Preprint: final version has been published as:
BLANDFORD, A., KEITH, S., BUTTERWORTH, R, FIELDS, B. & FURNISS, D. (2007) Disrupting Digital Library Development with Scenario Informed Design. *Interacting with Computers* 19. 70-82. DOI http://dx.doi.org/10.1016/j.intcom.2006.07.003

Disrupting Digital Library Development with Scenario Informed Design

Ann Blandford[a], Suzette Keith[b], Richard Butterworth[b], Bob Fields[b] & Dominic Furniss[a]
[a] UCL Interaction Centre, University College London, Remax House, 31-32 Alfred Place, London WC1E 7DP.
Email: A.Blandford@ucl.ac.uk
Tel.: +44 (0)20 7679 5288.
Fax: +44 (0)20 7679 5295.
[b] Interaction Design Centre, Middlesex University, The Burroughs, Hendon, London NW4 4BT.

Abstract

In recent years, there has been great interest in scenario-based design and other forms of user-centred design. However, there are many design processes that, often for good reason, remain technology-centred. We present a case study of introducing scenarios into two digital library development processes. This was found to disrupt established patterns of working and to bring together conflicting value systems. In particular, the human factors approach of identifying users and anticipating what they are likely to do with a system (and what problems they might encounter) did not sit well with a development culture in which the rapid generation and informal evaluation of possible solutions (that are technically feasible and compatible with stable system components) is the norm. We found that developers tended to think in terms of two kinds of user: one who was exploring the system with no particular goal in mind and one who knew as much as the developer; scenarios typically work with richer user descriptions that challenge that thinking. In addition, the development practice of breaking down the design problem into discrete functions to make it manageable does not fit well with a scenario-based approach to thinking about user behaviour and interactions. The compromise reached was scenario-informed design, whereby scenarios were generated to support reasoning about the use of selected functions within the system. These scenarios helped create productive common ground between perspectives.

Keywords: Digital Libraries; Scenario Based Design; usability evaluation; software development processes.

1. Introduction

The development of any complex interactive system is demanding. There are various factors to take into account: reliability, maintainability, interoperability, usability etc. It is very difficult for a development team to keep all of these requirements in focus simultaneously, or even to pay attention to each one as it becomes relevant. In particular, usability is often sidelined within traditional software development practice.

In the work reported here, we investigated ways of introducing user concerns into traditional software development processes that had previously focused on functionality and reliability. The approach taken was based on participant observation, in which a human factors specialist joined each development team at appropriate points within the development process and investigated the use of scenarios (Rosson & Carroll, 2002) and Claims Analysis (Carroll & Rosson, 1992) as techniques for including a stronger user focus within that development process. One of our partner organisations used both scenarios and Claims Analysis; the other only used scenarios. The findings related to Claims Analysis are reported elsewhere (Blandford, Keith & Fields, in press). In this paper, we focus on those relating to the development and use of scenarios.

Before presenting the study and findings, we summarise the background under three headings:

- Understanding the target system: design and usability evaluation of digital libraries;
- Scenarios; and
- Previous work on making usability usable.

1.1 The development context: development and usability of digital libraries

While the focus of this paper is not on digital libraries (DLs) per se, but on the experience of working with developers, DLs provide an essential strand to the context of this work.

Historically, much work on usability of DLs has been post hoc – at least in the sense that a fully functioning prototype system exists to be tested (e.g. Papadakis et al, 2002; Komlodi & Marchionini, 1998). More recently, there has been an interest in check-list based approaches that make explicit some of the desirable features of DLs (e.g. Kyrillidou & Giersch, 2005). While this work has advanced the understanding of usability issues for DLs, these approaches do not lead to techniques that can be directly embedded within development processes.

Few descriptions of DL development processes exist. Bates (2002) gives one account of why this might be, describing a digital library in terms of a "cascade" of interacting components. Although not a development model itself, Bates' work sets out to establish the relationships between the various entities (content, search mechanism, user interface etc.) that make up a DL system, with the intention that a DL designer who better understands what the entities are and how they interrelate is likely to come to better design decisions.

The majority of existing development descriptions are case studies giving accounts of the design and testing of individual DL projects, typically for use in tertiary education. In particular, various researchers (e.g. Champeny et al, 2004; Spasser, 2003) have given accounts of development processes that they have been directly involved in. Champeny et al focus on the overall design process, which they describe as involving "iterative, formative evaluation, a process in which system design is studied in parallel with user needs and requirements". In contrast, Spasser (2003) is more concerned with the selection of material to be included within the DL – i.e. with collection building.

Most (though not all) DL projects reported in the literature are experimental, exploratory research projects, and this may go some way towards explaining why there is a paucity of design process models reported for DL systems. A design process should reduce the risk of failure in a development project by steering the designers towards good practice. By its nature a research project is high risk, and therefore should not be bound by possibly restrictive design processes. However as DL systems transfer into public domains, it is more important to develop design processes which can ensure the delivery of working, usable DL systems that are fit for purpose.

One particular problem for DL systems implemented outside the educational domain is the difficulty in identifying well defined user groups. Users and their tasks can be fairly easily identified in an educational setting, but many libraries do not sit in such well defined contexts. Butterworth, Ghosh and Wise (2005) argue that many DL projects set out explicitly to 'broaden access' to library resources into new user domains, and that in this case it is theoretically impossible to collect a priori user requirements from an unidentified user group. Typical design processes (even sophisticated, iterative, requirements driven processes) start from the assumption of there being a user group which can be delimited by some homogeneous need, and are therefore not well suited to DL development. Butterworth et al's solution to the problem is similar to Champeny et al's: to iteratively develop a DL system in parallel with assumed user requirements, and to refine both the DL system and the assumptions about the users' requirements as part of the development process. While both of the development contexts studied here include iteration in design, there has historically been little user focus within those iterations: the use of scenarios as a means of taking a user perspective within an iterative development process is the focus of the work reported here.

1.2 Scenarios

The use of scenarios is an established approach to thinking about users within design. Indeed, Carroll (2002, p.621) asserts that "it is impossible to find HCI methods today that do not incorporate scenario-based design". However, a variety of approaches to generating and using scenarios can be found in the literature.

