Int J Digit Libr (2004) 4: 108–123 / Digital Object Identifier (DOI) 10.1007/s00799-003-0074-4

# Usability inspection of digital libraries: a case study

**H. Rex Hartson, Priya Shivakumar, Manuel A. Pérez-Quiñones**

Department of Computer Science – 0106, Virginia Tech, Blacksburg, VA 24061, USA;
E-mail: hartson@vt.edu, pshivaku@vt.edu, perez@cs.vt.edu

Published online: 23 July 2004 – 

Those that know the wheel of seasons
Notice nothing but its turning.
Time, they say, is love's dimension;
Love, the fruit of trial and tears.
*The Fruits of Time*, Robert K. France

**Abstract.** This paper reports a case study about lessons learned and usability issues encountered in a usability inspection of a digital library system called the Networked Computer Science Technical Reference Library (NCSTRL). Using a co-discovery technique with a team of three expert usability inspectors (the authors), we performed a usability inspection driven by a broad set of anticipated user tasks. We found many good design features in NCSTRL, but the primary result of a usability inspection is a list of usability problems as candidates for fixing. The resulting problems are organized by usability problem type and by system functionality, with emphasis on the details of problems specific to digital library functions. The resulting usability problem list was used to illustrate a cost/importance analysis technique that trades off importance to fix against cost to fix. The problems are sorted by the ratio of importance to cost, producing a priority ranking for resolution.

**Keywords:** Usability – Inspection – Digital library – Usability problems – Cost-importance analysis

## 1 Introduction and background

As part of our usability engineering role in the digital library project entitled *Collaborative Project: Core Integration for the National SMETE Digital Library* at Virginia Tech, we conducted a usability inspection of a digital library system called the Networked Computer Science Technical Reference Library (NCSTRL), to be found at: http://www.ncstrl.org/. As our charge was to perform a formative usability inspection, this paper reports a case study about lessons learned and usability issues encountered, and not a formal lab-based usability test or a summative study involving user performance measures and statistics. In the course of our work we have also performed a usability inspection of iLumina, found at http://www.ilumina-project.org/, a digital library of shareable undergraduate teaching materials for science, mathematics, technology, and engineering where the user can contribute, locate, and download resources. We also have looked at the usability characteristics of several other digital library systems. We have observed that many of the usability issues discovered in our inspection of NCSTRL are generally applicable to all the digital library systems that we have explored. Thus, while our report is worded specifically in terms of NCSTRL, many of the concepts apply to other digital library systems as well.

An important category of common problems that we observed involves an apparently functionally oriented design approach rather than an approach based on user task threads. Another category involves the way that most of these digital library systems handle the submission of documents and the corresponding concept of metadata (descriptive data formatted for searching). Finally, many of these systems share problems with the use of words (e.g. in labels), including a tendency to use jargon and designer-centered, rather than user-centered, terminology.

### *1.1 National SMETE Digital Library (NSDL)*

The National Science Foundation is developing the National SMETE (science, mathematics, engineering, and technology) Digital Library (NSDL, http://www.nsdl.org/).

The NSDL has a goal to create a community of users and information providers to enhance education at every level, including K-12, undergraduate, graduate, and lifelong learning. The NSDL will be a repository of information and a central place for collaboration in improving education, primarily in the USA, but indirectly around the world. It will offer portals and collections to support various user communities (K-12, undergraduates, professors, computer science, different branches of engineering, sciences, etc.).

### *1.2 Our role in NSDL evaluation activities*

Our area of work is human–computer interaction (HCI), especially usability engineering. In particular, we concentrate on formative evaluation in the usability engineering process [16].

All existing usability development methodologies agree that, to produce a product with high usability, a complete usability engineering process must be an integral part of the overall software-development process from the very beginning of the project. Typically, a usability engineering process involves client and user interviews, task analysis, user class definitions, usage scenarios, iterative usability design, prototyping, design walkthroughs, and usability evaluation. Unfortunately, many software developers think of usability engineering as only usability testing, to be done at the end of a project. Additionally, the reality of software-development cycles is such that time constraints, ill-specified requirements, and lack of usability-trained professionals often conspire to make usability issues an after-the-fact consideration. Thus, usability specialists are often called in at or near the end of the project to 'do some usability testing'. They are asked to help fix a user interface for which they had not participated in the design. They typically will identify problems that, had they been part of the development team, could have been addressed much earlier and much more economically.

The previous paragraph sums up our experience as practitioners representing usability in this project. We were not part of the design team that built NCSTRL, nor were we privy to the design discussions and design rationale that led to what we saw in our inspection. Although it would have been better to maintain close communication with the designers and developers while doing the inspection, this was not a practical option in our case. We were asked to conduct a usability evaluation after the system was deployed, and funding for further NCSTRL user interface work had expired. At the time of this inspection we hoped for additional funding, but it did not materialize. Currently, there is still some development being done on NCSTRL back-end functionality and we contacted a former developer for cost information for this paper but, in the main, we did our inspection more or less independently, without much interaction with the designers or developers.

These circumstances led to some disadvantages compared to the ideal situation but they, unfortunately, are not atypical:

- We, the inspectors, are usability people and not digital library people and that means that we are not intimately familiar with digital library conventions, designs, and issues. It is not unusual for a usability team unfamiliar with the application domain to be called in to do a usability evaluation.
- The inspection was not grounded in a deep understanding of NCSTRL's capabilities. Unfortunately this, too, happens frequently.
- The inspection did not ease communication between developers and usability people.

Despite these negatives, we feel there is value in reporting what we did, because it helps raise awareness of usability issues in the digital library domain. It also demonstrates that, even in difficult and highly constrained conditions, a usability inspection can yield useful information about usability defects. Finally, having done informal inspections of two other digital library systems, we think that NCSTRL inspection results are typical and representative of many different digital library systems and therefore valuable to designers, developers, and users of other digital library systems. We hope that, by reporting this work, we can contribute to the inclusion of usability practitioners in future digital library projects from the inception.

### *1.3 Usability inspection*

Lab-based usability testing is often the method of choice for thorough formative usability evaluation. A lab-based formative evaluation process would involve employing real and representative users performing benchmark tasks, collecting qualitative and quantitative data to aid redesign. After deployment, remote usability evaluation methods can be used to continue gathering formative usability data from real users doing real tasks in their daily work. We are currently developing a method for remote users to report their own critical incidents [14] and are adapting it to digital libraries.

Our limited resources did not allow for full formative usability testing, leading us to opt for the less expensive usability inspection. Many real-world teams must make this same choice for the sake of cost effectiveness. Usability inspection [29] is a usability evaluation method that:

- Applies to early designs (e.g. low-fidelity prototypes), well-developed designs (e.g. high-fidelity prototypes), and deployed systems.
- Is usually less expensive than lab-based usability testing.
- Does not employ real users (in contrast to lab-based testing).
- Is expert-based (conducted by trained usability engineering practitioners).

- Is often, but not always, guided by user tasks.
- Has the goal of predicting usability problems that users will encounter in real usage.

### *1.4 Networked Computer Science Technical Reference Library (NCSTRL)*

The Networked Computer Science Technical Reference Library (NCSTRL) [4] is a collaborative project involving NASA Langley, Old Dominion University, University of Virginia, and Virginia Tech. NCSTRL is a distributed system that provides a single point of access to the technical reports from participating international computer science departments and laboratories, through its Web interface at http://www.ncstrl.org/. Technical reports are gathered in NCSTRL via a process called 'harvesting' from other repositories. Harvesting is the process of issuing a request through a particular protocol to a data provider (a partner repository or archive) and using the returned metadata (data such as title, authors, date, etc., representing the document) as a basis for building value-added services, such as search engines, browsing capabilities, and output formatting. Submissions to NCSTRL are done indirectly through the Computing Research Repository (CoRR), an online repository of a partnership of ACM (Association for Computing Machinery), the Los Alamos e-Print archive, and NCSTRL.

