Claims Analysis 'in the wild': a case study on Digital Libraries

Ann Blandford, Suzette Keith,
Tamzyn Lodder & Bob Fields

Technical Report: IDC-TR-2004-011
March 2004

Interaction Design Centre

Interaction Design Centre
School of Computing Science
Middlesex University
Trent Park Campus
Bramley Road
N14 4YZ

For further details of this technical report series contact:
Paul Curzon (p.curzon@mdx.ac.uk)

Claims Analysis 'in the wild': a case study on Digital Libraries

Ann Blandford[1], Suzette Keith[2], Tamzyn Lodder[2] & Bob Fields[2]
[1] UCL Interaction Centre, University College London, Remax House, 31-32 Alfred Place, London WC1E 7DP
[2] Interaction Design Centre, Middlesex University, Trent Park, Bramley Road, London, N14 4YZ

Abstract

One of the most demanding tasks in HCI is transferring HCI capability into traditional design practice that is already facing enormous challenges in solving complex technical problems. We have worked with developers of digital libraries (DLs) to develop usability tools that support design practice. In particular, we have adapted the Claims Analysis technique to the DL context. This has posed many challenges, including having to work within a function-oriented design culture, finding ways to develop suitable scenarios that are 'pitched' at an appropriate level, and generating and evaluating claims. A scenario-based approach was found to be necessary to the effective generation of claims; this is entirely consistent with the published literature on claims, but is at odds with the way our collaborating developers thought about design. This posed substantial challenges in the way the claims approach could be communicated to the design team, or used within the ongoing design project. Claims Analysis was found to largely live up to the claims made for it by its own developers, but to be significantly more difficult to introduce into technology oriented design practice than anticipated.

Categories and subject descriptors: D.2.10: Design [*Methodologies*]; H.1.2: User/Machine Systems [*Human Factors*]; H.3.7: Digital Libraries; K.6.3: Software management [*Software Development*]

General terms: Human Factors; Design.

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

Introduction

The work reported here is an outcome of a 3-year project to develop tools and techniques to bring user concerns into the design process for digital libraries (DLs). In particular, we have applied Claims Analysis (CA: Carroll & Rosson, 1992; Carroll, 1996; Carroll, 2000; Rosson & Carroll, 2002; Carroll & Rosson, 2003) as an approach to accommodating user concerns within design. Here, we present an account of the trials, tribulations and achievements of including CA within the design process for three separate DL development projects. In particular, we reflect on the experiences of working with CA and with DL developers.

Digital libraries are large, structured repositories of electronic documents, in various media (text, sound, graphics, animation, etc.). They are widely perceived as being difficult to use – certainly when compared to finding information on the Web. However, their structure makes them, in principle, accessible to more powerful

searching techniques, and the fact that they are managed gives users assurance about the quality of the DL contents. In addition, many digital libraries include features that can enhance the user experience – for example, when compared to using a simple internet search or a traditional library. Examples of such enhancements include personalised document management systems and notification systems that allow the user to share a document discovery with other users; one such feature is discussed below.

The design practices of DL developers have not previously been studied in detail. This may be, in part, because a typical DL environment includes components from many different sources without any clear, structured development process. The scale of the problem is clearly outlined by Bates (2002), who describes the user's experience of interacting with a DL in terms of a 'cascade' of different interacting components. For example, the end user's experience of a DL is likely to be determined by the work of the developers of a web browser (through which the DL is accessed), the technical infrastructure (e.g. the networking and interoperability protocols), the document management software and the collection building system. Additional features may have been provided by yet other development teams. In this complex development environment, in which basic functionality and interoperability of components present substantial technical challenges, user concerns can become secondary: there is no point in focusing on user experience if the system is not well engineered, and consequently does not perform as intended or is unreliable. Two of the challenges within the work reported here were to understand the context within which design happens and to develop tools that DL designers could use to support their usability practices.

Before presenting the study and findings, we summarise the background under three headings: information seeking and usability of digital libraries; Claims Analysis; and, finally, the challenge of making usability usable.

Information seeking and usability of digital libraries: an overview

While the focus of this paper is not on DLs or information seeking *per se*, but on the experience of working with CA with developers, DLs provide an essential strand to the context of the work reported here. In particular, we briefly review work on information seeking and on usability of DLs.

Several models of the information seeking process exist (e.g. Kuhlthau, 1998; Ellis & Haugan, 1997; Bates, 1989; Wilson, 1999; Ingwersen, 1996; Vakkari, 2001; Sutcliffe & Ennis, 1998). These tend to differ mainly in focus and terminology rather than substance, so we briefly describe Kulhthau's (1998) Information Search Process (ISP) model as an example. This model identifies six stages of the information search process through which an information seeker moves on the path from uncertainty to a constructed understanding. Kuhlthau found that the information search process begins with vague thoughts and associated uncertainty (initiation), followed by identification of possible information sources (selection). The feelings of uncertainty soon dissipate as the information seeker assesses prospective topics. As the person explores general information on the chosen topic (exploration), feelings of uncertainty and confusion increase, and an inability to express precisely what information is required results in awkward communication between the user and the information system. The most critical part of the process is the point where the information seeker forms a focus for their task (formulation). Following formulation thoughts become clearer, and clarity

and confidence increase as the user gathers information (collection). Finally, a sense of relief is experienced as the search is completed (presentation).

This model is presented in a way that is independent of the context in which search takes place, but provides a good overview of the key stages of information seeking; research on user behaviour during detailed formulation of queries has been conducted by Blandford, Stelmaszewska and Bryan-Kinns (2001) and Stelmaszewska and Blandford (2002). These studies have shown that users experience difficulty in formulating effective queries and understanding search results with existing commercial DLs. As noted by Kuhlthau (1998), early on in a new search, users may not even know what they are looking for: problem formulation emerges through the dialogue between a partially specified topic (for example, an essay title) and a set of information sources; at this stage, the user may have difficulty assessing the relevance of documents to their information task. In principle, a browse-based interaction can be more effective than search at this stage, depending on the thematic organisation of the information sources. As the task becomes better understood, the bounds of the information problem become clearer; the issue becomes more one of understanding how to formulate a query for this particular search interface. For example, the user may need to understand Boolean logic or the use of different descriptor fields. When browse or search results are returned, the user has to be able to assess the relevance of each document rapidly and reliably. Each of these steps is demanding, and can pose such difficulties that users give up or leave with unsatisfactory results.