Young and Barnard (1987) propose and illustrate the use of scenarios to test cognitive theories, arguing that HCI scenarios can play an important role in science by challenging cognitive psychologists to give theory-based accounts of human behaviour (as encapsulated within those scenarios). More recently, the focus of scenario use has shifted towards supporting design rather than science. Carroll and Rosson (1992, p.185) describe a scenario as a description of "the activities a user might engage in while pursuing a particular concern". Different kinds of scenarios can be produced, focusing on existing or envisaged tasks and interactions, and described at different levels of detail according to purpose. Carroll and Rosson discuss the selection of scenarios in terms of typical and critical use scenarios: focusing on the tasks people perform most frequently and those that are most likely to cause problems. The main source of scenarios discussed by Carroll and Rosson is empirical studies of how users work; in principle, it is also possible to create them by reflecting on how users are intended to work (notably when creating new interaction possibilities or systems for new user populations). Sutcliffe and Carroll (1999) extend this to describe three different kinds of scenario: problem initiation (describing the situation prior to re-design); usage (describing a sequence of user–system interaction,

based on empirical evidence); and projected usage (describing anticipated interaction with a re-designed artefact).

Cooper (1999) proposes a slightly different approach to scenarios that focuses on "personas". He describes personas as "hypothetical archetypes of actual users". He argues strongly that personas should be at the centre of the design process, and that a system should be designed for a very small number of personas – ideally, just one. Personas, which include a detailed description of the hypothetical user, include information about how that user might work with a system. However, since the system might not exist yet, there is typically less emphasis on details of interactions, and more on user features and requirements.

Within the HCI literature, there are many reports on the use of scenarios within design; these take a variety of perspectives – from the use of scenarios to support requirements acquisition (e.g. Maiden & Robertson, 2005) to their use in fostering creativity and co-operation (e.g. Bødker et al, 2000). However, as far as we can establish, there have been no studies of how to incorporate scenarios within an established function-oriented design process – an approach that can be regarded as "scenario informed design" rather than the more familiar "scenario based design".

1.3 The challenge: making usability usable

The challenge of making usability-oriented design and evaluation techniques themselves useful and usable has been with us for a long time (e.g. Hammond et al., 1983). Bellotti (1989) observed that usability techniques from the HCI research community had poor take-up within design practice, and presented a range of reasons for this, from users being difficult to contact to many user-oriented techniques being inappropriate to particular design situations. The problem is highlighted by O'Neill (1998, p. 65) when he refers to "the largely undisturbed arsenal of system development methods". With a view to alleviating this problem, Rosson et al. (1988) and Bellotti et al. (1995) are amongst those that argue for a better understanding of the practitioners, their context and their tasks, so as to overcome barriers to method use. Buckingham Shum and Hammond (1994) discuss the same issue in terms of 'gulfs' between research and practice. There are arguably at least two gulfs: between research and practice and between protagonists with a user and a technology focus.

Heinbokel, Sonnentag, Frese, Stolte and Brodbeck (1996) go further than identifying gulfs, and actually suggest that user participation, or even a user orientation, within design practice can hinder software development. The measures of success they use explicitly avoid the ultimate fitness for purpose of the designed artefact (including, of course, its perceived usability and usefulness); rather, they use software engineering metrics. One interpretation of their findings is that actually designing for users makes design problems more complex, hence 'better' designs ignore how systems will ultimately be used. Suggested reasons for this include "higher levels of aspiration and, therefore, […] a higher degree of stressors and a lower degree of team effectiveness [particularly] when only little knowledge and experience are available on how to put user orientation into practice" (Heinbokel et al, 1996, p.234). In the study reported here, it has not been possible to assess the quality – on any measures – of the design revisions made as a consequence of introducing user-oriented development methods, but it has been clear that the additional demands placed on developers, who already had difficult technical problems to deal with, were disruptive, for reasons discussed below.

Even in the twenty-first century, there are surprisingly few accounts of the incorporation of HCI techniques within (non-research-focused) design practice, and what little evidence there is (e.g. Paddison & Englefield, 2003) points to the use of largely post-hoc, discount usability techniques such as heuristic evaluation (Nielsen, 1994) and lightweight empirical user testing (Vredenburg et al, 2002). Reliance on such techniques means that many inappropriate design decisions may have been made much earlier in the design process, and become design commitments, long before user perspectives are taken on board. There are counter-examples; for example, Spencer (2000) describes the incorporation of the Cognitive Walkthrough technique within an ongoing design process. However, Spencer's description shows that the process of assimilating CW with an existing design process demanded adaptation on both sides: of the CW technique (simplifying it to make it more efficient) and of the design practice.

Whether or not it is true (as Carroll (2002) asserts) that all HCI methods today incorporate scenario-based design, there are still design processes that are not user-centred; a shift towards scenario-based design has not been apparent in our studies – either of DL development (as in the study reported here) or of other systems. For example, Blandford and Rugg (2002) report on an industrial design process in which there was a strong focus on systems reliability, but minimal consideration of users and use – whether in requirements acquisition or subsequent design and evaluation of the product. In their study, Blandford and Rugg introduced a suite of user-oriented techniques within one design cycle, and design insights from the exercise were adopted within the following design iteration, but there was palpable resistance to further change of the established design process.

2. Methodology

The study reported here is located in the real world of system development (as advocated by John (1998)). This account draws on various experiences over a three-year project that started with the broad aim of developing user-oriented tools that were applicable in DL design practice. The work has been exploratory and participatory rather than following a more traditional science research paradigm.

In the early stages of the project, various user-oriented design and evaluation techniques were investigated, with two particular concerns in mind:

- They should be adaptable to fit within existing DL design practice: to make any impact at all, we needed techniques that were acceptable to DL developers, and that could co-evolve with their design practices.
- Techniques should be ones that could accommodate the kinds of issues that really make DLs difficult to use.

From the earliest days of the project, we were working with librarians and developers in a large telecommunications company (BT), learning about their design processes and their users' needs, and developing and testing candidate usability tools. Data collection and analysis were qualitative, based on interviews and observations.

Scenarios and Claims Analysis were adopted as a focus for the work because:

- Some other techniques that we had tested, including Cognitive Walkthrough (CW: Wharton et al, 1994) and Heuristic Evaluation (Nielsen, 1994), gave useful insights about superficial aspects of design, such as the suitability of labels and the quality of feedback, but did not give leverage on deeper

usability issues concerning information seeking (Blandford et al, 2004). Scenarios and Claims offered the promise of addressing these deeper issues.

- Rather than being purely evaluative, scenarios and Claims are explicitly designed to fit within ongoing design practice; thus, the analysts' reflection on design strengths and weaknesses could directly inform design changes.
- They appeared to be sufficiently light-weight to be acceptable to developers without a strong user-oriented analytical background.
- Compared to general design rationale (e.g. QOC: MacLean, Young, Bellotti & Moran, 1991), the psychological basis of Claims Analysis provides more explicit support for viewing design from a user perspective.

