Persistent problems and practices in information systems development

Karlheinz Kautz, Sabine Madsen & Jacob Nørbjerg

Department of Informatics, Copenhagen Business School, Howitzvej 60, DK-2000, Copenhagen, Denmark, email: khk.inf@cbs.dk

Abstract. *This paper identifies and discusses the persistent problems and development practices of information systems development (ISD). A critical examination and comparison of past times' 'traditional' and present-day 'web-based' development shows that contemporary ISD can be seen as an accentuated evolution – rather than a revolution – of well-known challenges and solutions. On this basis, (1) diversity; (2) knowledge; and (3) structure are identified as inherent and interrelated problems, while the practices for coping with these three challenges are (a) organization and specialization; (b) constant verbal communication and negotiation; and (c) pragmatic application of certain development methods and methodical concepts. We conclude that more research on the occurrence and interaction of problems and practices at, and between, different contextual levels (e.g. the business environment, company, project, team and individual levels) is needed to understand and assess (the gap between) 'observed practice' and 'good practice' across the many types of Web and non-Web ISD projects conducted today. We outline a possible research agenda to investigate these issues.*

Keywords: information systems development, empirical studies of ISD practice

INTRODUCTION

Information and communication technologies are rapidly evolving. Hardware processing and storage capacities have been increasing at tremendous rates and computer-based devices and applications are spreading into nearly every human life sphere. The tools, techniques and processes used for producing information systems (IS) are undergoing profound changes as well. High-level languages and program generators have replaced machine languages. Similarly, *ad hoc* approaches to information systems development (ISD) have been replaced by development methods supported by powerful CASE tools, and new systems are now rarely built from scratch, but based on standard software packages or assembled from module libraries.

Unsurprisingly, these changes are accompanied by frequent claims about their fundamental and paradigmatic nature. Already in the mid-1960s, early third-generation languages such as

FORTRAN and COBOL were expected to put an end to the programming profession – a prophecy that was repeated when CASE tools and fourth-generation languages appeared in the 1980s (Friedman, 1989). Structured analysis and design methods (Yourdon, 1989) and, later, Object Oriented Methods such as Rational's Unified Process (RUP) (Jacobson *et al*., 1999) have also been expected to revolutionize ISD and put an end to low quality systems, budget overruns, etc.

The rate of technological and developmental change obviously creates challenges for IS research. As an example, consider the World Wide Web. The World Wide Web has been labelled a fundamentally new medium of human communication, and many researchers see web-based systems as a special class of IS that require new concepts and development approaches (Lyytinen *et al*., 1998; Pressman, 1998; Bansler *et al*., 2000; Alter *et al*., 2001; Carstensen & Vogelsang, 2001; Murugesan *et al*., 2001; Baskerville & Pries-Heje, 2002; Holck, 2002). Murugesan *et al*. advocate Web engineering as a new, necessary and distinct area that addresses the unique challenges of Web-based development (Murugesan *et al*., 2001), while others introduce concepts such as development @Internet Speed and Short Cycle Time Systems Development to capture the special characteristics of contemporary ISD (Baskerville & Pries-Heje, 2002; Ramesh *et al*., 2002; Baskerville & Pries-Heje, 2004).

However, extending the position of Kautz & Nørbjerg (2003), we argue that the numerous changes in information technologies and applications have not significantly changed the underlying, persistent characteristics of ISD. We therefore consider it problematic for ISD researchers to concentrate only on the unique features and apparent newness of certain IS phenomena, such as all Web-related matters, while disregarding fundamental concerns and challenges (still) at the core of the field. In this paper, we set out (a) to identify and discuss the persistent problems of ISD and (b) to outline the contours of a possible research agenda that explores such problems and their intricate solutions.

The ISD literature contains numerous research contributions and empirical studies about ISD in the past and about the characteristics, development practices and problems of (Web-based) ISD today. What is missing is a conceptual integration and synthesis of the many, sometimes conflicting discussions and findings in the existing body of knowledge.

In this paper, we adopt an approach where we first provide the results of a careful study of ISD literature for the purpose of creating a consistent basis and reading of the field (Klein & Myers, 1999). As the ISD literature is vast and as it is one of the newest and recently most debated phenomena, we have opted for Web development as a case for identification and elaboration of the persistent problems of ISD based on a critical examination and comparison of past times' 'traditional' and present-day 'Web-based' development.

In keeping with the research topic and our interpretive approach, our analysis and our understanding of ISD has come about through an iterative process of interpretation, comparison and interlacing of prior research and empirical data. The empirical data are based on four empirical studies. Two studies have been performed by the authors of this paper (Nørbjerg, 1994; Kautz & Madsen, 2003), and two have been selected from the literature because of their thoroughness and high level of empirical grounding (Curtis *et al*., 1988; Baskerville & Pries-Heje, 2004). For this paper, all empirical data and earlier published write-ups have been reread and anal-

ysed anew by focusing on the differences and similarities between the cases and case summaries, and cross-case comparison tables have been outlined. The arguments for, and choice of, the theories and approaches that constitute our understanding are therefore equally informed by both the literature and practice, by deduction as well as induction.

The paper proceeds as follows. The next section contains a comparative survey of the literature on Web-based and historic/traditional ISD, and we uncover characteristics and claims that are made about them. On this broader basis, the third section then focuses more specifically on past and contemporary practice. It presents the four empirical case studies of historic/traditional ISD and contemporary Web-based development, as well as their inherent difficulties and coping mechanisms, respectively. In the fourth section, the identified fundamental and persistent problems and their current and potential solutions are discussed and areas for future research are proposed. The concluding section summarizes the outline of the suggested research agenda and sums up our research findings.

WEB DEVELOPMENT: REVOLUTION OR EVOLUTION

In this section, we compare the characteristics of, and the claims made about, Web development in relation to traditional ISD. Inspired by Walsham (1993) and following Kautz (2004) in considering ISD as an innovative activity that can be better understood by its content, context and processes, this section is structured around a focus on (1) technology and IS; (2) context and conditions; and (3) development practices.

Technology and IS

Internet technology supplies a unified and relative standard user interface across applications and platforms through the concept of a browser (Lyytinen *et al*., 1998; Deshpande & Hansen, 2001). Technologically, the internet is relatively simple, but from a social and organizational perspective it is a remarkable phenomenon (Vidgen, 2002), because, as Angehrn (1997) asserts, the internet has opened up a new marketing and communication channel that can be accessed flexibly, globally, and at high speed and low cost, allowing for new and extended business opportunities and interactivity that goes beyond the traditional physical and geographical boundaries. Internet technology makes possible a range of applications from the simple brochure-like Web presence to sophisticated systems that provide customized information as well as communication, distribution and transaction facilities (Angehrn, 1997). Web-based IS are evolving from simple, stand-alone Web sites into complex business critical systems, based on integration with databases, legacy systems and other back-office applications (e.g. Lyytinen *et al*., 1998; Pressman, 2000; Barry & Lang, 2001; Vidgen, 2002). Murugesan *et al*. (2001) explain that 'the Web has evolved very rapidly into a global environment for delivering all kinds of applications, ranging from small-scale, short-lived services to large-scale enterprise applications distributed across the internet and corporate intranets' (p. 5). Deshpande & Hansen (2001) suggest that two broad categories of applications can be identified: (a) informational

applications concerned with dissemination and presentation of information, and (b) software applications in the usual sense (however, no definition of 'software in the usual sense' is provided). A similar distinction is made in other contributions, where (a) web *sites* are perceived to be oriented towards the user interface with an emphasis on information publishing, advertising and branding, while (b) Web-based information *systems* typically are assumed to have more traditional back-end functionality (e.g. Isakowitz *et al*., 1995; Howcroft & Carroll, 2000; Carstensen & Vogelsang, 2001).