More generally, most work on usability of DLs has been *post hoc* – at least in the sense that a fully functioning prototype system exists. Much of this work has involved classical empirical studies of single libraries or features within libraries – generally conducted or commissioned by the systems' developers, to inform future developments (e.g. Papadakis *et al*, 2002; Komlodi & Marchionini, 1998). While this work has advanced the general understanding of usability issues for DLs, in terms of providing exemplar studies, the empirical approach does not lead directly to re-usable insights into DL design, or techniques that can be applied earlier in the design process, before a functioning prototype exists. This was one of the challenges for the project presented here, and Claims Analysis was the technique investigated most thoroughly as having the potential to deliver against this need.

Claims Analysis: an overview

Claims Analysis has been described in many sources by Carroll and co-workers. The most comprehensive account is provided by Carroll and Rosson (1992); a succinct summary is provided by Sutcliffe and Carroll (1999). Here we focus on the key points drawn from those and other sources.

Carroll and Rosson (1992) describe Claims Analysis as a form of 'psychological design rationale' – that is, a semi-structured approach to considering design from a user perspective. Claims are statements about the positive and negative effects of a design on the user within a particular context of use (a scenario). Thus, CA is embedded within scenario based design. It are also embedded within a particular view of design evolution which Carroll and Rosson term the 'task-artefact cycle'. Under this view, existing user tasks may be used to define requirements on new artefacts to support those tasks, but those new artefacts create new interaction possibilities that change users' tasks, so that we get an evolving interplay between design and use.

A scenario is a description of "the activities a user might engage in while pursuing a particular concern" (Carroll & Rosson, 1992, p.185). Different kinds of scenarios can be produced, focusing on existing or envisaged tasks and interactions, and 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).

Few of the descriptions of CA give explicit guidance on how to generate claims. The clearest guidance is provided by Carroll and Rosson (1992), who present a list of nineteen questions to ask as a basis for generating claims. These questions are organised according to the seven stages of Norman's (1986) action cycle, which describes user–system interaction from a user perspective as encompassing user goal formation, intention formation, action specification, action execution, perception of system state, interpretation of that state and evaluation of that state (with respect to user goals). For example, under the 'action execution' step, Carroll and Rosson propose three questions: how does the artefact make it easy or difficult to perform a task? What slips are most likely, or most costly? And how does the artefact indicate progress in task performance? Sutcliffe and Carroll (1999) propose something similar, but at a higher level of abstraction, suggesting (p.236) that an appropriate approach is "to apply an extended form of cognitive walkthrough to inductive claims analysis". Carroll (1996) extends the original nineteen questions with a further eighteen that help identify organisational claims, this time structured according to the organisational goals, and how an organisation (or team) engages in planning, execution, interpretation and evaluation of activities. Most other accounts of CA appear to assume that it is obvious to the reader how to generate claims, providing rich examples of the outcomes of CA, but negligible guidance on how to generate claims. As discussed below, learnability of CA proved to be a bigger challenge than initially anticipated – both for the research team and for communicating to DL developers.

Carroll, Koenemann-Belliveau, Rosson and Singley (1993) discuss the use of CA as an evaluative technique that supports an understanding of critical incidents (i.e. surprising incidents in an interaction – often ones where something has gone unexpectedly wrong). However, as Carroll (1998) make clear, he does not view CA as primarily an evaluation method. He states that claims "are intended to help developers think about what changes to specify by helping them become more aware of the tradeoffs among consequences of possible design changes for users and their activities" (Carroll, 1998, p.242).

As far as we can ascertain, few studies of CA have been conducted by anyone outside the original developer group. One of the few exceptions is work by John and Marks (1997), in which CA was one of six techniques adopted by a group of students to evaluate an interface to a novice programming environment. One aim of that work appears to have been to pit evaluation techniques against each other, deriving numerical scores for how many problems (categorised under various headings) each of the tested approaches identified, and how many of those problems were subsequently dealt with by the system developer. The authors conclude that "CA (circa 1991) did not lead to any design changes. We speculate that its 'claim … but' format, presenting both advantages and disadvantages of a design makes it difficult to

see clearly a better way to implement the system." (John & Marks, 1997, p.199). Their approach contrasts strongly with our own, where developers engaged in the process of generating claims, rather than being presented with them. However, the issue of the developers' perceptions of advantages and disadvantages is one we return to later, when discussing findings from this study.

Sutcliffe and co-workers have investigated reuse of claims. For example, Sutcliffe, Fickas, Sohlberg and Ehlhardt (2003) report on the use of claims to structure a contextualised interpretation of design principles to support reflection on multiple design alternatives in the context of designing assistive user interfaces for an electronic mail / communication program. They describe claims as a "set of trade-offs that have to be resolved" – in other words, as a tool to support reflection and re-design. Sutcliffe and Carroll (1999) focus more explicitly on structuring claims for re-use. Within the design projects we studied, we found limited scope for re-use; the assumptions about design processes that underpin a concern for reuse did not hold in this case study, for reasons discussed below.

On the basis of our reading, we believed that Claims Analysis should be able to address a broad range of user-oriented issues. For example, by drawing on theories of social structures such as Communities of Practice (Wenger, 1999), we could make theoretically grounded claims about how the deployment of digital libraries in an organisation may change organisational practice and relationships (Adams & Blandford, 2003). Drawing on theories of information seeking, as outlined above, we could make theoretically grounded claims about how well a particular library design supports common information seeking practices. Drawing on theories of cognition and interaction (such as Norman's (1986) action cycle), we could make claims about more detailed features of a system design. Below, we discuss to what extent we succeeded in this.

The challenge of 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. 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. 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. The second of these gulfs proved to be particularly difficult to bridge in the work reported here.

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), focusing on 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 conducting CA, but it has been clear that the additional demands placed on developers, who already had difficult technical problems to deal with, generated resistance, for reasons discussed below.

Even at the start of 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 points to the use of largely post-hoc, discount usability techniques such as heuristic evaluation (Nielsen, 1994) and lightweight empirical user testing. 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.

Carroll (2002, p.621) asserts that "it is impossible to find HCI methods today that do not incorporate scenario-based design". There are, however, 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 described below) 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.

