ECIS 2003 Proceedings

European Conference on Information Systems (ECIS)

2003

Finding a Home for Web Based Information Systems - Perusing the Landscape

Chris Barry
National University of Ireland, chris.barry@nuigalway.ie

Jeremy Brown
National University of Ireland, jeremy.brown@nuigalway.ie

Follow this and additional works at: http://aisel.aisnet.org/ecis2003

Finding a Home for Web-based Information Systems – Perusing the Landscape

Chris Barry
Department of Accountancy & Finance
National University of Ireland, Galway
Galway
Ireland
Tel.: +353 91 750301, Fax: +353 91 750565
chris.barry@nuigalway.ie

Jeremy Brown
Department of Accountancy & Finance
National University of Ireland, Galway
Galway
Ireland
Tel.: +353 91 750301, Fax: +353 91 750565
jeremy.brown@nuigalway.ie

Abstract

Information systems (IS) and software engineering (SE) have shared the domain of systems and software development for several decades with too little overlap in practice and research. The IS school has largely focused on in-house systems, concentrating on the human-computer aspects of systems development while SE attempts to apply engineering principles and 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. In this paper, the origins and the development of education of both fields are explored – then various aspects are compared and contrasted. If, as it would appear, recommended development methods are ineffective or simply not being used, is a new understanding of development practice that finds expression in creativity and improvisation the way forward, or is this just a new engineering problem to be solved? The authors conclude that we need fast and flexible methods that go beyond new SE techniques for the Web, reflecting the business imperative to quickly produce high-quality robust systems in competitive environments. Web-based systems development should be contextualized within IS theory - learning from the rigour of SE - but viewed definitively as part of a larger socio-technical system.

Keywords

Web-based Information System development, Information Systems development, Software Engineering, Web Engineering.

1 Introduction

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 IS school has largely focussed on in-house systems, concentrating on the socio-technical approach toward systems development (Avison, Fitzgerald & Powell 2001) while SE attempts to apply engineering principles to the production of software systems (Sommerville 2001). With the development of Web-based Information Systems (WIS) these schools are thrown together – IS Departments now need to develop systems that share both in-house usage and external commercial software characteristics, and demand the robustness and reliability of software written with SE methods. While research indicates that traditional IS and SE methods are not generally used in developing such systems, each academic field is laying strong claims to the area of WIS development. To yield some understanding of how WIS research and practice might go forward, the roots of the respective fields are examined and the authors then reflect on the nature of their relationship with and contribution to WIS development.

2 The Roots of Information Systems

2.1 The Early Days

The origins of IS in the academic world quickly followed the growth of data processing departments. 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 data processing department was within the accounting function. While initially hardcore programming staff came from scientific and engineering backgrounds, managers needed people with more rounded commercial and technical skills. Eventually third level academic programmes met this need.

2.2 Definition and Descriptive Explanation of IS

Definitions of IS are difficult because of the breadth of the field, reflected in the number of reference disciplines drawn on in the IS literature. The following definition comes from an introductory-level management information systems (MIS) text:

> *"An information system is a group of interrelated components that collectively work to carry out input, processing, output, storage and control actions in order to convert data into information products that can be used to support forecasting, planning, control, coordination, decision making and operational activities in an organisation".*

> (Bocij, Chaffey, Greasley & Hickie 1999).

This definition focuses on the composition of an IS and the process by which it delivers information to support problem-solving and decision-making needs of management and others. Support has been a key concept in IS literature over the past twenty years. Other authors prefer to use a descriptive explanation rather than a necessarily broad definition:

"Information systems are developed for different purposes. Transaction processing systems (TPS) function at the operational level of the organization; office automation systems (OAS) and knowledge work systems (KWS) support work at the knowledge level. Higher-level systems include management information systems (MIS) and decision support systems (DSS). Expert systems apply the expertise of decision makers to solve specific, structured problems. On the strategic level of management we find executive support systems (ESS). Group decision support systems (GDSS) and the more generally described computer supported collaborative work systems (CSCWS) aid group-level decision making of a semistructured or structured variety".