## 2 Related work

Usability in general has seen enormous coverage in the literature; the book on usability engineering by Nielsen [28] and the usability engineering process books by Hix and Hartson [17] and by Mayhew [26] are representative. Digital libraries represent an important application area for usability and user experience, but in the digital library literature a focus on usability is only beginning to emerge and has lagged the non-user-oriented technical topics. As Dillon [5] puts it, "From an HCI perspective, much of the literature on DL research is highly technocentric.". Dillon goes on to state that only one-third of the papers published in the 2000 ACM Digital Library Conference include user studies. The NSDL literature is similar, with significant efforts being devoted to creating collections [2, 8, 9, 18, 19, 23], protocols for sharing data [2, 3, 20, 22, 30], security issues [6, 11], and even user interface software infrastructure [11, 27, 35], but less to the usability aspects of the work [8].

One example of a brief evaluation of searching and browsing in NCSTRL, conducted by Theng [32], was aimed at exploring the problem of being 'lost in hyperspace' in the context of digital libraries. Theng evaluated searching and browsing by 10 participants, seven of whom were experienced users of digital libraries. The author reported that 30% of users experienced feeling lost, 40% could not identify where they were, and 80% could not return to previously visited information. To address these problems, the author recommends that developers must "take into consideration both users' and designers' goals". The paper does not recommend any more specific design changes to the NCSTRL interface to address the problems found in the study.

Another representative digital library usability report in the literature discusses the usability evaluation of Cypress, a digital library with approximately 13 000 color images and associated metadata from the Film Library of the California Department of Water Resources (DWR), a division of the California State Resources Agency [34]. The authors stress the importance of user-centered design applied to digital libraries, illustrating with issues and problems specific to the Cypress system. They grouped usability problems found (via user-based usability evaluation) by related usability principles.

In our work, we found that grouping usability problems by principles did not provide any new insights into the problems found. In our evaluation, we have grouped the problems by usability problem type and/or by system functionality. This method has helped us see clusters of problems and draw some important conclusions, which are described in Sect. 4.

The Cypress study also emphasizes user needs assessment, with which we strongly agree. Their iterative design process gives substantial up-front attention to the needs of users in the areas of content, resource discovery and retrieval methods, document analysis, interface design, and browsing.

Another usability report on digital libraries [33] is a preliminary report of the User Needs Assessment and Evaluation for the UC Berkeley Electronic Environmental Library Project, which is aimed at developing a massive, distributed, electronic, work-centered library of environmental information containing text, images, maps, sound, full-motion videos, numeric datasets, and hypertext multimedia composite documents to support actual environmental planning decisions. These researchers discuss the user needs assessment and evaluation components of the project: the underlying premises, methods, and initial findings.

A more formative approach, based on user studies, taken by the researchers at the Harvard–MIT Data Center (HMDC) at Harvard University and the University of Michigan's School of Information and College of Engineering, is presented in the paper entitled 'Usability Testing of the Virtual Data Center' published in the Joint Conference on Digital libraries (JCDL) in 2002. The Virtual Data Center (VDC) is an open-source, Web-based digital library for the management and dissemination of social science research data.

Usability testing and heuristic inspections were an integral part of the development process from the inception of the VDC project and throughout its evolution. Among other issues, the paper stresses the negative effects on usability that result from a presupposition that users have

high levels of technical expertise. We strongly agree with this and have found many such problems related to the use of technical terms and specialized language in digital libraries that are not user centered. We discuss these in later sections of this paper.

Interested readers can find an overview of the history of NCSTRL in Davis and Lagoze [4] and of performance evaluations for NCSTRL server activities in Powell and French [31].

## 3 Method

### *3.1 Inspection instance*

We performed the NCSTRL usability inspection in April 2002. It took about 24 person-hours for the inspection and about 16 person-hours to complete the original (online) report [16]. For the inspection, we used Internet Explorer version 5.50.4134.0600 on Microsoft Windows 2000. The evaluation was conducted on an AT/AT compatible Gateway computer with 130.6 MB RAM, using a 10-Mbps connection to the Internet. Our evaluation was conducted on NCSTRL as it existed around the time frame mentioned above, and we have not tracked changes or improvements made to the system since the inspection. We tried to preserve a copy of the evaluated version of NCSTRL for reference purposes. However, as it was not practical to establish a static version of the entire, database-intensive NCSTRL system merely as a reference for this report, we are unable to provide a persistent URL to the evaluated version of NCSTRL.

### *3.2 Pre-inspection preparation*

In preparation for the inspection, we compiled a list of generic and representative NCSTRL tasks, but not benchmark tasks [17] or detailed scenarios, to represent the breadth of usage possibilities. We browsed and explored the system to become familiar with the overall look and feel and range of functionality.

### *3.3 Co-discovery technique*

The primary method used for this usability inspection is sometimes called co-discovery [12] in which two or more evaluators work together. Co-discovery often leads to rich verbal protocol arising from the conversations among the evaluators. In our study we used three evaluators (the authors) with different characteristics, in order to get a broad representation of user perspectives. One evaluator was completely new to NCSTRL; another evaluator had enough experience with NCSTRL to know what kinds of tasks are appropriate; and the third had been working more extensively with digital libraries in general. Working in twos, or all three together, this variety in backgrounds allowed us to understand design constraints and conventions behind some usability issues while not overlooking impressions that a new user might form. Furthermore, one of the evaluators is involved in the design of CITIDEL and was, at the time of the work here reported, involved in design decisions that would produce a system similar in functionality to NCSTRL. This provided the team with extra expertise and understanding from an implementation point of view. This is a perspective that is often ignored in usability reports, which tend to focus only on the user experience. Having an understanding of the design decisions made or considered to reach a particular user interface implementation allowed us to provide better feedback on what possible approaches could make the user experience better.

Despite our range of digital library background knowledge, none of us works in digital libraries as a primary field of interest. This lack of depth in our experience with digital libraries was both a benefit and a disadvantage. Our relative naiveté with respect to digital libraries allowed us to operate without the biases of digital library tradition and conventions (of which, we learned, there are many) and perhaps to consider less constrained design ideas. In other cases, we could provide an independent confirmation of larger technical issues, such as those for combining search and browse functions, broadly discussed in the digital library community but of which we were not completely aware before the inspection. On the other hand, in some cases we could have used more knowledge about digital libraries. For example, it would have been useful to know when digital library conventions were strong enough to color user expectations to the point where they might override our perceived usability issues.

### *3.4 User classes*

The primary user class for NCSTRL is composed of scientific researchers in the field of computer science, who use NCSTRL to search and browse technical reports, retrieve documents, and submit their own technical reports via CoRR (Computing Research Repository), from which the reports are harvested by NCSTRL. This user class was the only one represented in our usability inspection.

A group of 'administrators' also uses NCSTRL, to keep the system updated by harvesting metadata on a regular basis from the various sources that support the Open Archives Initiative. However, the administrators use a command-line interface and not the usual Web-based user interface that we inspected.