In an earlier, smaller-scale study, Blandford, Buckingham Shum and Young (1998) studied the learnability of a theory-based user-centred evaluation approach, Programmable User Modelling (PUM: Young, Green and Simon, 1989). They found that with six hours of tuition, a group of HCI students could master the main points of the approach, but that at that point they had insufficient experience to be able to reliably produce user-centred descriptions that effectively supported design reasoning. Systematic studies of learnability and utility of techniques have also been conducted by John and others (e.g. John & Packer, 1995; John & Marks, 1997). These have shown that extensive exposure, to multiple descriptions of a technique and over weeks rather than hours or days, gives reasonable capability with a technique; however, as noted above, John generally assesses effectiveness in terms of number of usability problems identified by a technique – an approach that we did not wish to adopt in this work, as our focus was on how to learn and communicate the technique, and to understand its potential scope.

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

developmental rather than following a more traditional, investigative 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 was qualitative, based on interviews and observations.

A commitment was made to developing and contextualising Claims Analysis about nine months into the project. The main reasons for this choice were:

- 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. CA offered the promise of addressing these deeper issues.
- Rather than being purely evaluative, CA was explicitly designed to fit within ongoing design practice; thus, the analysts' reflection on design strengths and weaknesses could directly inform design changes.
- Techniques such as PUM (Blandford, Buckingham Shum & Young, 1998) were known to be unacceptably difficult and time-consuming to fit readily within design practice, unless the design team included a strong advocate for the approach.
- Compared to general design rationale (e.g. QOC: MacLean, Young, Bellotti & Moran, 1991), the psychological basis of CA provided more explicit support in viewing design from a user perspective.

Learning CA involved substantial reading and practice, and time spent encapsulating it in a form that supported communication to DL developers. A tutorial on CA for DLs was developed and delivered at two international DL conferences (Fields, Keith & Blandford, 2003). In parallel, we applied the approach with the BT developer/librarians, on various features of their library as it underwent development.

Once the method was reasonably well codified, we worked with a second development team – this time, developers of the Greenstone DL software, based at the University of Waikato. This study was unavoidably rather 'hit and run': 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.

In parallel, a final year student undertook a project comparing the findings of CA and Cognitive Walkthrough (CW: Wharton *et al*, 1994) for evaluating and re-designing a small DL maintained at the Museum of Domestic Architecture (MODA) in London.

While this study did not directly involve the original developers on the MODA DL, the student's experience of learning and applying CA provided a valuable perspective to complement the main studies.

We describe each of these studies in more detail here, before drawing out general findings and conclusions. However, first we should note refinements we made to the CA approach.

Codification of Claims Analysis

As far as possible, our description of CA followed that in the published literature as described above. Within the first year of the project, three refinements were made:

- To help DL developers "put themselves in the users' shoes", we embellished scenarios with personas, as described by Cooper (1999). Early on, we found that developers tended to unintentionally slip back into thinking that their intended users had the sophisticated understanding of DLs and search strategies that they themselves possessed. Personas helped them distance themselves and their expertise from that of their users.
- Extensive use was made of the information seeking literature, as summarised above, to provide background accounts of established information seeking behaviour. This could be used both for providing the 'bigger picture' for developing scenarios and personas, and for generating claims.
- We found the 'craft skill' approach to generating claims that was implicit in most accounts of scenario-based design and CA lacked the structure needed by our developers. Conversely, the 19 + 18 questions proposed by Carroll and Rosson (1992) and Carroll (1996) made the claims generation process too time-consuming and labour-intensive to be acceptable to our development teams. A 'middle way', in which we focused on three stages of Norman's action cycle, was adopted. In other words, we considered goal formation, action execution and evaluation of feedback for each stage of a scenario, and sought to generate both positive and negative claims for each of these phases of interaction. The net effect is similar to the approach suggested by Sutcliffe and Carroll (1999) of basing claims generation around the Cognitive Walkthrough questions.

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, such as INSPEC and ABI/INFORM databases and O'Reilly technical books. However, in contrast to many other organisations, 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 have the technical capability to deal with some aspects of the development (notably interface implementation), and who also have the usual kinds of contact with library users visiting the physical library, and with information suppliers. Development of underlying technical infrastructure is devolved to specialists. The DL has undergone three major upgrades, in terms of features offered and user interface, in the past six

years. Development does not follow any standard methodology; the BT librarians 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. In this paper, we refer to the library staff as developers when discussing their development role, and as librarians when referring to their librarianship skills; both terms refer to the same three individuals.

The developers consulted users in various ways. Around the time of each new release, they would conduct user trials, focusing on finding bugs and design flaws in the system; they tended to give users in these trials detailed instructions on what to do and what features to test. Their 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.

One key feature of the digital library – 'information spaces' – was the result of a marketing requirement: the initial information spaces were constructed to reflect the main market areas of the organisation, to enable marketing specialists to keep abreast of developments in their areas of specialism. 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.

In overview, the process of working with BT was as follows:

- Shortly after this project started, the developers introduced a new interface. We got involved with empirical evaluation of this interface. This consisted of user trials with six volunteers within the organisation, who were given guidance on performing set tasks with the new system. This allowed the developers to establish where users had difficulty finding or working with new features.
- Various usability evaluation techniques (including heuristic evaluation, cognitive walkthrough and CA) were applied to a local copy of the BT library, to establish the scope of these various techniques and establish requirements on evaluation techniques in terms of them giving useful leverage on DL usability difficulties. As discussed above, attention eventually fixed on CA.
- To better understand the nature of expertise in DL use, we conducted a detailed analysis of two of the BT librarians each acting as an information intermediary for three end-users. The six think-aloud protocols produced showed a sophisticated approach to formulating queries and interrogating results. This study complemented the literature on information seeking, representing one perspective on information use; in particular, it highlighted a sophisticated approach to information seeking with the BT DL in particular, and provided the basis for some searching scenarios.
- Having practiced CA on the whole of the BT library in the privacy of our own offices – an approach that is necessarily limited because it is divorced from the design process and the designers' reasoning – we engaged in a first CA exercise with the BT developers, focusing on the design of a new feature called Jasper, as described in more detail below.
- A further CA session was conducted some time later on the 'information spaces' feature of the library, as described below. The main aim of this session

was to test the feasibility of the method as we had codified it, rather than focusing on a particular feature of the library.

- A final debriefing session was held.

Now we focus in more detail on the two CA sessions, with Jasper and the information spaces.