Later in the project, we worked with a second development team – this time, developers of the Greenstone DL software, based at the University of Waikato. This study was necessarily time-limited: initial meetings were held with the lead developer when he was in the UK, but the main study was conducted in a series of interviews and team meetings over a 2-week period in New Zealand.

Each study comprised three main phases:

1. Initial exploratory participant observation work that investigated the design approach taken by the development team and collaboratively developed scenarios. By coincidence, this phase with each team focused on the development of a novel feature (Jasper at BT and the Gatherer in Waikato).
2. Formal training in development of scenarios and Claims Analysis. In particular, this training focused on the idea of a simple action cycle, in which users form goals, execute actions and receive feedback from the system to structure some of the discussion of scenarios.
3. More structured investigation of scenarios, in which the developers created their own scenarios of use and the discussion focused on the implications for design. Again, these sessions took a participation observation approach. These sessions happened to focus on review and redesign of existing system features with both development teams.

Clearly, the approach employed in this study is not a standard scientific method. It is not reproducible: if a different team of people had conducted the study, the process and findings would have been different in detail. The 'evaluator effect' (Hertzum & Jacobsen, 2001) has been found to apply even to one individual evaluating a stable system, so it is unavoidably much greater where teams are involved in evaluating an evolving system. This is action research, in which insights have been achieved through the process of working with developers and reflecting on the experience. As we show in the accounts that follow, we have not chosen design problems to be particularly well suited to the approach adopted; neither have the development processes adopted by our collaborators been textbook development lifecycles. We have adapted our investigation to fit with development constraints, and the development teams, in turn, have worked hard to accommodate us and create constructive interactions. Thus, although the work has limited reproducibility, it has much higher ecological validity than most studies that have looked at applicability of user-oriented techniques in non-user-centred design practice.

We describe each of the studies in more detail here, before drawing out general findings and conclusions. As noted above, the Claims Analysis work with BT is

described elsewhere (Blandford, Keith & Fields, in press); here we focus on their generation and use of scenarios. The Greenstone developers did not engage with Claims Analysis, preferring to work with only scenarios. To better understand how scenarios fit within the development context, we start each section by outlining key features of the development team, their access to users and the processes they follow.

2.1 Work at BT: the library and Jasper

Like many public- and private-sector organisations, BT provides a DL interface to a range of digital resources to which it subscribes on behalf of its staff. It also aims to provide 'value added' services. These include a set of 'Information Spaces' that allow users to keep track of new information in their specialist areas, and collaborative tools enabling users to share information they find.

The development is coordinated by three key members of library staff, who deal with some aspects of the development (notably interface implementation) themselves, and who also have regular contact with library users visiting the physical library. Development of underlying technical infrastructure is devolved to specialists. The DL had undergone three major upgrades, in terms of features offered and user interface, in the previous six years. Development does not follow any standard methodology; the BT developers described their development process as very informal – that they tended to have an idea, implement a prototype and then discuss the design-as-instantiated with colleagues to get reactions.

The developers consulted users (i.e. BT employees) in various ways. Around the time of each new release, they would conduct user trials, focusing on finding flaws in the system; users in these trials were given detailed instructions on what to do and what features to test. The developers' knowledge of user difficulties and user requirements was derived primarily from users calling them or coming into the library with queries and requests. They also accessed system logs, to ascertain levels of use and which facilities or resources were being most extensively used.

2.1.1 Development of a novel feature: Jasper

The preliminary study at BT focused on the design of Jasper. This was a web site notification feature that enabled users to send interesting URLs to nominated colleagues, including notes about why they were interesting. It had been developed by another group within BT and tested by them as a stand-alone item, and the plan was now to incorporate it within the library. There was no data about user experience working with Jasper to inform scenario development.

The question was where to locate it within the library, and how to present it to users. In particular, the developers believed that the data entry form should be redesigned, and they wanted to focus on that. The first step was to create a scenario accounting for how a user would have reached this point. The scenario developed was very informal, but involved a user surfing the web, seeing a page that was interesting, and then entering the digital library (while keeping track of the interesting URL) to access the Jasper feature. This highlighted the fact that finding and using Jasper would be non-trivial, and that the user would have to be aware of this feature and how it functions before it could be used. Further analysis also highlighted the fact that the user was given no feedback when they had completed and sent the form to confirm their action. The latter was a solvable problem, so one of the BT developers immediately went to

the whiteboard to map out how he could create a preview / feedback page that addressed the identified difficulty.

Even this initial case study highlighted several important features of the way the developers integrated the scenario with their design reasoning:

- The scenario was developed around a pre-selected function and questions about that function.
- The scenario evolved in parallel with reasoning about usability insights from it.
- Solution generation was also interleaved with scenario development; in fact, the development team expressed the view that they did not want to find out about problems unless they could also identify solutions. Thus, they were highly solution-focused.