### *3.5 User tasks*

Our approach to usability inspection was task-driven and we used the following high-level categories of representative tasks to organize our inspection process:

- **T1 – Search** for technical reports based on a set of criteria.

**T1a – Simple Search** involves searching all bibliographic fields, grouping the results by archive, and sorting.

**T1b – Advanced Search** provides for searching on specific bibliographic fields with several filter options.

- **T2 – Browse** the NCSTRL collection.
- **T3 – Register** with NCSTRL to publicize (advertise) reports through NCSTRL.
- **T4 – Submit** a technical report to CoRR.
- **T5 – Harvest** from NCSTRL using the Open Archives Initiative protocol suite.

Because of the specialized task domain requirements of task categories T4 and T5, our inspection involving these categories was limited and we recommend further usability testing with real users doing real document submission and harvesting tasks.

## 4 Results and discussion

As is usually the case for a usability inspection, or any method for formative usability evaluation, the results were in the form of a list of usability problems, each problem a candidate to be fixed in an effort to improve usability in the interaction design. In our presentation of these problems, we have grouped them by usability problem types (such as wording problems and consistency problems) and/or by system functionality (such as search function and browse function), rather than by severity (until the cost–importance analysis in Sect. 5).

Wherever usability problem numbers (e.g. Overall.1 and T1a.2) appear (e.g. in the cost–importance analysis spreadsheets shown as tables in Sect. 5), the problem numbers indicate the related task per the task labeling given in Sect. 3.5. These are also the same problem numbers used in the sample-evaluation report [16].

### *4.1 Good points about the NCSTRL design*

The purpose of usability inspection is to find problems. Thus, a list of usability problems at the end of the day is an indication of success for the process and should not spell embarrassment or criticism for the designers themselves. However, because a list of problems can often be perceived as only negative and critical, especially by developers who have an 'ownership' stake in the design, usability evaluators wishing to avoid being perceived as the 'usability police' should not overlook the good points. To this end, we must say that we did find NCSTRL to be generally easy to understand and not difficult to use. The system functionality seems appropriate and the user interface is aesthetically pleasing. In fact, we found no 'show-stopper' usability problems that must be fixed, regardless of resources available.

The goal of a usability report is, however, to list what the evaluators perceive to be usability problems, and that is the thrust of this paper. Although the discussion of the problems here is specific to NCSTRL, since we have found that many of the problems reported here are common to numerous digital library systems we feel that this discussion will also be useful in a more general context of digital library interaction design and evaluation. It is our hope that this report will help raise usability awareness, leading to a clearer understanding of user experience issues among digital library system designers.

### *4.2 Organization of usability problems for reporting*

The raw output of a usability evaluation (lab-based or usability inspection) is an unorganized 'laundry list' of usability problem descriptions. Our definition of a usability problem is anything that impacts the user's task performance or satisfaction. This takes the concept beyond the popular misconception that usability problems are only about look and feel, to problems for the user relating to system functionality and its usefulness [24].

We carried out the analysis by organizing the problems around usability issues and then using the problem descriptions to illustrate and relate to these issues. We began by printing out each usability problem description on a separate piece of paper. We then labeled each one with one or more 'keywords' that we thought were the best descriptors of the problem type, and sorted them by type. We found that some of these categories were what we called 'the usual kind', problems commonly found in most applications (especially GUIs) and not specific to a particular NCSTRL function. Other problems were very closely tied to the design for a digital library function. We further sorted the former class into problems about precise wording, consistency, graphic layout, and organization, and the user's model of the system. The latter class was primarily about browsing, filtering, searching, and document-submission functions.

This simple sorting provided a modest chance at manual data visualization and confirmed to us the value of data visualization for usability for showing the broader picture of relationships among the data. Despite doing the evaluation, writing up the separate problem reports, and organizing and posting the whole thing on our project Web site, we were never completely aware of the nature of the data we had collected. Perhaps this was because, in such projects, the evaluators often find and report each problem in isolation.

In any case, we were surprised to discover that problems about wording, not directly associated with a particular function, accounted for 10 of our 28 total problems, or almost 36%. To someone not familiar with cognitive affordances [13] in interaction design, problems with wording may seem unimportant because they are 'just about words'. However, precise use of words in user interfaces is one of the utmost important design considerations for usability. Clear, complete, and correct wording of button and tab labels, menu choices, and Web links is crucial to helping users, especially new users, learn and

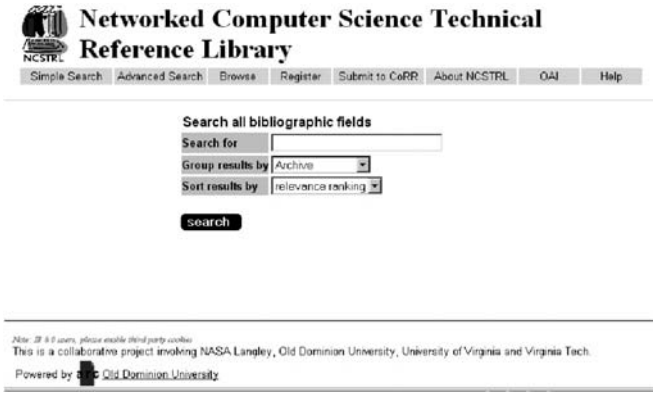


**Fig. 1.** The NCSTRL user interface

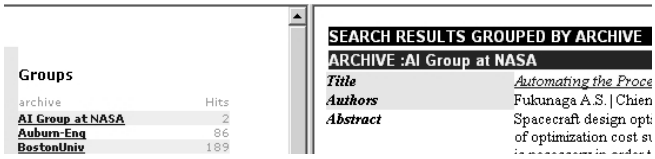


**Fig. 2.** Inconsistent use of terms **Groups** and **Archive** referring to institution

understand the functionality behind those interface controls. Additionally, since fixing wording often yields the greatest improvement in usability for the least cost, problems about wording are among the most important kinds of usability problems to solve. NCSTRL's high incidence rate of wording problems pointed out the value of having a person with strong writing and word skills on the interaction design team.

Another realization, perhaps less surprising but interesting, was that we had five problems (almost 18%) concerning the search and browse functions. This helped us look at those problems together and realize that the interaction design (and possibly underlying functionality) for these functions should be reconsidered together and would probably benefit significantly from a completely new, more integrated, design. The manual data-visualization exercise helped us see this need and the redesign requires a level of integration developers cannot get from looking at the problem descriptions one at a time. Furthermore, from an implementation perspective, searching and browsing are usually thought of as two separate functionalities. But, from a user perspective, they are so closely related to a single task (e.g. finding a resource) that they should be designed together.

### *4.3 The usual suspects*

Finding usability problems of the 'usual kind' indicates that digital libraries are susceptible to the same kinds of problems as are many other applications. As context for the various problem descriptions, Fig. 1 shows the NCSTRL user interface and its three frames, a small one at the top for the tabbed menu of functions, a small one at the bottom for information on the project collaborators, and the large one in the middle for the main interaction. The top and the bottom frames remain constant throughout the interaction.

#### 4.3.1 Problems with consistency

The terms **Group** and **Archives** seem to have been used interchangeably (Overall.5), both referring to institutions and to collections belonging to institutions. (Note that wording quoted from the NCSTRL user interface is distinguished by boldface font.) For example, the middle of the three search parameters in Fig. 1 shows **Group results by Archive**. At the left in Fig. 2 the term **Groups** is used to refer to an institution with a collection of reports. At the right, however, the term **ARCHIVE** is used instead. Two or more terms referring to the same concept can be confusing to the user and can impair learning. This kind of inconsistency can at least have a mild impact on new and returning users, who will often have to take the time out from a task to look for evidence to confirm that these terms are, in fact, used in the same way. This kind of terminology variation occurs naturally in a project before the terminology is 'standardized', but designers must root it out, to avoid confusion to the users. Both consistency and precision might be gained by using the term 'institution', for example, in both places. The final choice for a consistent term (as for any terminology question) is, of course, up to the users, who are subject-matter experts.

