Methodologies for Developing Information Systems: A Historical Perspective

David Avison[1] and Guy Fitzgerald[2]
1 David Avison, ESSEC Business School, Department of Information Systems and Decision Sciences (SID), 95021 Cergy-Pontoise, France.
avison@essec.fr, WWW home page:
http://domservices.essec.fr/domsite/cv.nsf/WebCv/David+Avison
2 Brunel University, Department of Information Systems, Computing and Mathematics, Uxbridge, UB8 3PH, UK. guy.fitzgerald@brunel.ac.uk
WWW home page: http://www.brunel.ac.uk/~csstggf

Abstract. For the past 30 years and more, Information Systems Development (ISD) has been at the heart of the study and practice of Information Systems (IS). This paper examines the history of ISD methodologies and looks at some of the trends and issues concerning ISD, and shows how these have been reflected in methodologies and how organizations use (or do not use) them. Discussion of the present state of the field is followed by a discussion of possible future directions.

1 Introduction

In this paper we celebrate the 30th anniversary of IFIP Technical Committee 8, which through its working groups (especially, but not limited to, WG 8.1 and WG 8.2) has put ISD amongst its major work and contribution. We also reflect on the coincidental publication of the 4th edition of [1], a book which has a history of merely 18 years. These reflections enable us to build on and bring up to date our short *Communications of the ACM* paper [2] to examine the history of methodologies for ISD as well as reviewing the current position and suggesting some pointers to the future.

Systems development activities have been around for as long as computers but although the development of technology has been phenomenal, the development of a generally-accepted systematic approach or approaches to utilize that technology effectively has been slower and this may have been to some extent a limiting factor on the speed of progress in the use of the technology. In some other practical domains there is a 'one correct way of doing something' – why has this not been the same for ISD?

This paper examines some of the trends and issues related to ISD over time. We identify four eras: pre-methodology, early methodology, methodology and post-methodology. This could be perceived as a ‘maturity model for ISD’ as some organizations may be in different stages in the same countries, whereas different countries may be in general in front of or behind others. Thus it is risky, if appropriate at all, to put actual dates on the ‘eras’ as they are more stages of ISD practice. Nevertheless we do suggest approximate decades in which each was at the fore in North America, Europe and Australia. The current era has been one of the most difficult to deal with as it is not at all clear how it will pan out. Unlike for previous eras, we do not have the benefit of hindsight. However, it would appear that the period is perhaps surprisingly one of much greater stability - methodologies are not being invented (or reinvented) as before, many methodologies discussed in previous eras do not now have much following in practice and there is some consolidation in the field. Where development is not outsourced in some way, there is emphasis on approaches which aim at developing a product with greater speed and flexibility.

2 Pre-Methodology Era

Early computer applications, up to around the time TC8 was established, were implemented without an explicit ISD methodology. We thus characterise this as the pre-methodology era. In these early days, the emphasis of computer applications development was on programming. The needs of the users were rarely well established with the consequence that the design was frequently inappropriate to the application needs. The focus of effort was on getting something working and overcoming the limitations of the technology, such as making an application run in restricted amounts of memory. A particular problem was that the developers were technically trained but rarely good communicators. The dominant ‘methodology’ was rule-of-thumb and based on experience. This typically led to poor control and management of projects. For example, estimating the date on which the system would be operational was difficult, and applications were frequently delivered late and above budget. Programmers were usually overworked, and spent a large proportion of their time correcting and enhancing the few applications that were operational. These problems led to a growing appreciation of the desirability for standards and a more disciplined approach to the development of IS in organisations. Thus the first ISD methodologies were established. Although this era was common in many large European and North American organizations of the ‘60s, the characteristics can be seen in some companies developing applications on PCs nowadays.

3 Early Methodology Era

As a reaction to the failings of the pre-methodology era:

1. There was a growing appreciation of that part of the development of the system that concerns analysis and design and therefore of the potential role of the systems analyst.

2. There was a realisation that as organisations were growing in size and complexity, it was desirable to move away from one-off solutions to a particular problem and towards more integrated IS.
3. There was an appreciation of the desirability of an accepted methodology for the development of IS.

