Requirements Eng (2007) 12:203–220
DOI 10.1007/s00766-007-0052-2

ORIGINAL ARTICLE

Web-based systems design: a study of contemporary practices and an explanatory framework based on "method-in-action"

Michael Lang · Brian Fitzgerald

Received: 6 January 2006 / Accepted: 7 June 2007 / Published online: 6 July 2007

Abstract This paper reports the findings of a detailed study of Web-based systems design (WBSD) practices in Ireland based on data collected over a 3-year period (2002–2005), the objectives of which were to (1) contribute towards a richer understanding of the current "real-world" context of WBSD by characterising the profile of a typical project (team size, timeframe, nature of requirements, etc.) and identifying the key challenges, constraints, and imperatives (i.e. "mediating factors") faced by Web-based system designers, and (2) understand how those contextual parameters and mediating factors influence the activity of WBSD as regards the selection and enactment of whatever design practices are therefore engaged (i.e. the use of methods, procedures, etc.). Data was gathered through a survey which yielded 165 usable responses, and later through a series of semi-structured qualitative interviews. Using grounded theory, an explanatory conceptual framework is derived, based on an extension of the "method-in-action" model, the application of which to WBSD has not been previously investigated in depth. It is proposed that this framework of WBSD issues is valuable in a number of ways to educators, researchers, practitioners, and method engineers.

Keywords Web-based systems design practices · Method-in-action · Grounded theory

M. Lang (✉)
Business Information Systems Group,
Cairnes School of Business & Public Policy,
National University of Ireland – Galway,
University Road, Galway, Ireland
e-mail: Michael.Lang@nuigalway.ie

B. Fitzgerald
University of Limerick, Limerick, Ireland

1 Introduction

Keen [1] cautions IS researchers against making rash proclamations of "newness", emphasising the need to always retain an historical consciousness. Nevertheless, there are many instances throughout the relatively short history of computer-based systems development where technological innovations were eagerly pronounced as "paradigm shifts". The spectacular arrival of Web-based systems in the mid-1990s was similarly greeted, it giving rise to a tide of popular academic opinion that "traditional" methods and techniques are poorly suited to the distinctive design challenges raised by these types of systems. On such a premise, Murugesan and Deshpande [2] called for a "new concept and discipline of Web Engineering" and affirmed that there was a "pressing need for new methods and tools" [3]. In similar vein, Oinas-Kukkonen et al. [4] claimed that "systematic analysis and design methodologies for developing Web information systems are necessary and urgently needed among practitioners". Speculation was rife of an imminent "Web crisis" on foot of a prevalent view that industry development practices in general were unsystematic and unreliable. Regretably, arrogant and misplaced proclamations by academia that systems development practice is "sloppy" are not without precedent [5]. However, history shows that practitioners have often overcome adversity by independently devising workable solutions to urgent problems, and also that the lessons of leading-edge best practice have often paved the way for the subsequent advancement of theory [6–8]. Now again, Web designers in industry are going about their business, successfully producing Web-based systems for, literally, the world to behold, mostly without recourse to academic intervention. Bearing this observation in mind, if academic researchers wish to make useful contributions to Web-based systems design (WBSD) practice, perhaps the best place to start is by learning from practice through grounded empirical research.

Midst the initial flurry that is typical for a new research topic, a substantial number of empirical studies of WBSD were published between 1998 and 2002. These focused on such issues as:

- Profile of the development environment (e.g. team size, project duration, issues and challenges, etc.) [9–18]
- Development processes (high-level overview of tasks and phases) [11, 13, 19–22]
- Roles and responsibilities of the design team [14–17, 20, 21, 23]
- Interactions and work practices within design teams [14, 23–27]
- The use of methods and techniques [12, 13, 15–18, 25, 28–36]
- The use of tools [9, 11, 15–18, 27, 33]
- Requirements definition [22, 37, 38]
- Methodology of high-speed development [39–42]
- Skills and knowledge profiles of developers [43]

However, setting aside general HCI research on the effectiveness/usability of Web sites and the mainly experimental and basic research contributions of the Web Engineering community [44] (much of which focuses on implementation technologies), very few empirical studies of actual practices in the world of commercial WBSD have since appeared [18, 38, 41, 42]. In the intervening years since the aforementioned early studies, the Web has progressed from the margins of primitive "brochureware" into the mainstream of business information systems and is now integrated with back-end databases and organisational processes. After the abatement of the pre-Y2K "dot.com" hysteria, there ensued an industry upheaval whereby many of the firms engaging in shoddy or casual practices were found wanting and did not survive. Development technologies have since advanced remarkably, and many Web development firms originally established in the mid- to late-1990s have at this stage attained process maturity. It is therefore a timely juncture to once again look at the state of WBSD practices a few years on. This paper presents the findings of a recently completed study done in Ireland based on data gathered over a 3-year period (2002–2005).

In addition to providing an up-to-date analysis of current practices, this paper makes a novel theoretical contribution in its presentation of an extended variant of the "method-in-action" model of systems development [45], as applied to the domain of WBSD. As such, unlike much previous work in this area, this paper is not merely descriptive in outlining modern practices, but also explanatory in discussing the rationale and motivation for these practices.

Before proceeding, some definitional points must be clarified. "Web-based system" is a rather loose term which necessarily implies nothing about a system except that it is somehow Web-enabled. This could include not just interactive e-commerce systems, but also plain-text legacy databases with visually primitive front-ends, Web crawlers, middleware, server software, and a miscellany of other categories. In the sense that the term "Web-based system" is used in this paper, it refers to that class of systems which have visually-rich graphical user interfaces, robust functional back-ends, and interactive purposefully-designed navigation/information-seeking mechanisms, i.e. what more accurately might be called Web-based *hypermedia* systems.

Though at certain junctures in this paper we alternately refer to the terms "design" and "development", for two principal reasons our general preference is to use the former. Firstly, when we speak of Web-based systems "design", our intended meaning has a dual sense: the target system is a purposefully designed solution, but the means of arriving at that solution is also designed (e.g. "method tailoring" and "situated action"); simply put, "design" encompasses both the product and the process. Secondly, the term "development" bears connotations of coding, construction and back-end software engineering. Our study did not look in any great depth at the physical technologies used by Web developers; these are transient and undergo frequent and dramatic change. Rather, we focused more on design processes and the underlying logic and forces which shape those processes. Hence, it is more correct for us to talk of "Web-based systems design" (hereafter abbreviated as "WBSD") than "Web-based systems development".

2 Research questions and approach

2.1 The research questions were as follows:

RQ1. What is the profile of a typical Web development project?
RQ2. What are the main challenges, constraints and imperatives faced by Web-based systems designers?
RQ3. What practices are being engaged to address these challenges?
RQ4. What factors influence or drive the selection and enactment of design practices?
RQ5. Where formalised design guidance is in place, what is its nature?

The survey research method is especially well suited to RQ1, for it involves the enumeration of the characteristics of a population. RQ2 is also amenable to survey research, because surveys can be useful in "indicating the extent to which perceived problems actually exist, for whom they exist, and their intensity or pervasiveness" [46]. However, surveys are less useful for RQ3, RQ4 and RQ5, because these questions entail not merely descriptive data but also the generation of explanatory insights. As Whitley [29] puts it, "investigating what developers do 'in the wild' is not something that can be adequately addressed by large scale postal surveys … what is required is a more qualitative approach which is able to pick up the nuances of the situation". Case study research is suitable for such purposes, but single-site studies are limited because findings cannot be generalised. A multi-site field study is better where the objective is to generate rich insights and build explanations sufficiently robust to hold across a variety of situations.

A three-phase research approach was therefore taken, as shown in Fig. 1. At the outset, a number of informal meetings were held with a few experienced Web designers to help solidify the research objectives, assess the salience and relevance of certain aspects raised by the literature, and uncover any major topical issues of which we were unaware.

The second phase consisted of a dual-mode (postal and Web-based) survey of 438 organisations. The sampling frame included organisations engaged in bespoke software application development; those specialising in Web or interactive multimedia systems design; companies from traditional media that had branched into "new media"; and large organisations with internal IT departments. The survey received an overall response rate of 52%, ultimately yielding 165 usable responses. The profile of responding organisations is shown in Table 1.

The third and final phase was a follow-up field study, consisting of semi-structured qualitative interviews with 15 Web designers. The selection of interviewees was theoretically driven, chosen so as to seek out similarities and dissimilarities, looking at both typical and atypical cases [47, 48]. They varied according to organisational size, organisational type, application domains, client location (in-house versus external Web development houses), and the interviewee's professional background (see Table 2). Data gathering continued until a point of adequate "theoretical saturation" was reached [48]. Many of the interviewees had recently won or been nominated for awards at prestigious national ceremonies. It was assumed that award winners would be more forthcoming, knowledgeable and insightful, and also that they exemplify best practice. In most of the organisations visited, one personal interview was conducted with the design team leader, typically convened during the mid-day break so as not to encroach upon busy work schedules. In two organisations, multiple interviews took place. Where available, secondary data sources were also consulted. Additionally, follow-up telephone calls and e-mail messages were exchanged in a number of cases to seek clarifications.