Jasper

Jasper was a new feature that was being added to the library. It had been developed by another group within BT, and tested by them as a stand-alone item. Jasper was a web site notification feature that enabled the user to send an interesting URL to nominated colleagues, including a note about why the user thought it was interesting. Jasper also took a snapshot of the web site and had an automatic summarisation feature that it stored in case the website vanished.

The question now was *where* to locate it within the library, and how to present it to the user. In particular, the developers believed that the data entry form was poorly designed and should be redesigned, and they wanted to focus on that. However, it was not possible to say much of any significance about the form without creating a story (or scenario) about how a user would have reached this point. Creating such a story highlighted the fact that a user might just be surfing the web, see a page that was interesting, and then have to enter the digital library (while keeping track of the interesting URL) to access the Jasper feature. Thus, the user would have to be aware of the Jasper feature and of the way it functions (notably requiring the URL) before it could possibly 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. He was not entirely happy with this solution as it added a step to the process of working with Jasper, making it less efficient, but the process of generating claims had persuaded him that it was a sensible addition to the system.

The CA session with Jasper lasted about 2 hours. The session was led and structured by a human factors specialist (the second author on this paper), who worked with the developers to create stories ‘on the fly’.

At this stage, the development team had great difficulty in perceiving the design from a user’s perspective. Since it was always their intention to design systems well, including to be easy to use, in practice they found it much easier to identify the positive features of a design than the negative ones. Having said that: they described positive features with more hope than confidence – hope that they would be useful and usable, without either theory or empirical studies to back up that hope. Their normal reviewing practice was devoted to finding bugs or identifying where things did not work as expected; they expressed the view that they did not want to find out about problems unless they could also identify solutions to those difficulties. Thus, they were highly solution-focused.

Information spaces

The final CA session at BT was structured into three phases: presentation of a shortened version of the tutorial; development of personas and scenarios; and CA.

The CA considered the whole library in various degrees of detail, but focused particularly on the information spaces.

Three personas and scenarios 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, but also drew on theory to deliberately construct less skilled users because they were aware that it is the less skilled users that have most problems.

Similarly, the scenarios were based on their own experience of recent enquiries and problems that people had mentioned. They favoured producing problematic scenarios (ones where the information seeking was less than instantly successful) on the basis that such scenarios would provide more useful insights into likely problems and solutions. Scenarios tested several parts of the library, including the keyword browser and the information spaces. Scenario construction drew in particular on Bates' (1989) berry-picking model of information seeking, testing the 'search for similar' feature of the library.

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. Having worked with information spaces for several years, this was the point where both the developers and the human factors specialist realised explicitly why they did not provide good support for novices.

Overall, our impression is that this CA episode enabled the developers to get a richer sense of the problems the less experienced users face and a richer understanding of the user experience than they previously had.

In going through the scenarios, the team (developers and human factors specialist) tried to identify which features of the system were causing user difficulties and to relate those to the information seeking models. In part, they compared the three different personas, the different ways they would search, and the effects of the design on the users. However, the focus remained on problems that were in principle solvable, as the developers did not perceive any value in identifying problems that could not be fixed.

Work on Greenstone

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 1 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.

[1] www.greenstone.org, accessed 30/12/03
[2] www.nzdl.org, accessed 30/13/03

Figure 1: 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 writing, 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.

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 work on and contribute functions according to their particular specialities.

In overview, the process of working with the Greenstone core team was as follows:

- Remote analysis was conducted of one of the existing DL collections. This provided basic familiarity with the structures and features of a library collection built using Greenstone.
- One developer was interviewed when he visited the UK, focusing on the developers' perspective and culture, to establish what their requirements were and what they expected from a usability inspection method. He also outlined current and projected developments to provide a context for future studies.
- The lead developer, D, also visited the UK, on sabbatical. At that time, he was developing a tool called the Gatherer. The experience of conducting a CA with D on the Gatherer is described below.
- Finally, the second author, S, visited Waikato for a fortnight. While there, she interviewed most of the development team individually (3 of these were primarily user-oriented and 3 technology-focused); presented a short version

of the tutorial, tailored to the team's interests as established in the interviews; then conducted three team sessions on evaluation and design of Greenstone 3.

- Debriefing was conducted by email.

The first three of these stages preceded the final session with the BT developers. We now consider in more detail the analysis of the Gatherer and the three analysis sessions on Greenstone 3.

The Gatherer

The Gatherer was under development while D was on sabbatical in London. Thus, it was at a point of development where decisions were still being made and CA could be used in its intended context of informing design and encouraging reflection within the design process.

The Gatherer is a tool that supports a collection-builder in gathering together and organising documents to create a digital library collection. Thus, theory about information seeking models was irrelevant to the design, and could not be used for developing personas or scenarios.

Initially, the developer (D) and human factors specialist (S) 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 the 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, D and S constructed a more user-centred description of how the person would create a collection, including where they would get documents from and how they would collect them onto the site and what they would be expecting to do at each point in the interaction.

While working, S focused on establishing the developer's design intentions and what he thought the user's goals were. Thus, she was trying to split the user view from the developer's view. This required her to draw out from the developer how he had made and was making decisions, and how he intended the user to work with the system.

At this stage, the focus was on telling stories (i.e. creating scenarios), rather than CA *per se*. Together, S and D were trying to find a way of talking about what the effects of a design on a user would be – what that actually meant, and how the developer could comprehend what the effect of a design was on a user. 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. However, the argument still did not fall into the standard form for claims: rather than being able to say what the effect of the design would be on the user, it focused more on what the consequences of user actions would be on the system *and* on the user.

Thus, it continued to be difficult to create a statement of the form "my decision to do X has this effect on the user", with or without scenarios – i.e. to create claims of the

form described by Carroll and Rosson (1992) or Sutcliffe and Carroll (1999). Nevertheless, 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.

At this stage, in our view, it would not have been possible for the developer to work independently with CA; there was a substantial cultural gap between the HF and developer perspectives on design. Nevertheless, understanding the user as being goal directed proved invaluable, and helped the developer to move away from his pre-existing conception of 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 knowledge they need. Through the process or generating personas and scenarios, and trying to develop claims, they created a user who knows what they want to do but does not know how to do it. Scenarios and the action cycle provided a useful vocabulary for understanding what the user wanted to do.