Web sites and systems are by now a part of the way a company presents itself and markets its products to the rest of the world. This leads to new kinds of requirements to the 'look and feel' and marketing capability of such systems. It also means that new development activities are included into the life cycle to design, produce and implement a pleasant and consistent interface able to attract and keep the attention of users. These interfaces often include advanced graphics and multimedia features such as sound, animation and video streaming (Bansler *et al*., 2000; Carstensen & Vogelsang, 2001; Murugesan *et al*., 2001; Holck, 2002; Vidgen, 2002).

Web development projects also often include content production. Web development projects not only produce technical frameworks and facilities for information storage and presentation, but they produce the information as well (Murugesan *et al*., 2001). In this sense, Web development seems to transcend the traditional boundary between, on the one hand, producing software with data processing and storage capabilities, and on the other, using the software to enter, process and retrieve data. The result, according to some, is that the boundary between designers and users of Web-based systems becomes blurred (Bansler *et al*., 2000).

The context and conditions of Web development

New skills and organizations

To meet the new types of demands, Web development projects depend on an increasing and increasingly varied sets of skills besides the technical skills needed in 'traditional' ISD. Web development projects are staffed with new kinds of systems developers recruited from areas such as marketing, graphics design, and video or film production, not to mention the end users and the specialists needed to produce and communicate content within various domains, e.g. health and law (Greenbaum & Stuedahl, 2000; Hansen *et al*., 2001; Vidgen, 2002). Often these new systems developers lack basic information technology (IT) skills (Carstensen & Vogelsang, 2001). At the same time, new specializations and job descriptions emerge to meet some of the special requirements of web-based ISD, e.g. information architect, multimedia designer and multimedia project manager. An organization developing and using corporate intranets for information storage and retrieval furthermore faces issues and conflicts related to responsibilities for information ownership, production and maintenance (Bansler *et al*., 2000; Deshpande & Ginige, 2001).

All this leads to the inclusion of more tasks and specializations in the development life cycle and an increasingly complex organization of Web development projects. It also means that

user representatives are included as active participants in the development project, sometimes using advanced tools to independently produce complete or substantial parts of a web-based system or application. Project organization and management furthermore tends to be unstable as IT organizations struggle to define the job categories, training programmes, and development and management practices required to meet the needs of Web development (Greenbaum & Stuedahl, 2000).

We do not question these observations regarding skill needs, the role of the user or the difficulties involved in organizing and managing Web development projects. We do, however, argue that what is observed are evolutions or continuations of trends and changes that have taken place in ISD over the last 50 years, and not signs of a significantly different situation.

In his account of the history of ISD, Friedman illustrates how IS have penetrated deeper and deeper into the world of users (Friedman, 1989). The use of IS has expanded from the first scientific applications built by the individuals who required them, over the very early business applications to automate distinct and routine clerical processes, to present-day applications and systems that are deeply integrated into almost all aspects of the working – and often social – lives of humans. This development has been accompanied by a continuous expansion of the core issues and concerns of IS development away from the computing machines towards the world of the users, ongoing and interdependent changes in the internal organization of IS projects and departments, and changes in the relations between users and systems developers.

First, the main issues and concerns of ISD have gradually moved away from what Friedman calls the computer systems core. In the early days of computing, the primary concern was one of efficient utilization of the very costly, complex and limited hardware resource. Decreasing hardware costs, more powerful development tools, e.g. operating systems and compilers, and the increasing diversity and importance of IS, shifted concerns away from the computing machines towards user-related issues, such as business analysis, organizational implementation and impact of IS, and, later, user interface and ergonomic issues.

Second, these changes have been accompanied by a need for more and more skills in ISD and an increasing specialization. The original project manager/analyst/programmer/tester/maintainer primarily concerned with the computing machine has been replaced by a plethora of specializations that reflect the increasing complexity of ISD as well as the inclusion of new tasks and concerns in the development life cycle. ISD organizations have always had to recruit staff from 'outside', to answer the need for new skills in ISD. Often the people thus recruited did not have basic ISD skills.

Third, ISD organizations have always had to define and redefine project organization, ISD tasks, job categories, divisions of labour, development practices and management principles in ISD in light of the changes described earlier (cf. Kraft, 1976; Greenbaum & Stuedahl, 2000). This is not, and never has been, an easy task as illustrated by Pettigrew's study of an IS department in the 1960s (Pettigrew, 1973), or the software engineering literature in general (cf. Naur & Randell, 1969; Swanson & Beath, 1990; van Genuchten, 1991).

Hence, we argue that what has unfolded in Web-based ISD is a continuation of the historic trends already described. Web-based systems represent the latest expansion of IS into the user domain, and this creates – as has always been the case – new issues and concerns for

ISD, new tasks in the life cycle, a need for new specializations and skills in ISD, new challenges for organization and management of development projects, as well as a need to reconsider user roles and the boundary between systems use and development. The challenge is, as always, to identify the special requirements and problems of the situation, and to learn from the past in terms of how similar problems and challenges have been dealt with previously.

A widening user population

Web-based IS on the public part of the internet can be used by potentially everyone with access to the internet and a browser. Internal (corporate) systems are likewise accessible from all computers on the corporate intranet. This means that Web development projects must take into account the needs and capabilities of a very large, diversified and, for the most part, unknown user group when designing a system and particularly its user interface. Several authors note the size and diversity of the user base as one of the distinctive characteristics of Web-based sites and systems (Deshpande & Ginige, 2001; Murugesan *et al*., 2001).

A large and diversified user base does create challenges for an ISD project in terms of identifying, describing and managing requirements as well as designing appropriate user interfaces. However, this is not a new situation, neither is it particularly associated with Web-based systems. Several studies have addressed the issues involved in developing IS for a large and diversified user base, including the difficulties of managing vague and unstable requirements (cf. Pape & Thoresen, 1985; Curtis *et al*., 1988). Grudin (1991) distinguishes between in-house and contract development for an identified set of users as opposed to development of a commercial software product for an unidentified user group, i.e. standard off-the-shelf software such as office automation applications. In commercial product development, the user group is relatively unknown and the potential users are not required to buy and use the system. This means that the system must appeal to people, resulting in increased focus on the look and feel of the user interface (Grudin, 1991). The same argument can be made for Web development aimed at a divergent, external user group, which does not have any obligations to use a given Web application. Based on Grudin (1991), it can therefore be argued that some Web development projects have characteristics of commercial product development, even when they take place in an in-house or contract development context. But a diverse and unknown user group is not in itself a new and unique challenge. The fundamental question of how to define and manage requirements and test a system intended for a large and partly unknown user population is independent of whether the system in question is a 'traditional' information system, a product for a market, or a Web-based application.

Development practices

Researchers in ISD generally agree that Web-based systems are produced in an *ad hoc*-like manner, without much attention to development methodologies, systematic planning and management practices, quality assurance, and process and product measurements (Murugesan *et al*., 2001; Avison & Fitzgerald, 2002). A study of Web development projects in the USA and

Denmark observes, for example, high time pressure created by a desperate rush to market, use of prototyping and parallel development, the need for good people and a flexible approach to quality assurance (Baskerville & Pries-Heje, 2002). According to this and similar studies, this means that traditional ISD methods and management techniques are unfit for the development of Web-based applications. Therefore, there is a need for new methods and tools for Web development and Web engineering (cf. Murugesan *et al*., 2001; Baskerville & Pries-Heje, 2002).