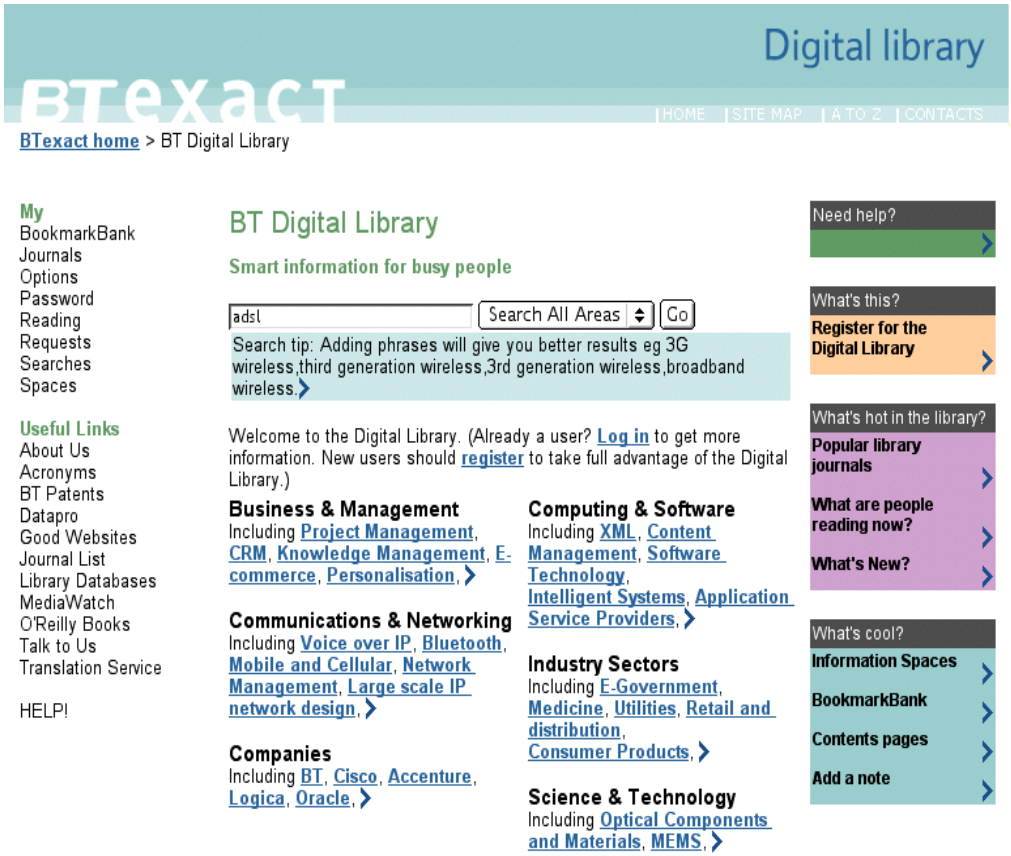

Figure 1: The BT Library interface

2.1.2 Development of an existing resource: Information spaces and other features

A subsequent, more structured, session at BT focused on existing features such as the Information Spaces (shown in the centre of the screen in Figure 1). An information space is a browsable collection of documents on a specified topic, to which new documents are regularly added. Each is specialised so that it will not grow unmanageably large.

Three scenarios, based around personas, were developed by the BT developers. The personas covered a less skilled user, one more demanding, and one in the middle; in creating them, the developers drew heavily on people they knew (exploiting the fact

that they have contact with their end users). They named their three personas after the Goodies (a UK TV show from the 1970s).

- Tim was an expert in his subject, and had done a preliminary search on credit card fraud. He had requested help on following links from a relevant journal.
- Graeme was an expert in his field but new to the DL. Working against a tight deadline, he had no time to waste. The scenario involved him conducting a basic search and then selecting appropriate resources, changing key terms, and using the keyword search feature.
- Bill was doing an MBA, and therefore using the resource for professional development, pushing the boundaries of the resources beyond the core business content. He had a preference for books, and therefore wanted to search the book catalogue, but was having difficulty using phrase search.

These scenarios, generated by the developers, share some notable properties:

- They all reflect real cases where users had come to the library seeking help;
- The formulated scenarios were (possibly as a direct consequence) very sketchy;
- They were all based on existing library features that were known to be difficult to use;
- All the personas lacked a good understanding of the search process, library features, the structure of the database, and the choice of resources; however, two of the personas represented people who were experts in their own fields.

Because the scenarios were based on real incidents, there were times when the scenario as discussed merged into memories of real incidents. For example, while discussing Graeme, one of the developers remembered:

> D1: What actually happened was that at this stage we showed him how to use 'browse by key words' to pick a better word.

Similarly, while discussing Tim, two of the developers tried to recall the user's experience:

> D1: If it was credit card fraud, he went into places he knew, though, because I think you said that it's a similar
>
> D2: That's a similar one, but he didn't find much: looked and searched
>
> D1: How did he go in, do you know? Ranking technology or
>
> D2: I really don't know. I think he probably just did a keyword search, really tie it down. I think he probably just did keyword searching
>
> D1: but a bad choice of keywords perhaps
>
> D2: Yeah because the terms, I don't think he was familiar with them

The developers expressed a strong preference, as exemplified in the descriptions of Tim, Graeme and Bill, for generating scenarios where the information seeking was less than instantly successful, because such scenarios would provide more useful insights into problems and solutions. Scenarios tested different parts of the library, including the keyword browser, the 'search for similar' feature and the Information Spaces.

Although the scenarios developed look impoverished, because they referred to incidents of which the developers were all aware, there was substantial common ground on details that were never explicitly articulated. Thus, the scenarios are richer than they appear, but the developers did not perceive any value in producing the enriched descriptions that are advocated in texts on scenario based design. The initial sketchy descriptions were gradually embellished through the dialogue, and (implicitly) through the shared knowledge of the developers.

The scenarios, described from a user perspective, were then tested against the system description to assess whether users with the specified skills sets (as defined within the personas) would know what to do and would understand the results they were getting back. For example, when analysing the Information Spaces, it became apparent that they are much more appropriately regarded as a monitoring feature for expert users than a good place for a novice user to start unless the user happens to be lucky and have a requirement that neatly matches an existing topic area.

In practice, the developers got distracted several times by trying to enact the scenarios as proposed and having difficulty understanding how the system returned results. For example:

> D1: I was trying to think of something that would turn up. (typing) I know we de-emphasised databases we thought were less interesting, I can't remember if it's a set order or if it's. Yep, we must have, I think we some time manually set this, perhaps when ABI was new and we wanted to make sure it was at the top, so I think we emphasised certain databases here and de-emphasised others.

This illustrates one difficulty with working with scenarios in a domain such as DL development: that the effects of user actions can be unpredictable to the developers, so that describing the scenario at an appropriate level of detail is a real challenge: very abstract scenarios can make assumptions (e.g. about the success of a search or the appropriateness of the resources used) but lack the detail needed to support design reasoning, while very detailed scenarios can become behaviour traces, which typically results in the detail obscuring the principles.

Quite soon, the developers extended the discussion into problem solutions:

> D1: Um, well if you don't find anything, there's a link to us for help.
>
> D2: We should make that more prominent for a start!
>
> D1: "I suggest you check your spelling and use the phone number for help".

In discussing solutions, they were considering the costs of making changes as well as the benefits:

> D1: We could easily do that.
>
> D2: I thought that would be quite difficult.
>
> D1: The difficulty would be if you went down to the issue level: then the search box would, you think you were just searching for that issue, because I couldn't do that, I could only have search making strategies. I couldn't have searched for January/ February 2003 issues.
>
> D2: Yes, you'd have to qualify it, wouldn't you? Interesting thought.

The developers worked readily with the scenarios, and switched seamlessly between scenarios, problem diagnosis and solution generation. Through their choice of scenarios, the developers focused on problems, and in the discussion their interest was

in problems that were in principle solvable, as the developers did not perceive any value in identifying problems that could not be fixed.

This session enabled the developers to get a richer sense of the difficulties less experienced users face and a richer understanding of the user experience than they previously had. Personas (based on real users) and scenarios (based on real incidents) were central to this richer understanding.