These reflections led to the evolution of the Systems Development Life Cycle (SDLC) or waterfall model as the approach to develop IS. This was an early methodology, although at the time it was not yet known as such. It included phases, procedures, tasks, rules, techniques, guidelines, documentation, training programs and tools. The waterfall model consisted of a number of stages of development that were expected to be followed sequentially. These stages typically consisted of feasibility study, systems investigation, analysis, design, and implementation, followed by review and maintenance, and this was the approach widely used in the 1970s and even some of the 1980s, and is still a basis for many methodologies today.

The SDLC has been well tried and tested and the use of documentation standards helps to ensure that proposals are complete and that they are communicated to users and computing staff. The approach also ensures that users are trained to use the system. There are controls and these, along with the division of the project into phases of manageable tasks with deliverables, help to avoid missed cutover dates and disappointments with regard to what is delivered. Unexpectedly high costs and lower benefits are also less likely. It enables a well-formed and standard training scheme to be given to analysts, thus ensuring continuity of standards and systems.

However, there are serious limitations to the approach along with limitations in the way it is used. Some potential criticisms are: Failure to meet the needs of management (due to the concentration on single applications at the operational level of the organization); Unambitious systems design (due to the emphasis on ‘computerizing’ the existing system); Instability (due to the modelling of processes which are unstable because businesses and their environments change frequently); Inflexibility (due to the output-driven orientation of the design processes which makes changes in design costly); User dissatisfaction (due to problems with the documentation and the inability for users to ‘see’ the system before it is operational); Problems with documentation (due to its computer rather than user orientation and the fact that it is rarely kept up-to-date); Application backlog (due to the maintenance workload as attempts are made to change the system in order to reflect user needs); and the Assumption of ‘green field’ development (due to the tradition of a new IS ‘computerizing’ manual systems, an assumption inappropriate as IS now largely replace or integrate with legacy systems).

4 Methodology Era

As a response to one or more of the above limitations or criticisms of the SDLC, a number of different approaches to IS development emerged and what we term ‘the methodology era’ began. Methodologies can be classified into a number of movements. The first are those methodologies designed to improve upon the traditional waterfall model. A second movement is the proposal of new methodologies that are somewhat different to the traditional waterfall model (and from each other).

Since the 1970s, there have been a number of developments in techniques and tools and many of these have been incorporated in the methodologies exemplifying the modern version of the waterfall model. The various CRIS conferences of IFIP WG8.1 were important here (see, for example [3], published following the third of these conferences and provided an excellent overview of earlier ISD and the early shoots of more sophisticated approaches). Techniques incorporated include entity-relationship modelling, normalisation, data flow diagramming, structured English, action diagrams, structure diagrams and entity life cycles. Tools include project management software, data dictionary software, systems repositories, drawing tools and, the most sophisticated, computer-assisted software (or systems) engineering (CASE) tools (now broadened in scope and more frequently referred to as toolsets). The incorporation of these developments addresses some of the criticisms discussed in section 3. The blended methodologies Merise [4], SSADM [5] and Yourdon Systems Method [6] could be said to be updated versions of the waterfall model. The later method engineering movement (see for example [7], a collaboration of IFIP WG 8.1 and WG 8.2) developed the practice of blending methods and techniques further. Although these improvements have brought the basic model more up to date, many users have argued that the inflexibility of the life cycle remains and inhibits most effective use of computer IS.

It is possible to classify alternative approaches that developed during the 1980s and beyond within a number of broad themes including: systems, strategic, participative, prototyping, structured, and data. Each of these broad themes gave rise to one or more specific methodologies.

General systems theory attempts to understand the nature of *systems*, which are large and complex. Organisations are open systems, and the relationship between the organisation and its environment is important. By simplifying a complex situation, we may be *reductionist*, and thereby distort our understanding of the overall system. The most well-known approach in the IS arena to address this issue is Checkland's soft systems methodology (SSM) [8]. It includes techniques, such as rich pictures, which help the users understand the organisational situation and therefore point to areas for organisational improvement through the use of IS.

Strategic approaches stress the pre-planning involved in developing IS and the need for an overall strategy. This involves top management in the analysis of the objectives of their organisation. These approaches counteract the possibility of developing IS in a piecemeal fashion. IBM's Business Systems Planning is an early example of this approach and business process re-engineering [9] is part of this overall movement.