Again, we do not question the results from these studies of Web-based ISD and we agree that many so-called traditional ISD methods may be unfit for the development of Web-based systems. We do not, however, agree that this implies a need for a special 'Web-based' approach to ISD.

First, the challenges and practices of Web development are not different from what has been consistently reported from studies of ISD for more than 30 years. According to Avison & Fitzgerald (2002), for example, there are clear parallels between current practices in Web development and the way IS were developed already in the 1970s. Recent capability maturity surveys of software development organizations, on the lowest maturity level, indicate practices similar to those reported from Web development projects (Herbsleb *et al*., 1997). Unrealistic and hard deadlines have also been repeatedly reported in the past (DeMarco & Lister, 1987; Greenbaum & Stuedahl, 2000). These observations therefore point to general issues of ISD project planning and management.

Second, the practices observed in current Web development projects may deviate from the methods recommended in textbooks, but that has also been consistently reported in several studies of ISD in the past. ISD methods, especially those that presume a sequential progression of a predefined set of tasks, are, as a rule, not followed in practice. Instead, developers are reported to combine elements of different methods and tools based on prior experience and intuition (Bansler & Bødker, 1993; Fitzgerald, 1996). These findings are confirmed by more recent research into the use of modern, more flexible, methods as well (Madsen & Kautz, 2002). As a consequence, some researchers suggest to abandon the idea of planned methodical ISD and talk about emergent and amethodical ISD instead (Truex *et al*., 1999; Truex *et al*., 2000). This, in our view, indicates a general, i.e. not Web related, concern with the mainstream conceptualization of ISD methods and their use in practice.

Third, researchers of Web-based development note that sequential approaches to ISD, like the waterfall model, are inappropriate for the highly dynamic environment, short development cycles and unstable requirements typical of Web-based ISD. Instead, these researchers recommend the use of alternative approaches based on evolutionary or incremental software process models (Baskerville & Pries-Heje, 2002). We totally agree. In fact, several researchers and practitioners question the appropriateness of sequential software process models within many development contexts. The alternatives are well described, have been known for years (cf. Floyd, 1984; Boehm, 1988; Floyd *et al*., 1989) and have proven their usability in a variety of settings (Korsaa *et al*., 2001; Nørbjerg, 2002). Thus, the challenge seems not to be the need for evolutionary and incremental approaches unique to Web development, but the dissemination of knowledge about how to use these approaches in a planned, manageable and consistent way (Nørbjerg, 2002).

Finally, and on a more general note, we must be careful not to equate *observed* practice in Web-based ISD with *good* practice. It appears that many organizations that develop Web-based applications and systems have been able to cope with a highly pressurized market by accepting tight deadlines, short development cycles and *ad hoc* approaches to planning, development and quality.

We have already shown how these Web practices resemble what has been observed in other types of ISD in the past, but we should also remember that many organizations did not survive and that such practices may be considered undesirable because of the predicaments they create, e.g. poor planning and control, missed deadlines, unstable systems, and high pressure on developers (Paulk *et al*., 1993; Pressman, 2000). The challenge remains, however, to define what constitutes 'good practice' and how to assess it across the many different types of ISD projects, Web or non-Web based, that exist today (Bach, 1994; Iversen *et al*., 1998; Nielsen & Nørbjerg, 2001).

DEVELOPMENT IN PRACTICE: PAST AND PRESENT

In Table 1, we have summarized the characteristics of, and claims made about, traditional ISD and Web development in the ISD literature. Some differences emerge with regard to the under-

Table 1. Summary of historic and contemporary characteristics and claims about information systems development (ISD)

	Historic claims	Web claims
Technology and information systems	• IS expands into all aspects of human living	• Cheap internet technology, which offers unlimited possibilities • Distinction between Web sites vs. Web-based systems • Important/distinguishing features of Web applications: 'look and feel', user interface, marketing capability, multimedia content and content production
Context and conditions	• ISD focus from computer core to business analysis, implementation and users • (Re)definition of project organization, skills, tasks, jobs, etc. is ongoing • Commercial software products are developed for diverse and unknown users	• New skills and jobs needed • More tasks and specializations in the development life cycle • The organization of Web development projects is (increasingly) complex • The user base is diverse and unknown
Development practices	• Problems with *ad hoc* development and time pressure frequently reported • ISD methods not followed as prescribed, but adapted to particular situation • Sequential ISD methods unfit for dynamic environments and projects; evolutionary and incremental methods advocated	• Characteristics of Web development: *ad hoc*, time pressure, prototyping, parallel development, need for good people and negotiable quality • ISD methods unfit for Web development • New, unique Web approaches required

lying technology and the resulting IS; however, with regard to the context and the conditions as well as the development processes and practices, striking similarities can be observed.

To further investigate these differences and similarities, we now focus on the inherent problems of ISD and the routines and procedures applied to deal with such problems in practice. As explained earlier for this purpose, we use four empirical studies. Two of the studies were undertaken by the authors (Nørbjerg, 1994; Kautz & Madsen, 2003), while the remaining cases (Curtis *et al*., 1988; Baskerville & Pries-Heje, 2004) were chosen from the literature because of their thoroughness and excellent empirical foundation. The section of the paper is structured in two parts. First, Curtis *et al*.'s (1988) and Nørbjerg's (1994) studies are used to present the main characteristics, problems and coping mechanisms of practice in the past. Second, Baskerville & Pries-Heje's (2004) and Kautz & Madsen's (2003) investigations of characteristics, difficulties and development practices in contemporary ISD are outlined.

Practice in the past

In their seminal paper from 1988, Curtis *et al*. (1988) interviewed personnel in 17 large projects in a number of business areas such as computer manufacturing, telecommunications, consumer electronics and aerospace. The projects concerned the development of a variety of very different application types, such as real-time, distributed or embedded operating, transaction processing, or defence systems. In general, the projects concerned commercial contract development of specialized systems for individual customers or development of software products for the market. The purpose of the study was to explore the behavioural processes and problems that hinder software productivity and quality.

Curtis *et al*. (1988) introduce a layered model to identify and understand such processes and problems. The model consists of five levels: (1) business environment (external influences from co-contractors and customers); (2) company (internal influences from corporate politics, culture and procedures); (3) project (interteam group dynamics); (4) team (intrateam group dynamics); and (5) individual (cognition and motivation). Curtis *et al*. showed that problems related to the various contextual levels and conditions operate through different mechanisms; may require different solutions and development practices; and typically will emerge from processes at one level of the layered model, but will affect several levels.

The three salient problems of ISD are identified as (a) the thin spread of application domain knowledge; (b) vague and conflicting requirements; and (c) communication bottlenecks and breakdowns. The problem with the thin spread of domain knowledge starts at the individual level, where few people understand the application domain and lack the ability to map between the domain and the computer structures (e.g. data, architectural and control structures) that can support it. To overcome this problem, truly exceptional people, i.e. project gurus, with great domain knowledge, visualization, integration and communication skills are needed. These people are not necessarily good programmers. Instead, they spend most of their time in face-to-face interaction with people at one or more levels to negotiate and communicate a shared understanding of the system, thereby gaining even better insight into the particular application

domain. Exceptional people play a crucial role for the effective communication, coordination and integration of people and code, but may in turn become communication bottlenecks.