(Kendall & Kendall 2002).

What is clearly evident from the above is that IS are characterised as an inclusive expression for many types of information systems. A look back over earlier descriptive explanations of IS reveals how novitiate business information systems are keenly incorporated into the welcoming ministry of IS literature.

2.3 The Development of IS Education

By the 1960s business schools began to incorporate IS-type and general computing courses into academic programmes. The perspective was firmly business-oriented and distinctions were made between the requirements of business applications and those of engineering or science. Business schools quickly appreciated that issues other than programming were critical to the success of computer-based applications - systems analysis and design, project management and information management soon became essential elements of business computing courses. IS academics soon saw programming as a relatively minor part of a 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 more holistic perspective was fostered by IS academics, placing the emphasis on the use of information in an organisational context. The nature of programming for in-house IS was also different to SE. It was generally accepted, quite reasonably, that for most business applications you did not need to produce perfect code.

Early calls for better management information began a debate (Ackoff 1967, Rappaport 1968) about the nature of information provision and decision-making. Models that differentiated 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). New journals were appearing and IS was emerging as a strong academic and professional discipline. Its applied nature and management-focus clearly distinguished it from computer science.

By the late 1980s decision support application and end user computing were key IS issues. However, by the 1990s the growth of real-time business applications and the increased reliance on mission-critical IS led to pressures on IS Departments to develop more robust and secure systems. The tolerance for IS that were unstable diminished quickly. Furthermore the desire to have reliable, secure systems that dealt with the complexity of organisational operations and information provision was manifestly expressed in outsourced systems.

2.4 Uncertainty in the IS Field

For IS academics, outsourcing has its dangers. As other disciplines invade the general IS territory (management via strategy, marketing via e-commerce, IT via software development) the IS field may face a severe contraction, serving the traditional, but disappearing, base of the user organisation. Other arguments can be made that technology is becoming ubiquitous and is an embedded subject in most disciplines within which unique and distinctive issues are best discussed, thus dissipating core IS issues. In an analysis of the challenges facing the IS academic field, Lynn Markus believes the new mission for IS is 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).

3 The Roots of Software Engineering

3.1 The Early Days

In its infancy programming was an activity that took second place to the construction of hardware systems. Systematic programming methods did not exist – in truth programming was undisciplined and more likely to yield good software by trial and error or the application of intellectual brute force. It was a new field in which some gifted individuals fashioned a mystique about the creative process. However by the late 1960s the "software crisis" (NATO 1968) had arrived and mainframe applications had grown to unmanageable proportions. In response, new languages that used structured programming concepts were adopted to improve software quality and maintainability. More complete life-cycle based development models evolved and innovative approaches such as Boehm's spiral model were proposed (Boehm 1988). 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 to improve standards.

3.2 Definitions of SE

For the most part, the SE community see it as a distinct field dealing only with the production of computer software and taking its cue from the broader philosophy and principles of engineering. The following definitions describe SE as:

> *"The establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines".*

(Naur & Randell 1969).

> *"The application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software, i.e., the application of engineering to software".*
>
>   *(IEEE 1990).*

In both definitions, there is an obvious preoccupation with the metaphor of engineering and the use of hard science in problem solving.

3.3 SE Education

Like IS, SE education evolved closely with that of industry. Initially the main focus was on hardware systems not software. As programmes were introduced it was widely accepted that the academic domain had its roots planted firmly in Computer Science, Computer Engineering and Mathematics. There remains a debate today as to the relevance of the subjects being taught within SE courses (Lethbridge 2000). Parnas discusses the differences and similarities between traditional computer science and engineering programmes, and argues for SE programmes that follow a traditional engineering approach to professional education (Parnas 1999). He contests that Computer Science and SE should be considered two related yet independent disciplines. There is also a debate as to whether it is yet a mature field. 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, Griss & Lutz 2000).

3.4 Disjoint in the SE Field

Recent research points to a serious disjoint between research and the state of practice in SE (Glass, Vessey & Ramesh 2002). In an extensive study of six leading research SE journals, researchers 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.