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

Greenstone 3

The initial intention for the study at Waikato had been to take a particular library and analyse it using the method; however, the lead developer (D, now back in New Zealand) felt that was inappropriate, since current development effort was on Greenstone 3. As outlined above, this is not a DL, but a suite of software tools to enable a collection-builder to create a collection (or a DL, depending on your terminology). Consequently, once again, the information seeking models could not be used as a theoretical basis for generating personas and scenarios.

D had planned to review the strengths and weaknesses of Greenstone 2 and consider what needed to be changed before they mapped those features into Greenstone 3; S's presence provided an opportunity and a technique for structuring that review.

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 Waikato development team and end-users. There was therefore relatively little group knowledge to inform scenario development for CA.

The review took place in three stages, defined by D and the development team, rather than by S.

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, the intention was that they should be retained in Greenstone 3. A function was only 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). Although not expressed in terms of claims, the positive features of a function could be regarded as positive consequences.

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.

In this first review session, CA was not used explicitly, although functions were divided into either positives or negatives. Another consideration at this stage was that the function had some kind of impact on the user. So, for example, although DL protocols were mentioned, and they pose huge technical challenges, these were not considered in detail because their effect on the user is indirect.

In the second session, again most of the development team participated. A set of eleven scenarios, including personas for different kinds of users, were constructed collaboratively. 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 were all problematic scenarios, 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 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. As in the Gatherer study, it proved impossible to work out exactly what the effect on the user would be of particular design features. In practice, the developers were not particularly interested in what the effect on the user would be; it was enough for them to know that the user was likely to have a problem. Once they were sure the user was likely to have difficulty, 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 fact that something is causing the user a problem on goal formation or causing the user a problem in relation to some higher level information processing requirement is not perceived as contributing to their understanding of what it is they need to change at the software level.

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 a problem solving strategy that they often use, namely to consider 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 from CA: the focus shifted 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). In considering the effects on users, S invoked theory, while recognising that the only way to be sure would be to implement the new solution and test it with users.

Overall, structuring the review around claims, scenarios and personas made it possible to introduce principles of cognition and models of information seeking in a way that enabled 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 and claims to guide design is difficult. And yet, even now (reflecting on the project), we believe that this approach was more effective in bringing usability into design than any other strategy we could have adopted.

Student project at the Museum of Domestic Architecture (MODA)

The final study we consider is substantively different from the other two. It was an undergraduate project conducted by T, the third author on this paper. She set up her project with the goal of evaluating the effectiveness of Claims Analysis (CA) and Cognitive Walkthrough (CW) as approaches for evaluating and guiding the redesign of the MODA DL (Lodder, 2003). Thus, while it did not involve interaction between human factors specialists and other software developers, it gives an additional, complementary, view on the learnability and scope of CA.

The MODA DL[3] is effectively a catalogue of many items in the physical MODA. For most items, a picture and short description are available. Users can search the DL by any available criteria, including artist/maker, theme and date of production. There is no general browse facility, but the user can browse within search results.

Because T was testing the learnability of techniques, including the usability of extant descriptions of them, she worked entirely independently of the other authors of this paper, except for using short guidelines prepared by Keith. (Her project was supervised by another HCI specialist, but one who was unfamiliar with the details of CA.) Other resources she used for CA included Carroll (2000) and Rosson and Carroll (2002).

The project involved several phases: learning and applying CW and CA (in that order) to the design of the existing MODA DL; redesigning the system as far as possible in the light of findings; then reapplying the same evaluation techniques to the revised interface – thus starting to explore iteration and re-use as discussed by Sutcliffe and Carroll (1999). In contrast to the other two studies, this one did not explicitly involve the original DL development team; rather, T took the role of analyst and developer. Early on, she also noted all the problems she herself encountered while learning to use the DL – a kind of 'expert walkthrough'.

To develop scenarios, two sources of data were used. The first was an interview with the museum curator, who keeps a log of visitors and could provide information on what kinds of people typically visit the museum and what kinds of information they look for. The second, and more influential, was a series of observational studies of

[3] See http://monet.mdx.ac.uk/ , accessed 30/12/03

users searching for information in the physical museum, selecting searches of their own choice. These varied from the very specific (a particular object type from a particular maker in a particular year) to fairly general searches (e.g. on the 'Arts and Crafts' movement). Interviews with these users were recorded and transcribed, and detailed notes were taken on their interactions with the DL. Interviews covered issues such as what they were looking for and what they would be doing with the information.

Scenarios were based on this data; both positive and negative scenarios were generated (negative ones being ones where the user's search was unsuccessful, whether or not the information they wanted was in principle available in the DL). T found scenario generation difficult – particularly in terms of knowing what level of detail to include. In practice, during analysis, scenarios were made richer – to the point where they became very detailed accounts of interactions.

T made substantial design changes following the initial evaluations of the system, but found no further difficulties with the revised version. There are possible explanations for this:

- Any particular evaluation technique will only help identify certain classes of difficulty (those that lie within its scope). Since the same evaluation techniques were used on both iterations, the difficulties readily identified by those techniques were eliminated in the re-design. In particular, T had difficulty generating new scenarios that might highlight different design issues.
- More critically, the same individual was involved (alone) through the entire exercise, so that her personal insights had 'dried up' by the time of re-evaluation. As discussed further below, designers found it very difficult to identify negative claims regarding their own designs, and by this time, T was too 'close' to the design.

At a more detailed level: T created a set of scenarios to cover all the key features of the system. For each scenario, T walked through every step of the interaction, and for each step she tried to identify positive and negative claims pertaining to the user's goals and actions and the system feedback. Thus, the process became very systematic.

Reflecting on the process and outcome in a debriefing session with the first author, T felt that the approach "helped me find things I wouldn't have found otherwise". She also believed it would scale up to larger systems with more optionality, though this would be better done with a team of analysts, as the process became rather tedious over time. Overall, she was enthusiastic about CA, saying "I actually thought it was really good" even though "I found it hard to get my head around".

Reflections on the method

Clearly, the approach employed in this whole 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, though probably not in the general themes that emerged. Neither is it entirely inspectable: we cannot produce a full audit trail of the data generated in three years of activity, and have chosen to focus in this paper on a qualitative account of experience rather than a data-driven account of analysis. This is action research, in which insights have been achieved through the process of being engaged in learning and codifying a particular user-

oriented technique (CA) and working with developers, and reflecting on the experience. 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, the work has much higher ecological validity than most studies that have looked at learnability and applicability of user-oriented techniques in non-user-centred design practice. We now move on to consider the main findings from this study.