The problem of vague and conflicting requirements often starts at the business environment and company level and results in the project members holding misconceptions about the application domain and incomplete information about relevant market factors (e.g. technological advances, competitive products and regulatory constraints) and internal issues (e.g. corporate politics, marketing plans and financial conditions). An important part of any project is learning about the application domain, requirements and constraining market and internal factors even though it is often not explicitly planned or budgeted for. This process of learning is itself a major source of requirements fluctuation. Coping with learning and incomplete information necessitates negotiation, constant communication, and exploration and validation of the project members' understandings of potential problems and solutions via simulation and prototyping.

The problem of communication bottlenecks and breakdowns occurs at all levels, often because of the lack of a single representative. Project members are, for example, typically not communicating with *the* customer, but with many customer representatives. This hinders the definition of stable requirements and increases communication and negotiation costs. An important insight from Curtis *et al*.'s (1988) study is that large projects (all projects we add) require extensive communication, which is not and cannot be reduced by documentation. To anticipate such problems, the early phases of a project see much time spent on defining terms, coordinating representational conventions and creating channels for the flow of information. Moreover, boundary spanners and the establishment of informal networks are important to ensure communication and coordination at, and between, different levels.

In line with Curtis *et al*. (1988), similar conclusions about application domain, requirements and verbal communication have been reported by one of the authors (Nørbjerg, 1994). Using semistructured interviews carried out between January and August 1992, Nørbjerg (1994) provides detailed case-study accounts of two ISD projects, hereafter referred to as Projects A and B. Project A was performed by six people (i.e. the department head, two systems analysts and three programmers) in a company that primarily develops and maintains IS for Danish municipalities. The project concerned the development of a centralized standard application for administration of unemployed citizens, their skills and course activities. Project B was undertaken by six people (i.e. the department head, two programmers, two user representatives and a database administrator) in a large Danish telecommunications company and concerned the development of an information system for internal administration of the company's technical installations (cables, traffic, etc.). The purpose of the research was to understand how systems developers' learning and knowledge sharing in ISD projects are facilitated or frustrated by organizational structures (i.e. division into departments, job categories and tasks, and development phases) at the company and project team levels.

Nørbjerg (1994) conceptualizes ISD as a (1) combined design and construction process, where the systems developers have to learn about and build an individual, as well as a shared, understanding of the application domain and the IS under development, and (2) as a process that requires knowledgeable developers and two distinct sets of skills, i.e. for design, detailed knowledge of the application area and analysis and design abilities are needed, while con-

struction requires more technically oriented skills. This conceptualization is reflected in the two case studies, where both development organizations were divided into departments according to the application area of the systems; each department was responsible for development and maintenance of the systems within the particular application area; and each department consisted of a department head as well as a group of systems analysts and a group of programmers. The organization into departments, jobs, tasks and phases helps the developers to specialize and build skills, but means that learning from each other and knowledge-sharing across divisions is required during the development process. Both development organizations aimed to use a phased development process with a traditional task division and handover of work (in the form of documents) from systems analysts to programmers and with management follow-up at phase completion. However, in both projects A and B, this phased approach was not followed because of time pressure and efficient use of programmer resources, respectively.

Project A was performed under time pressure as it was known that competing systems were under development and would soon reach the market. The project manager therefore decided that the systems analysts and programmers should work together during analysis and design to avoid the misunderstandings and iterative corrections to documents, which was a normal part of the handover in the phased approach. Together with the collaboration between analysts and programmers in the specification phase, a four-layered architecture facilitated parallel work. Thus, after a shared overall understanding of the system had been reached, the programmers started program construction, while the analysts continued the detailed specification of more program aspects. To further cope with the need to get something ready for the market, system scope and content were negotiated with sales personnel and customer representatives. It was decided to structure the development into a release-oriented process with three versions completed at different points in time within a 2-year time frame. In Project B, the traditional division of work between systems analysts (in this case, the department head, the user representatives and the database administrator) and programmers was maintained to a greater extent. However, the continuous involvement of user representatives and the existence of an in-house developed and well-documented standard for the user interface meant that the systems analysts were able to outline and pass on parts of the system specifications to the programmers already *during* the specification phase. In this way, parallel work and the efficient use of programmer resources were made possible.

Ten to 15 years on from Curtis *et al*.'s (1988) and Nørbjerg's (1994) empirical studies, there is a remarkable similarity in the characteristics, problems and practices that are presented in the two case studies about ISD today, in the internet era.

Practice at present

In their paper 'Short Cycle Time Systems Development', Baskerville & Pries-Heje (2004) report from an interview study performed with personnel in three Danish and nine US companies. The authors 'selected a diverse set of companies . . . achieved a good variety of application areas, including third generation (interactive data exchange such as business-to-business) Web applications, components for such applications or systems software for internet-based applications

. . . [and] also achieved a good variety in development operation size, company size and capitalization (both venture start-ups and new divisions in old companies)' (Baskerville & Pries-Heje, 2004, p. 239). The purpose of the study was to identify the characteristics of short cycle time development, which the authors describe as a new and clearly distinguishable form of ISD.

Baskerville & Pries-Heje (2004) show that the context and conditions of short cycle time development can be characterized by (a) time pressure and a desperate rush to market; (b) vague requirements; and (c) and a new type of software market. The issue of time pressure and prime focus on minimizing the time from concept to customer is grounded on tight deadlines, technological change, the developers' and customers' fear of business and technological obsoletion, as well as customer expectations about expedient results. Baskerville & Pries-Heje (2004) further explain that an inability to predefine requirements is the central, defining constraint of internet time development. Time pressure and vague requirements, they argue, have contributed to the creation of a software market for non-critical applications (i.e. bugs, lacking features, etc. are acceptable), where requirements and quality are negotiated with the customer on an ongoing basis with a constant eye to time and technology.

This context, or new software market, has led to the deployment of a set of development practices. Baskerville & Pries-Heje (2004) suggest a package of five practices that include a focus on (1) completion speed; (2) release-oriented parallel prototyping; (3) adherence to a fixed architecture; (4) negotiable quality; and (5) an ideal workforce. A focus on speedy completion of releases (often every 2–3 weeks) via parallel prototyping is necessary to cope with time pressure. Adherence to a fixed layered architecture helps impose some structure on the process and facilitates coordination, division of labour and parallel development. To cope with customer expectations, unknown users and vague requirements, the developers rely on close verbal interaction (not written documentation) and, if possible, colocation with customers. Prototyping and prioritization of requirements and features are used to validate, refine and negotiate meaning and quality throughout ISD and use. An ideal workforce with good coding and coordination capabilities is essential to handle the pressure and fuzziness of short cycle time development.

Baskerville & Pries-Heje (2004, p, 260) conclude that while 'time pressure is certainly not new . . . [and] there is a long history of prototyping (cf. Alavi, 1984) and architecture (cf. IBM, 1989) as a basis for fast application development . . . it would appear that the collective set of characteristics . . . do indicate that something very new and different has happened in practice'. Thus, the authors argue that it is the particular collection of (the five) characteristics and development practices that is unique to internet development processes across companies, projects and cultures. In contrast, Kautz & Madsen (2003) conclude that there are very different types of internet development when one looks beyond development practices and considers the applications under development and the context in which the development takes place.