2.2 Work on Greenstone

To test the broader validity of the findings from the work with BT, a similar approach was taken with the developers of the Greenstone system. Greenstone is not a DL per se, but a "suite of software for building and distributing digital library collections"[1]. A digital library may consist of any number of collections; for example, Figure 2 illustrates the home page[2] of the New Zealand Digital Library, showing six separate collections; however, in much DL development, the term DL may be used to refer to either a collection or a set of collections.

Figure 2: six collections in the NZDL

Greenstone (Witten, Bainbridge & Boddie, 2001) is developed by the New Zealand Digital Library Project at the University of Waikato; at the time of the study, Greenstone 2 was in widespread use, including by international organisations such as UNESCO and the Human Info NGO. Greenstone 3 was under development. This was envisioned to be a substantial re-design that retains the essential features of the current version with an improved system architecture and some re-designed or new features.

There were two main phases of interaction between Greenstone developers and the human factors team. The first occurred when the lead developer visited the UK, on

[1] www.greenstone.org, accessed 14/12/5
[2] www.nzdl.org, accessed 14/12/5

sabbatical. At that time, he was developing a tool called the Gatherer (Bainbridge et al, 2003), which was the focus for one analysis phase.

The second occurred when the second author, referred to below as HF, visited Waikato for a fortnight to work on the development of Greenstone 3. At the time of the visit, the core development team for Greenstone comprised seven academic staff and research fellows. This team is supplemented by Masters students, who complete projects based around Greenstone, and a worldwide network of developers who contribute functions according to their particular specialisms.

2.2.1 Analysis of a new feature: The Gatherer

The Gatherer is a tool that supports a collection-builder in gathering together and organising documents to create a digital library collection. It was at a point of development where scenarios could be used for informing design and encouraging reflection within the design process. Data collection and analysis remained informal throughout this interaction.

Initially, the developer and human factors specialist worked through some of the screens and the developer told her about the pages he was having difficulty with. Over time, they constructed a representation of a person who was using the Gatherer to collect information together and what they were going to be doing. The developer's first description of what the user would be doing was in terms of procedures: they would look at this and they could add in a few documents and then they can go to this bit which is the metadata and then they can come to this page and redesign it, etc. Gradually, the developer and HF specialist constructed a more user-centred description of how the person would create a collection, including where they would get documents from, how they would collect them onto the site and what they would be expecting to do at each point in the interaction.

At this stage, the focus was on telling stories about use and the user experience. HF and the developer investigated ways of talking about what the effects of a design on a user would be. The approach eventually taken was to consider the user's goals, intentions and actions in the situation of working with the Gatherer. Together, they worked out what the user's actions were and what had triggered them, what the designer expected the user to do and what the user was able to do. By this route, they created an integrated account that related the developer's intentions and user's behaviour. In this phase, there was a substantial cultural gap between the HF and developer perspectives on design. Both HF specialist and developer had much to learn from each other, which involved substantial negotiation over shared understanding and values.

The process of analysing the design in this way was generating useful insights – for example, in identifying pages where the user would be unable to predict the effects of actions or choose a sensible action at all. The developer started with two conceptions of users, as being either the exploratory user who will click boxes and just happen to see what is there or the skilled user who already has all the system knowledge they need. Understanding the user as being goal directed proved invaluable for considering a richer spectrum of users. Through the process of generating scenarios, they created a user who knows what they want to do but does not know how to do it.

The Gatherer was intended to become a component of Greenstone 3, to which we now turn our attention.

2.2.2 Greenstone 3

In contrast to the BT developers, the Greenstone development team's contact with end-users of libraries built using Greenstone is mixed: a couple of team members have structured interactions with end users, but most only ever interact directly with collection builders, who therefore act as intermediaries between the development team and end-users. There was therefore relatively little group knowledge to inform scenario development.

In a two week visit to Waikato, the HF specialist engaged with the developers in a systematic (user-focused) review of Greenstone. The review took place in three stages, defined by the development team.

The first session was a functional review in which the development team listed out the different functions of Greenstone 2 on a whiteboard, noting whether they were part of the core development or prototypes that had not yet been integrated into the core system, and discussing whether or not they were perceived as being successful. If they were considered successful, they were to be retained in Greenstone 3. A function was considered successful if the software was reliable and functionally correct, and users were able to use it fruitfully (i.e. it was perceived as being both usable and useful). A function was regarded negatively if any team member came up with a reason why it was not working well; this might be that the code was unreliable, that users were known to have difficulty working with it, or that collection-builders had to write substantial extra code to turn it into exactly what they wanted.

Functions that were to be considered further were retained for subsequent group sessions.

In the second session, again, most of the development team participated. A set of eleven scenarios, including personas for different kinds of users, was constructed by the team members. The users included novices and others with more knowledge, and end-users as well as collection builders. At this stage, the scenarios were fairly general, avoiding too much detail about individual user actions since these would depend in part on the content of a particular library. From the set of eleven, three scenarios were selected for detailed analysis. These represented situations where the developers agreed it would take some work to fix known problems with the system, but also situations that would affect many users, so that there would be a high gain in correcting known difficulties. For example, one of these scenarios was as follows:

> Scenario 5 -- Historic manuscripts
>
> > Victoria and Albert are developing a new collection of historic manuscripts and books, digitised from original source material. They want to maintain a close fidelity to the pagination and other physical properties of the originals. Victoria and Albert are working in a small archive/library of interest both to expert professional researchers, family historians and those with an interest in 19th century literature - the latter including adults and, especially, children and students of literature. Victoria and Albert want to create a collection which will attract these users, some of whom already visit the physical library, both to the online and physical collections. Three typical users include Janet, a regular visitor (two or three visits a year), an English Literature academic who is particularly interested in the manuscripts of poets; Jeremy, who wants to find out if his ancestors are mentioned in the memoirs of a writer; Sally and Paul, two 13 year-olds who are working on a project for their English literature class on the use of industrial motifs in 19th century literature.

One of the developers almost immediately launched into design solutions:

DA: To me, if I'm doing this, I want plug-ins. You could say plug-ins are a good feature that helps and means the system is adaptable to changes in document format.

The discussion developed fluidly covering design solutions, requirements and scenario embellishment. For example, a bit later they further considered the likely IT skills of the personas and more about how they would use the system:

DA: V and A: do we think they have a lot of IT skills? They are working in a small archive.

Group response: No, not high IT.

DA: Do we think this is something they would have to attempt using the Gatherer only?

DB: Ideally the gatherer should have everything they want to do – that rather than command line: they ain't going to hack that.

DA: I guess the strong specialist feature is that the pagination in the digital library is preserved.

