(7), 78-80.

Jackson, M. (1998). Will There Ever Be Software Engineering? IEEE Software, 15(1), 36-39.

Keen, P. and M. Scott Morton (1978). Decision Support Systems, An Organisational Perspective. Addison-Wesley, Reading, MA.

Koufaris, M., T. Isakowitz and M. Bieber (1999). Web Information Systems: Introduction to the Minitrack. In Proceedings of the 32nd Hawaii International Conference on System Sciences, IEEE Press, Hawaii.

Lethbridge, T., C. (2000). What Knowledge Is Important to a Software Professional? Computer, (May), 44-50.

Lynn Markus, M. (1999). Thinking the Unthinkable - What Happens if the IS Field as we Know it Goes Away? In Rethinking Management Information Systems (Currie, W. and B. Galliers Eds.), p. 175-203, Oxford University Press.

Mason, R. (1969). Basic Concepts for Designing Management Information Systems. AIS, Research paper no. 8.

Matsubara, T. and C. Ebert (2000). Benefits and Applications of Cross-Pollination. IEEE Software, 17(1), 24-26.

Murugesan, S., Y. Deshpande, S. Hansen and A. Ginige (1999). Web Engineering: A New Discipline for Web-based Systems Development. In Workshop on Web Engineering, ICSE, p.1-10, ICSE, Los Angelas.

Nierstrasz, O., S. Gibbs and D. Tsichritzis (1992). Component-oriented software development. Communications of the ACM, 35(9), 160-165.

Parnas, D. (1997). What We Teach Software Engineers in the University: Do we Take Engineering Seriously? In Proceedings of the 6th European Conference on the Foundations of Software Engineering, (Maibaum, T. Ed.), p.40-50, Springer-Verlag New York, Inc., Zurich, Switzerland.

Pour, G., M. Griss, L and M. Lutz (2000). The push to make software engineering respectable. Computer, (May), 35-43.

Powell, T., D. Jones and D. Cutts (1998). Web Site Engineering: Beyond Web Page Design. Prentice-Hall, New Jersey.

Pressman, R. (2000a). Software Engineering: A Practitioner's Approach. McGraw-Hill, Singapore.

Pressman, R. (2000b). What a Tangled Web We Weave. IEEE Software, 17(1), 18-21.

Rappaport, A. (1968). Management Misinformation Systems - Another Perspective. Management Science, 15(4), 133-136.

Rockart, J. and M. Treacy (1982). The CEO Goes On-Line. Harvard Business Review, 60(1), 74-79.

Schwabe, D. and R. de Almeida Pontes (1998). OOHDM-WEB - Rapid Prototyping of Hypermedia Applications in the WWW. Dept. of Informatics, Pontificia Univeridade Catolica do Rio de Janeiro.

Sommerville, I. (2001). Software Engineering, 6th Edition. Addison-Wesley, Harlow.

Sprague, R. (1980). A Framework for the Development of Decision Support Systems. MIS Quarterly, (December), 1-26.

SWEBOK (2001). Guide to Software Engineering Body of Knowledge. http://www.swebok.org/

Wang, W. (2002). Beware the Engineering Metaphor. Communications of the ACM, 45(5), 27-29.

Wasserman, A. (1996). Toward a Discipline of Software Engineering. IEEE Software, (November), 23-31.

Watson, H., K. Rainer and C. Koh (1991). Executive Information Systems: A Framework for Development and a Survey of Current Practices. MIS Quarterly, (March), 12-30.

Whitten, J., L. Bentley and K. Dittman (2001). Systems Analysis and Design Methods. Irwin McGraw-Hill, Boston, MA.