In *participative* approaches, the role of all users is stressed, and the role of the technologist may be subsumed by other stakeholders of the information system. If the users are involved in the analysis, design and implementation of IS relevant to their own work, particularly if this takes the form of genuine decision-making, these users are likely to give the new IS their full commitment when it is implemented, and thereby increase the likelihood of its success. ETHICS [10] stresses the participative nature of ISD, following the socio-technical movement and the work of the Tavistock Institute and embodies a sustainable ethical position.

A *prototype* is an approximation of a type that exhibits the essential features of the final version of that type. By implementing a prototype first, the analyst can show the users inputs, intermediary stages, and outputs from the system. These are not diagrammatic approximations, which tend to be looked at as abstract things, or technically-oriented documentation, which may not be understood by the user, but the actual data on computer paper or on terminal or workstation screens. Toolsets of various kinds can all enable prototyping. These have become more and more powerful over the last few years. Rapid Application Development [11] is an example of an approach that embodies prototyping.

Structured methodologies are based on functional decomposition, that is, the breaking down of a complex problem into manageable units in a disciplined way. These approaches tend to stress techniques, such as decision trees, decision tables, data flow diagrams, data structure diagrams, and structured English, and tools such as systems repositories.

Whereas structured analysis and design emphasises processes, *data analysis* concentrates on understanding and documenting data. It involves the collection, validation and classification of the entities, attributes and relationships that exist in the area investigated. Even if applications change, the data already collected may still be relevant to the new or revised systems and therefore need not be collected and validated again. Information Engineering [12], for example, has a data approach as its centre.

In the 1990s there was what might be perceived as a second wave of methodologies. *Object-oriented* ISD became another ‘silver bullet’ [13] and has certainly made a large impact on practice. Yourdon [14] exposition argues that the approach is more natural than data or process-based alternatives, and the approach unifies the ISD process. It also facilitates the realistic re-use of software code. Coad and Yourdon [15] suggest a number of other motivations and benefits for object-oriented analysis, including: the ability to tackle more challenging problem situations because of the understanding that the approach brings to the problem situation; the improvement of analyst-user relationships, because it is not computer-oriented; the improvement in the consistency of results, because it models all aspects of the problem in the same way; and the ability to represent factors for change in the model so leading to a more resilient model. To some extent, therefore, it has replaced the singular process and data emphases on ISD.

Incremental or evolutionary development (often including prototyping) has also been a feature of 1990s development. Incremental development has the characteristic of building upon, and enhancing, the previous versions rather than developing a new system each time. Incremental development aims to reduce the length of time that it takes to develop a system and it addresses the problem of changing requirements as a result of learning during the process of development (‘timebox’ development, see [11]). The system to be developed is divided up into a number of components that can be developed separately. This incremental approach is a feature of DSDM [16]. Recently developing applications from components from different sources has gained popularity [17] as has obtaining open source software components (reflected in [18,19]).

Some methodologies have been devised for specific types of application. These specific-purpose methodologies include Welti [20] for developing ERP applications; CommonKADS [21] for knowledge management applications; Process Innovation

[22] for business process reengineering applications, Renaissance [23] supporting the reverse engineering of legacy systems and WISDM [24] for web development.

We characterise the above as the methodology era because of the apparent proliferation of different types of methodologies, and their increasing maturity. The work of IFIP WG 8.2 has tended to emphasize the human and organizational aspects of ISD (see for example [18,25]).

Many users of methodologies have found the waterfall model and the alternative methodologies outlined above unsatisfactory. Most methodologies are designed for situations, which follow a stated, or more usually, an unstated 'ideal type'. However, situations are all different and there is no such thing as an 'ideal type' even though situations differ depending on, for example, their complexity and structuredness, type and rate of change in the organisation, the numbers of users affected, their skills, and those of the analysts. Further, most methodology users expect to follow a step-by-step, top-down approach to ISD where they carry out a series of iterations through to project implementation. In reality, in any one project, this is rarely the case, as some phases might be omitted, others carried out in a different sequence, and yet others developed further than espoused by the methodology authors. Similarly, particular techniques and tools may be used differently or not used at all in different circumstances.

There have been a number of responses to this challenge. One response is to suggest a *contingency approach* to ISD (as against a prescriptive approach), where a structure is presented but stages, phases, tools, techniques, and so on, are expected to be used or not (or used and adapted), depending on the situation. Those characteristics which will affect the choice of a particular combination of techniques, tools and methods for a particular situation could include the type of project, whether it is an operations-level system or a management information system, the size of the project, the importance of the project, the projected life of the project, the characteristics of the problem domain, the available skills and so on. Multiview [26] is such a contingency framework.