4 Contrasting IS and SE

4.1 Differences Between the Fields

That one field, either IS or SE is an interloper on the stage of systems development is not being contended in this paper. The distinctions between the fields have long been there, reflected in literature, practice, academic forums and professional associations. Arguments can be made that each field is a legitimate specialisation working on differing aspects of software and systems development and that therefore the separateness in both literature and practice is natural. Indeed it is suggested that there needs to be differences between the fields of SE and IS (Avison & Wilson 2001). IS schools have focussed on in-house systems development, looking at socio-technical systems 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. End users typically initiate a systems project, are closely involved during the development process and are often physically near at hand during development.

SE differs in that it normally has a product rather than a service focus. The software is often commercial in nature, with an obligation to ensure systems are extremely stable and reliable. The user is normally a "client", outside of the organisation. Generally the functional requirements specification for software engineers is much tighter than its IS equivalent.

Typically, IS Projects are:	**Typically, SE Projects are:**
• In-house bespoke business systems development • Outsourced bespoke systems • Integration and installation of commercial off the shelf software (COTS) • Systems (including software) maintenance	• System software • Scientific and engineering software • Embedded software • Real-time software • COTS (of any software type) • Third-party commissioned products (of any software type)
IS are characteristically:	**SE are characteristically:**
• Made up of software, hardware and people • Open systems • Control more of the end user environment • Subject to change on a regular basis • Developed with languages like Basic, Cobol, RPG or 4GLs and typically use relational databases	• Made up of software • More closed in nature • Control less of the end user environment • Less likely to be changed on a regular basis • Developed with languages like Fortran, Pascal, C and C++

Table 1: IS and SE Projects

From the analysis of the nature of IS and SE projects in Table 1, it is clear that while both involve the production of computer programs, 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 in IS (illustrated in Table 2). The breadth of the IS "Body of Knowledge" (we term ISBOK) is reflected in both IS curricula and research subjects sought for IS journals.

Knowledge Areas	Related Disciplines
• IS development (ranging from "hard" to "soft" approaches) • Database design and management • Technology management • Specialised decision support applications (DSS or ESS) • IT/IS strategy • Knowledge management • End user computing (EUC) • E-commerce	• Organisational theory • Communications • Managerial decision-making • Management science and operations research • Human computer interaction • Software engineering • Computer science

Table 2: ISBOK Knowledge Areas and the Related Disciplines

The Software Engineering Body of Knowledge (SWEBOK) project (Bourque, Dupuis, Abran, Moore, Tripp & Wolff 1999) results, shown in Table 3, demonstrate all the knowledge areas are software subjects but surprisingly the related disciplines do not include IS. This is peculiar on several levels: many SE projects are commissioned by MIS or IS Departments; programming is typically an integral part of any IS development project; and many SE projects need to be integrated with other, larger IS application architectures.

Knowledge Areas	Related Disciplines
• Software configuration management • Software construction • Software design • Software engineering infrastructure • Software engineering management • Software engineering process • Software engineering evolution and maintenance • Software quality analysis • Software requirements analysis • Software testing	• Computer sciences and human factors • Computer engineering • Computer science • Management and management science • Mathematics • Project management • Systems engineering

Table 3: The SWEBOK Knowledge Areas and the Related Disciplines

The SWEBOK suggests there is a broadly agreed understanding of the boundaries of SE amongst the academic and professional communities. There are even codes of ethics suggested by both the ACM and IEEE for software engineering. This stands in some contrast to the IS field that finds division within its academic community about the very nature of systems development, especially debates about 'hard' and 'soft' development approaches. Furthermore there is uncertainty about the theoretical foundations of IS.

4.2 Shared Subjects in IS and SE

Despite a clear sense of difference between the two academic communities the distinction can seem less obvious when the shared subjects are outlined. 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). The common areas include:

- Requirements determination and analysis

- Programming or SE development methods and techniques

- Modelling approaches (data, process, OO)

- Metrics

- Project management

Interestingly, where cross-fertilization does occur it is typically unidirectional – from SE to IS. Thus, while the IS field may be guilty of embracing too few experiences of SE, SE appears largely cocooned from the IS domain.