Findings

We structure the findings according to the two main themes that have emerged from the work: the experience of learning and applying CA, and the cultural gulf between the development and human-factors teams.

Learning and applying CA

As noted above, one of the real challenges for HCI researchers is developing design and evaluation techniques that are theoretically grounded, but that are also taken up and worked with by others. Put another way: to have any impact, HCI techniques need to be usable, useful and used – by people other than their developers. So one of the first questions about any technique is: what are the resources available for learning it, and which aspects are easy or difficult to learn (and why)?

For CA, the best learning resource we found was Carroll and Rosson (1992); Sutcliffe and Carroll (1999) provided a useful alternative perspective, although the focus of that paper is more on re-use than initial generation of claims. Most of the other publications that discuss CA at all do so peripherally, leaving the reader with the impression that generating claims is so easy that it does not even need to be explained explicitly. Carroll (2002) contrasts scenario-based design with task based design; scenarios effectively provide examples of how the system is intended to be used, whereas tasks prescribe the what and how of implementation. Some people can think entirely in terms of examples or of procedures, but most learn best with a mix of both. The presentation of examples divorced from the cultural context in which they were generated can make it difficult to generalise from them. We found the shortage of procedural descriptions of how to apply CA a disadvantage, and therefore generated our own procedural description, codified in the tutorial (Fields, Keith & Blandford 2003).

T's experience is telling in this context. She had participated in three 30-hour modules on HCI prior to her project, but then had to learn and apply CA within a relatively short time (a matter of weeks). She found learning CA difficult; in particular, she found the original CA material, based on examples such as learning Smalltalk and developing a community network system, hard to transfer to the digital libraries context. For this, she needed the support of the project material. At the end of her project, she felt that she would have struggled with CA had she not had this contextualised resource. She also felt that her prior HCI experience was essential to her being able to apply CA effectively; this is consistent with the view of Carroll and Rosson, that CA is highly dependent on craft skill.

Applying CA presented two particular challenges: developing good scenarios, and identifying the effects on the user (i.e. generating claims). We consider each of these in turn.

Scenarios and personas

Had we been operating, as Rosson and Carroll (2002) advocate, within a scenario-based design context, the challenge of generating scenarios might have been smaller, but in function-oriented design, 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.

Sutcliffe and Carroll (1999) describe three different kinds of scenario that can be used with CA: *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). Even within the development contexts where much of the design activity could be characterised as conforming to the task–artefact cycle of iterative design (Information Spaces and Greenstone 3), this distinction proved too subtle to be useful. Empirical studies were used at BT (of both novice and expert searchers), together with theory about information seeking, to generate scenarios there. We did not have access to such empirical resources in relation to Greenstone, and had to rely on anecdotal evidence of how users worked with Greenstone 2; we also had less recourse to theory, since there is no established theory on how people create digital library collections. At BT, the developers favoured 'problematic' scenarios of known 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.

The other two design situations (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. Again, 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.

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. The three–stage version of Norman's action cycle and personas were the two thinking-tools that supported us, collectively, in shifting towards user-centred scenarios.

In the DL context, there are several factors that greatly influence the searching experience, relating to both the user – e.g. familiarity with the topic of the search, general searching skills or knowledge of the particular library – and to the search problem – e.g. more or less well defined, well or poorly supported by the DL being used. To capture key differences, we found personas invaluable. They helped to frame thinking about different kinds of user experience (different scenarios) when working with the library features under consideration. This is an extension to the perspective

presented by Carroll and co-workers, probably because of the differences between the kinds of examples they work with and the DLs that we were studying.

In addition, as noted above, DL developers typically thought of two kinds of users: the exploratory user whose entire focus was on exploring the DL and its features, with no external task driving their interaction, and the knowledgeable user, who had a deep understanding of the DL, its features and the kinds of material available in it. Personas enabled them to view the DL additionally from the perspective of externally-motivated but novice users.

The human factors specialist (S) who was involved in all the studies with developers had a strong background in scenario-based design (acquired prior to this project). Therefore, generating and working with scenarios was not difficult for her – the difficulty was in communicating that understanding to developers who came from a different development culture. In contrast, T was a novice user of scenarios, and she found it very difficult to create scenarios that captured insights at an appropriate level of detail. She refined her scenarios several times, generally adding detail from her empirical studies. By the end, other authors of this paper found T's scenarios surprisingly detailed, highlighting differing views on what constitutes a 'good' scenario and the difficulty of communicating and reaching consensus on this craft skill.

Generating and evaluating claims

Carroll and Rosson (1992, p.193) describe claims analysis as an "analysis of tradeoffs": that an artefact feature or technique causes desirable psychological consequences but may also cause undesirable psychological consequences, noting also that the consequences (or effects on the user) occur within the context of a particular use scenario. Sutcliffe and Carroll (1999) expand this description of a claim to encapsulate thirteen aspects for every claim to support re-use. For pragmatic reasons, we used the simpler description. Even so, it was often difficult to work out what the positive and negative consequences on a user would be.

Sutcliffe and Carroll (1999) include examples of claims that draw on learning theory to predict effects on users of different kinds of learning environments; thus, the effects are clearly learning effects on the user. In searching, the primary effect appeared to be that users succeeded (with concomitant sense of satisfaction) or failed (with concomitant frustration). This is overly simplified, of course, but one key difficulty of current DLs is that they provide minimal support for users in *learning* about effective search strategies or even the structure of a particular DL.

If, as generally happens in search-based interfaces, the user is simply presented with a search box, what is the effect on the user? It clearly depends heavily on who that user is – if we assume that the user recognises a search box for what it is, there are still questions about how the user will formulate a query, so much non-determinism in what they might do, and how they will perceive and interpret the results they get back. Even with the help of information seeking theory and basic cognitive theory, we found ourselves struggling over this kind of scenario. Maybe we got the level of description of the scenario wrong: we could answer this kind of question if our scenarios were as detailed as T's; however, if we tried to generalise over cases, it was difficult to be confident about effects.

When it came to working with developers, they found it much easier to identify positive effects of their own designs (though in the tutorial examples, they were better at identifying the negative effects of the exemplar designs!). Developers are highly motivated to deliver usable, effective design solutions, and are much more aware of their positive motivations than of any difficulties users might experience. The personas, scenarios and three-stage process of generating claims about goals, actions and feedback helped to provide theoretical context for identifying claims; however, developers found it hard to identify weaknesses of their favoured solutions, when they have already struggled to work out the technical feasibility of those solutions. This view was echoed by T who, following her experience of re-design, felt that the roles of evaluator and designer were incompatible, and that CA should have been separated from design activity – a perspective that is at odds with that espoused by Rosson and Carroll (2002).