Another design point (T1a.4) about consistency relates to the difference between the **Simple Search** tab (far left in the top tab bar in Fig. 1) and the main label for the simple search function (at the top of the middle frame in Fig. 1), which is **Search all bibliographic fields**.

Usability designers view this technically as an inconsistency because the label at the 'point of departure', the **Simple Search** tab, and the label at the 'destination point', **Search all bibliographic fields**, are not close enough as a match for all users to be sure they arrived at the right place. It is easy to make a better match, boosting user confidence that they have arrived in the correct place, by saying 'Simple Search: Search all bibliographic fields', as shown in Fig. 3.

#### 4.3.2 Problems with feedback

Feedback by highlighting the currently selected task or function is fundamental for providing a 'You are here' style of navigational grounding for the user. Unfortunately, clicking on a 'tab' (a menu button in the top frame of Fig. 1) does not result in highlighting or any kind of feedback as an indication of which tab is the currently active choice (Overall.2). This is typical of HTML designs that do not include some form of server processing to detect which page is being served and disable the link or change the color of the button (see Sect. 4.3.4 below for other problems with this navigational bar).


**Fig. 3.** Proposed matching of departure tab and arrival screen title

The design also has an instance of non-persistent highlighting as feedback (T1a.10). On the search-results page, the institution (archive) selected in the left-hand frame is highlighted when clicked, indicating the source of the contents displayed in the right-hand frame. However, the highlighting is not persistent and disappears after a few subsequent user actions. The solution is to maintain the link highlighting until another link in that group or at a higher level is clicked. In general, feedback used as long-term state information should persist beyond the immediate user action that sets it.

### 4.3.3 Problems with wording

In several places within the design, wording for labeling (especially when used as cognitive affordances for new users) uses jargon or slang or is unclear or missing. In some cases, where software code is apparently being reused, the wording is even incorrect.

*Unclear, imprecise, and incomplete wording.* A good example of terminology that was unclear to us, as new users, and that appeared to be jargon, or at least did not seem to us to be user-centered, is the term **Discovery Date** (Overall.3 and T1b.3). This term is used in several places in the interface; one example is the second choice in the pull-down menu **Sort Results By** for the simple search function, as shown in Fig. 4.

We were told by an NCSTRL developer that **Discovery Date** is the date that the author (or contributor) creates the NCSTRL record representing the document. This is not necessarily the same as the date on which that record is actually entered into the digital library of NCSTRL, which is called the accession date. The difference is due to the harvesting process that occurs between record creation and accession into NCSTRL. Use of the word 'discovery' in the user interface is enigmatic, as the usual concept of discovery does not seem to apply. It is probably a developer's term showing through to the user, and a more user-centered term, such as 'record creation date', is indicated. Designers might have to survey some users to determine what term the average user is likely to use here. Furthermore, it is not clear to us why users would care when the record was created. It seems more likely that users will search for the publication date and not the date a record was created in a database. This information might be used for administrative and/or record-keeping purposes, but would then play a role only with administrators and not with scientific researchers as users (see Sect. 3.4 for the user classifications).

Another way wording can be unclear and imprecise to new users is by being incomplete. An example is the **Submit to CoRR** tab at the top that does not say what is to be submitted and would be more complete if it said 'Submit Technical Report(s) to CoRR' (T4.1). Another example is the label, **Search all bibliographic fields**, in the simple search that does not say what or where the user is searching and would be more complete if it said, including the suggestion made earlier, 'Simple Search: Search all bibliographic fields in selected archive' (T1a.6) or, per the discussion in Sect. 4.3.1, 'Simple Search: Search all bibliographic fields for selected institution'. Additionally, there is plenty of space also to add a sentence about what the term 'archive' refers to in this context – for example, 'Each archive contains the technical reports from one institution'. Although long labels can be unwieldy, they are sometimes necessary for increased completeness and clarity for the user.

This added wording might also be part of the solution for the problem (Overall.5) of using the terms **Group** and **Archives** referring to collections belonging to institutions, discussed earlier in Sect. 4.3.1 and seen in, for example, the heading of the left-hand column at the top of the left frame of the browse page, as shown in Fig. 2.

Another example (T1b.3) of wording that should be made more complete is shown in Fig. 5, where the terms **Archive's Set**, **DateStamp**, and **Discovery Date** (discussed above) are anything but self-explanatory and obvious in their meaning to users.

All these terms seem to be designer-centered terms that need to be changed to user-centered terms and briefly explained. The term **Archive's Set** is a technical term and stands for the 'sets' concept in OAI-PMH (The Open Archives Initiative Protocol for Metadata Harvesting), which is not user-centered. In OAI-PMH, a set is an optional construct for grouping items for the purpose of selective harvesting.

One could argue that users who are doing the tasks involving these terms will know what the terms mean. However, making the words more complete and precise makes a better fit to the tasks and will give new users faster recognition as a match to a given task. Even if new


**Fig. 4.** Use of the term **Discovery Date** in a simple search menu

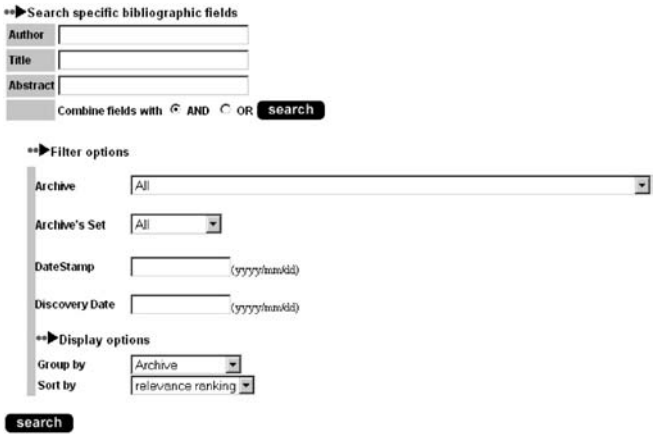


**Fig. 5.** Advanced search, as seen in NCSTRL

users are likely to make the right choices eventually, they might first have to look for and eliminate other possibilities to be sure that this is the best match to the task, incurring an unnecessary productivity penalty.

*Incorrect or inappropriate wording.* On clicking the **Browse** tab and selecting an institution at the left (apparently any institution except the first one), the labels **SEARCH RESULTS GROUPED BY ARCHIVE** and **hits (1–n) of total xxx hits** displayed at the top of the right hand side (see Fig. 6) are incorrect (T2.1). The top label refers to the search function, even though we are in the browse function and the second label refers to the pages and reports as though they have been retrieved via a search when they really are just reports available for browsing. These words appear to come from code shared with the search design. The words should say 'Browse reports, grouped by institution' and 'This is page 1 of a total of 42 pages for the selected institution, reports 1–10 of a total of 418 reports'.

Finally, in several places in the NCSTRL interface, the term 'hit' or 'hits' is used to refer to documents retrieved, as seen for example in Fig. 6 (T2.4). While this is not an uncommon term in the world of information and retrieval, in the usability world it is considered inappropriate slang. At best, it is unattractive and, to some, it is slightly offensive. Although this is only an aesthetic or cosmetic consideration, we do recommend using something more descriptive, such as 'Matches with search term'.