Although data gathering for the survey and field study phases was done in chronological sequence, data analysis was an iterative and parallel activity, involving both inductive and deductive reasoning in a grounded, reflective process. Through this triangulation of methods and data,

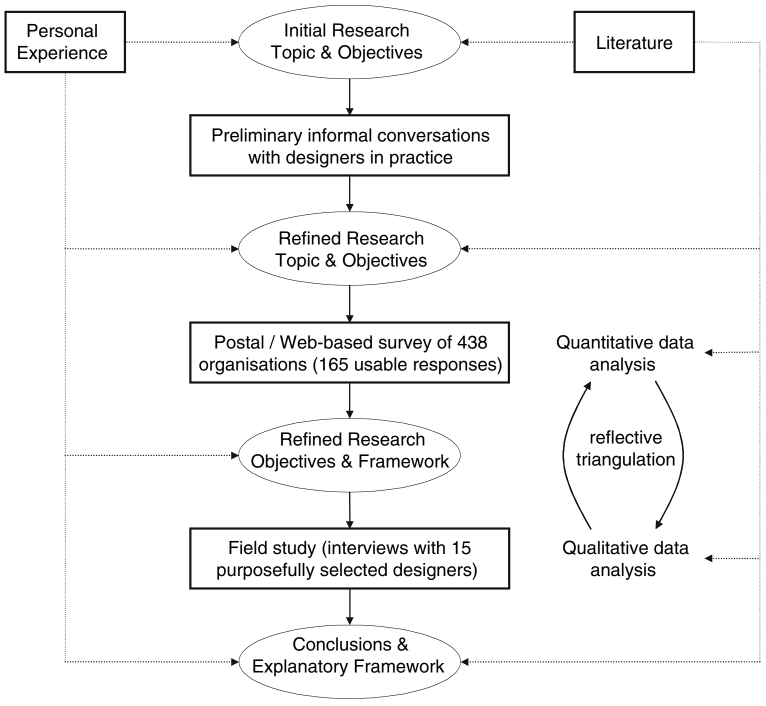
Personal Experience
Initial Research Topic & Objectives
Literature
Preliminary informal conversations with designers in practice
Refined Research Topic & Objectives
Postal / Web-based survey of 438 organisations (165 usable responses)
Quantitative data analysis
reflective triangulation
Refined Research Objectives & Framework
Field study (interviews with 15 purposefully selected designers)
Qualitative data analysis
Conclusions & Explanatory Framework

Fig. 1 Research approach

Table 1 Size and primary business of survey respondents ($n = 164$)

Primary business	Organisation size (number of employees)					Total
	1–20	21–50	51–100	101–500	>500	
Web development	42	2				44 (27%)
IT/software development	9	4		4	5	22 (13%)
Graphic design/visual communications	22					22 (13%)
Multimedia development	13	1				14 (9%)
Management consultancy	4			2	3	9 (5%)
e-Learning/CBT	4	3	2			9 (5%)
Financial services		1	1	1	6	9 (5%)
Public sector			1		6	7 (4%)
Traditional media	1	1		1	3	6 (4%)
Miscellaneous	15	4	2		1	22 (13%)
Total	110	16	6	8	24	164 (100%)

One organisation returned two responses, hence n is 164 here, not 165

Table 2 Profile of field study interviewees

Org #	Industry (all private sector unless otherwise specified)	No. of employees	No. of developers	Interviewee(s) job title	Interviewee background	Interviewee experience (years)
1[a]	University (public sector/in-house)	1,300	2	(1) Chief Web Technologist (2) Web Editor	(1) Software development (2) Physics/Web development	(1) 9 (2) 10
2	Web design agency	10	8	Creative Director	Graphic design	9
3	Graphic design	5	5	Managing Director	Graphic design	12
4	On-line recruitment firm (in-house)	50	3	Web Project Manager	Software development	5
5	Web development	5	4	Managing Director	Computer games development/Software engineering	10
6	Web development	25	20	Commercial Director	Business studies	10
7	Broadcast media (public sector/in-house)	2,000	8	Web Project Manager	Industrial design	6
8	Visual communications	8	8	Managing Director	Film-making/Journalism	>10
9	Web development	30	10	Managing Director	Software development	10
10	Web portal	2	1	Managing Director	Software engineering	15
11	Web design agency	12	7	Senior Designer	Graphic design	7
12	Web development	7	5	Creative Director	Industrial design	10
13 [a]	Web development	60	40	(1) MIS Applications Architect (2) QA Manager	(1) Software development (2) Industrial engineering	Unknown

[a] Multiple interviews took place in these organisations

the inherent weaknesses of individual methods are reduced, strengthening the validity and reliability of findings.

3 Analytical methodology

The analytical methodology that we followed is illustrated in Fig. 2, based on an adaptation of grounded theory. Grounded theory (GT) was chosen for the following reasons:

- It is a systematic yet pliable method. Strauss and Corbin [49] explain that in their formulation of GT, the recommended procedures "were designed not to be followed dogmatically but rather to be used creatively and flexibly by researchers as they deem appropriate".
- GT is analytically *rigorous* but also practically *relevant*. The rigour of GT derives from the internal logic of its procedures, and its relevance derives from its capability to "develop a theoretical account of the

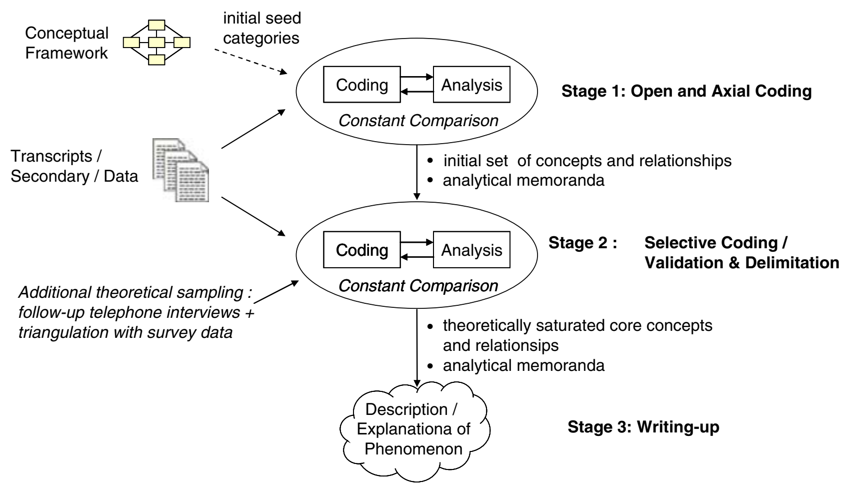
Conceptual Framework
initial seed categories
Coding
Analysis
Constant Comparison
Stage 1: Open and Axial Coding
Transcripts / Secondary / Data
• initial set of concepts and relationships
• analytical memoranda
Coding
Analysis
Constant Comparison
Stage 2 : Selective Coding / Validation & Delimitation
Additional theoretical sampling : follow-up telephone interviews + triangulation with survey data
• theoretically saturated core concepts and relationsips
• analytical memoranda
Description / Explanationa of Phenomenon
Stage 3: Writing-up

Fig. 2 Qualitative data analysis approach followed

general features of a topic while simultaneously grounding the account in empirical observations or data" [50].

- GT is well suited to the investigation of situated processes such as systems design and can produce a multi-faceted explanation of organisational action in context.
- GT can be used not merely to build new theories, but also to extend existing theory and make it more dense by filling in empty or shallow gaps [51].

Walsham [52] recommends that "researchers need to reflect on their own philosophical stance, which should be stated explicitly when writing up their work". Accordingly, we declare the beliefs underpinning this study to be those of the interpretivist paradigm. Quite at odds with the ontology of interpretivism, the original version of GT [47] and the variant subsequently developed by Glaser [53] assume the independent existence of an external reality and the objective neutrality of the researcher, and strive for replicability and verifiability even in social science studies [54]. However, the strand emanating from Strauss and Corbin [55] requires that "interpretations and perspectives of actors on their own and others' actions become incorporated into our own interpretations", which is consistent with an interpretivist research approach. We therefore chose to follow the variant of GT formulated by Strauss and Corbin [49], but made the following procedural adaptations:

- In its original form, GT rejects a priori theorising. However, we chose to construct an initial conceptual framework of seed categories rather than starting from an entirely blank canvas. This decision was guided by Miles and Huberman's [48] advice that some prior specification of existing theory serves to narrow and direct data gathering and analysis, for otherwise one runs the risk of being overwhelmed by the sheer volume of unstructured data.
- The rationale of "theoretical sampling" and "constant comparison" in GT is that the qualitative researcher can avoid becoming swamped with data by continuously developing and refining interpretations in an ongoing process of observation and data collection. However, as Curran and Blackburn [56] point out, the reality of researching small businesses is that data collection cannot be an ongoing process with frequent returns to the field because this is both time-consuming and intrusive, perhaps even disruptive. Access to research sites can be difficult to negotiate because many small businesses are understandably reluctant to commit staff time to external research projects where the benefits to them are perceived at best to be gradual and indirect. Accordingly, just one or two participants were solicited within each organisation contacted in both the survey and field study phases of this research, and follow-up communications were restricted to brief telephone calls and email messages.
- Curran and Blackburn [56] also make the point that data collection in GT may run ahead of theory building because the sheer rate of accumulation offers so many possible alternative interpretations which need to be explored. As anticipated, such was our experience in this project, so our selection of interviewees was driven by the *extant* theory moreso than by the *emerging* theory. As much analysis as possible was done between interviews (e.g. analytical memoranda, repeated listening to digital audio recordings), but of necessity most analysis was done after all the interviews had finished and were fully transcribed.

4 Explanation of empirically-grounded framework

Anselm Strauss, one of the original advocates of GT, has affirmed that it can be used not merely to build new theories, but also to extend existing theory by filling in gaps [51]. The framework derived by this study is an extended variant of the "method-in-action" model [45], the application of which to WBSD has not yet been investigated in depth.

As well as being based on the method-in-action model, the conceptual framework used in this study also synthesises a number of elements drawn from other existing frameworks. The names of the categories (or "bins") into which these concepts are re-organised in Fig. 3 are indicated in the following passages by the use of square brackets [ ].