Many attempts have been made to compare and contrast this diversity of methodologies. Olle [27] provides one example emanating from IFIP WG 8.1. Avison and Fitzgerald [1] compare methodologies on the basis of philosophy (paradigm, objectives, domain and target); model; techniques and tools; scope; outputs; and practice (background, user base, players, and product). In relation to the number of methodologies in existence, some estimates suggested that there were over 1,000 brand name methodologies world-wide, although we are rather skeptical of such a high figure, there is no doubt that methodologies had proliferated, although many of these were similar and differentiated only for marketing purposes. However, the characterization of this as the methodology era does not mean that every organization was using a methodology for systems development. Indeed, some were not using a methodology at all but most, it seems, were using some kind of in-house developed or tailored methodology, typically based upon or heavily influenced by a commercial methodology product.

5 Post-Methodology Era

We identify the current situation as the post-methodology era, in the sense that we now perceive methodologies as having moved beyond the pure methodology era. Now it seems that although some organisations still use a methodology of some kind there is enough of a re-appraisal of the beneficial assumptions of methodologies, even a backlash against methodologies, together with a range and diversity of non-methodological approaches, to justify the identification of an era of reflection.

Methodologies were often seen as a panacea to the problems of traditional development approaches, and they were often chosen and adopted for the wrong reasons. Some organisations simply wanted a better project control mechanism, others a better way of involving users, still others wanted to inject some rigour or discipline into the process. For many of these organisations, the adoption of a methodology has not always worked or been the total success its advocates expected. Indeed, it was very unlikely that methodologies would ever achieve the more overblown claims made by some vendors and consultants. Some organisations have found their chosen methodology not to be successful or appropriate for them and have adopted a different one. For some this second option has been more useful, but others have found the new one not to be successful either. This has led some people to the rejection of methodologies in general. In the authors' experience this is not an isolated reaction, and there is something that might be described as a backlash against formalised ISD methodologies.

This does not mean that methodologies have not been successful. It means that they have not solved all the problems that they were supposed to. Many organisations are using methodologies effectively and successfully and conclude that, although not perfect, they are an improvement on what they were doing previously, and that they could not handle their current systems development load without them.

Yet in the post-methodology era, there are many reasons why organizations are questioning the need to adopt any sort of methodology, as follows: *Productivity:* The first general criticism of methodologies is that they fail to deliver the suggested productivity benefits; *Complexity:* Methodologies have been criticized for being over complex; *'Gilding the lily':* Others argue that methodologies develop any requirements to the ultimate degree, often over and above what is legitimately needed.; *Skills:* Methodologies require significant skills in their use and processes; *Tools:* The tools that methodologies advocate are difficult to use, expensive and do not generate enough benefits; *Not contingent:* Methodologies are not contingent upon the particularities of the project; *One-dimensional approach:* Methodologies usually adopt only one approach to the development of projects, which does not always address the underlying issues or problems; *Inflexible:* Methodologies may be inflexible and may not allow changes to requirements during development; *Invalid or impractical assumptions:* Most methodologies make a number of simplifying yet potentially invalid assumptions, such as a stable external and competitive environment; *Goal displacement:* This refers to the unthinking use of a methodology and to a focus on following the procedures to the exclusion of the real needs of the project being developed. De Grace and Stahl [28] have termed this 'goal displacement' and Wastell [29] talks about the 'fetish of technique', which inhibits creative thinking; *Problems of building understanding into methods:* Introna and

Whitley [30] argue that some methodologies assume that understanding can be built into the method process. They call this ‘method-ism’ and believe it is misplaced; *Insufficient focus on social and contextual issues:* The growth of scientifically based highly functional methodologies has led some commentators to suggest that we are now suffering from an overemphasis on the narrow, technical development issues and that not enough emphasis is given to the social and organizational aspects of systems development [31]; *Difficulties in adopting a methodology:* Some organizations have found it hard to adopt methodologies in practice, partly due to the resistance of users to change; *No improvements:* Finally in this list, and perhaps the acid test, is the conclusion of some that the use of methodologies has not resulted in better systems, for whatever reasons. This is obviously difficult to prove, but nevertheless the perception of some is that ‘we have tried it and it didn’t help and it may have actively hindered’. The work of IFIP WG 8.6 on the diffusion of technology has much to teach us here.