4.3 Literature and Professional Membership

From quite an early time IS literature began to differ from that of SE. MIS Quarterly, Management Science and Interfaces and to a lesser extent the Harvard Business Review became key forums for the discussion of IS issues and research. The more technical software-related issues were discussed in places such as the IBM Systems Journal, the IEEE (e.g., IEEE Transactions on Software Engineering) and the ACM (e.g., Communications of the ACM). In more recent times, the IEEE and its various journals and magazines, Information and Software Technology, and the Journal of Systems and Software are where many SE issues are discussed (Glass et al. 2002). The ACM has a somewhat broader reach that attracts IS researchers and professionals, as well being of interest to SE. Now, mainstream IS journals include Information Systems Research, Information Systems Journal, Journal of Management Information Systems, Information & Management and Decision Sciences. Regional variations are demonstrated in a study that ranks journals in the US, Europe and Australasia (Mylonopoulos & Theoharakis 2001).

In the US, SE professionals are more likely to join the IEEE while IS professionals tend towards the ACM. In the UK software engineers would tend to join the BCS (British Computer Society) or possibly the IEE (Institution of Electrical Engineers). Systems analysts and IS academics lean toward the ACM, the AIS or to a lesser extent the UKAIS, mainly an academic support group (Avison & Wilson 2001). In Ireland, professionals can join the ICS (Irish Computer Society) but would also look to UK and US organisations.

5 IS, SE and the Web

5.1 Information Systems and the Web

Today the most profound effect on the way in-house IS are developed are the technologies of the World Wide Web. Almost every organisation, public and private, has developed a Web-based system, many on their third or fourth iteration. What is the future of IS if it does not encompass the design and development of e-Commerce and WIS? Such systems surely are making demands IS developers have not faced before – rock-solid software systems, integration across a range of systems, the management of outsourced components and new forms of IS/IT governance. Nonetheless, the domain of WIS development must fall within the intellectual compass of IS academe. It is as central to the field as any other type of IS.

5.2 Software Engineering and the Web

There are conflicting views as to whether WIS development should be a central part of the SE school. Some argue that applying engineering discipline to WIS development is inherently sensible, to ensure systems quality and maintainable software. Others state that we have persisted for too long in the illusion that there can be universal methods to develop software. Jackson believes we must develop and apply contemporary techniques based upon the new Web-based technologies, and that these activities should be viewed as separate and distinct to software engineering (Jackson 1998).

6 Web-based Systems Development

6.1 Web-based Systems – What are they?

In the introduction to this paper it was asserted that the separateness of IS and SE in both literature and practice is, for the most part, natural but that Web-based systems projects have characteristics from conventional in-house development and commercial software, aimed at an external audience. The coming together of these two fields presents great challenges to both. There are now numerous terms that have been used to describe the new phenomenon of WIS development. Some of these are: Web-based Information Systems (WIS), Web Site Engineering or WebApps. Conceptually, it has been called a new "Web Application Paradigm" (Enguix & Davis 1999), and described by some as a new, quasi-engineering field called "Web Engineering" (Murugesan, Deshpande, Hansen & Ginige 1999). Others are more sanguine and doubtful. In their essence, how new or unique are WIS? Looking at established IS models and frameworks there is little new in any theoretical sense. For example, WIS comfortably span Mason's continuum of information systems (Mason 1969), Gorry and Scott Morton's framework easily accommodates Web-based applications (Gorry & Scott Morton 1971) and contingency models from Davis (1982) stand up well to scrutiny. A more detailed analysis of how the IS literature has been able to absorb WIS into the family of information systems without too much difficulty can be seen in Barry (2000).

Nonetheless it is clear that WIS have real differences with 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, Jones & Cutts 1998).

Such a view suggests an obvious role for Software Engineers but there are several other indicated roles such as Graphic Designers, Systems Analysts, Audio Producers, Scriptwriters, Video Producers, Technical Writers and Human Factors Engineers. We need development methods and techniques that cater for the differing roles of individuals within Website development teams and to assist their collaborative effort.