Kautz & Madsen's (2003) research was undertaken in four companies, i.e. in a small Web agency, two medium-sized Web agencies and a large consultancy company. The companies were selected to cover the development of a variety of internet applications, ranging from smaller information publishing- and advertising-oriented Web sites to transaction-oriented

Web-based systems. The purpose of the study was to investigate how internet applications were developed in practice.

The main findings from the study are:

1 Internet development is characterized by diversity – in project scale, scope, technical complexity and development focus (front-end vs. back-end) as well as in project duration, team size, and educational background and level of experience of the individuals involved.

2 *How* internet development is performed can be explained by not just the development practices applied, but by the complex interplay of application type characteristics, organizational structures and conditions, and development practices – their interaction and relationship to each other, in general (i.e. at the business environment and company level) and in the particular project.

Thus, in the small Web agency, the average project concerned the development of an information publishing or advertising Web site, lasted 1 month and involved a project manager, a Web designer, a text writer and an HTML programmer. The deadline was normally determined in advance by the customer, and therefore no formal project plan was outlined. In the concept phase, the project team came up with the idea for the site using group meetings and brainstorming techniques. Based on these meetings, a PowerPoint mock-up was iteratively developed and discussed with the customer until all Web pages were complete and fully designed. Subsequently, the PowerPoint mock-up was handed over to the HTML programmer, who carried out the actual coding. In contrast, the consultancy company's projects lasted 12–20 months, involved 10–20 people, concerned the development of complex transaction-oriented Web-based systems with enterprise resource planning integration and followed a traditional waterfall model supplemented with tools and techniques from RUP. Between these two examples of how internet development is conducted in very different ways in practice, the medium-sized agencies specialized in the development of both information publishing and transaction-oriented Web-based sites and systems. Project and team size were reported to vary greatly. Some projects lasted no more than 14 days involving only one systems developer, while other projects last 4–6 months involving six to eight people full time. Both companies had developed their own in-house method to accommodate both front-end and back-end oriented development issues into their work practices. The two methods were divided into distinct phases. However, in practice the development processes were primarily driven by iterative prototyping supplemented with written documents such as project plans, requirement specifications, use cases and database models, as well as the design of an overall systems architecture.

DISCUSSION: PERSISTENT PROBLEMS AND PRACTICES

Table 2 provides an overview of the presented studies of ISD in practice. The table shows that despite major differences in the times, types of applications and market conditions under investigation, as already established through the literature review, there are a number of remarkable

Table 2. Overview of past and contemporary practice

	Past studies		Contemporary studies	
	Curtis *et al.* (1988)	Nørbjerg (1994)	Baskerville & Pries-Heje (2004)	Kautz & Madsen (2003)
Characteristics of the study	• Interview study of personnel in 17 projects	• Longitudinal interview study of two projects	• Interview study of personnel in 12 companies	• Interview study of personnel in four companies
Technology and Information Systems	• Large projects, diversity in application area	• Commercial software product; In-house project	• Varying project size, diversity in application area	• Varying project size, Web sites and systems
Context and conditions	• Thin spread of application domain knowledge • Vague and negotiable requirements • Communication bottlenecks and breakdowns	• Division into departments, jobs, tasks and phases • Time pressure • Efficient use of programmer resources	• Great diversity in types of organizations, projects and teams • Time pressure • Vague and negotiable requirements • A new type of software market	• Great diversity in types of organizations, projects and teams • Different types of internet development
Development practices	• Exceptional people • Documentation not enough, constant verbal communication required • Learning via exploration, simulation and prototyping	• Knowledgeable developers • Release-oriented development • Negotiable quality • Fixed architecture • Documentation not enough, verbal communication required	• An ideal work force • Verbal interaction and colocation with customers • Parallel prototyping • Fixed architecture and parallel work and parallel work	• Iterative prototyping • Identification of overall architecture • Verbal communication and documentation

similarities across the four case studies. Thus, the five characteristics of time pressure, parallel work and prototyping, negotiable quality, fixed architecture, and reliance on good people are in no way exclusive to contemporary internet development, regardless of whether the practices are viewed separately or as a collective package. Instead, these practices seem to have been and to still be quite consistently applied to many different types of ISD. Rather than confirming a new paradigm (Baskerville & Pries-Heje, 2002), our comparison of past and contemporary practice points to a number of issues inherent to ISD in general. From this conclusion, we extend our argument one step further and suggest that it is necessary, and possible, to go beyond actual problems and practices (e.g. time pressure and reliance on good people) as they are currently coined in the literature to form a more deeper theoretical understanding of ISD. However, the ISD research community lacks a research agenda that distinguishes between, and aims at, identifying and conceptualizing the persistent problems and the situated practices applied to cope with these problems as they emerge in different contexts. We therefore propose and discuss such a research agenda here.

The discussion is structured around three inherent and interrelated problems areas of ISD: the *diversity* of ISD projects, the dependence on developers' *knowledge* and the relationship between *structure* and ISD practice. These problem areas have been identified from our literature review and the analysis of the presented empirical studies. They are discussed at five contextual levels, which are inspired by Curtis *et al*. (1988). This enables the development of a comprehensive research agenda, which facilitates a focus on the occurrence and interaction of problems and practices at and between different levels.

Coping with diversity

Information systems development projects have always been and are now becoming increasingly diversified in terms of size, application domain, underlying technology, and the number, knowledge, needs and requirements of people involved in (developers, customers, co-contractors, future end users, etc.), and affected by (end users), a given project. Thus, adaptation of organizations, methods and development practices to a broad and very diversified set of conditions and environments is a continual challenge.

This challenge is handled in different ways at different contextual levels by either *absorbing* or *reducing* diversity. At the business environment level, diversity is absorbed by conceptualizing and creating new and clearly distinguishable forms of development and software markets. Web and open source development are more recent examples. This structuration (Giddens, 1984) of the marketplace helps reduce diversity, because it allows for organizations and individuals to specialize in particular types of technology, application domains, IS and user groups (e.g. advertising- and information publishing-oriented Web agencies). Also, at the level of the particular company, there is a need for ways of coping with diversity to keep up with the market and because project scale, scope, duration, technical complexity, team size as well as the educational background and level of experience of the people involved can vary greatly (Kautz & Madsen, 2003). Again, this diversity is absorbed and reduced via organization into (new) departments, jobs, tasks and specializations. At the project and team levels, it is often neces-

sary to cope with many, vague and conflicting requirements as well as varying developer, customer and user skills and expectations. To this end, the coping mechanisms are communication, coordination, division of labour and exceptional people who have become so through participation in projects that allow them to build up a repertoire of knowledge on how to deal with all types of situations (Schön, 1983; Curtis *et al*., 1988; Baskerville & Pries-Heje, 2004). For the individual, diversity can be a stressful experience as it might mean that he or she is working with many different people on several projects during a short time period – sometimes over the course of just 1 day (Greenbaum & Stuedahl, 2000). The individual developer therefore continuously has to stay up-to-date on the project(s), the market, and new and changing technology to be a qualified and attractive project team member in the short and long term.

It seems reasonable to conclude that diversity is a fundamental challenge in ISD generally – one challenge that has to be, and which is, dealt with in many ways and at many levels. We have outlined a tentative understanding of this problem and its current solutions based on past and contemporary ISD literature. However, existing literature primarily focuses on the project and the individual developers as its units of analysis. It also has some focus on team organization and dynamics, but the influence of the external business environment and internal company factors are largely ignored. There is therefore a need for more research about the true nature and severity of the diversity issue via thorough exploration of research questions such as: How is diversity absorbed and reduced at, and between, different contextual levels? And what kind of theoretical and empirical research contributions might describe and prescribe this absorption and reduction of diversity in practice?