We thus find that for some, the great hopes in the 1980s and 1990s, that methodologies would solve most of the problems of ISD have not come to pass. Strictly speaking, however, a distinction should be made in the above criticisms of methodologies between an inadequate methodology itself and the poor application and use of a methodology. Sometimes a methodology vendor will argue that the methodology is not being correctly or sympathetically implemented by an organization. Whilst this may be true to some extent, it is not an argument that seems to hold much sway with methodology users. They argue that the important point is that they have experienced disappointments in their use of methodologies.

One reaction to this is to reject the methodology approach altogether. A survey conducted in the UK [32] found that 57% of the sample were claiming to be using a methodology for systems development, but of these, only 11% were using a commercial development methodology unmodified, whereas 30% were using a commercial methodology adapted for in-house use, and 59% a methodology which they claimed to be unique to their organization, i.e. one that was internally developed and not based solely on a commercial methodology.

A variety of reactions to the perceived problems and limitations of methodologies exist and we now examine some of these. We begin by considering external development, but if the choice is made to develop internally, then users may demand that the methodology that they do use needs to be refined and improved (just as they were in the methodology phase). On the other hand, users may prefer to adapt the methodology according to the particular needs of each circumstance following a contingency approach, or even more informally and risky, an ad hoc approach. In some organizations speed as well as flexibility has become watchwords, and rapid and agile approaches have gained more adherents and the tendency towards more user and customer involvement strengthened. Finally we suggest that we are in a more stable environment than in any time since the early days of ISD methodologies and the foundation of IFIP TC8, and we see the immediate future being one of consolidation.

5.1 External Development

Some organisations have decided not to embark on any more major in-house system development activities but to buy-in all their requirements in the form of packages. This is regarded as a quick and relatively cheap way of implementing systems for organisations that have fairly standard requirements. A degree of package modification and integration may be required which may still be undertaken in-house. Clearly the purchasing of packages has been commonplace for some time, but the present era is characterised by some organisations preferring package solutions. Only systems that are strategic or for which a suitable package is not available would be considered for development in-house. The package market is becoming increasingly sophisticated and more and more highly tailorable packages are becoming available. Sometimes open source components can be 'packaged' to form the application.

Enterprise resource planning (ERP) systems have become particularly popular with large corporations since the mid '90s. The key for these organisations is ensuring that the correct trade-off is made between a 'vanilla' version of a standard package, which might mean changing some elements of the way the business currently operates, and a package that can be modified or tailored to reflect the way they wish to operate.

For others, the continuing problems of systems development and the backlash against methodologies has resulted in the outsourcing and/or offshoring of systems development. The client organisation no longer has any great concern about how the systems are developed. They are more interested in the end results and the effectiveness of the systems that are delivered. This is different to buying-in packages or solutions, because normally the management and responsibility for the provision and development of appropriate systems is given to a vendor. The client company has to develop skills in selecting the correct vendor, specifying requirements in detail and writing and negotiating contracts rather than thinking about system development methodologies.

5.2 Continuing Refinement and Improvement

One reaction to the criticisms that users of methodologies make is for authors and suppliers to 'get methodologies right'. For some there is the continuing search for the methodology holy grail. Methodologies will probably continue to be developed from time to time and, more likely, existing ones evolve. Most methodologies have some gaps in them or, if not complete gaps, they have areas that are treated much less thoroughly than others. For example, rich pictures, cognitive mapping, lateral thinking, scenario planning, case-based reasoning, and stakeholder analysis represent some of the techniques that are rarely included in methodologies, but we see good reasons for their inclusion [1]. Adams and Avison [33] suggest how analysts may choose between techniques as well as potential dangers in their use. Similarly, toolsets have developed greatly over the period from simple drawing tools to very comprehensive toolsets, some designed to support one particular methodology and others to support ISD as a whole.