### 4.3.4 Problems with layout and graphic design

Many of the visual cues new users get from GUI and Web designs are from the appearance of the graphic design, especially from how things are located and grouped by related function on the screen. Because of the importance of visual design to user performance, software-development teams increasingly are adding graphic designers and 'information designers' to the group doing interaction design. The top menu bar of NCSTRL (Fig. 1)

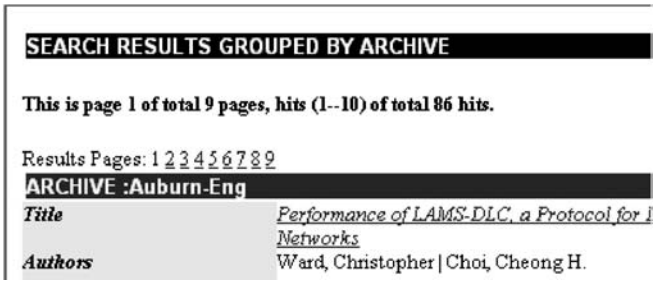


**Fig. 6.** Incorrect labeling for browse function

is a good example of a visual design that needs reorganization in terms of information content (Overall.7). Often menu items get added to a design in the order the functions are considered by designers, an order that might not mean much to the user. After the menu choices are established, designers can improve usability by reorganizing them and grouping by related task or functionality. Organizing tasks by major task categories can present a more structured system model and reduce cognitive workload for users. Choices in the NCSTRL menu (Fig. 1) lead to a mixture of both information-seeking tasks and other interactive functionality.

A natural abstraction of these tabs might be represented by groups labeled user tasks and information links. The user tasks would then contain Simple Search, Advanced Search, Browse, Register, and Submit Technical Report to CoRR. This last item has other problems of its own (T4.2) (see Sect. 4.3.2). The information links would contain the items About NCSTRL, About CoRR, OAI, and Help as shown in Fig. 7 below.

The menu tabs have a further graphical problem. The gray background to the links in the top menu bar makes them look like buttons, as shown in Fig. 1, but they do not behave like buttons (Overall.6). The user might click on the background surrounding a label, assuming it is a button, but not get any result. Because the 'button' is actually just a hyperlink, it requires clicking exactly on the text in order to get a result. We call this a false affordance, since it has a false visual appearance that elicits an erroneous user action. Real graphical buttons are relatively easy to use in place of these links.

The **Advanced Search** tab clearly supports an advanced search as a primary task. However, as another example of graphical design as a cognitive affordance for the user, designers should consider adding an Advanced Search button to support an advanced search as an extension of a simple search (T1a.5). When a user tries a simple search and finds it inadequate, the next obvious step is to try the advanced search. At that point of this task sequence, having an Advanced Search button at hand would support the user's workflow or task thread continuity. It would be easy to include an Advanced Search button on the simple search page, as shown in Fig. 8. Although redundant with the **Advanced Search** tab at the top, this button will appear at the right time and place


**Fig. 7.** Proposed main menu, reorganized by user tasks and information links


**Fig. 8.** Proposed addition of an Advanced Search button to support task thread continuity for the user extending the simple search

for a user who needs to extend a simple search into an advanced one.

Another graphical design issue is about how the proximity of elements in interaction design layouts is used to indicate associations and relatedness. As shown in the top part of Fig. 5, the Search button (for the top section of the advanced search) is located very close to the **OR** choice and in the line labeled **Combine fields with** (T1b.1).

However, the **Search** button is not intended to be associated directly with either the **OR** choice or the **Combine fields with** label. The **Search** button is intended to be associated equally with everything in the **Search specific bibliographic fields** box; i.e. the button goes with the entire search function or task. This broader association can be indicated by moving the button further away and up to the middle, as in Fig. 9.

Our final example of the need for graphic design to improve the layout to group user interface elements by related function is seen for the advanced search in Fig. 5 (T1b.4).

The pieces of the display that support the advanced search task should be organized and grouped together by related function and the **Search** button should be moved to show its association with the entire search function. Figure 10 shows a possible design solution.

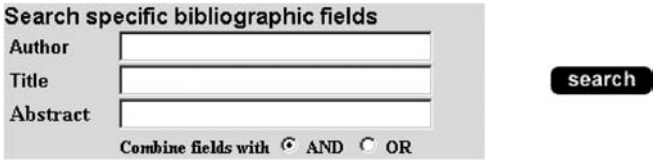


**Fig. 9.** Proposed improved layout location of search button to associate it with the entire search function

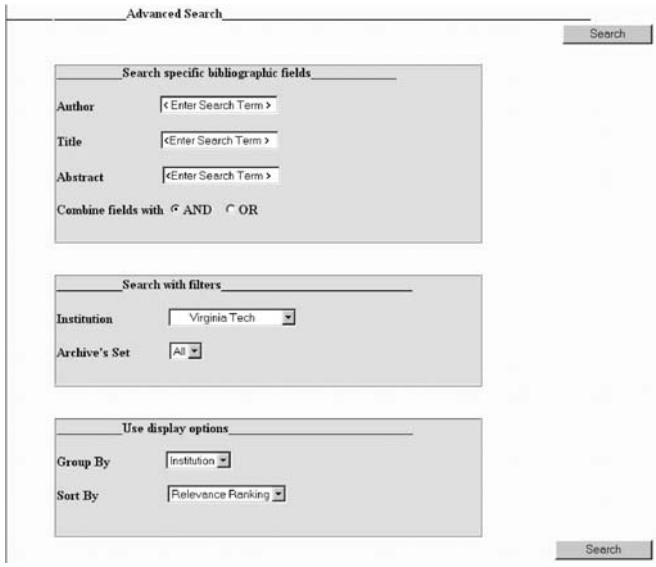


**Fig. 10.** Possible design solution for grouping by function within advanced search layout

In this design solution, the user would immediately know that the search button is associated with the entire set of fields, whereas the original design implies two separate searches, one a plain bibliographic field search and the other with **Filter options** and **Display options**, which is not actually the case. Also, the indentation of the **Filter options** and **Display options** could cause confusion to the user. The **Filter options** are indented one step and the **Display options** are indented further within the **Filter options**, which does not convey any necessary meaning.

### 4.3.5 Problems with user's system model

New users often expect the home page of an application to be a kind of 'splash page' that introduces the application and tells users how they can use it, thereby helping the user with planning about how to use the system to achieve goals in the work domain. In contrast, the NCSTRL has no home page, as such; it uses the simple search screen as the default screen, as seen in Fig. 1, and in so doing makes simple search the hidden (implicit) default task or function for the user (Overall.1 and T1a.1).

New users, arriving at the simple search screen without clicking on anything, may wonder what that is and how they arrived at it. A design that starts the site off with a function that wasn't requested by the user presents a confusing model of the system and can seem to some users as though the system has taken control from the

user and presented the simple search function possibly as though the user must start with a search.

If the simple search could be justified as the default function upon entering the system, then the design should follow the usability principle of making default choices clear. The redesign solution could be as simple as highlighting the **Simple Search** tab initially. At least that would show that the simple search is selected upon arrival, explaining why users see what we see in Fig. 1 immediately upon arrival.

However, we recommend, instead, designing a separate home page for NCSTRL to help new users understand what NCSTRL is and how it can be used. Expert users can by-pass the home page with one click on the tab of their choice.