The NIMSAD model [57] treats systems development as essentially an evaluative problem-solving activity centred upon the tripartite relationship between the problem situation, problem solver, and methodology. Likewise, the conceptual framework of this study views design practice as being situated within the confluence of [Project Factors]/[Mediating Factors] ↔ [Designer] ↔ [Formalised Design Guidance]. Hence, as in the Method-in-Action model, which also draws influence from NIMSAD, the following relationships are posited:

- [Designer] analyses [Project Factors];
- [Project Factors]/[Mediating Factors] shapes [Situated Design Practices];
- [Designer] enacts [Situated Design Practices]

Similar to NIMSAD, Multiview/WISDM [58, 59] sees the enacted ISD methodology as emerging out of the developers' interpretation of the design context, but in addition it acknowledges the multiple perspectives and different thinking styles of individual developers [Designer]. Furthermore, the Multiview framework explicitly recognises that the emergent methodology may utilise methods, techniques and method fragments drawn from *different* sources rather than any single base methodology [Formalised Design Guidance].

Kumar and Bjørn-Andersen's model [60] of the role of designer values in ISD recognises that the shape of the design approach is to a large extent determined not just by the chosen methodology, but also by designers' values, which are derived from their backgrounds [Designer]. Another factor impacting the design approach is organisational control and reward structures, which have both a direct effect on the enacted approach by endorsing desired behaviour and also an indirect effect by conditioning designers' values [Mediating Factors]. From here, the following relationship is derived:

- [Mediating Factors] influences-behaviour-of [Designer]

Gasson's social action model of ISD [61] is quite elaborate, so only certain elements of it are re-used in the conceptual framework for this study:

- [Designer]: The constraining influence of past experience/existing practice on the choice and interpretation of design methodologies is acknowledged, as well as the effect of education, training and backgrounds on individual problem-solving perspectives;
- [Mediating Factors]: The potentially dominant influence of the lead designer on the actions of the design team is recognised (locus of power), as is the possible existence of political pressure from stakeholders to achieve rapid closure.

From Fitzgerald and Fitzgerald's [62] framework for analysing the practice of ISD, the following elements (italicised) are reused:

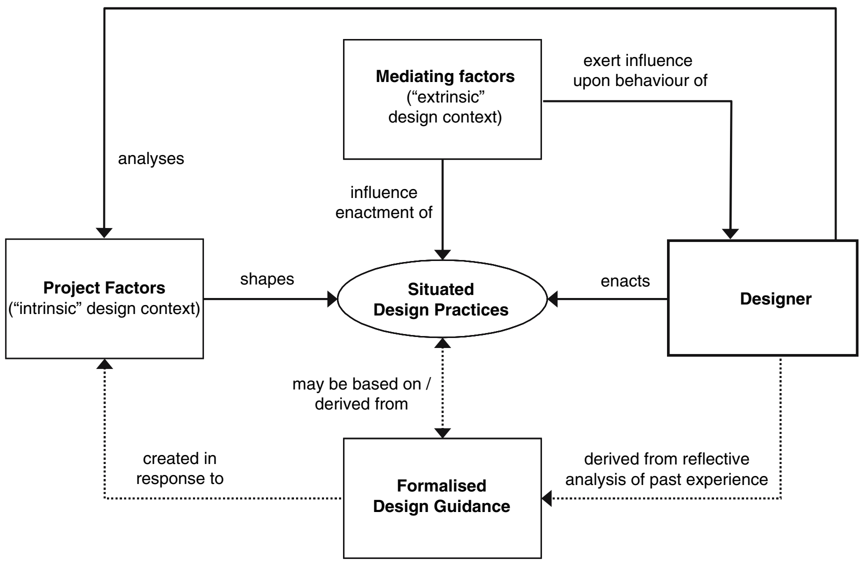
Mediating factors
("extrinsic" design context)
exert influence upon behaviour of
analyses
influence enactment of
Project Factors
("intrinsic" design context)
shapes
Situated Design Practices
enacts
Designer
may be based on / derived from
created in response to
Formalised Design Guidance
derived from reflective analysis of past experience

Fig. 3 Situated model of Web-based design practices

- [Project Factors]: *Where*, e.g. industry sector, nature of business environment, size of development team, location of end users (internal or external), application criticality; *What*, e.g. characteristics of projects; nature of system requirements; *When*, e.g. time-scale for development;
- [Designer]: *Who*, e.g. developer characteristics, developer motivation/values;
- [Formalised Design Guidance]: *How*, e.g. nature of methodology usage; how is development organised?; emphasis on process or product?;
- [Mediating Factors]: *Who*, e.g. nature of client-developer relationship; *Why*, e.g. formality of culture; locus of power.

The iterative GT technique of "constant comparison" was used firstly to synthesize the main concepts of these existing models into a coherent unified framework, and then to mould this initial framework into the empirically-grounded model which emerged as the sense-making tasks of data gathering and analysis progressed. The resulting framework is presented in Fig. 3 and its constituent parts are described in the following sections. At its heart, design practices are regarded as situated actions, purposefully enacted by knowledgable actors who analyse the design context and act accordingly, drawing upon their own experiences to choose an appropriate method.

The foundation of the "situated action" view of design is that, "rather than attempting to abstract action away from its circumstances and represent it as a rational plan, the approach is to study how people use their circumstances to achieve intelligent action" [63]. It rejects the "technical rationalist" assertion that formalised design methods can be executed objectively. Rather, design methods must always be uniquely interpreted; as Essinck [64] puts it, "in a real life project one has to puzzle together one's own specific method, tuned to the problem at hand and the situation the designer is in".

4.1 Project factors (intrinsic design context)

All of the models from which the initial conceptual framework was constructed recognise that design practices must consider the specific situation at hand, variously referred to as the "problem situation"/"methodology context" (NIMSAD), "situation" (Multiview), "context" (Kumar and Bjørn-Andersen), and "business/development context" (method-in-action). Here, the design context is represented by the categories labelled "Project Factors: intrinsic design context" (explained in this section) and "Mediating Factors: extrinsic design context" (explained in Sect. 4.4). The main project-related contextual factors that can affect design practices were found to be: application characteristics, the project timeframe, the profile of the design team, and the nature of requirements.

4.1.1 Application characteristics

Most interviewees and survey respondents had experience of developing Web-based systems in a variety of application domains (e.g. Web design agencies with clients in different industry sectors, or in-house Web teams serving different organisational functions). The general impression which emerged is that most organisations use largely the same design process for all types of applications, regardless of delivery platform or application domain. Some evidence was found that in highly specialised areas such as interactive e-learning/CBT applications, a proprietary domain-specific method might be used, and also that in some industry sectors (e.g. Financial Services/Electronic Banking) there tends to be a greater emphasis on processes and procedures (e.g. detailed functional specifications, formalised organisational guidelines).

While the general development process may be very similar across all projects regardless of application domain, the rigour with which its sub-tasks are executed varies, as one would expect, in accordance with application size/complexity and application criticality. This relationship is well typified by the following transcript excerpt:

> "If a system is critical, it's going to be one of our larger budget developments, in which case its criticality is accommodated by the fact that it already gets the rigour that we apply to all large scale projects. Nearly all of the large scale projects that we're involved in are mission critical anyway, so project scale and system criticality are almost inextricably linked. But the level of process rigour tends to be driven by project scale".

The focus of systems development activity (i.e. in-house versus external client) was also found to impact design practices. Whereas Web design agencies can agree plans with clients and negotiate with them over who pays for subsequent over-runs, in-house development teams are in a "hands tied" situation, meaning that project planning is necessarily done very differently.

4.1.2 Profile of design team

Consistent with the findings of previous studies [11, 12, 16, 65], it was found that most Web design teams are small. Only 7% of survey respondents indicated that they normally work in teams of more than 10, and in almost two-thirds of cases there are less than 5 team members.

Similarly, all field study interviewees indicated that they generally work in design teams of about three to five members. One commented that “project management skills are the most lacking, so keeping a team small is the best way to control the chaos”. Indeed, the survey found a very low incidence of major problems to do with controlling project tasks, managing team collaboration, and communicating with team members within WBSD projects. Generally, there are clearly delineated boundaries between the tasks performed by each member of the team. This is now streamlined through the use of technologies such as CSS and XHTML, whereby front-end and back-end design are cleanly separated, meaning that they can proceed in parallel. This speeds up the development process, facilitates easier maintenance, and reduces team management problems by de-coupling task interdependencies.

The literature suggests that as teams become larger, there is a greater need for formalised processes and procedures, and conversely that “light” methodologies are better suited to small teams [66]. Our survey findings support this view, because as team size increased, a greater propensity to use documented guidelines and procedures was observed ($p < 0.01$; $r_s = 0.25$). One respondent commented that their in-house development approach was tailored so as to be “small enough to be useful for a small company”. Another remarked that “we work in small groups and this tends to obviate formal working methods.” However, as teams grow in size, knowledge becomes fragmented. As revealed by our interview findings, this can lead to wasteful inefficiencies due to “re-inventing the wheel”, which in turn leads to over-worked employees and low morale. There consequently arises a need to formalise and standardise working methods (e.g. conventions for collective code ownership) in order to sustain high-speed high-quality development.

Unlike earlier research [10, 23, 28, 67], conflict between Web designers from different professional backgrounds was not found to be much of a problem in practice in this study. This can be explained by the reality that the once rival factions of software engineering and graphic design have over time come to gain an appreciation of each others’ perspectives and priorities (as evidenced by a considerable degree of cross-skilling), and it is now easier to separate front-end and back-end Web design into different layers than it was a few years ago.