In particular, methodologies are now appearing to deal with systems development for the web. This, it is argued, has some special characteristics, which make traditional methodologies inappropriate. Baskerville and Pries-Heje [34], for example, list these as time pressure, vague requirements, prototyping, release orientation, parallel development, fixed architecture, coding your way out, negotiable quality, dependence on good people, and the need for structure. The WISDM methodology [24] also addresses web development. Some of the methodologies devised for web development use the term 'agile' to characterise the need for flexibility and adaptability in web development which distinguishes them from traditional approaches (see section 5.4).

5.3 Ad-hoc Development and Contingency

This might be described as a return to the approach of the pre-methodology days in which no formalized methodology is followed. The approach that is adopted is whatever the developers understand and feel will work. It is driven by, and relies heavily on, the skills and experiences of the developers. Truex et al. [35] represents part of this backlash against conventional methodologies as they talk of amethodological and emergent ISD. This is perhaps an understandable reaction, but it runs the risk of repeating the problems encountered prior to the advent of methodologies.

We see a contingent approach as providing a positive response and see this as offering a good balance. A contingency approach to ISD presents a structure to help the developers, but tools and techniques are expected to be used or not (or used and adapted), depending on the situation. Situations might differ depending on, for example, the type of project and its objectives, the organization and its environment, the users and developers and their respective skills. The type of project might also differ in its purpose, complexity, structuredness, and degree of importance, the projected life of the project, or its potential impact. The contingency approach is a reaction to the ‘one methodology for all developments’ approach that some companies adopted, and is recognition that different characteristics require different approaches and we see it gaining increasing importance.

5.4 Agile Development

When following agile development, requirements are ‘evolved’ and, as the agile manifesto’ [36] suggests, the approach emphasizes the involvement of users and customers in a joint approach to ISD more than processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation and responding to change over following a plan (see also [37]. Working software is delivered in smaller chunks than traditionally, but in a much shorter time span. Changing requirements are accepted as the norm and even welcomed. These principles conform more to today’s ISD needs than many of the ISD methodologies of the ‘methodology era’, for example reacting to ‘Internet speed development’ [34]. These features are found in extreme programming (XP) and SCRUM as well as ISD approaches, such as DSDM [38].

5.5 Consolidation

In the previous three previous editions of Avison and Fitzgerald [1] published in 1988, 1995 and 2002, we discussed 9, 12 and 34 themes; 8, 11 and 37 techniques; 7, 6 and 12 tools; and 8, 15 and 32 methodologies respectively. Despite our best research endeavors, the numbers have not increased in the 2006 edition, indeed there has been a decline in numbers as some methodologies (and their associated techniques and tools) fall into disuse. However, this does not necessarily indicate a fall into disuse of frameworks and methodologies for ISD as a whole, but rather a consolidation process, indeed we see some methodology-era methodologies being used effectively and successfully as well as agile and contingent approaches to ISD. This may also suggest greater maturity in the field of IS generally and we see this consolidation process continuing.

6 Conclusion

This paper has attempted to review, albeit briefly, the history and drivers of ISD methodologies. We have used our analysis to reflect on and discuss the current situation, identified as the post-methodology era. This has involved the identification of various eras of methodologies. Our present era is perhaps best described as an era of methodology reappraisal, resulting in a variety of reactions. Although we believe that it is unlikely that any single approach will provide the solution to all the problems of ISD, we do now see a change. Diversity of methodologies and multiplication of similar methodologies has been replaced by some consolidation: ISD has entered a maturing phase of greater stability.

References

1. D.E.Avison and G.Fitzgerald. Information Systems Development: Methodologies, Techniques and Tools. 4th edition, McGraw-Hill, Maidenhead. (2006).

2. D.E. Avison and G. Fitzgerald. Where now for Development Methodologies?, Communications of the ACM, (January, 2003).

3. T.W. Olle, H.G. Sol, and A.A.Verrijn-Stuart (eds). Information Systems Design Methodologies: Improving the Practice, North Holland, Amsterdam (1986).

4. P.T. Quang and C. Chartier-Kastler. Merise in Practice. Macmillan, Basingstoke (1991).

5. M. Eva. SSADM Version 4: A User's Guide. McGraw-Hill, Maidenhead. (1994).

6. Yourdon Inc. Yourdon Systems Method: Model-Driven Systems Development. Yourdon Press, Englewood Cliffs (1993).

7. S. Brinkkemper, K. Lyytinen, and R.J. Welke (eds). Method Engineering: Principles of Method Construction and Tool Support, Kluwer, Boston (1996).