### *4.4 Problems more specific to digital library functions*

Some of the usability problems in the results were more specific to digital libraries. These problems also tended to be more complex and often related to how some of the core services [21, 22] are presented to users. We are now aware that these problems are difficult and that the information-retrieval community has been struggling with some of them for years. We think that it is important, however, to show that an independent evaluation not biased by a detailed knowledge of these struggles has pointed out the same problems, confirming their importance as ongoing research issues.

#### 4.4.1 Searching, filtering, and browsing

In our usability inspection, we found several aspects of the searching, filtering, and browsing functions to merit consideration for redesign. It is possible that a single integrated redesign could solve most of these problems at once, given significant collaboration with designers and other developers.

Searching, filtering, and browsing are core services to almost all digital libraries and portals. As Robert France put it [7], users see searching, filtering, and browsing just as ways of selecting and navigating through documents and document surrogates to find information or resources to meet a need in their work domain. In this perspective, search criteria and filters look pretty much the same. Each is a logical criterion (query) that in some sense defines a certain subset of the full collection that the user expects to be 'interesting'. Each is performed indirectly by the user in that it is automated by the system on behalf of the user.

In contrast, browsing is performed directly by the user through explicit acts of navigation within information structures. During this navigation, the user manually decides which documents encountered are 'interesting'. One of the most common types of browsing is navigating through lists of results retrieved by a search. In this case, one could say that the search was used to filter the information structure and the browsing was used to navigate the filtered information structure. Another kind of browsing is navigation through a list (or other structure) of documents in a given category. Filtering can be applied to either searching or browsing. If a filter is applied to a search, the result to the user is usually the logical AND of the search criteria and the filter criteria. That result can be interpreted as the filter pruning the search space for the user. If applied to browsing, the filter prunes the browsing space for the user.

To developers, these issues are somewhat different. Searching and browsing seem to be provided as separate functions in many digital library systems, as they are in NCSTRL. We believe this is often because developers see these functions as having different underlying mechanisms and follow a functional, rather than a task-oriented, approach to interaction design. The difference between a filter and a search might be that search criteria include attributes for which developers can build effective indexes to speed internal searching. Filters, especially for post-search pruning, might contain attributes for which record-by-record matching is necessary. By following a functional view of keeping them separate (e.g. invoked by different tabs), the user may be discouraged from using them together in a more integrated way where the two functions can work together.

*Combining searching, filtering, and browsing.* The above discussion leads us to one design consideration we suggest that most broadly affects many different digital library interfaces: the consideration to combine the search, filter, and browse functions into a single, powerful data-selection and navigation capability. This approach has been suggested for other kinds of information systems and database systems [25] but, for digital libraries, this point is only conjectural and would have to be discussed at length with digital library designers and users, taking into account other factors such as tradition, conventions, user expectations, and other task threads that might require separate search, filter, and browse functions.

Although it is a different domain and a different kind of interaction, perhaps digital library designers can take a suggestion from the domain of Web searching. Increasingly Web users are demanding more flexibility in searching and more power to customize searches to match their own information needs [10].

*Supporting iterative query refinement.* As shown in Fig. 11, the display of query results does not also show the query that led to those results (T1a.2).

This search appears to be designed as a one-shot function where the user submits a query, gets the results, and is done. However, from empirical usability data, we know that when users submit queries, the results are not always what they expected. The next logical step for the user in this task thread is to compare the query and the results to see how the query should be iteratively modified to get the desired results. Going back and forth between

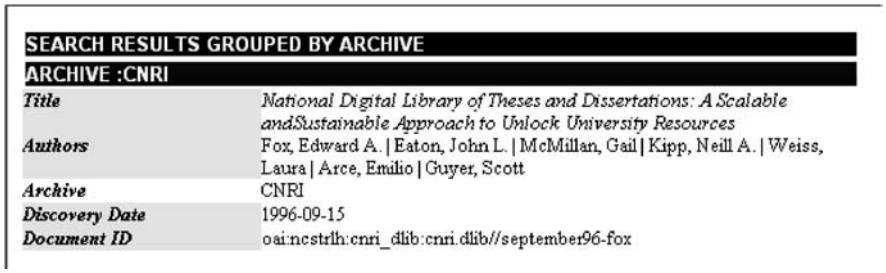


**Fig. 11.** Display of query results

the query and the results on different screens introduces a workload on the user's limited human memory capacity. Having the query shown along with the results supports direct iterative query refinement.

Further, users often wish to prune the retrieved set of documents by applying a subsequent query to the results themselves. The need for this capability is quite common, being essential to support a divide-and-conquer approach to information seeking. But NCSTRL does not directly support the ability to search again, using a different query, within the results of a previous query (T1a.8). A design that supports the continuity of this essential query refinement task threads will show the query along with the results and will allow applying a query to the results in addition to searching the entire archive.

A minor additional limitation is that NCSTRL allows browsing only for documents grouped by archive (institution) and does not allow browsing technical reports grouped by author, year of publication, or other attributes (T2.3). We do not know how important these tasks are to most digital library usage, but the user community of most digital library systems is very broad, requiring broad task support in the system.

#### 4.4.2 Inter-system navigation: the 'portal pass-through' problem

According to the glossary found at http://www.auburn.edu/helpdesk/glossary/portal.html, a portal is an entry point or starting site for a portion of the World-Wide Web, combining a mixture of content and services and attempting to provide a personalized 'home base' for its users with features like customizable start pages, filterable e-mail, a wide variety of chat rooms and message boards, personalized news and sports headlines options, gaming channels, shopping capabilities, advanced search engines, and personal home-page construction kits. Most of these portals started simply as search engines, but now have added other services such as email, chat, etc., and have thus transformed themselves into Web portals to attract a large audience.

In the context of digital libraries a portal could be thought of as a single point of access to distributed systems that provides services to support user needs to search, browse, and contribute content, often linking to shared existing functionality at other sites.

The practical effect in NCSTRL (and many other digital library systems) is a single portal Web page through which each participating site or system is accessed. Just listing the participating sites and giving links to them on this portal page makes the page little more than a high-level index to the other sites. In order to present a more integrated view of functionality across the systems of the whole portal, sometimes specific functionality is mentioned in the portal, but only as a link to that functionality in the other sites. This is the source of the 'pass-through' usability problem, a large issue for NSDL and other distributed digital library facilities, and one that we were not able to resolve entirely. It is especially important with respect to the Integration Project, for which we have some responsibility.

*Half-way link to Submit to CoRR.* The specific case of the NCSTRL link to CoRR for technical report submission is a good example of the portal pass-through problem (T4.2). The NCSTRL link **Submit to CoRR** is indirect (a 'part-way-there' link), leading not directly to a CoRR page for submission, but to the home/welcome page of CoRR, which in turn has a link for submitting a technical report, embedded within its text, as shown in Fig. 12.