4.1.3 Project timeframes

It was found that the duration of a typical Web development project is of the order of 2–3 months, which is consistent with other studies [9, 11, 12, 17, 18, 65]. At first glance, such short cycles might convey a sense of a hectic work schedule, so-called “Web time”. In the late 1990s, this environment was supposedly characterised by “guerilla programming in a hostile environment using unproven tools, processes, and technology” [68]. However, almost 70% of survey respondents reported having no or minor problems coping with the accelerated timescales of Web development, it being a “major” issue for a meagre 4%. From the interviews, it was clear that the imperative to deliver systems quickly is as much if not more driven by the desire of Web design agencies to maximise throughput as by any sense of genuine urgency on behalf of clients. Delivery cycles of 3 months or less, until recently at least, were unprecedented in traditional software development and are made possible in WBSD by a combination of factors. Firstly, the Web is an immediate delivery medium which, unlike traditional IS and off-the-shelf software applications, is not impeded by production, distribution and installation delays. Secondly, the other major enabling factor has been dramatic gains in recent years in developer productivity, coupled with ever more efficient and refined development processes. These gains have been achieved through the widespread practice of using high-speed rapid application tools, templates and wizards for automatic code-generation, plug-and-play database connectivity, and libraries of pre-fabricated components and applets. This has been refined to a point where most development time is now invested into the ongoing evolution of an out-of-the-box solution, such as advanced content management functionality. Code production for a project has moved from crude cut-and-paste re-use to instant automatic generation, meaning that most of the standard back-end functionality required for any given project is up and running within a day. The visual design of the GUI front-end, like the traditional production process for commercial art, can also be done within a very short timeframe. A fully-proven working prototype can therefore be very quickly launched, which can later be modified and enhanced in such a manner that end-users may be largely oblivious to the ongoing changes. As such, short-cycle evolutionary and incremental development approaches are a natural fit to the Web environment.

Consistent with the previous work of Baskerville and Pries-Heje [40, 41], this study found, as one would expect, that time pressure is the central determinant of design practices. However, there are discrepancies between this research and that of Baskerville and Pries-Heje, most notably with their finding that developers may resort to the practices of “coding your way out” and “negotiated quality” because of the pressures of high-speed development environments. Whereas in Baskerville and Pries-Heje’s study such practices were endemic, in this research hardly any such incidents were discovered. This can be explained in a number of ways. Firstly, the interviewed companies were mostly award-winners, a likely indicator that they make special efforts to strive for excellence and quality. Secondly, the marketplace has become more competitive in recent years and users are much less tolerant of unprofessional standards of work, meaning that expectation levels have risen. Thirdly, as already mentioned, the use of pre-fabricated "productised" solutions that are already fully tested means that robust systems can be rapidly delivered without compromising cost or quality. Even in the worst case scenario for a development team, where they face the dreaded "backs-to-the-wall" combination of acute time and resource constraints, a tactic of "pragmatic satisficing" is engaged, meaning that a tried-and-tested solution is re-used, albeit it may not be the best possible outcome.

4.1.4 Nature of requirements

The clarity and stability of requirements is an age-old issue in systems development, so it came as little surprise to find that the most acute problem in WBSD is controlling project scope/feature creep. 43% of survey respondents regard this issue as moderately problematic, and it poses major problems in a further 17% of cases. The other principal challenge is the preparation of time and cost estimates. Nevertheless, project managers seem to be faring quite well in this latter regard, because 67% of projects are delivered within the agreed budget and 33% are delivered on time. From interviews, it would appear that time slippages are mostly caused by procrastination and indecision by the client, rather than initially poor estimates. Of course, time/cost over-runs and scope creep are intrinsically linked. A major cause of scope creep is that projects often kick-off with a very vague idea of the requirements. As one interviewee explained,

> "When you go into a pitch for a job you'll say 'Yeah, we can turn this around in 6 weeks', but at that stage you don't know what their specific requirements are. So, that timeline will be altered after we find out what they want, and if they're happy to extend the timeline, based on their testing criteria"

The planning/requirements definition phase is the most resource-intensive part of the WBSD process. In comparison with traditional software development, it was found that a greater weighting of time in WBSD is spent on analysis and design as opposed to coding. Requirements analysis is the most time-consuming phase of all in Web development, whereas coding can actually be very quick. Though most of the functional requirements for a Web-based system are typically standard and can therefore be readily described, the bespoke elements take time to specify, as does a considered analysis of the fine details of the overall package including the "non-functional" requirements (usability, accessibility, security, performance levels, etc.). In high-speed environments it is important to "nail" a prioritised list as quickly as possible. Accordingly, as initially revealed by the survey and later substantiated by follow-up interviews, it is common practice to produce and sign-off a detailed requirements specification before commencing full scale production, the purpose of which is to keep feature creep in check and compel clients to make firm decisions. 87% of survey respondents indicated that for their most recently completed project, written specifications were prepared, these typically being about 30 to 40 pages in length. As remarked by one respondent, the requirements specification is essentially a pseudo-legal bargaining chip because "you need sign-off on a project to control creep, cost, and scheduling, but mostly to ensure that you have got a clear brief that you can defend".

4.2 Formalised design guidance

Departing slightly from the original method-in-action framework, the term "formalised design guidance" is used here in preference to "formalised method" because this study found that, even where Web-based systems designers have process documentation in place, it is usually not at the comprehensive level of "method" but more commonly takes the form of high-level process models supplemented by standard operating procedures (SOPs), rules of good practice, heuristics and guidelines, or intranet-based "how-to" WIKIs.

It has been frequently asserted in the literature that Web-based design practice is in a state of "crisis", characterised by sloppy, "quick and dirty" activity [2, 69–71]. A recent study of Web development practices in Norway, based on a convenience sample of 11 organisations, found that in most cases process models do not exist, and even where they do, they are not adhered to [18]. On the contrary, the findings of our study suggest that WBSD in Ireland is quite disciplined and systematic. Eighty-three percent of survey respondents indicated that their organisation has a clearly defined development process, although the process is *explicitly* formalised in only half of these cases. This is consistent with the view that systems design activity may often appear to be ad hoc or perhaps even somewhat chaotic, but beneath the surface is implicitly guided by the purposeful actions of the design team [72]. Interestingly, it was found ($p < 0.05$) that processes tend to be more formalised in Web Development companies than in traditional IT/Software Development companies. A possible explanation for this is the sales-driven high-speed nature of work practices in Web design agencies, as described by an interviewee:

> "You have to streamline how you do things. You have to build processes, put them in place, and just

follow them … So when a Web site comes in, you know exactly what to do, you take it, and you go bang-bang-bang-bang."

The level of formality of development processes was found to be negatively correlated to the level of severity of problems raised by a number of selected development issues (e.g. user interface design, database design, project co-ordination, time and cost estimation), suggesting that formalised processes and procedures can help reduce the incidence of such issues. However, to the extent that process documentation is itself the by-product of experience and trial-and-error, it is erroneous to deduce that the mere existence of formalised process guidelines of their own right lead to more efficient and less troublesome projects. Rather, it is more likely the case that it is the reflective act of producing such documentation that is beneficial. This interpretation is supported by a survey finding that the usefulness of formalised procedures as epistemological instruments is actually quite limited, they being regarded as one of the least useful sources of design know-how (compared to books, on-line forums, observation of colleagues, etc.).

Survey participants were also asked, in an open-ended question, to list the names of any development methods or approaches they had used. Consistent with the findings of previous comparable studies [11, 16], the top response category was in-house methods (23%, $n = 78$). These were mainly either proprietary methods or tailored hybrids. For those about which some detail was provided, they tended to consist of internal procedures built around HCI principles/guidelines, or else were combinations of generic computer-based systems development (CBSD) methods, such as [SSADM + Yourdon + XP], [Waterfall + Spiral + Prototyping], and [RUP + XP]. In quite a few cases, respondents indicated that their in-house method was founded on research, experimentation and experience, and where such in-house methods are in place, they tend to be the only method being used. This suggests that Web-based systems designers, rather than shunning method, are purposefully assembling fragments of public domain methods and distilling the most useful elements to form customised in-house approaches, tuned to the typical demands of the design context. Interestingly, these in-house approaches integrate method fragments drawn from apparently incompatible paradigms (e.g. traditional versus agile, structured versus object-oriented) and as such are a combination of the old and the new.

Some authors have claimed that traditional CBSD methods are not well suited to WBSD [4, 12, 73], so it is notable that this was the second highest response category (22%). Most of these were derivatives of SSADM or SDLC/Waterfall, with Yourdon and Jackson Structured Programming also receiving mention. Notwithstanding its long acknowledged failings, the SDLC/Waterfall model has proven to be quite versatile over time. Indeed, Powell et al. [74] are of the view that a modified version of the Waterfall model "that allows rapid minor changes to the site within a larger general phased effort" is still the most appropriate model for Web development. Of course, it may well be the case that where the SDLC/Waterfall model is being used for WBSD, it is as a project management and pseudo-legal framework rather than an endorsement of any underlying philosophy [16], and covert political motives can also be a factor [5], as later discussed.

The other main response categories were: rapid/agile development methods (15%), approaches focused around the use of tools (14%), incremental/evolutionary methods (13%), and object-oriented methods (8%). However, it seems likely that these categories are under-represented because it is reasonable to assume that many proprietary in-house methods would involve some component of rapid/agile, incremental/evolutionary, or object-oriented methods. Iivari and Maansaari [75] make a relevant point that developers might follow, for example, an object-oriented "approach" comprising a number of "techniques", as opposed to a specific "method". So, while only 8% of respondents indicated that they use an object-oriented *method*, it is of note that a substantially larger cohort have used object-oriented *techniques* such as use case diagrams (72%), class diagrams (62%), and state diagrams (50%).