6.2 The Use of Existing and New Methods

For more "conventional systems" the recent past has been dominated by structured methods for large-scale systems development projects and by visual-oriented or object-oriented methods for interface design and specialised systems. Problems in using existing approaches towards WIS development seem widespread and it has been argued that traditional development processes are simply inappropriate (Lowe & Hall 1999). It would appear that there is a return to the pre-methodology era when ad-hoc and trial and error characterised IS development (Avison & Fitzgerald 2003). While it might be expected that practitioners would be informed by new and innovative development methods (Isakowitz, Stohr & Balasubramanian 1995, Gellersen, Wicke & Gaedke 1997) research indicates that practitioners are not making use of new multimedia and web development methods and techniques (Barry & Lang 2001). If recommended development methods are ineffective or simply not being used, is a new understanding of development practice that finds expression in creativity and improvisation the way forward (Ciborra 1999, Lang 2001)? On the other hand is this just a new engineering problem to be solved? Do software engineering principles simply need to be improved and re-cast as Web Engineering (Murugesan et al. 1999)? Perhaps more recent approaches may offer more promise. Agile software development (ASD) approaches, that include XP, Scrum, Adaptive Software Development, Crystal Methods, Feature-Driven Development (FDD) and Dynamic System Development Methodology (DSDM) (Boehm 2002) are typically being used by small teams developing software for quick-to-market applications (Reifer 2002). Certain WIS development projects may be well suited to the use of ASD approaches. The first principle of the Agile Software Manifesto (Highsmith & Cockburn 2001) states that "our highest priority is to satisfy the customer through early and continuous delivery of valuable software." The emphasis is on individuals and interactions rather than processes, tools and project plans. Another recent approach is the Web IS Development Methodology or WISDM (Vidgen, Avison, Wood & Wood-Harper 2003) that adapts the Multiview framework (Avison & Wood-Harper 1990) to develop WIS. It stresses the importance of jointly considering the technical, organisational and personal perspectives in developing IS. However, WISDM on its own is insufficient to guide developers across the entire systems development process since it does not cover systems construction and subsequent activities, although it can be demonstrated that the method may be combined with new technologies to produce a working software system (Vidgen et al. 2003).

7 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 specialisation that grew over time. Now WIS 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 WIS development needs to draw water from the wells of both IS and SE theory and practice. Furthermore other disciplines, especially graphic design, have roles to play in improving the practice of WIS development. The multidisciplinary nature of Web team composition inevitably leads to cross-cultural and indeed philosophical differences. There is a need to reconcile the differing language and development environments of various individuals working on Web-based projects.

In considering the need to improve the quality of WIS an interesting relationship with a system's life cycle emerges. The demand for high quality WIS applications that have a relatively short life cycle (illustrated in the lower left quadrant in Figure 1) contrasts with in-house IS that traditionally have had an acceptably lower level of reliability but a longer life cycle (illustrated in the upper right quadrant). This perspective also demonstrates 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. From this analysis, the authors suggest that there is an evident life cycle for Web-based IS and it appears to be very short. Most organisations have been through at least two and some as many as four WIS in as many years. If, as seems likely, Websites continue to be frequently re-developed, unit costing such systems may reveal them in an unfavourable light. How the challenge of continuous evolution should be handled in business, technical and development terms requires further research.

		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 1: Quality versus Lifespan of IS and SE Applications

It is widely believed that poor Website design stems from ad hoc development practices and poor project management. If, as it was reported in section 6.2, recommended development methods are not being used, and paradoxically there is ample evidence there are many excellent Websites, how exactly are they being developed? Many professionals working in the field of WIS do not have backgrounds in either IS or SE development, so it is unsurprising that what they do fails to resemble traditional development practices. Notwithstanding this phenomena, it should not be assumed that because WIS are a recent arrival using new technologies, that IS or SE practises cannot assist. The authors presented earlier the SE knowledge areas and reference disciplines (SWEBOK) and their own, tentative, equivalent IS version (ISBOK). These are starting points for identifying the key knowledge areas and reference disciplines that will assist WIS development. It should be made clear however that the authors are not forging a path for the resurrection of heavyweight methodologies that have had limited success elsewhere. Indeed, the notion of improvisation, where problem solving is teleological and seems to be influenced by chance as well as design (Ciborra 2002), also begs for further research.