This indirection offers poor support for the user's task thread. Upon arrival at the CoRR home page via the **Submit to CoRR** NCSTRL link, the user is expecting immediate access to functionality for submitting a document. Once it is apparent that this is only the CoRR home page, the CoRR link for submission may still not be immediately obvious. The user who can eventually find it, however, is again deflected from the task at hand, because that link actually leads to a dialogue box for logging in. Nor does this terse dialogue box explain that logging in is required here or why; it just offers the possibility to log in and stands in the way of doing anything else – an unarticulated indication to the user that logging in is required before submitting. Of course, logging in requires a password, having a password requires being registered as a user, and so on, moving further and further from the submission task, walking the user backwards through a task sequence revealed only a step at a time while requiring users to stack the intervening unexpected tasks in their mental models. There is much information overloading occurring in the login dialogue box and it does not support the user's task thread well. Among the usability

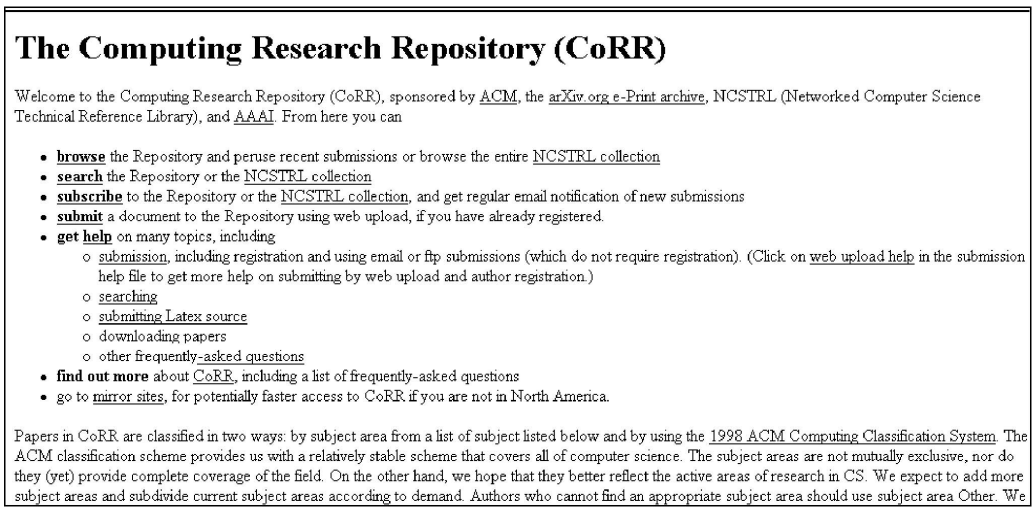


**Fig. 12.** Partial view of CoRR home page, containing the link to submit a document

guidelines that apply to this case are: use precise labels, ensure that responses to user actions are predictable, and avoid function overloading.

The portal pass-through problem is a very difficult problem, the most significant part of which is a complex system model that derives from using a portal to combine multiple systems, each with a potentially different method of operation, but without any real integration. Additionally, the model is more or less hidden from users. Just as an example of the hidden relationship between NCSTRL and CoRR, once a user has submitted a technical report to CoRR, it does not show up in NCSTRL until it is later 'harvested' from CoRR. Thus, regardless of the rest of the solution, an up-front description of how it all works is essential, perhaps from a tab or button labeled 'About submitting reports'.

Beyond this up-front explanation of the system model, a possible solution could employ a 'Submit TR to CoRR' tab taking the user to a new page *within NCSTRL* that has its own 'Submit TR to CoRR' button, that explains the situation, and that serves as a 'base of operations' for going back to register, logging in, and moving forward to submit. In particular, this page in NCSTRL should:

1. Explain that submitting a technical report is a function within CoRR, not NCSTRL.
2. Explain that clicking the 'Submit TR to CoRR' button here will take you to the CoRR home page, where you will be in a different system.
3. Explain that a TR submitted to CoRR will not show up in NCSTRL until later, after a 'harvest' from CoRR.
4. Explain that the CoRR page will open in a new window and the back button will not return you to NCSTRL.
5. Explain that you will then have to click on the 'Submit' link in the CoRR home page.
6. Explain the need to log in to CoRR, in order to submit.
7. Offer a button leading directly (in CoRR) to the dialogue box for logging in.
8. Explain the need to be registered/subscribed as a CoRR user in order to log in.
9. Offer a button to go directly (in CoRR) to register.
10. Offer a button labeled 'Submitting Technical Reports to CoRR', to be used when the pre-requisite conditions regarding logging in and registering are already satisfied, to go directly (in CoRR).

This solution helps the user with the complicated mental model and supporting the user's task stream. The extra click and page it requires is more than compensated for, and is not costly to the user since this task will not occur frequently.

Interestingly, developers of NCSTRL encountered one other form of the portal pass-through problem in regard to the formatting of distributed search results [4]. When a search from one NCSTRL site found documents on another site, the reports were displayed using the local site's look and feel, that is, the look and feel of the site at which the document was stored. Davis and Lagoze [4] reported that the NCSTRL software was updated to produce a more consistent look and feel across sites. We expect that NSDL will follow suit once the effect on users is clear.

## 5 Sample cost/importance analysis

After data collection, we had typical usability data in the form of usability problem reports, along with our severity ratings of each problem. We then contacted a developer for estimates of the cost to fix each of the problems, after which we could perform a sample cost–importance analysis. The importance (to fix) ratings and cost (to fix) estimates here are just representative, to illustrate the analysis process. Each digital library project will have its

own goals and criteria and would assign its own importance ratings and cost estimates, plus their own rationale for these figures.

The purpose of cost–importance analysis is to help manage the design-change process in a realistic project environment where budget and schedule are limited, as explained in Chapter 10 of Hix and Hartson [17]. Because most development teams cannot afford to fix all usability problems found, they need a method for prioritizing, to provide solutions that offer the most effect on increased usability per unit of cost.

Cost–importance analysis is done in a spreadsheet where, for each problem or group of related problems to be considered together, developers estimate the importance to fix and cost to fix, calculate ratios of importance to cost, and sort by priority ranking. By drawing a cutoff line just before where cumulative cost meets or exceeds available resources, developers can delimit the problems which realistically can be fixed. All usability problems below the cutoff line are saved to fix if resources permit or are tabled until the next design iteration or next product version.

### *5.1 Importance (to fix)*

The rating of importance (to fix) a usability problem is an engineering estimate and usability engineering practitioners get better at making this kind of estimate with experience. Because the importance rating is an estimate, it is best to keep the basis simple. We have chosen a scale of five integer values, allowing for fractions or decimals by interpolation [15]. In addition, the highest importance rating is a non-integer value of importance = M, meaning *must fix*, regardless of cost. This rating is used when a usability problem is a show-stopper or affects primary mission-critical or safety-dependent tasks. The numeric importance ratings are:

- Importance = 5: the problem has a major impact on task performance (e.g. the user cannot complete a key task), is expected to occur frequently, causes costly errors, or is a major source of dissatisfaction. Major problems that cannot be overcome by user learning are included (e.g. lack of feedback for an operation).
- Importance = 3: the user can complete the task, but with difficulty (e.g. it caused confusion and required extra effort) or the problem was a source of user dissatisfaction. Includes problems that might otherwise be rated as 5 but can be overcome by user learning (e.g. which button to click to do a certain operation).
- Importance = 1: the problem did not impact task performance or dissatisfaction much (e.g. was an irritant or a cosmetic problem), but is still worth listing.

The numeric values in between (2 and 4) are for interpolation in cases where one of the other ratings needs to be down-graded or up-graded due to, for example, a high or low frequency of occurrence, the number of user classes affected, or the amount of user effort to recover. For example, a problem that is seen to block task completion is given an initial rating of 5 but, since the situation will not arise frequently and the task is not critical, the rating is down-graded to 4.

### *5.2 Cost (to fix)*

Developers next consider possible solutions for each problem and estimate the cost to solve each, in person-hours. This is a stage at which it is often useful to combine related problems to be considered for a single solution that solves the whole group. Additional training is often suggested here as a solution, but is seldom a good solution, since the cost must be repeatedly incurred, for each new user, whereas an interaction design change that obviates the need for this training has a cost paid only once.