In the follow-up field study interviews, the nature of in-house methods was investigated in greater depth. When asked to do so, nearly all interviewees were able to clearly articulate the development process used in their organisation, and these descriptions were all remarkably similar. A typical process might run as follows:

0. Project initiation: Specification of the "brief", statement of project goals, outline of timeplan and budget, kick-off meetings. Output: Project Initiation Document (signed-off). Typical turnaround cycle: 1–2 days.
1. Planning/Requirements definition: Review of business strategy and IT/digital communications strategy, profiling of end-users, analysis of competitors, background research, detailed functional specification, outline of information architecture and search/navigation mechanisms, exploration of graphic design "concept", branding/marketing requirements. Outputs: e-Business Plan [optional], Requirements Specification, and Graphic Design Brief (all signed-off). Typical turnaround cycle: 3–4 weeks.

2a. User interface design: Design of GUI look-and-feel, detailed information architecture, usability/accessibility considerations, brand and logo design. Output: Visual templates and style sheets, user interface "skins" (signed-off). Typical turnaround cycle: 3–5 days.

2b. Technical design/Production: Coding, database design, detailed architectural design, linkage of front- and back-end, unit and integration testing. Typical turnaround cycle: 1–2 weeks.
3. Delivery: Upload of system to Web server, entry of content, user training, final testing and quality/usability audits, final "tweaking", full launch. Typical turnaround cycle: 1–2 weeks.
4. Maintenance and marketing: Ongoing support and enhancement, promotion of Web site.

The multidisciplinary nature of WBSD is clearly reflected by the form of this generic process, for it is an amalgam of the traditional processes of software development, graphic design, strategic marketing, and industrial design. There are similarities to the traditional SDLC/Waterfall model in so far as phase products are signed off, there is "big up-front design", and full scale development does not start until the requirements specification is frozen. However, this generic Web design process also differs from the traditional software development process in a number of important ways:

- Because such "non-functional" requirements as branding and the visual look-and-feel are of paramount importance for Web-based systems, the design of the graphical user interface (GUI) is done at an early stage, either before or in parallel with technical design. In the traditional software development process, GUI design was generally relegated as a lesser concern (if it was a concern at all), deferred to the end, and given little forethought.
- Because the Web is an open, distributed delivery platform, the potential audience of end-users is much wider and more diverse than was traditionally the case with business information systems, and for the same reason rival organisations suddenly find themselves competing side-by-side on-line. All interviewees stressed the importance of user profiling, competitor analysis, and explicit strategic planning as key elements of the WBSD process. Traditionally, these aspects need not have explicitly received such detailed consideration.

Returning to the conceptual framework in Fig. 3, a number of relationships were found to exist between Formalised Design Guidance, the Design Context, the Designer, and Situated Design Practices. These are explained as follows:

- [Formalised Design Guidance] created-in-response-to [Design Context]: In cases where organisations had developed their own in-house methods and procedures, these came from a reflective evaluation of recurrent challenges and constraints encountered within previous projects. In the words of one designer,

  "… it was only when issues and problems arose that we had to sit down and say 'right, this is the way to do it', and from our experience, this is the process that we should follow. It was an organic process, it continually changed, depending on what technologies were out there."

  In the literature, there exists a multitude of formalised methods specifically intended for the design of Web-based systems (e.g. EORM, RMM, OOHDM, WSDM, WISDM, W3DT, VHDM, Fusebox). Notably, the level of usage of such methods was found to be negligible in this study. This begs the question if these methods, most of which are academically-produced, are indeed well suited to the real-world context of WBSD. It would seem from the findings of this study that many of these academic WBSD methods have omitted to adequately consider the realities of the context of WBSD in actual practice.

- [Formalised Design Guidance] may-be-based-on/derived-from [Situated Design Practices]: Departing slightly from the original method-in-action model, the relationship between Formalised Design Guidance and Situated Design Practices is indicated here by a *two-way* arrow in explicit recognition that not only can practices be guided by formalised methods, but also that those very methods may themselves be the product of an experiential learning process whereby repeatable courses of action which are discovered to be useful in practice are documented and progressively refined [76, 77]. Substantial evidence of this loop between Formalised Design Guidance and Situated Design Practices was found in the interview transcripts, perhaps best illustrated by the following passage:

  "Design knowledge is mostly gained through experience, but only if the designer is a structured thinker. An ad hoc thinker might just say 'Wow, that was scary, I hope we don't get another one like that', but a structured thinker will say 'This is a difficult situation, if I can list the means I use to get out of this, I'll have a template for dealing with any similar situations in future'."

4.3 Designers

Though some WBSD tasks can be formalised to a high degree of precision, or even automated, there remains an essential need for creative human intervention and the exercise of judgement in the overall process. A strong theme which emerged from interviews was the *importance* *of experience* in knowing how to tailor the process to the situation at hand. As one project manager put it:

> "I've no right and wrong answers but over time you just learn by client and by project what applies to different clients to help get them to the end position as efficiently as possible … The more senior and experienced people in our business would certainly know the right questions to ask. We've fallen into the pit before, so we know not to fall in it again."

Important types of knowledge mentioned were: application domain knowledge, knowledge about development tools/environments and technical standards, knowledge about design methods and techniques, and knowledge of core design principles. Furthermore, knowledge is a critical asset in a development environment characterised by high-speed work practices because it contributes to productivity. More knowledgeable employees are able to work faster because they are equipped with a repertoire of time-efficient "tricks", heuristics, and patterns acquired along the downward traverse of the learning curve. Most award-winning companies have mechanisms in place to facilitate and encourage the management of design knowledge, with rewards and bonuses accruing to employees who use slack time to acquire and exchange useful knowledge. A number of companies schedule regular time slots for research activity, setting aside normal development work. Where practiced, this policy is said to enhance creativity and innovativeness.

A number of authors have mentioned that it would be interesting to investigate the practices of Web designers from backgrounds other than computer-based systems development, so as to build a broader, richer understanding [12, 78]. However, this issue has received very little attention thus far. In view of this gap in the literature, a comparison of the methods and approaches used by designers from different *professional backgrounds* was one of the main concentrations of this study.

In the survey phase of this research, the cover letter attached to the questionnaire simply requested that it be completed by someone in a design role, the rationale being to capture a cross-section of respondents across the various disciplines that contribute to WBSD. As expected, two dominant disciplinary groupings emerged: computer-based systems development (CBSD), and visual design (VD). Differences in priorities and preferences were observed, apparently influenced by the historical practices in each field. For example, the VD group were considerably more lax than the CBSD group as regards requirements documentation (perhaps reflecting the difference between a traditional visual design "brief" and a functional software specification), and were also generally very loose concerning the use of "approaches" and "methods". Indeed, the notion of a design "method" seemed to be alien to many of the VD group. On the other hand, the CBSD group were mostly comfortable with the idea of a systematic process for WBSD, such processes mainly being adaptations of traditional software development methods and techniques.

In the follow-up field study, the influence of professional background on design practices was probed in greater depth. Interestingly, a number of different problem-solving perspectives were discovered, each clearly shaped by the various priorities and orientations of the respective disciplines. The perspectives identified were: WBSD as the design of a functional software application (emphasis on back-end functionality); as the design of an interactive tool (emphasis on ergonomics); as the design of a directed communicational dialogue (emphasis on audience engagement); and as an extension of branded graphic design (emphasis on visual presentation).

The framework therefore recognises that a designer's professional background and education can shape his "world view" by conditioning him to think and behave in certain ways [79, 80]. While different perspectives and orientations were found to exist, it would seem that, at least in the field of practice, there is a growing degree of pluralism, as was evidenced by a substantial degree of cross-skilling and cross-pollination of techniques.

The other main designer-encapulated factor which emerged in this study was *individual commitment*. Again, like knowledge, this is critical in order to be able to sustain a continuous pace of high-speed delivery. Such issues as organisational culture, appropriate reward mechanisms, and the adoption of practices to eliminate morale-sapping overtime were found to be important in this regard.

4.4 Mediating factors (extrinsic design context)

Enacted design practices can sometimes be affected by the intervention of extraneous factors, the influence of which may be to cause designers to pursue a course of action they might not otherwise have taken. For example, there may be a *mandate by the client* that certain procedures are to be rigidly followed (e.g. because of statutory requirements to comply with certain standards, or the existence of binding protocols for procurement or software testing), or not to be followed (e.g. political pressure to complete, "just do it!").

As was previously observed by Powell et al. [74], we found that the *locus of power* within organisations can also significantly influence the design approach. Fledgling in-house Web development units often have to resort to pragmatic satisficing behaviour such as "negotiated quality" [39, 40] because they are under-resourced. In Web design agencies, a typical cause of conflict is the competing motives of the sales team (revenue maximisation) and the design team (quality optimisation), this argument is usually won by the sales team, and programmers might end up being coerced into taking shortcuts to meet targets. The locus of power is also a common issue for client organisations, where the politics, indecision, and communicative difficulties arising from the "design-by-committee" syndrome can frustrate even the best-laid project plans.

Reward and control systems, which are intrinsically tied to organisational priorities, were also found to have an influence on Web development practices. Prerogatives such as "perpetual immediacy" (i.e. constantly pressing deadlines), statutory and regulatory imperatives, a commercial desire to maximise revenue/throughput, a need to be internally flexible with schedules and requirements, or a focus on quality above time and cost considerations can impact development processes by directing priorities. Similarly, the culture of an organisation, as reinforced by control and reward mechanisms, is also a relevant issue (e.g. emphasis on individual *accountability* as opposed to responsible *autonomy*).