It is reasonable to propose that a more sophisticated and inclusive approach is needed using fast, dynamic and flexible methods, reflecting the business imperative to respond quickly in a competitive environment. Whatever approach is chosen (ASD, WISDM or some other) it needs to be broader than a set of new SE Web-oriented techniques since there may well be a need for an attitude change toward traditional SE practices (Carstensen & Vogelsang 2001). It must start with the establishment of a business case and include requirements determination and analysis, systems design as well as the adoption of SE techniques to ensure a robust and reliable software system. Furthermore the software system must be viewed as purposeful, part of a larger information system that is consistent with organisational objectives. Suggesting that this is just a new engineering problem to be solved and that SE principles need only be modified to Web-engineer systems is unlikely to yield a rich development environment where all the new, and old, issues are resolved. When recalling the SWEBOK guide does not include IS as a "related discipline" – it is clear that cross-fertilization from other areas and the courage to "go beyond traditional boundaries" must be embraced by both disciplines (Matsubara & Ebert 2000).

The authors contend that a broader perspective is called for - one that sees WIS development contextualized within IS theory. While the authors call for the IS community to learn from the practice and rigour of SE, it is essential that traditional, conceptual IS frameworks - that view software as part of a larger socio-technical system - prevail.

References

Ackoff, R (1967), 'Management Misinformation Systems', *Management Science*, vol. 11, no. 4, pp. 147-156.

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

Avison, D & Wood-Harper, T (1990), *Multiview: An Exploration of Information Systems Development*, Blackwell Scientific Publications, Oxford.

Avison, D, Fitzgerald, G & Powell, P (2001), 'Reflections on Information Systems Practice, Education and Research', *Information Systems Journal*, vol. 11, no. 1, pp. 3-22.

Avison, D & Wilson, D (2001), 'A viewpoint on software engineering and information systems: what we can learn from the construction industry?' *Information and Software Technology*, vol. 43, no. 13, pp. 795-799.

Avison, D & Fitzgerald, G (2003), 'Where Now for Development Methodologies?' *Communications of the ACM*, vol. 46, no. 1, pp. 78-82.

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

Barry, C & Lang, M (2001), 'A Survey of Multimedia and Web Development Techniques and Methodology Usage', *IEEE Multimedia*, vol. 8, no. 3, pp. 52-60.

Bocij, P, Chaffey, D, Greasley, A & Hickie, S (1999), 1st edn, *Business Information Systems - Technology, Development and Management*, Pearson Education, Harlow, England.

Boehm, B (1988), 'A Spiral Model for Software Development and Enhancement', *Computer*, vol. 21, no. 5, pp. 61-72.

Boehm, B (2002), 'Get Ready for Agile Methods, With Care', *IEEE Computer*, vol. 35, no. 1, pp. 64-69.

Bourque, P, Dupuis, R, Abran, A, Moore, J, Tripp, L & Wolff, S (1999), 'The Guide to the Software Engineering Body of Knowledge', *IEEE Software*, vol. 16, no. 6, pp. 35-44.

Carstensen, P & Vogelsang, L (2001), 'Design of Web-based Information Systems - New Challenges for Systems Development?' *in 9th European Conference on Information Systems*, ECIS, Bled, Slovenia, pp. 536-547.

Ciborra, C (1999), 'A Theory of Information Systems Based on Improvisation' in *Rethinking Management Information Systems* eds. Currie, W & Galliers, B, Oxford University Press, pp. 136-155.

Ciborra, C (2002), *The Labyrinths of Information*, Oxford University Press, Oxford.

Davis, GB (1982), 'Strategies for Information Requirements Determination', *IBM Systems Journal*, vol. 21, no. 1, pp. 4-30.