This last statement is an example of discussion that teeters on the borderline between being an element of a scenario (that Victoria and Albert are working with facsimile pages) and a design requirement.

Others returned to discussing particular system features (or potential features) several times through the discussion. Another soon gave an account of the source of inspiration for this particular scenario:

DB: I wrote this scenario – confession. I was thinking of something that looks like the Niupepa Collection[3], where you have got something that looks like the original thing, not just the raw text.

At one point, the team focused on Sally and Paul, considering their specific needs. This was a very tightly coupled discussion with four participants contributing ideas:

DB: Classifiers.

DC: Yes.

DB: Seems damn important because probably want to classify by who was mentioned, who wrote it.

DC: Date, location, particularly for the 13 year olds, probably aren't going to be.

DB: Classifiers are going to be pretty big, I think.

DA: Do we feel that classifiers are done in a good way? We could obviously reconstruct the Niupepa classifiers by date quite easily. However, if you wanted to do.

DC: Different things.

DB: We could improve on how we do that, we could do things nicer and make the collection nicer as a finished result. It would make their job a lot easier. A lot of what they are going to end up doing is after they have done the OCR, is the classifiers.

DD: Provide some automated help in checking the quality of the data that we are getting out of OCR process.

Similar (though less tightly coupled) exchanges occurred for the other two scenarios considered.

[3] A collection of Maori newspapers.

Overall, the design team collaboratively embellished the scenarios (though these embellishments were never written down) through the process of considering designs and requirements, and the three kinds of activity were interwoven in a reflective design interaction. The tightly coupled discussions ensured that common ground between participants was reinforced.

Personal experience was also invoked routinely to inform consideration of design possibilities and costs. For example:

> DA: The next goal is to brand the website – that is what the scenario is about.
>
> DC: Out of interest, so how hard was this?
>
> DA: This was very close to something. I did it - so it wasn't hard at all.

For each scenario, the team considered what functions users would make use of and which functions were causing problems, in order to focus on the most critical ones that needed attention. In practice, the developers were not particularly interested in what the consequences of a problem would be for the user; it was enough for them to know that the user was likely to have a problem. Once they were sure this was the case, they simply needed to know enough about the nature of that difficulty to be able to make design changes so that the user would be successful in future.

The third session moved beyond evaluation into the next stage of development. There was extensive discussion of the difficulties experienced by novice and inexperienced searchers in developing an iterative search (such users typically receive a results set that is not quite right for their needs but do not have the skills to refine the search). The meeting achieved consensus that this was indeed a problem that needed to be tackled. At that point, the team resorted to considering how other developers have tackled the same difficulty and whether those other solutions could be exploited in Greenstone. From that point on, the discussion moved away to new design solutions and the practicalities of implementing them. The discussion considered costs (the likely time to implement new solutions) and benefits (what user problems the new solutions would fix).

Overall, structuring the review around scenarios made it possible for the more technically oriented developers to gain a new perspective on how a DL would be used, particularly by relatively inexperienced users. The discussion also enabled the team to identify feasible design changes that would improve matters for users.

Greenstone 3 is a challenging object of study. It is not a system that is directly used by end-users. Some of the development team view it simply as a toolset that is made available so that others can create good collections, so that their responsibility is to make collection building as straightforward as possible. Other team members argue that there should be defaults built in to Greenstone to enable collection builders to quickly create collections that are likely to be easy to work with. This is a development context where responsibility is shared across remote design teams who may have minimal contact with each other, but where the decisions of one team constrain and inform what is possible for another; in this context, the generation and use of scenarios to guide design is difficult. Nevertheless, we believe that this "scenario informed design" approach was more effective in bringing usability into design than any other strategy we could have adopted.

3. Discussion

We structure the findings according to the two main themes that have emerged from the work: the use of scenarios and the cultural gulf between the development and human-factors teams.

3.1 The generation and use of scenarios

Within this project, scenarios were found to be more difficult to work with than anticipated. Had we been operating, as Rosson and Carroll (2002) advocate, within a scenario-based design context, the challenge of generating scenarios would probably have been smaller. In function-oriented design, for a complex and partly unpredictable system, defining scenarios at an appropriate level of detail proved difficult. As a craft skill, of course, there is no such thing as a 'correct' scenario: just one that is more or less useful.

For both sets of developers, in the early stage of the collaboration it proved difficult to establish common ground between their traditional technical orientation and our user-centred orientation. The early scenarios they generated were technology-focused: assuming a user with perfect knowledge, or an exploratory user without particular information goals, and considering what was possible or sensible with the existing system. Personas and a vocabulary of goals, actions and feedback were the two thinking-tools that supported us, collectively, in shifting towards user-centred scenarios. In particular, personas enabled them to view the DL from the perspective of externally-motivated but novice users.

The early design situations with each development team (Jasper and Gatherer) were novel ones: designs that were creating new interaction possibilities rather than upgrading existing ones. For these, the only option was to create projected usage scenarios. The developers' strong preference was to focus on scenarios that predicted user difficulties that could be fixed with obvious design changes. They were not interested in either scenarios that highlighted no user difficulties or ones that highlighted difficulties that they could not address.

The other two cases (the Information Spaces and Greenstone 3) involved iterative development so that there was an existing system to refer to. At BT, the experiences of real users were the main source for generating scenarios. The development team had difficulty divorcing the abstract scenario from the details of real user interactions and low level implementation details that typically have a substantial influence on the user experience (e.g. in determining how search results are ranked). There were no such empirical resources in relation to Greenstone, so that the team had to rely on anecdotal evidence of how users worked with Greenstone 2. At BT, the developers favoured scenarios of known problematic situations that could be used to directly inform re-design; the Greenstone team favoured relatively abstract scenarios – but again ones that gave insights into problems with the current system.

In this "scenario informed design", all the scenarios that were generated originated in the system features that were the current focus of concern, rather than the more abstract user considerations that might be the focus in scenario based design. These developers had no interest in completely transforming their design processes to make them user centred. The function-centred, scenario-informed design compromise appears to have achieved an acceptable balance between technology- and user-centred design. In addition, there was no text-book progression from problem statement to solution generation: from almost the beginning of each session, design possibilities

were being generated alongside the development and discussion of the scenarios. Where possible, these solutions drew on the developers' prior experience. Personal experience was clearly a highly valued resource for considering both problem and solutions.

3.2 Bridging gulfs between perspectives

Although the focus of this study was on contextualising and testing scenarios in DL development, key issues also emerged regarding the contrasting perspectives of the human factors team and the two development teams we worked with.