Consistent with the original method-in-action model [45], it was also found in this study that design methods may fulfil *covert political roles*. These included: establishing a power-base for method champions (e.g. the XP, WAI, or BS7799 "expert"), maintaining a transparent and accountable audit trail of the design process as a protective fallback (e.g. the in-house "blame game", or negotiating responsibility for change requests or delays with clients), providing assurance that correct and "proper" practices are being followed (e.g. public-sector tenders), and helping to raise the status of in-house Web development departments (e.g. the creation of internal policies to "legitimise" or "professionalise" operations).

5 Discussion and implications of framework

In the area of software design models, the "not invented here" syndrome has unnecessarily led to a proliferation of different notational representations, many of which are essentially very similar. This same complaint can hardly be made of the existing store of conceptual frameworks to describe the activity of information systems development, there being only a few in the literature, none of which have ever been applied to the domain of WBSD. These existing frameworks vary in detail from concise high-level abstractions [57, 81] to elaborate diagrams [61], and in coverage from broad inclusive frameworks [59, 62] to models concentrating on specific issues (e.g. [60]). Nevertheless, whenever a "new" framework is proposed it is legitimate to ask: what is different about it and what additional contribution does it make? We submit that our framework is valuable in a number of ways:

- Because it synthesises, juxtaposes and extends concepts derived from existing frameworks of information systems development, it represents a simple yet reasonably comprehensive structure which can help researchers to order their thoughts, make sense of data, and communicate findings, all the while providing visible anchors that link back to key reference points in the literature.
- Essentially, the framework is a "map" that serves as a mental model of the subject area. Because it draws upon previous frameworks, it is mostly constituted of concepts that already feature in one or more of those frameworks. That said, benefit can often be gained where a body of knowledge is re-packaged and visually presented in a simple form. Just as in cartography, where there are many different ways of drawing a map that represents the same thing, usefulness is determined by such concerns as ease-of-use, intuitive appeal, legibility, visual clarity, and an appropriate abstract balance as regards scale and level of detail. We informally tested our framework by showing it to a number of academic colleagues, all of whom found it a useful and easy to use mechanism to guide the discussion of issues and challenges in WBSD.
- The framework provides a macro-level overview of the main problems and issues in WBSD and how they are inter-related. These issues are often looked at in isolation, rather than within the broader systemic context. While specialised research must understandably be narrowly concentrated in order to delimit scope and achieve anything, a criticism that can be made of much "Web engineering" research, particularly that which concentrates on design methods, is that problems are often investigated without due consideration of their "natural" context in the real-world environment of practice. While there is a vast array of academically-produced WBSD methods in the literature, very few of these are being used in industry. Of course, there are many reasons why this may be so, lack of awareness being one, but the long-standing criticism [82] remains that many of these have only been validated in restricted experimental settings or pilot studies. It would be far more beneficial to report lessons learned from the success, or failure, of these methods as applied to industrial-strength projects. The framework is helpful in this regard by providing academic researchers and method developers with a view of the over-arching context of WBSD, thereby encouraging systemic thinking and "big picture" problem-solving, which ultimately should lead to research products that are more attuned and adaptable to the demands of practice.
- The cumulative body of literature in the field of WBSD is continually expanding, all the more since the

establishment of a number of dedicated journals and conferences. In this regard, the framework could usefully form the basis of a bibliographic taxonomy. Setting out to conduct a literature search of WBSD is daunting because of the depth and breadth of specialised work that exists in the area. A conceptual map of the territory is therefore beneficial, both to novices and those broadly familiar with the literature, in helping to categorise, synthesise and discuss the results of previous and ongoing research. Such an exercise was attempted by Bahli and Di Tullio [44], but their work is now in need of extension and revision.

- Following on, the framework can help to identify aspects of WBSD that have received little attention in the literature, which in turn can inform the choice of relevant directions for future research. It effectively serves as the outline of a research agenda, in the sense that the framework posits a number of inter-related factors and variables that influence WBSD practices.

5.1 Implications for education

Historically, IS/IT graduate programmes placed substantial emphasis on formalised design methods and techniques as described in standard textbooks, neglecting or entirely ignoring the factors which impact the use of those methods and techniques in practice. This limited one-dimensional perspective meant that perplexed graduates straight out of college often found themselves at a loss to understand how so much of the material they had diligently studied seemed to be irrelevant in the "real world". The conceptual framework derived by this research is therefore potentially valuable for educators because it constitutes the outline for a revised and extended curriculum which treats WBSD as a situated contextually-sensitive activity.

A number of particular issues stand out. In the industry of WBSD, competitive advantage does not derive from slavish adherence to prescriptive working methods, but rather from creative and innovative thinking, the efficient exchange of valuable design know-how, and experienced staff who pay meticulous attention to critical details. While graduates almost inevitably have to "go in at the deep end" when recruited into industry, they can best be prepared by using pedagogical approaches (e.g. problem-based learning, service learning) that as close as possible simulate the demands and realities of a commercial work environment. This means that course work should be designed to provide practical coverage of issues drawn from right across the conceptual framework, as opposed to merely focusing on the box labelled "Formalised Design Guidance". For example, an interesting approach to teaching WBSD involving the use of "blogs" as a reflective tool is described by Hollyhead and Cox [83].

In respect of the varying influences of different professional backgrounds on systems design practices, Kumar and Bjørn-Andersen [60] recommended that curricula for teaching, training, and socialising systems designers should be re-designed "to introduce them to design issues and choices other than those with which they are currently familiar". Even yet, a criticism that can be made of many higher-level educational programmes—be they in computer-based systems development, visual design, or other area related to WBSD—is that they produce graduates whose skill sets and problem-solving perspectives are rather narrow. What many employers in the WBSD industry desire are holistic cross-disciplinary programmes with a suitably proportioned blend of IT/technical, systems analysis and design, graphic design, HCI, and business/marketing skills.

Much can be gained by reaching out to the various reference disciplines that contribute to WBSD and defining an integrated body of core knowledge. As Checkland [84] puts it,

> "What we need is not interdisciplinary teams but transdisciplinary concepts, concepts which serve to unify knowledge by being applicable in areas which cut across the trenches which mark traditional academic boundaries"

For example, as was remarked by one interviewee, essential principles of design such as "beauty" and "aesthetics" are universal, relevant not just to graphic design but also to other aspects such as software design. Though there exists a substantial body of literature on design theory—the common ground for various branches of design such as architecture, graphic design, engineering design, and industrial design—the field of WBSD and more generally software design is notable by its extended absence from such journals as *Design Studies* and *Design Issues*. A number of authors have mentioned how, for example, architectural design might serve as a useful reference point for information systems development or software engineering [85–89]. However, the fundamentals of general design theory rarely appear on syllabii for systems analysis and design courses, nor do they explicitly feature in the IS 2002 curriculum [90] or SWEBOK [91].

In addition to the traditional skills and aptitudes required of students, two others stand out: the ability to efficiently seek out answers to technical problems by consulting online information sources, and the ability to rapidly build applications by reverse engineering and customising existing code from in-house libraries or open source modules. This has significant implications for the way in which applications programming is taught and learnt.

5.2 Implications for practice

While the maxim that "there is nothing so practical as a good theory" is oft proclaimed, the reality of such academic rhetoric is another issue. The conceptual framework derived in this research is probably of more use to researchers and educators than to practitioners, but nevertheless it has a number of potential direct applications:

- It can help to identify areas where there are knowledge gaps within the design team, which in turn can be used to plan focused training and education programmes.
- The framework could possibly be used by Web-based system designers in practice as a self-assessment tool, whereby it serves to highlight important management issues and pertinent aspects of organisational processes that are in need of explicit consideration. For example, it could assist the roll-out of a process improvement programme by directing attention to critical aspects and prompting what questions to ask.
- Whitley [29] makes the point that "in order to be able to use a method appropriately, it is necessary to have an understanding of the context in which it is being used". The framework can assist method users and method engineers to gain a better understanding of context because it provides an outline template of issues that may need to be taken into consideration in any particular situation. Of course, many factors will likely be common from one project to the next, but the framework nevertheless focuses attention on the idea that design practices are situated within an environment where there are numerous systemic interdependencies. Organisational processes and working methods therefore need to be purposefully tailored to fit the distinct circumstances of any given project.
- Following on, the framework could form the basis of a knowledge-base schema for a repository of tried-and-tested patterns and combinations of method fragments that over time are found to work effectively, thereby facilitating the retention of organisational memory.

6 Conclusions

This paper presented the main findings of an extensive study of WBSD practices, from which was derived an empirically-grounded conceptual framework, an extended variant of the method-in-action model.

Given the high-speed nature of WBSD, the emphasis of formalised design guidance is very much on agility, speed, efficiency and productivity. Streamlined processes are necessary in order to maximise throughput, and also to sustain a continual pace by eradicating the need for ongoing overtime (which has fatiguing and demoralising effects). Interestingly, many of the participants in this study have independently evolved practices that are remarkably similar to those of the "agile" methods family, such as: collective code ownership; an emphasis on simplicity; the use of regular informal team briefings; insistence on a close working relationship with the client; the pursuit of continuous process improvement through reflective evaluation; and a general emphasis on people, communication, and working software over processes, documentation, and adherence to a plan. A key finding that emerged from our interview data is that where WBSD processes and procedures exist in organisations, their purpose is to serve as flexible templates to guide and assist (rather than constrain and direct) the essentially creative tasks of analysis and design.