Coping with knowledge

The issue of knowledge is closely related to diversity. Diversity in available technology, types of application domains and applications, as well as in the numerous involved and affected peoples' skills, needs and (work) practices, leads to different demands of knowledge. As a consequence, systems developers have to be knowledgeable about, among others, the three areas of technology, applications and people. The diversity in technology, application and people results in vague and conflicting requirements, which have to be discovered and negotiated throughout ISD and use. In all types of ISD and ISD projects, it is therefore necessary to determine what kind of knowledge is needed, how much is enough and how it can be acquired and communicated.

The challenge of *acquiring* and *negotiating* knowledge is substantial and takes place at all contextual levels. At the business environment level, stratification of the marketplace leads to specializations at all levels, which consequently helps reduce the need for knowledge. However, at the company level, it is crucial to follow the developments in the market (e.g. for a particular area such as web development) with regard to technology, applications and their new 'standard features', legislation, and development methods and buzzwords, etc. At the project and team levels, it is necessary to negotiate the terms, representational conventions and communication channels that will support the flow of information, and create and coordinate the

interfaces between people and code (Curtis *et al*., 1988; Button & Sharrock, 1995). To cope with incomplete information about system requirements (i.e. about the product) and therefore about how to proceed (i.e. about the process), constant verbal communication, negotiation and explorative prototyping are deployed (Curtis *et al*., 1988; Baskerville & Pries-Heje, 2004). Also, the individual developer has to keep up with the external market (or a stratum in the marketplace), the internal company situation and the particular project(s). To this end, personal strategies for acquiring knowledge through formal training, the 'right' kind of project experience and the building of informal networks become important.

Despite numerous empirical studies that report otherwise, the mainstream ISD literature is still dominated by rational and methodical assumptions about ISD: how it is and should be performed in practice (Truex *et al*., 2000). This methodical stream of literature promotes formal ISD methods as *the* coping mechanism that provides all the knowledge, which IS developers need (Truex *et al*., 2000; Avison & Fitzgerald, 2003). Consequently, much less attention has been given to human behaviour and processes of communication, negotiation and learning (Curtis *et al*., 1988). As yet, we therefore know little about issues such as what kind of knowledge is acquired and negotiated at and between different contextual levels, how this takes place in practice, and how theoretical and empirical research might assist ISD practitioners in processes of knowledge acquisition and negotiation.

Coping with structure

Structure is often introduced to cope with diversity and incomplete knowledge; a written requirements specification is such an example. However, structure is a controversial issue. Some view it as an inhibiting factor; see, e.g. Wastell's (1996) account of how formalized methods can become a fetish. Others see structure as a necessary means to achieve a disciplined development approach and quality IS. Coping with different perceptions about and the institution of only partly controllable organizational, technological (e.g. bandwidth), application (e.g. a fixed architecture) and methodical structures as well as time and market pressures is not a trivial matter.

Structure is *perceived* and *established* differently at different contextual levels. At the business environment level, there is increased demand for more formalization, methodical discipline and software process improvement according to software capability maturity models, quality standards, etc. (Fitzgerald *et al*., 2002) But there is also a growing opposition that proposes competing theoretical ideas and conceptualizations about amethodical development (Truex *et al*., 2000), agility (Cockburn, 2002; Highsmith, 2002), complex adaptive systems development (Highsmith, 2000), etc. The perceived need for formalization and methodical structure varies greatly. Web development is, for example, seen by some as an innovative, non-critical application area performed by smaller organizations and teams and therefore with less need for structure (Baskerville & Pries-Heje, 2004), while other 'software markets' for defence and medical systems are perceived as high-risk applications domains with much formalization required. At the company level, structure can be viewed as an emergent property that is influenced and shaped by the complex interplay between the market in which the company oper-

ates, the type of technology, the application domain and the applications under development. Structure can also be influenced by the type of organization, its cultural and political situation and perceptions about the need for methodical discipline (Kautz & Madsen, 2003). At the project and team levels, a variety of means such as technology, systems architecture, the formal project and team organization, the individual developers' skills and experiences, social relationships, the choice and use of particular formalized (and more or less structured) ISD methods, and time and budget constraints impose structure on, and create the landscape, within which ISD can take place. Lastly, the individual IS developer's assumptions, skills and practical experience also serve as structural influences, because they shape the perception of the problem situation and the solutions that he or she can think of (Schön, 1983).

In the field of ISD, structure has traditionally been associated with formal project organization and methodical discipline. However, it is increasingly recognized that there are many elements, including both formally introduced as well as only partly controllable social structures, that influence and shape ISD in practice. Yet more work is demanded to move beyond a focus on *that* it is so to dynamic questions of *how*: How are formal and social structures perceived and established at and between different contextual levels? How do such structures influence and shape ISD in practice? And how might theoretical and empirical research help answer questions about the perception and establishment of structure and achievement of the 'right' level of structure?

CONCLUSION

Information systems development is a rapidly changing area prone to fads, fashion and frequent claims about *the revolutionizing nature* of the latest developments in IT, business opportunities and development methods. However, based on a review of older ISD literature compared with more recent ISD literature that focuses on web-development, this paper shows that contemporary ISD can be seen as *an accentuated evolution* of long-since identified problems and practices.

We conceptualize the persistent problems of ISD in terms of (1) diversity; (2) knowledge; and (3) structure, and propose that these three inherent and interrelated challenges are handled in different ways at different contextual levels (e.g. the business environment, company, project, team and individual levels). The development practices currently applied to cope with the diversity in, among others, technology, applications and people; the need for knowledge; and (re)creation of formal, organizational and informal, social structures are identified as (a) organization and specialization; (b) constant verbal communication and negotiation; and (c) pragmatic application of certain methods and concepts such as prototyping and a layered architecture.

More theoretical and empirical conceptualization of the occurrence and interaction of inherent problems and applied practices are needed to understand and assess 'observed practice' and 'good practice' across the many types of ISD projects conducted today. We propose a focus on dynamic research questions such as: How is and how can diversity be absorbed and

Table 3. Research agenda

Core problems/ Contextual levels	Diversity	Knowledge	Structure	Necessary studies
Business environment Company	What are the different types of ISD and software markets? How is diversity absorbed and reduced in and by different types of ISD and software markets?	What are the different kinds of knowledge and skills needed in different types of ISD and in different software markets?	Are structures perceived and established differently in different types of ISD and software markets?	There is a general paucity of ISD research; theory and studies of (longitudinal processes of) organization, specialization and institutionalization in ISD are needed
Project Team	What are the collective mechanisms for absorption and reduction of diversity?	What kind of knowledge and skills are needed? How are they acquired and negotiated?	Which formal and informal structures can be identified? How are they perceived and established? What is their influence on work and social interaction?	Only little ISD research goes beyond ISD methods; there is a need for theory and studies about social behaviour and processes of communication, negotiation, and learning and their relation to the broader (historical, political and social) context
Individual	What are the individual mechanisms for absorption and reduction of diversity?	What kind of individual learning and sense-making takes place? How do they take place and how are they informed by a particular context?	What is the influence of structure on the involved individual's knowledge, preconceptions and actions?	There is a paucity of ISD research that relates individual knowledge, learning and sense-making to the broader (historical, political and social) context; this kind of theory and studies is needed

reduced at and between different contextual levels? How is and how can knowledge be acquired and negotiated at and between levels? How are and how can formal, organizational and informal, social structures be perceived and established at and between the different contextual levels? And, lastly, how can theoretical and empirical research best assist practitioners in understanding the underlying problems and their potential solutions? Table 3 summarizes this research agenda and underlines, in line with others (see, e.g.Nygaard, 1986; Hirschheim *et al*., 1991; Hirschheim *et al*., 1996), the necessity to strengthen ISD research from a social science perspective.