In practice, it proved to be much easier to think in terms of causes of user difficulties. Both teams of developers had enough contact with users, as well as access to system logs, to be able to identify places within their existing designs where users often had difficulties or did inappropriate things or failed to use particular features that could help them. Drawing on psychological or information seeking theory, it was possible to argue that a user could not perform an action because they had not formed an appropriate goal or they lacked a knowledge of the domain or they lacked information retrieval skills, or whatever. But that was an explanation of why the user was having a problem; it was not saying what the effect of the design was on the user.

It was possible to extend that to make basic claims of an effect that this particular part of the interface does not convey to the user what to do. This may be considered a negative effect, although it is perhaps more accurately an absence of effect: the user receives neither positive guidance nor misinformation about what they might do next. A lot of the problems that the users experienced were situations where the tools were available to them, but they did not know how to use them, so the tool was not affecting the user: the tool was not providing the support the user needed because it never occurred to the user to use it.

Thus, ultimately, the attention of the project shifted somewhat from generating claims in terms of effects on users to seeking cause and effect within the scenario – e.g. that a user might have an inappropriate goal, rather than an outcome being a direct result of the system design.

Understanding the developers' perspective

While investigating CA in naturalistic design contexts (focusing on DLs) was the main focus of this study, key issues also emerged regarding the contrasting perspectives of the human factors team and the two developer teams we worked with.

Both development teams shared a functional approach to thinking about design: both had difficulty perceiving how, from the user's perspective, all these functions need 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. The human factors specialist could not work in this way – could not answer the developers' questions as posed, without probing the context and (effectively) creating scenarios.

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 is hardly a new insight, but it made the early stages of working together challenging – rather like two worlds colliding and trying to get along. Over time, scenarios, personas and claims were used to create common ground but the process of building this ground was slow and difficult. And the process of generating scenarios was reversed from the text-book account, in that 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 CA 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.

Discussion and Conclusions

In electing to work with CA, our initial assumption was that it should be possible to create a library of scenarios and claims that could then be readily re-used by DL developers. As has become obvious through the preceding account, this assumption was flawed. We have identified several reasons for this:

- The cultural gulf between the development teams we worked with and ourselves was substantial. They were function- and solution-focused, whereas we were scenario- and user-focused. Scenarios and claims 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.
- Half 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 claims.
- For reasons outlined above, CA proved more difficult to learn and apply than expected.

Nevertheless, the use of personas, scenarios and claims helped deliver important design insights, and to bridge the gulf between the conflicting perspectives. Using claims as a communication tool proved to be more effective than just writing them down: the process of discussing the scenarios and who the users are and what they are trying to do generated the most productive dialogue between the developers and human factors specialists. Thus, the power is in the dialogue and potential for creating shared understanding: although we have codified CA for the DL context, we have not established that the collaborating development teams could take this codification and work with it without human factors input. While others (e.g. Iverson, 2003) have

reported taking up CA in practice, the BT development team have explicitly stated that scenarios are very useful, but generating claims is "too academic".

At the end of this study, we are in a position to reflect back on CA and the intentions of *its* developers. The most explicit claims about CA are to be found in the work of Sutcliffe and Carroll (1999, pp.214-215). Here we consider those claims and our findings in relation to them.

"**Claims can cover a wider span of design problems**" [than cognitive modelling approaches to evaluation]. We found this to be true. In particular, we could invoke empirical findings and information seeking theory to address broader concerns. As well as a wider span, we also felt that CA enabled us to probe issues in more depth – certainly when compared to CW or Heuristic Evaluation. Although we have not explicitly applied them, we believe that other cognitive modelling techniques such as GOMS (Card, Moran & Newell, 1983) and PUM (Young Green & Simon, 1989) would also only address comparatively superficial aspects of users' interactions with DLs because they demand an explicit statement of user knowledge in a form that is 'missing the point' for DL users.

Claims can "**deliver HCI knowledge to designers**". Again, this is a reasonable claim. However, as discussed above, this was more difficult than anticipated, and we are not sure that the developers we worked with could or would use claims unsupported. Nevertheless, one of the Greenstone team noted in feedback that: "claims analysis stretched our ideas for future improvements to Greenstone […and…] will prove a beneficial tool in future."

Although CA is less formal than predictive models, claims "**provide knowledge that is more tractable**". In the context of DLs, this is certainly true. One of the Greenstone development team expressed this more strongly: "we simply don't know enough about the world in which we're trying to operate to use more formal approaches". Conversely, however, we found that CA, as presented in the literature, provides less support to the analyst than more formal approaches generally do. The counter-argument might be that Carroll and Rosson (1992) and Carroll (1996) provide, between them, 37 questions to structure the interaction; in the situation in which we were working, this was *too* much structure.

Claims "**associate theoretically motivated and empirically justified knowledge of human computer interaction with a designed artefact and context of use**". This is true where relevant theory exists. At times, we were working beyond this boundary, and therefore claims were much more speculative. Of course, applying theory presupposes that an analyst is sufficiently familiar with all relevant theory, which may not be the case in all development contexts.

"**A given claims analysis can be reused and generalised within the scope of a particular design project**". While this may be true in theory, and it was one of our motivations in applying claims, we did not find this to be true in the contexts in which we were working. Developers were most interested in insights on aspects of design about which they felt less sure – which were typically aspects where the design was more speculative. The developers were not developing whole library systems *ab initio*; they were mainly developing novel features, for which no library of scenarios or personas or claims would ever be adequate. Every design is exploratory, user task possibilities being defined less from pre-determined requirements than from a recognition of possibilities – what Rzevski (1990) calls design revolution rather than

evolution. In more structured evolutionary design (which conforms more closely to the 'task-artefact cycle'), it is possible that reuse might work, but we do not have evidence to support this claim.

"**Claims have a domain-specific anchor in the artefact context**". This is true of the way we applied claims, and is one of the strengths of claims as compared to purely cognitive or system-centred approaches.

In summary, our experience is necessarily coloured by the nature of the development environments in which we were working. DL development is unavoidably ill-structured. There are many layers (even that term makes it sound more structured than it is in reality) of interdependent development. 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). This complexity 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: developers are interested in solutions to design problems – they don't want problems, they want solutions. 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 function correctly together, 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.