Though most of the participants in this study have a clearly understood way of working, in very many cases development processes are not explicitly documented. Design know-how is most efficiently transmitted and acquired by working "on the job", rather than from perusal of formalised procedures or attending training programmes. Most organisations use a "home-cooked" in-house development process that is founded on research, experimentation and reflective analysis of past experience. On the basis of interview findings, these in-house "methods" seem not to be complete end-to-end solutions, but more of a high-level process model within which there is a pick-and-mix selection of low-level techniques to support phase tasks. They are mainly hybrids and custom-tailored variants, based on combinations of internally devised guidelines and public domain methods, informed by an awareness of best industry practice as gleaned from handbooks or on-line forums, and supported by or based around useful tools. This is consistent with the concept of "bricolage" whereby Web-based systems designers, rather than shunning method, judiciously assemble fragments of methods and distil the most useful elements into a flexible custom-made approach which is then applied depending on the needs of the particular situation at hand. Interestingly, these approaches combine the old with the new, retaining the orderly benefits proferred by legacy methods such as the "Waterfall" model while adding the advantages of speed and flexibility made possible by rapid/agile development methods. Additionally, influences from disciplines other than computer-based systems development are being brought into the mix. It is in this coming together and fusion of heretofore disparate fields that WBSD is indeed very different from what has gone before.

Ironically, while there is a vast and ever-growing "jungle" of academically-produced Web/hypermedia systems design methods in the literature, none of which are being used to any significant extent in practice, out in the real world a single generic process dominates, it resembling a modified derivative of the traditional "Waterfall" software development model wedded to an amalgam of sub-processes inherited from the fields of graphic design, HCI, strategic marketing/brand design, and industrial design. What differentiates one company from the next is therefore not the overall shape or format of their development process—notwithstanding the fact that many companies do indeed present their process as a unique selling point—but rather the way in which the finer points of that process are uniquely interpreted by their design team in the specific context of a particular project. A useful direction for further research into WBSD practices would be to move the focus away from methods and method tailoring to concentrate more on technologies and processes which support effective knowledge management, innovation, and creativity within Web design teams. Rather than unnecessarily adding to the existing heap of academically produced methods, it would be far more useful for method researchers to base their future work on "tales from the trenches" of practice.

In addition to the form of the generic Web development process model—which represents a merger of approaches drawn from a variety of sources—the influence of multiple disciplinary fields on the practice of WBSD is evidenced by the finding that all interviewees, regardless of their professional backgrounds, found that the same methods and techniques they had formerly used in their "native" discipline transferred across to Web design. This suggests that wholly new methods and techniques for WBSD are neither necessary nor appropriate. For example, traditional software development techniques (e.g. ERDs for database-driven applications) are still relevant and adequate as regards the specification of back-end functionality, but there is now also an essential need for front-end design techniques drawn from the field of visual communications, such as storyboards and "mood boards". Here too is an area that is worthy of further research: the integration of complementary WBSD techniques drawn from different professions.

While this research project was concerned with WBSD, the point should be made that the design of Web-based systems is more alike than different from what might be called "traditional" or "conventional" non-Web-based systems. In the first instance, Web-based systems are primarily *systems*, regardless of the fact that they are deployed via the Web, and therefore share the same general concerns that are common to the design of business information systems or off-the-shelf software applications. Secondly, as most information systems and many software applications are somehow becoming Web-enabled, the distinction between "Web-based systems" and non-Web-based systems is being eroded. Indeed over time the former term may be cast aside as a redundant anachronism, and already more general terms such as "public information systems" and "wide audience information systems" are beginning to appear. Much of the knowledge that now applies to commercial WBSD can certainly be carried forward to future and emerging technologies, such as the design of fourth-generation mobile information services ("m-commerce") and interactive TV applications ("t-commerce"). Though the empirically-grounded conceptual framework that emerged out of this research was derived and illustrated by reference to an analysis of WBSD practices, it would seem to be broadly applicable to software development in general.

Finally, it should be noted that most of the design teams studied in this research were small. Of course, this is the trend not just in WBSD but across software and information systems development in general. To that extent, many of the design practices described herein are probably broadly applicable. However, the findings cannot be reliably argued to apply to large teams.

References

1. Keen PGW (1991) Relevance and rigor in information systems research: improving quality, confidence, cohesion and impact. In: Nisson H-E et al (eds) Information systems research: contemporary approaches and emergent traditions. Elsevier, Amsterdam, pp 27–49
2. Murugesan S, Deshpande Y (1999) Preface to ICSE'99 workshop on Web engineering. In: 21st international conference on software engineering (ICSE), Los Angeles, May 16–22, pp 693–694
3. Murugesan S, Deshpande Y, Hansen S, Ginige A (1999) Web engineering: a new discipline for development of Web-based systems. In: 1st ICSE workshop on web engineering, Los Angeles, May 16–17, pp 1–9
4. Oinas-Kukkonen H, Alatalo T, Kaasila J, Kivelä H, Sivunen S (2001) Requirements for Web engineering methodologies. In: Rossi M, Siau K (ed) Information modeling in the new millennium. Idea Group, Hershey, pp 360–382
5. Fitzgerald B (1996) Formalised systems development methodologies: A critical perspective. Info Sys J 6(1):3–23
6. Jackson M (1998) Will there ever be software engineering? IEEE Softw 15(1):36–39
7. Glass RL (1989) The temporal relationship between theory and practice. J Sys Softw 10(1):65–67
8. Glass RL (1990) Theory versus practice—revisited. J Sys Softw 12(2):81–82
9. McClure S (1998) Web application development developer perspectives: an IDC white paper. International Data Corporation, Framingham
10. van Aalst JW, van der Mast CAPG (1998) Creating the multimedia project experience database. In: Sutcliffe A et al (eds) Designing effective and usable multimedia systems: Proceedings of the IFIP working group 13.2 conference on designing effective and usable multimedia systems, stuttgart, germany, september 1998. Kluwer, London, pp 117–129
11. Vora P (1998) Designing for the Web: a survey. ACM Interactions 5(3):13–30
12. Russo NL, Graham BR (1999) A first step in developing a Web application design methodology: understanding the environment.

In: Wood-Harper AT et al (eds) Methodologies for developing and managing emerging technology based information systems: 6th international BCS information systems methodologies conference. Springer, London, pp 24–33
13. Rodriguez-Garcia D, Harrison R (2000) Practitioners views on Web development: an industrial survey by semi-structured interviews. In: 13th international conference on software and systems engineering and their applications (ICSSEA 2000), Paris, December 5–8
14. Greenbaum J, Stuedahl D (2000) Deadlines and work practices in new media development: its about time. In: Cherkasky T et al (eds) PDC 2000 participatory design conference, New York, November 28–December 1, pp 70–77
15. Lang M, Barry C (2001) Techniques and methodologies for multimedia systems development: a survey of industrial practice. In: Russo NL et al (eds) Realigning research and practice in information systems development: the social and organizational perspective. IFIP WG8.2 conference, Boise, Idaho, USA, 27–29 July 2001. Kluwer, Boston, pp 77–86
16. Barry C, Lang M (2001) A survey of multimedia and Web development techniques and methodology usage. IEEE Multimedia 8(3):52–60
17. Barry C, Lang M (2003) A comparison of "traditional" and multimedia information systems development practices. Info Softw Tech 45(4):217–227
18. Zhou J, Stålhane T (2004) Web-based system development: Status in the norwegian IT organizations. In: Bomarius F, Iida H (eds) Product focused software process improvement: Fifth international conference, PROFES 2004, Kansai Science City, Japan, April 5–8, 2004, proceedings. Springer, Heidelberg, pp 363–377
19. Oinas-Kukkonen H (1994) Lessons learned from developing hypertext applications. In: Chrisment C (eds) Information systems design and multimedia. Cépadués-Editions, Toulouse, France, pp 255–261
20. Liu M, Jones C, Hemstreet S (1997) A study of the multimedia design and production process by the practitioners. In: ED-MEDIA/ ED-TELECOM World Conference on Educational Multimedia, Hypermedia and Telecommunications, Calgary, Canada, June 14–19, pp 634–639
21. Liu M, Jones C, Hemstreet S (1998) Interactive multimedia design and production processes. J Research Computing Educ 30(3):254–280
22. Lowe DB, Eklund J (2002) Client needs and the design process in Web projects. J Web Eng 1(1):23–36
23. Carstensen PH, Vogelsang L (2001) Design of Web-based information systems—new challenges for systems development? In: Ninth European conference on information systems (ECIS), Bled, Slovenia, June 27–29, pp 536–547
24. Gallagher S, Webb B (1997) Competing paradigms in multimedia systems development: Who shall be the aristocracy? In: Galliers RD et al (eds) Fifth European Conference on Information Systems (ECIS), Cork, June 19–21, pp 1113–1119
25. Newman MW, Landay JA (2000) Sitemaps, storyboards, and specifications: A sketch of Web site design practice. In: ACM symposium on designing interactive systems, Brooklyn, New York, August 17–19, pp 263–274
26. Jonasson I (2002) Trends in developing Web-based multimedia information systems. In: Kirikova M et al (eds) Information systems development: Advances in methodologies, components and management. Kluwer, New York, pp 79–85
27. Klemmer SR, Thomsen M, Phelps-Goodman E, Lee R, Landay JA (2002) Where do Web sites come from? Capturing and interacting with design history. In: ACM SIGCHI conference on human factors in computing systems. Minneapolis, April 20–25
28. Britton C, Jones S, Myers M, Sharif M (1997) A survey of current practice in the development of multimedia systems. Info Softw Tech 39(10):695–705
29. Whitley EA (1998) Method-ism in practice: Investigating the relationship between method and understanding in Web page design. In: 19th international conference on information systems (ICIS), Helsinki, December 13–16, pp 68–75
30. Paynter J, Pearson M (1998) A case study of the Web-based information systems development. Department of Management Science and Information Systems. University of Auckland, New Zealand
31. Eriksen LB (2000) Limitations and opportunities for system development methods in Web information system design. In: Baskerville R et al (eds) Organizational and social perspectives on information technology, IFIP TC8 WG8.2 international working conference on the social and organizational perspective on research and practice in information technology, June 9–11, 2000, Aalborg. Kluwer, Boston, pp 473–486
32. Alatalo T (2001) Lessons learned from putting a Web engineering approach to practice. In: 24th information systems research seminar in Scandinavia (IRIS). Ulvik in Hardanger, Norway, pp 11–14
33. Taylor MJ, McWilliam J, Forsyth H, Wade S (2002) Methodologies and website development: a survey of practice. Info Softw Tech 44(6):381–391
34. Vidgen R (2002) Constructing a web information system development methodology. Info Sys J 12(3):247–261
35. Vidgen R (2002) What's so different about developing Web-based information systems. In: Wrycza S (eds) 10th European conference on information systems (ECIS), Gdansk, June 6–8, pp 262–271
36. Safieddine F (2003) Survey summary. Ph.D. Research, University of East London
37. Lowe D (2003) Web system requirements: an overview. Requirements Eng 8(2):102–113
38. Bleek W-G, Jeenicke M, Klischewski R (2004) E-prototyping: Iterative analysis of Web user requirements. J Web Eng 3(2):77–94
39. Baskerville R, Levine L, Pries-Heje J, Ramesh B, Slaughter S (2001) How internet software companies negotiate quality. IEEE Computer 34(5):51–57
40. Baskerville R, Pries-Heje J (2001) Racing the e-bomb: How the internet is redefining information systems development methodology. In: Russo NL et al (eds) Realigning research and practice in information systems development: the social and organizational perspective. IFIP WG8.2 conference, Boise, 27–29 July 2001. Kluwer, Boston, pp 49–68
41. Baskerville R, Pries-Heje J (2004) Short cycle time systems development. Info Sys J 14(3):237–264
42. McDonald A, Welland R (2005) Agile Web engineering (AWE) process: perceptions within a fortune 500 financial services company. J Web Eng 4(4):283–312
43. Taylor MJ, England D, Gresty D (2001) Knowledge for Web site development. Internet Research 11(5):451–461
44. Bahli B, Di Tullio D (2003) Web engineering: an assessment of empirical research. Commun AIS 12(14):203–222
45. Fitzgerald B, Russo NL, Stolterman E (2002) Information systems development: methods in action. McGraw-Hill, London
46. Kraemer KL, Dutton WH (1991) Survey research in the study of management information systems. In: Kraemer KL (eds) The information systems research challenge: survey research methods. Harvard Business School, Boston, pp 3–58
47. Glaser BG, Strauss AL (1967) The discovery of grounded theory: strategies for qualitative research. Aldine de Gruyter, New York
48. Miles MB, Huberman AM (1994) Qualitative data analysis: an expanded sourcebook, 2nd edn. Sage, Thousand Oaks