REFERENCES

Alavi, M. (1984) An assessment of the prototyping approach to information systems development. *Communications of the ACM*, **27**, 556–563.

Alter, S., Ein-Dor, P., Markus, L.M., Scott, J. & Vessey, I. (2001) Does the trend toward e-business call for changes in the fundamental concepts of information systems? A debate. *Communications of the Association for Information Systems*, **5**, 1–59.

Angehrn, A. (1997) Designing mature internet business strategies: the ICDT model. *European Management Journal*, **15**, 361–369.

Avison, D. & Fitzgerald, G. (2002) Reflections on information systems development 1988–2002. In *Proceedings of the 11th Information Systems Development Conference (ISD2002)*, Riga, Latvia.

Avison, D.E. & Fitzgerald, G. (2003) *Information Systems Development: Methodologies, Techniques and Tools.* McGraw-Hill, New York, NY, USA.

Bach, J. (1994) The Immaturity of the CMM. *American Programmer*, **7**, 13–18.

Bansler, J.P. & Bødker, K. (1993) A reappraisal of structured analysis: design in an organizational context. *ACM Transactions on Information Systems*, **11**, 165–193.

Bansler, J.P., Damsgaard, J., Scheepers, R., Havn, E. & Thommesen, J. (2000) Corporate intranet implementation: managing emergent technologies and organizational practices. *Journal of the Association for Information Systems*, **1**, 1–39.

Barry, C. & Lang, M. (2001) A survey of multimedia and web development techniques and methodology usage. *IEEE Multimedia*, **8**, 52–60.

Baskerville, R. & Pries-Heje, J. (2002) Information systems development @internet speed: a new paradigm in the making! In: *The Xth European Conference on Information Systems (ECIS 2002*), Vol. 1, Wrycza, S. (ed.), pp. 282–291. University of Gdansk, Gdansk, Poland.

Baskerville, R. & Pries-Heje, J. (2004) Short cycle time systems development. *Information Systems Journal*, **14**, 237–254.

Boehm, B.W. (1988) A spiral model of software development and enhancement. *IEEE Computer*, **21**, 61–72.

Button, G. & Sharrock, W. (1995) Practices in the work of ordering software development. In: *The Discourse of Negotiation – Studies of Language in the Workplace*, Firths, A. (ed.), pp. 159–180. Elsevier Science, New York, NY, USA.

Carstensen, P. & Vogelsang, L. (2001) Design of web-based information systems – new challenges for systems development? In *Proceedings of the 9th European Conference on Information Systems,* Bled, Slovenia, pp. 536–547.

Cockburn, A. (2002) *Agile Software Development*. Addison-Wesley, Boston, MA, USA.

Curtis, B., Krasner, H. & Iscoe, N. (1988) A field study of the software design process for large systems. *Communications of the ACM*, **31**, 1268–1287.

DeMarco, T. & Lister, T. (1987) *Peopleware. Productive Projects and Teams.* Dorset, New York, NY, USA.

Deshpande, Y. & Ginige, A. (2001) Corporate web development: from process infancy to maturity – a case study. In: *Webengineering 2000*, Murugesan, S., Deshpande, Y. & Hansen, S. (eds), pp. 36–47. Springer-Verlag, Berlin, Germany.

Deshpande, Y. & Hansen, S. (2001) Web engineering: creating a discipline among disciplines. *IEEE Multimedia*, **8**, 82–87.

Fitzgerald, B. (1996) Formalized systems development methodologies: a critical perspective. *Information Systems Journal*, **6**, 3–23.

Fitzgerald, B., Russo, N.L. & Stolterman, E. (2002) *Information Systems Development. Methods in Action.* McGraw-Hill, New York, NY, USA.

Floyd, C. (1984) A systematic look at prototyping. In: *Approaches to Prototyping*, Budde, R., Kuhlenkamp, K., Matthiassen, H. & Zullinghoven, H. (eds), pp. 1–18. Springer, Berlin, Germany.

Floyd, C.F., Reisin, F.-M. & Schmidt, G. (1989) STEPS to software development with users. In *Proceedings of ESEC 1989,* University of Warwick, Coventry, UK.

Friedman, A.L. (1989) *Computer Systems Development. History, Organization and Implementation.* John Wiley and Sons, Chichester, UK.

Giddens, A. (1984) *The Constitution of Society.* Polity Press, Cambridge, UK.

Greenbaum, J. & Stuedahl, D. (2000) Deadlines and work practices in new media development. In: *Proceedings of IRIS 2 3*, Svensson, L., Snis, U., Sørensen, C. *et al.* (eds), pp. 537–546. University of Trollhättan Uddevalla, Uddevalla, Sweden.

Grudin, J. (1991) Interactive systems: bridging the gaps between developers and users. *IEEE Computer*, **24**, 59–69.

Hansen, S., Deshpande, Y. & Murugesan, S. (2001) A skills hierarchy for web-based systems development. In: *Webengineering 2000*, Murugesan, S., Deshpande, Y. & Hansen, S. (eds), pp. 223–235. Springer-Verlag, Berlin, Germany.

Herbsleb, J., Zubrow, D., Goldenson, D., Hayes, W. & Paulk, M.C. (1997) Software quality and the capability maturity model. *CACM*, **40**, 31–40.

Highsmith, J.A. (2000) *Adaptive Software Development: A Collaborative Approach to Managing Complex Systems.* Dorset House, New York, NY, USA.

Highsmith, J.A. (2002) *Agile Software Development Ecosystems.* Addison-Wesley, Boston, MA, USA.

Hirschheim, R., Klein, H.K. & Newman, M. (1991) Information systems development as social action: theoretical perspective and practice. *OMEGA*, **19**, 587–608.

Hirschheim, R., Klein, H.K. & Lyytinen. K. (1996) Exploring the intellectual structures of information systems development: a social action theoretic analysis. *Accounting Management and Information Technology*, **6**, 1–64.

Holck, J. (2002) 4 perspectives on web information systems. In: *Proceedings of the 25th Information Systems Research Seminar in Scandinavia*, Bødker, K., Kühn Pedersen, M., Nørbjerg, J., Simonsen, J. & Vendelø, M.T. (eds), CD-ROM. Copenhagen, Denmark.

Howcroft, D. & Carroll, J.M. (2000) A proposed methodology for web development. In *8th European Conference on Information Systems (ECIS 2000),* Vienna, Italy.

IBM (1989) Systems application architecture: ADI/cycle concepts. IBM Corporation Department J57, San Jose, CA, USA.

Isakowitz, T., Stohr, E.A. & Balasubramian, P. (1995) RMM: a methodology for structured hypermedia design. *Communications of the ACM*, **38**, 34–44.

Iversen, J.I., Nielsen, P.A. & Nørbjerg, J. (1998) Problem diagnosis in software process improvement. In: *Information Systems: Current Issues and Future Changes*, Larsen, T.J., Levine, L. & DeGross, J.I. (eds), pp. 111–130. IFIP, Laxenburg, Austria.