Enguix, C & Davis, J (1999), 'Filling the Gap: New Models for Systematic Page-based Web Application Development and Maintenance' *in International Web Engineering Workshop, WWW8*, Toronto, Canada, pp. 1-9.

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

Gellersen, H, Wicke, R & Gaedke, M (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*, Vol.29, pp. 1429-1437.

Glass, R, Vessey, I & Ramesh, V (2002), 'Research in software engineering: an analysis of the literature', *Information and Software Technology*, vol. 44, no., pp. 491-506.

Gorry, G & Scott Morton, M (1971), 'A Framework for Management Information Systems', *Sloane Management Review*, vol. Fall, no., pp. 55-70.

Highsmith, J & Cockburn, A (2001), 'Agile Software Development: the Business of Innovation', *IEEE Computer*, vol. 34, no. 9, pp. 120-122.

IEEE (1990), IEEE Standard Glossary of Software Engineering Terminology. IEEE Standards Press, New York.

Isakowitz, T, Stohr, E & Balasubramanian, P (1995), 'RMM: A Methodology for Structured Hypermedia Design', *Communications of the ACM*, vol. 38, no. 8, pp. 34-44.

Jackson, M (1998), 'Will There Ever Be Software Engineering?' *IEEE Software*, vol. 15, no. 1, pp. 36-39.

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

Kendall, K & Kendall, J (2002), 5th edn, *Systems Analysis and Design*, Prentice Hall, New Jersey.

Lang, M (2001), 'Issues and Challenges in the Development of Hypermedia Information Systems' *in Proceedings of 11th Annual Business Information Technology Conference*, Manchester, England.

Lethbridge, T (2000), 'What Knowledge is Important to a Software Professional?' *Computer*, vol. no. May, pp. 44-50.

Lowe, D & Hall, W (1999), *Hypermedia and the Web: An Engineering Approach*, Wiley and Sons.

Lynn Markus, M (1999), 'Thinking the Unthinkable - What Happens if the IS Field as we Know it Goes Away?' in *Rethinking Management Information Systems* eds. Currie, W & Galliers, B, Oxford University Press, pp. 175-203.

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

Matsubara, T & Ebert, C (2000), 'Benefits and Applications of Cross-Pollination', *IEEE Software*, vol. 17, no. 1, pp. 24-26.

Murugesan, S, Deshpande, Y, Hansen, S & Ginige, A (1999), 'Web Engineering: A New Discipline for Web-based Systems Development' *in Workshop on Web Engineering at the International Conference on Software Engineering*, ICSE, Los Angelas, pp. 1-10.

Mylonopoulos, N & Theoharakis, V (2001), 'Global Perceptions of IS Journals', *Communications of the ACM*, vol. 44, no. 9, pp. 29-33.

NATO (1968), First NATO Software Engineering Conference. NATO, Garmisch, Germany.

Naur, P & Randell, B (1969), Software Engineering: A Report on a Conference Sponsored by the NATO Scientific Committee 1968, Garmisch, Germany.

Parnas, DL (1999), 'Software engineering programs are not computer science programs', *IEEE Software*, vol. 16, no. 6, pp. 19-30.

Pour, G, Griss, M, L & Lutz, M (2000), 'The push to make software engineering respectable', *Computer*, vol. no. May, pp. 35-43.

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

Rappaport, A (1968), 'Management Misinformation Systems - Another Perspective', *Management Science*, vol. 15, no. 4, pp. 133-136.

Reifer, D (2002), 'How good are agile methods?' *IEEE Software*, vol. 19, no. 4, pp. 16- 18.

Rockart, J & Treacy, M (1982), 'The CEO Goes On-Line', *Harvard Business Review*, vol. 60, no. 1, pp. 74-79.

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

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

Vidgen, R, Avison, D, Wood, B & Wood-Harper, T (2003), *Developing Web Information Systems*, Butterworth Heinemann.

Wasserman, A (1996), 'Toward a Discipline of Software Engineering', *IEEE Software*, vol. 13, no. 6, pp. 23-31.