49. Strauss A, Corbin J (1998) Basics of qualitative research: techniques and procedures for developing grounded theory, 2nd edn. Sage, Thousand Oaks
50. Martin PY, Turner BA (1986) Grounded theory and organizational research. J Appl Behav Sci 22(2):141–157
51. Strauss AL (1970) Discovering new theory from previous theory. In: Shibutani T (eds) Human nature and collective theory. Prentice Hall, Englewood Cliffs, pp 46–53
52. Walsham G (1995) Interpretive case studies in IS research: nature and method. Eur J Info Sys 4(2):74–83
53. Glaser BG (1992) Basics of grounded theory analysis. Sociology Press, Mill Valley
54. Charmaz K (2000) Grounded theory: objectivist and constructivist methods. In: Denzin NK, Lincoln YS (eds) Handbook of qualitative research. 2nd edn. Sage, Thousand Oaks, pp 509–535
55. Strauss A, Corbin J (1994) Grounded theory methodology: an overview. In: Denzin NK, Lincoln YS (eds) Handbook of qualitative research. Sage, Thousand Oaks, pp 273–285
56. Curran J, Blackburn RA (2001) Researching the small enterprise. Sage, London
57. Jayaratna N (1994) Understanding and evaluating methodologies, NIMSAD: a systemic framework. McGraw-Hill, London
58. Avison DE, Wood-Harper AT, Vidgen RT, Wood JRG (1998) A further exploration into information systems development: the evolution of Multiview2. IT People 11(2):124–139
59. Vidgen R, Avison D, Wood B, Wood-Harper T (2002) Developing Web information systems: from strategy to implementation. Butterworth Heinemann, Oxford
60. Kumar K, Bjørn-Andersen N (1990) A cross-cultural comparison of IS designer values. Commun ACM 33(5):528–538
61. Gasson S (1999) A social action model of situated information systems design. DATA BASE 30(2):82–97
62. Fitzgerald B, Fitzgerald G (1999) Categories and contexts in IS development: making sense of the mess. In: Pries-Heje J et al (eds) Seventh European conference on information systems (ECIS). Copenhagen, June 23–25, pp 194–211
63. Suchman LA (1987) Plans and situated actions: the problem of human-machine communication. Cambridge University Press, Cambridge
64. Essinck LJB (1988) A conceptual framework for information systems development methodologies. In: Bullinger H-J et al (eds) Information technology for organisational systems. North-Holland, Amsterdam pp 354–362
65. Glass RL (2001) Who's right in the Web development debate? Cutter IT J 14(7):6–10
66. Cockburn A (2000) Selecting a project's methodology. IEEE Softw 17(4):64–71
67. Koechlin O (1997) Methods and tools to improve software quality for multimedia productions. Final report ESSI project no. 21545. European System and Software Initiative
68. Thomas D (1998) Web time software development. Softw Dev Mag 6(10):78–80
69. Lowe D, Hall W (1999) Hypermedia and the Web: an engineering approach. Wiley, Chichester
70. Pauen P, Voss J, Six H-W (1998) Modeling hypermedia applications with HyDev. In: Sutcliffe A et al. (eds) Designing effective and usable multimedia systems: Proceedings of the IFIP working group 13.2 conference on designing effective and usable multimedia systems, Stuttgart, Germany, September 1998. Kluwer, London, pp 23–40
71. De Troyer O (2001) Audience-driven Web design. In: Rossi M, Siau K (eds) Information modeling in the new millennium. Idea Group Publishing, Hershey, pp 442–461
72. Ciborra CU (1999) A theory of information systems based on improvisation. In: Currie WL, Galliers B (eds) Rethinking management information systems. Oxford University Press, Oxford, pp 136–155
73. Siau K, Rossi M (2001) Information modeling in the internet age—challenges, issues and research directions. In: Rossi M, Siau K (eds) Information modeling in the new millennium. Idea Group, Hershey, pp 1–8
74. Powell TA, Jones DL, Cutts DC (1998) Web site engineering: beyond Web page design. Prentice Hall, Upper Saddle River
75. Iivari J, Maansaari J (1998) The usage of systems development methods: are we stuck to old practices? Info Softw Tech 40:501–510
76. Schön DA (1983) The reflective practitioner: how professionals think in action. Temple Smith, London
77. Shaw M, Garlan D (1996) Software architecture: perspectives on an emerging discipline. Prentice Hall, Upper Saddle River
78. Jonasson I (2000) Developing the information systems of tomorrow—competencies and methodologies. University of Skövde, Sweden
79. Sahraoui S (1998) Is information systems education value neutral? J Comput Info Sys 38(3):105–109
80. Kuhn TS (1996) The structure of scientific revolutions, 3rd edn. University of Chicago Press, Chicago
81. Lyytinen K (1987) A taxonomic perspective of information systems development: theoretical constructs and recommendations. In: Boland RJ, Hirschheim RA (eds) Critical issues in information systems research. Wiley, New York, pp 3–41
82. Fitzgerald G (1991) Validating new information systems techniques: a retrospective analysis. In: Nissen H-E et al (eds) Information systems research: contemporary approaches and emergent traditions. Elsevier B.V., North-Holland pp 657–672
83. Hollyhead A, Cox S (2006) Using blogs to encourage reflection in the undergraduate teaching of internet application development. In: Lang M et al (eds) AIS SIGSAND European symposium on systems analysis and design: practice and research. Galway, June 6, pp 47–55
84. Checkland PB (1976) Science and the systems paradigm. Intl J General Systems 3(2):127–134
85. Zachman JA (1987) A framework for information systems architecture. IBM Sys J 26(3):276–292
86. Lee AS (1991) Architecture as a reference discipline for MIS. MIS Research Center. University of Minnesota, Minneapolis
87. Introna LD (1996) Notes on ateleological information systems development. Info Tech People 9(4):20–39
88. Ebert C (1997) The road to maturity: navigating between craft and science. IEEE Softw 14(6):77–82
89. Atkinson C, Avison D, Wilson D (2003) Images of information systems development in the practice of architecture. Commun AIS 12(18):283–300
90. Gorgone J, Davis GB, Valacich JS, Topi H, Feinstein DL, Longenecker HE (2003) IS 2002 model curriculum and guidelines for undergraduate degree programs in information systems. Commun AIS 11(1):1–63
91. IEEE Computer Society (2004) Guide to the software engineering body of knowledge, http://www.swebok.org