Both development teams shared a functional approach to thinking about design; Because of this functional perspective, both had difficulty perceiving how, from the user's perspective, all the functions needed to be joined up into a continuous interaction experience. Both teams started several discussions by presenting the human factors specialist with a particular issue – for example, where should this page be linked in to the rest of the system, or how should it be laid out? – without considering the broader context of how the user would have got to that point in the interaction, or why they were there. This was inconsistent with a scenario based approach.

There was a cultural gulf to be bridged: the human factors team had difficulty comprehending the interaction from a technical perspective, while the developer teams had equal difficulty in comprehending it from a user perspective. This made the early stages of working together challenging, as there were substantial communication difficulties. Over time, scenarios were used to create common ground but the process of building this ground was slow and difficult. This is consistent with the view put forward by Bødker and Christiansen (1997) that scenarios can serve as 'boundary objects' between members of a design team who come from very different backgrounds. It is worth noting, however, that the process of generating scenarios was reversed from the text-book account: we typically worked from functions that developers were concerned about to generate scenarios that involved users working with those functions.

As well as thinking about design in terms of the functionality a system supports, another important characteristic of the developer perspective was that it was solution-focused. Developers were, frankly, not interested in post hoc reflection on the qualities of a design; consequently, they only really engaged with scenario-based discussion when it could help identify solvable problems, and analysis was often interleaved with solution generation and discussion. They valued the insights from theory, but only when they could guide problem-fixing.

4. Conclusions

Scenarios explicitly set out the assumptions that designers make about the users and their tasks. Where possible, the developers chose to base scenarios on instances from their own prior experience. However, in other cases there was no a priori empirical evidence gathered from actual DL system users, and therefore developers had to rely on their intuitions and make design guesses. Clearly doing so is not ideal, but the problems in doing so are mitigated by making those assumptions explicit and testable. Scenarios would seem to be a good vehicle for doing this, but our work has exposed various practical difficulties in doing so.

In electing to work with scenarios, one initial assumption was that it should be possible to create a library of scenarios that could then be readily re-used by DL developers. This proved more challenging than anticipated for several reasons:

- The cultural gulf between the development teams and ourselves was substantial. They were function- and solution-focused, whereas we were scenario- and user-focused. Scenarios proved useful for creating common ground between these perspectives, but the gulf remains.
- DL development is even more ill-structured than most other software development, typically involving interoperating components developed by distributed teams who may have no direct contact with each other. At the least, this creates distributed responsibility for delivering a user experience that is an emergent product of a complex web of design decisions.
- Two of the functions we worked with were novel designs that created new interaction possibilities, so that there was neither relevant theory nor empirical data on which to base the design of scenarios.
- Developers readily invoked their own prior experience (of observing users, developing functions, experiencing other people's designs as users themselves), but showed much less inclination to draw on theory or the empirical findings of others.

Nevertheless, the use of scenarios helped deliver important design insights, and to bridge the gulf between the conflicting perspectives. In particular, the process of discussing the scenarios and who the users are and what they are trying to do generated a productive dialogue between the developers and human factors specialists. Thus, the power is in the dialogue and potential for creating shared understanding (a point also noted by Bødker et al (2000)).

Scenarios and personas enabled developers to explore the 'hidden middle' of user profiles: the users who are neither simply exploring nor ready-made experts. Cooper (1999), in his discussion of programming practice, asserts that programmers tend to think in terms of three numbers: 0, 1 and infinity (e.g. if something can be done more than once, it can typically be done an infinite number of times), and that this prevents them from thinking of good design solutions based around smallish numbers. There is a parallel here with thinking about users: good designs are often those that assume users have some knowledge (and purpose), but not infinite knowledge.

Our experience is necessarily coloured by the nature of the development environments in which we were working. DL development is unavoidably ill-structured. There is a web of interdependent developments. The whole process is much more organic and opportunistic than that assumed in descriptions of design processes, including scenario-based design (Rosson and Carroll, 2002). Design and analysis are interspersed fluidly, with each other and sometimes even with exploration of the current system. Both of the design teams we worked with had a strong culture of close team working, as illustrated in some of the transcripts above where individuals work in tightly coupled ways, often even contributing elements of the same sentence. The rich complexity of the development process makes it difficult to apply scenario-based design in anything like a textbook fashion. More critically still, the cultures of the development processes we have investigated are highly function-focused, and developers are interested in solutions, not problems. The sheer complexity of the technical challenges that underlie the development of well

engineered DLs (i.e. ones that work at a functional level, with components that interoperate correctly, with a minimum of bugs) demands a strong technical focus, which is culturally at odds with user-centred design processes. This technical orientation is essential to creating products that are well enough engineered to be useful or usable; the need is not to change the development culture to a user-centred one, but to value both cultures and find ways to enable them to work together. We have called this 'scenario informed design'.

Making usability usable demands that it is acceptable and valued. There are many different development contexts, and those we have been involved in are very different from those described by Carroll et al. There are good reasons for retaining a strong technology perspective in development and for focusing on the development of individual functions. Newell and Card (1985) propose that "The race is between the tortoise of cumulative science and the hare of intuitive design". In the work reported here, it would appear that the hare is still leading the race – that, in the design contexts studied, intuition and solutions continue to take precedence over science and understanding problems. What is needed is not for the hare and tortoise to continue to compete, but for them to find ways of co-existing and working together more effectively.

5. Acknowledgements

We are grateful to the development teams at BT and Waikato who worked with us on this project and created such fruitful interactions. David Bainbridge and George Buchanan gave useful feedback on a draft of this paper. The work was funded by EPSRC Grants GR/N37858 and GR/S67494.

6. References

Bainbridge, D., Thompson, J. and Witten, I.H., (2003) Assembling and Enriching Digital Library Collections. In *Proc Joint Conference on Digital Libraries.* 323-334, Houston, Texas.

Bates, M. (2002) The cascade of interactions in the digital library interface. *Information Processing and Management* 38:381-400. Available from www.gseis.ucla.edu/faculty/bates/articles/cascade.html

Bellotti, V. (1989) Implications of Current Design Practice for the Use of HCI Techniques. In D. Jones & R. Winder (Eds.) *People and Computers IV, Proceedings of HCI'89,* pp. 13-34. Cambridge University Press.

Bellotti, V., Buckingham Shum, S., Maclean, A. & Hammond, N. (1995). Multidisciplinary Modelling in HCI Design… in theory and in practice. *Proceedings of CHI '95 ACM Conference on Human Factors in Computing*. 146 – 153.