Did CA 'work' in this context? Certainly not in the smooth way projected in publications by Carroll, Rosson, Sutcliffe and their co-workers. It proved to be difficult to learn, communicate effectively to developers, and apply within their ongoing design processes. Nevertheless, the cultural shift in being forced to consider personas and scenarios as a precursor to generating claims provided a strong foundation for creating common ground between divergent perspectives on design. And to that extent, CA was, indeed, effective, even if the resulting claims would probably be regarded as somewhat haphazard and ill-structured by the developers of CA.

Acknowledgements

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

References

ADAMS, A. & BLANDFORD, A (2003) Social empowerment and exclusion: A case study on Digital Libraries. Submitted for journal publication. Draft available from www.uclic.ucl.ac.uk/annb/DLUsability/ClinicalDL.html

BATES, M. (1989) The Design of Browsing and Berrypicking techniques for the Online Search Interface. *Online Review*, 13,5. 407-424.

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/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.

BLANDFORD, A. E., BUCKINGHAM SHUM, S. & YOUNG, R. M. (1998) Training software engineers in a novel usability evaluation technique. *International Journal of Human-Computer Studies*, 45(3), pp. 245-279.

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.

BLANDFORD, A. E., STELMASZEWSKA, H. & BRYAN-KINNS, N. (2001) Use of multiple digital libraries: a case study. In *Proc. JCDL 2001*. ACM Press. pp 179–188.

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.

CARD, S. K., MORAN, T. P. & NEWELL, A. (1983) *The Psychology of Human Computer Interaction*, Hillsdale NJ: Lawrence Erlbaum.

CARROLL, J. M. (1996) Becoming social: expanding scenario-based approaches in HCI. *Behaviour and Information Technology*. 15.4. 266-275.

CARROLL, J. M. (1998) On an experimental evaluation of claims analysis. *Behaviour and Information Technology*, 17(4), 242-243.

CARROLL, J. M. (2000) Making use: scenario based design of human computer interaction. MIT Press

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

CARROLL, J. M., KEONEMANN-BELLIVEAU, J., ROSSON, M . B. & SINGLEY, M. K. (1993) Critical Incidents and Critical Themes in Empirical Usability Evaluation. In J. ALTY, D. DIAPER AND S. GUEST, Eds. *People and Computers VIII, Proceedings of HCI'93,* 279 - 292. Cambridge: Cambridge University Press.

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.

CARROLL, J. M. & ROSSON, M. B. (2003) Design Rationale as Theory. In J. M. Carroll (Ed.) *HCI Models, Theories and Frameworks: Towards and Multidisciplinary Science*. San Francisco: Morgan Kaufmann. pp. 431 - 461.

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

ELLIS, D & HAUGAN, M. (1997) Modelling the information seeking patterns of engineers and research scientists in an industrial environment. *Journal of Documentation* Vol. 53, No. 4. pp. 384-403

FIELDS, B, KEITH, S. & BLANDFORD, A. (2003) *Usability Evaluation of Digital Libraries: A Tutorial*. Middlesex University Interaction Design Centre Technical Report: IDC-TR-2003-001. Presented at JCDL'03 and ECDL'03. Available from http://www.cs.mdx.ac.uk/research/idc/techreports.htm.

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.

INGWERSEN, P. (1996) Cognitive perspectives of information retrieval interaction: elements of a cognitive IR theory. *Journal of Documentation* 52 (1) 3-50

KUHLTHAU, C.(1988) Longitudinal case studies of the information search process of users in libraries. *Library and information science research* 10 (3) pp. 257-304

IVERSON, E. (2003) The Starting Point Project: Moving Beyond Usability and Web Logs to Refine Evaluation. Position statement for NSDL workshop. Available from http://eduimpact.comm.nsdl.org/evalworkshop/_iverson.php (viewed 29.1.04).

JOHN, B. & MARKS, S. (1997) Tracking the effectiveness of usability evaluation methods. *Behaviour and Information Technology* 16, No. 4/5, 188-202.

JOHN, B. E. & PACKER, H. (1995) Learning and using the Cognitive Walkthrough method: A case study approach. In *Proceedings of CHI'95.* pp.429-436. ACM Press: New York.

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

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

LODDER, T. (2003) *To evaluate the information retrieval interface of the Museum of Domestic Architecture (MoDA) Digital Library using the Digital Library usability evaluation techniques developed by Keith (2003), and evaluation of the usability evaluation techniques used*. CMT3992 Undergraduate Computing Project, Middlesex University.

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.

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

NORMAN, D. A. (1986) Cognitive Engineering. In D. A. Norman and S. W. Draper, Eds. *User Centered System Design,* Hillsdale NJ: Lawrence Erlbaum, 31-62

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.

RZEVSKI, G. (1990) Engineering Design Methodologies and Artificial Intelligence. *Proceedings of International Conference on Engineering Design, Dubrovnik,* ICED.

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

STELMASZEWSKA, H. & BLANDFORD, A. (2002) Patterns of interactions: user behaviour in response to search results. In A. Blandford & G. Buchanan (Eds.) *Proc. Workshop on Usability of Digital Libraries* at JCDL'02. 29-32 Available from http://www.uclic.ucl.ac.uk/annb/DLUsability/JCDL02.html

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

SUTCLIFFE, A. & ENNIS, M. (1998) Towards a cognitive theory of information retrieval. *Interacting with computers* 10. 321-351

SUTCLIFFE, A., FICKAS, S., SOHLBERG, M. M. & EHLHARDT, L. (2003) Investigating the usability of assistive user interfaces. *Interacting with computers* 15. 577-602

VAKKARI, P. (2001) A theory of the task-based information retrieval process: a summary and generalisation of a longitudinal study. *Journal of Documentation.* 57(1) 46-60.

WENGER, E. (1999) *Communities of practice: Learning, meaning and identity*. Cambridge: Cambridge University Press.

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.

WILSON, T. (1999) Models of information behaviour research. Journal of documentation, Vol. 55, No. 3, pp. 249-270.

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., GREEN, T. R. G. & SIMON, T. (1989) Programmable User Models for Predictive Evaluation of Interface Designs, in K. Bice. & C. Lewis (eds.), *Wings for the Mind: CHI '89 Conference Proceedings,* pp.15-19.