Jacobson, I., Booch, G. & Rumbaugh, J. (1999) *The Unified Software Development Process.* Addison-Wesley, Boston, MA, USA.

Kautz, K. (2004) The enactment of methodology: the case of developing a multimedia information system. In *25th International Conference on Information Systems (ICIS04),* AIS, Washington, DC, USA, pp. 671–683.

Kautz, K. & Madsen, S. (2003) Web development – the differences, similarities and in-betweens. In: *Constructing the Infrastructure for the Knowledge Economy, Methods and Tools, Theory and Practice – Proceedings of the 12th International Conference on Information Systems and Development*, Linger, H., Fisher, J., Wojtkowski, W.G., Wojtkowski, W., Zupancic, J., Vigo, K. & Arnold, J. (eds), pp. 495–505. Kluwer Academic, Melbourne, Australia.

Kautz, K. & Nørbjerg, J. (2003) Persistent problems in information systems development. The case of the world wide web. In: *11th European Conference on Information Systems (ECIS'03)*, Ciborra, C., Mercurio, R., De Marco, M., Martinez, M. & Carignani, A. (eds), CD-ROM. Universita di Napoli Federico II, Naples, Italy.

Klein, H. & Myers, M.D. (1999) A set of principles for conducting and evaluating interpretive field studies in information systems. *MIS Quarterly*, **23**, 67–93.

Korsaa, M.R., Vinter, O. & Olesen, O. (2001) *Iterative Software Development. A Practical View*. Datateknisk Forum, Hørsholm, Denmark.

Kraft, P. (1976) *Programmers and Managers. The Routinization of Computer Programming in the United States.* Springer Verlag, New York, NY, USA.

Lyytinen, K., Rose, G. & Welke, R. (1998) The Brave New World of development in the internetwork computing architecture (InterNCA): or how distributed computing platforms will change systems development. *Information Systems Journal*, **8**, 241–251.

Madsen, S. & Kautz, K. (2002) Applying system development methods in practice – the RUP example. In *Proceedings of 11th International Conference on Information Systems Developments, Methods and Tools*

– *Theory & Practice,* Riga, Latvia, 12–14 September 2002.

Murugesan, S., Deshpande, Y., Hansen, S. & Ginige, A. (2001) Web engineering: a new discipline for development of web-based systems. In: *Webengineering 2000*, Murugesan, S., Deshpande, Y. & Hansen, S. (eds), pp. 3–13. Springer-Verlag, Berlin, Germany.

Naur, P. & Randell, B. (1969) *Software Engineering*. NATO Scientific Affairs Division, Brussels, Belgium.

Nielsen, P.A. & Nørbjerg, J. (2001) Assessing software processes: low maturity or sensible practice. *Scandinavian Journal of Information Systems*, **13**, 23–36.

Nørbjerg, J. (1994) Skills and forms of co-operation in systems development. PhD thesis, Report 94/28, Computer Science Department, University of Copenhagen, Denmark.

Nørbjerg, J. (2002) Managing incremental development: combining flexibility and control. In: *Proceedings of the Xth European Conference on Information Systems (ECIS 2002)*, Vol. 1, Wrycza, S. (ed.), pp. 329–339. University of Gdansk, Gdansk, Poland.

Nygaard, K. (1986) Program development as a social activity. In: *Proceedings of the IFIP 10th World Computer Congress*, Kugler, H.-J. (ed.), pp. 189–198. Elsevier Science Publishers, North Holland, The Netherlands.

Pape, T.C. & Thoresen, K. (1985) Development of common systems by prototyping. In *Proceedings of the Working Conference on Development and Use of Computer Based Systems and Tools,* Aarhus University, Aarhus, Denmark.

Paulk, M.C., Curtis, B. & Chrissis, M.B. (1993) *Capability Maturity Model for Software, v. 1.1*. Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, USA.

Pettigrew, A.M. (1973) Occupational specialization as an emergent process. *Sociological Review*, **21**, 233–278.

Pressman, R.S. (1998) Can internet-based applications be engineered? *IEEE Software*, **September/October**: 104–110.

Pressman, R.S. (2000) *Software Engineering. A Practitioner's Approach. European Adaptation*. Mc-Graw-Hill, Maidenhead, UK.

Ramesh, B., Pries-Heje, J. & Baskerville, R. (2002) Internet software engineering: a different class of processes. *Annals of Software Engineering*, **14**, 169–195.

Schön, D.A. (1983) *The Reflective Practitioner. How Professionals Think in Action.* Basic Books, New York, NY, USA.

Swanson, E.B. & Beath, C.M. (1990) Departmentalization in software development and maintenance. *CACM*, **33**, 658–667.

Truex, D.P., Baskerville, R. & Klein, H. (1999) Growing systems in emergent organizations. *Communications of the ACM*, **42**, 117–123.

Truex, D., Baskerville, R. & Tavis, J. (2000) Amethodical systems development: the deferred meaning of systems development methods. *Accounting, Management and Information Technologies*, **10**, 53–79.

van Genuchten, M. (1991) Why is software late? An empirical study of reasons for delay in software development. IEEE *Transactions on Software Engineering*, **17**, 582–590.

Vidgen, R. (2002) What's so different about developing web-based information systems? In: *Proceedings of the Xth European Conference on Information Systems (ECIS 2002*), Vol. 1, Wrycza, S. (ed.), pp. 262–271. University of Gdansk, Gdansk, Poland.

Walsham, G. (1993) *Interpreting Information Systems in Organizations.* Wiley, Chichester, UK.

Wastell, D.G. (1996) The fetish of technique: methodology as a social defence. *Information Systems Journal*, **6**, 25–40.

Yourdon, E. (1989) *Modern Structured Analysis.* Prentice Hall, Englewood Cliffs, NJ, USA.

Biographies

Karlheinz Kautz is a Professor of Systems Development & Software Engineering at Department of Informatics, Copenhagen Business School, Denmark. Previously, he has been employed as a senior researcher at the Norwegian Computing Center and as a lecturer at universities in Germany, Norway and England. He is the chair of the IFIP TC8 WG 8.6 on Diffusion, Transfer, and Implementation of IT. His research interests are in systems development and system development methodologies in practice, diffusion and adoption of IT innovations, knowledge management and software quality and process improvement. He has published in these areas in journals like *Information and Software Technology*; *Information*, *Technology & People*; *Scandinavian Journal of Information Systems*; *Software Process: Improvement and Practice*; *IEEE Software*; *Journal of Knowledge Management*; *Journal of Information Systems*; *Journal of Informing Science*; and *Journal of Information Technology Cases and Applications*. E-mail: khk.inf@cbs.dk.

Sabine Madsen is an Assistant Professor at the Department of Informatics, Copenhagen Business School. She has recently defended her PhD dissertation on emerging ISD methods in practice and continues her research along these lines. E-mail: sma.inf@cbs.dk.

Jacob Nørbjerg is an Associate Professor at the Department of Informatics, Copenhagen Business School. His research is directed towards the organization and management of systems development, knowledge management and software process improvement. Much of his research is carried out in close collaboration with industrial partners. He has published at several international conferences and journals, including International Conference on Software Engineering, European Conference on Information Systems (ECIS), the *Database for Advances in Information Systems* and *Scandinavian Journal of Information Systems*. He has served as a member of programme committees of international conferences, including ECIS, Information Resources Management Association and ISD and a reviewer for *MIS Quarterly*, *Scandinavian Journal of Information Systems*, the International Conference on Information Systems and others. E-mail: jan.inf@cbs.dk.