Blandford, A., Keith, S., Connell, I. & Edwards, H. (2004) Analytical usability evaluation for Digital Libraries: a case study. In *Proc. ACM/IEEE Joint Conference on Digital Libraries*. 27-36.

Blandford, A., Keith, S. & Fields, B. (in press) Claims Analysis 'in the wild': a case study on digital library development. To appear in *International Journal of Human–Computer Interaction.* 21.2. 197-218.

Blandford, A. & Rugg, G. (2002) A case study on integrating contextual information with usability evaluation. *International Journal of Human-Computer Studies*. 57.1, 75-99.

Bødker, S. & Christiansen, E. (2000). Creativity, cooperation and interactive design. *Proc. DIS 2000.* ACM. 252-261.

Bødker, S., Nielsen, C. & Petersen, M. G. (1997). Scenarios as springboards in design. In Bowker, G., Gasser, L., Star, S.L. & Turner, W. (eds.), *Social science research, technical systems and cooperative work*. Erlbaum, pp. 217-234.

Buckingham Shum, S. & Hammond, N (1994) Transferring HCI Modelling & Design Techniques to Practitioners: A Framework & Empirical Work. *Proceedings of HCI'94: People and Computers IX,* Glasgow, 23-26 August, 1994. Cambridge University Press: Cambridge, pp. 21-36.

Butterworth, R., Ghosh, M. & Wise, C. (2005). Designing for broadening access: model based iterative design of digital library resources for life-long learners. Presented at *Digital Resources for the Humanities 2005*. Extended abstract available from: http://ahds.ac.uk/drh2005/viewabstract.php?id=24&cf=1

Carroll, J. M. (2002) Making use is more than a matter of task analysis. *Interacting with Computers*. 14. 619-627.

Carroll, J. M. & Rosson, M. B. (1992) Getting around the task-artifact cycle: how to make claims and design by scenario. *ACM Transactions on Information Systems*, 10(2), 181-21.

Champeny, L., Borgman, C., Leazer, G., Gilliland-Swetland, A., Millwood, A., D'Avolio, L., Finley, J. & Smart, L. (2004) Developing a Digital Learning Environment: An Evaluation of Design and Implementation Processes. In *Proc. JCDL 2004*. 37-46.

Cooper, A. (1999) *The Inmates are Running the Asylum*. Indianapolis: Sams Publishing.

Hammond, N., Jorgensen, A., MacLean, A., Barnard, P. & Long, J. (1983) Design Practice and Interface Usability. *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*. 40 - 44.

Heinbokel, T., Sonnentag, S., Frese, M., Stolte, W. & Brodbeck, F. (1996) Don't underestimate the problems of user centredness in software development projects – there are many! *Behaviour and Information Technology*. 15.4. 226 - 236.

Hertzum, M., and Jacobsen, N.E. (2001) The Evaluator Effect: A Chilling Fact about Usability Evaluation Methods. *International Journal of Human-Computer Interaction*, 13(4), 421-443.

John, B. (1998) On our case study of claims analysis and other usability evaluation methods. *Behaviour and Information Technology* 17, 4. 244-246.

Kyrillidou, M. & Giersch, S. (2005) Developing the DigiQUAL protocol for digital library evaluation. In *Proc. 5th ACM/IEEE-CS joint conference on Digital libraries*, 172 - 173

Komlodi, A. & Marchionini, G. (1998) Key frame preview techniques for video browsing. In *Proc. ACM Digital Libraries 1998*. 118-125.

MacLean, A., Young, R. M., Bellotti, V. & Moran, T. (1991) Questions, Options, and Criteria: Elements of Design Space Analysis. *Human-Computer Interaction*, 6 (3 & 4), 201-250. Special Issue on Design Rationale, (Eds.) Carroll J. M. & Moran T. P.

Maiden, N. & Robertson, S. (2005) Developing use cases and scenarios in the requirements process. *Proc. ICSE 2005*, 561-570.

Newell, A. & Card, S. K. (1985). The prospects for psychological science in human-computer interaction. *Human Computer Interaction*, 1, 209-242.

Nielsen, J. (1994) Heuristic Evaluation. In J. Nielsen & R. Mack (Eds.), *Usability Inspection Methods* (pp. 25-62) New York: John Wiley.

O'Neill, E. (1998). *User–developer cooperation in software development: building common ground and usable systems*. PhD Thesis. Queen Mary and Westfield College University of London.

Paddison, C. & Englefield, P. (2003) Applying heuristics to perform a rigorous accessibility inspection in a commercial context. *Proc. CUU 2003*. 126-133.

Papadakis, I., Andreou, I. & Chrissikopoulos, V. (2002) Interactive Search Results. In M. Agosti & C. Thanos (Eds.) *Proc. ECDL 2002*. Springer LNCS 2458. 448-462.

Rosson, M. B. & Carroll, J. M. (2002) *Usability Engineering*. San Francisco: Morgan Kaufmann.

Rosson, M., Maass, S. & Kellogg, W. (1988). The Designer as User: Building requirements for the design tools from design practice. In *Communications of the ACM*, Vol. 31, No. 11 (page 1288 – 1298).

Spasser, M. (2003) The Flora of North America Project: Making the case [study] for Social Realist

Theory. In A. P. Bishop, N. A. Van House & B. P. Buttenfield (Eds.) *Digital Library Use*. Cambridge, MA: MIT Press.

Spencer, R. (2000) The Streamlined Cognitive Walkthrough Method, Working Around Social Constraints Encountered in a Software Development Company, *Proc. CHI'2000*. pp. 353-359.

Sutcliffe, A. G. & Carroll, J. M. (1999) Designing claims for reuse in interactive systems design *Int J Human-computer studies* 50 213-241

Vredenburg, K., Mao, J-Y., Smith, P. & Carey, T. (2002) A survey of user-centred design practice. *Proc. ACM CHI 2002*. 471-478.

Wharton, C., Rieman, J., Lewis, C. & Polson, P. (1994) The cognitive walkthrough method: A practitioner's guide. In J. Nielsen & R. Mack (Eds.), *Usability inspection methods* (pp. 105-140) New York: John Wiley.

Witten, I.H., Bainbridge, D., and Boddie, S.J. (2001) Greenstone: Open-source digital library software with end-user collection building. *Online Information Review*, 25 (5) 288-298.

Young, R. M. & Barnard, P. J. (1987) The use of scenarios in human computer interaction: turbocharging the tortoise of cumulative science. in Carroll, J. M. & Tanner, P. P., Eds. *Proc CHI+GI '87—Human factors in computing systems and Graphics interface,* 291-296 New York: ACM.