8. P. Checkland and J. Scholes. Soft Systems Methodology in Action. Wiley, Chichester (1990).

9. M. Hammer and J. Champy. Reengineering the Corporation: A Manifesto for Business Revolution. Harper Business, New York (1993).

10. E. Mumford. Effective Requirements Analysis and Systems Design: The ETHICS Method. Macmillan, Basingstoke (1995).

11. J. Martin. Rapid Application Development. Prentice Hall, Englewood Cliffs (1991).

12. J. Martin. Information Engineering. Prentice Hall, Englewood Cliffs (1989).

13. G. Booch. Object Oriented Design with Applications. Benjamin/Cummings, Redwood City (1991).

14. E. Yourdon. Object-oriented Systems Design, An Integrated Approach. Prentice Hall, Englewood Cliffs (1994).

15. P. Coad and E. Yourdon. Object Oriented Analysis. Prentice Hall, Englewood Cliffs (1991).
16. DSDM Manual Version 3 DSDM Consortium, Tesseract, Surrey (1998).
17. V. Sugumaran and V.C. Storey. A semantic-based approach to component retrieval, Database for Advances in Information Systems, 34, 3 (2003).
18. N.L. Russo, B. Fitzgerald, and J. DeGross (eds) Realigning Research and Practice in Information Systems development. Kluwer, Boston (2001).
19. J. Feller and B. Fitzgerald. Understanding Open Source Software Development, Addison Wesley, Harlow (2002).
20. N. Welti. Successful SAP R/3 Implementation, Addison-Wesley, Harlow (1999).
21. G. Schreiber, H. Akkermans, A. Anjewierden, R. de Hoog, N. Shadbolt, W. Van de Velde, and B.J. Wielinga. Knowledge Engineering and Management: The Common KADS Methodology, MIT Press, Cambridge (2000).
22. T.H. Davenport. Process Innovation, Harvard Business School, Boston (1993).
23. I. Warren. The Renaissance of Legacy Systems, Springer-Verlag (1999).
24. R. Vidgen, D.E. Avison, R. Wood, and A.T. Wood-Harper. Developing Web Information Systems, Butterworth-Heinemann, London (2002).
25. D.E. Avison, J. Kendall, and J. DeGross (eds). Human, Organizational and Social Dimensions of IS Development. North Holland, Amsterdam (1993).
26. D.E. Avison, A.T. Wood-Harper, R. Vidgen, and R. Wood. Multiview: A Further Exploration in IS Development, McGraw-Hill, Maidenhead (1996).
27. T.W. Olle. Information Systems Methodologies: A Framework for Understanding, Addison Wesley, Harlow (1988).
28. P. De Grace and L. Stahl. The Olduvai Imperative: CASE and the State of Software Engineering Practice. Prentice Hall, Englewood Cliffs (1993).
29. D. Wastell. The Fetish of Technique: methodology as a social defence. Information Systems Journal, 6, 1 (1996).
30. L. Introna and E. Whitley. Against method-ism: Exploring the limits of method, Information Technology and People, 10, 1, 31-45 (1997).
31. R. Hirschheim, H.K. Klein, and K. Lyytinen. Exploring the intellectual structures of information system development: A social action theoretic analysis,
Accounting, Management and Information Technologies, 6, 1/2 (1996)
32. G. Fitzgerald, A. Philippides, and P. Probert. Information Systems Development, Maintenance and Enhancement: Findings from a UK Study, International Journal of Information Management, 40 (2), 319-329 (1999).
33. C. Adams and D.E. Avison. Dangers Inherent in the Use of Techniques: Identifying Framing Influences, Information Technology and People, 16, 2 (2003).
34. R. Baskerville and J. Pries-Heje. Racing the e-bomb: How the Internet is redefining IS development methodology, in N. L. Russo, et al. (2001)
35. D.Truex, R. Baskerville, and H. Klein. Growing Systems in Emergent Organizations, Communications of the ACM (42:8), (1999), pp. 117-123.
36. K. Beck et al. Agile Manifesto, available at http://agilemanifesto.org/ (2001).
37. J. Highsmith. Agile Software Development Ecosystems, Addison-Wesley, Harlow (2002).
38. J. Stapleton. DSDM: A Framework for Business Centred Development, Addison-Wesley, Harlow (2002).