Cost figures given to us by NCSTRL developers, for example, were as low as ‘a couple of minutes’ to as high as a month. To allow for some overhead in considering each new problem and to dodge the problem of guessing how many minutes an estimate such as ‘several minutes’ means, we adopted the approach of rounding all cost amounts under one person-hour up to a value of one. Where the designers’ estimate was ‘several hours’, we went with a half-day (four person-hours). Similarly, we used four days (32 person-hours) for estimates of ‘several days’. Table 1 shows the major problems with their importance ratings and cost estimates.

### *5.3 Priority rankings*

Before computing priority rankings, developers move all problems rated as M (something we did not have in this evaluation) to the top of the priority list. Sometimes the cumulative cost of the M-rated problems exceeds available resources, causing difficult management decisions and precluding fixing any other problems.

Using the problem data of Table 1, we then computed the priority ratios as (importance/cost)×1000. A higher ratio means more impact for the cost of fixing the problem. The figure of 1000 is just a scaling factor that ensures most ratios will be easy-to-use integers. We then sorted the problems of Table 1 by descending priority ratios, obtaining ordinal priority rankings, but the result was an ad hoc mixture of various seemingly unrelated problems. By grouping the problems according to the categories discussed in Sect. 4.2 we obtained a more well-behaved analysis, as shown in the spreadsheet of Table 2.

Suppose, just as an example, that the development team has two-and-a-half people available for one week to fix the problems. That is 100 person-hours and will just about cover all but the last group of problems, as seen in the final column of Table 2, the cumulative cost. Unfortunately, in this case, because of their high cost, the

**Table 1.** Cost-importance ratings for usability problems

| UP number | UP description | Importance (to fix) | Cost (to fix, in person-hrs) |
|---|---|---|---|
| Overall.1 | Home-page design | 2.0 | 36.0 |
| Overall.2 | Tabs need feedback | 3.0 | 1.0 |
| Overall.3 | Jargon (discovery date) | 3.0 | 1.0 |
| Overall.4 | Inappropriate term (hits), slang | 1.0 | 1.0 |
| Overall.5 | Inconsistent terms | 2.0 | 1.0 |
| Overall.6 | Links that look like buttons, but don't behave like buttons | 1.0 | 2.0 |
| Overall.7 | Tabs need organization | 3.0 | 6.0 |
| Overall.8 | Consider combining search and browse functions | 5.0 | 120.0 |
| T1a.1 | Hidden default for function/task | 3.0 | 1.0 |
| T1a.2 | Can't see query and result together | 5.0 | 1.0 |
| T1a.3 | Hidden default for search criteria | 2.0 | 1.0 |
| T1a.4 | Simple search label needs to match tab | 1.0 | 1.0 |
| T1a.5 | Advanced button in simple search | 2.0 | 1.0 |
| T1a.6 | Search label not clear | 1.0 | 1.0 |
| T1a.7 | Need for search box entry | 1.0 | 1.0 |
| T1a.8 | Missing functionality | 5.0 | 120.0 |
| T1a.9 | Needs user control of result display | 1.0 | 3.0 |
| T1a.10 | Non-persistent highlighting | 3.0 | 1.0 |
| T1b.1 | Spacing of search button | 2.0 | 1.0 |
| T1b.2 | Imprecise filter options label | 2.0 | 1.0 |
| T1b.3 | Unclear wording | 4.0 | 1.0 |
| T1b.4 | Page layout for advanced search needs grouping by function | 4.0 | 1.0 |
| T2.1 | Incorrect wording – search results | 1.0 | 2.0 |
| T2.2 | Incorrect wording – result pages | 2.0 | 2.0 |
| T2.3 | Limited browsing functionality | 4.0 | 160.0 |
| T2.4 | Inappropriate wording – number of 'hits' | 4.0 | 1.0 |
| T4.1 | Incomplete wording in a label as a cognitive affordance | 1.0 | 1.0 |
| T4.2 | Half-way link to Submit to CoRR | 5.0 | 36.0 |

**Table 2.** Cost-importance ratings for groups of related usability problems

| UP group | Avg importance (to fix) | Total cost (to fix, person-hrs) | Priority ratio | Priority rank | Cumulative cost (person-hrs) |
|---|---|---|---|---|---|
| Feedback | 3.0 | 2.0 | 1500.0 | 1 | 2.0 |
| Consistency | 1.5 | 2.0 | 750.0 | 2 | 4.0 |
| Layout, graphic design | 2.4 | 11.0 | 218.2 | 3 | 15.0 |
| Wording | 2.0 | 12.0 | 166.7 | 4 | 27.0 |
| Navigation | 5.0 | 36.0 | 138.9 | 5 | 63.0 |
| System model | 2.3 | 38.0 | 61.4 | 6 | 101.0 |
| Search, browse functionality | 4.0 | 404.0 | 9.9 | 7 | 505.0 |

troublesome search and browse functions fall below this cutoff line and will not get addressed, even though they have relatively high importance. Nonetheless, developers can begin discussing the redesign, in case they have a chance to make it in the next version.

## 6 Conclusion

Digital library systems are a relatively new area of application for usability design and evaluation. Although we report on the specifics of a usability inspection of a particular system (NCSTRL), the results and accompanying design discussion apply well to most other digital library systems, as well as other systems featuring search and browse functions and concepts such as portals.

## 7 Future work

As future work, we are developing a tool, the Usability Problem Inspector based on the User Action Frame-

work [1], specifically designed to support usability inspections, but this tool was not ready to use for this project. We were, however, able to draw on the concepts behind the Usability Problem Inspector tool and we believe those concepts helped us conduct a more comprehensive inspection than we otherwise would have done.

Someday, we hope to have the kind of usability problem diagnosis, usability database, and data-visualization tools that would have supported our small-scale evaluation and analysis activities in this project, and that we think would contribute enormously to evaluation projects with large numbers of usability problem descriptions.

*Acknowledgements.* We give our thanks to the editor and reviewers for the very thorough and helpful feedback for our revisions. We gratefully acknowledge support from NSF via Grant Nos. NSF DUE-0136690 and DUE-0121679 under supervision of Edward Fox. Our thanks go to Kurt Maly, who heads the NCSTRL interface development team at Old Dominion University, for permission to publish a critical review of the system. We also are grateful to Xioming Liu, who works with Dr. Maly, and who provided us with cost estimates during our analysis.

We acknowledge the influence of ideas from Dr. Robert France, who has contributed for years to the areas of digital libraries and usability, among his other contributions to the world. We included some of his words about searching and browsing and believe he would have been a co-author of this paper, had his life not been tragically cut short last year. This paper is dedicated to the memory of Dr. Robert France.

## References

1. Andre T, Hartson HR, Belz S, McCreary F (2001) The user action framework: a reliable foundation for usability engineering support tools. Int J Hum–Comput Stud 54(1):107–136
2. Bergmark D (2002) Models and tools for generating digital libraries: collection synthesis. In: Proceedings of the second ACM/IEEE–CS joint conference on digital libraries (JCDL '02). ACM Press, New York, pp 253–262
3. Chen S-S, Choo C-Y (2002) A DL server with OAI capabilities: LOVE. In: Proceedings of the second ACM/IEEE–CS joint conference on digital libraries (JCDL '02). ACM Press, New York, p 388 (poster)
4. Davis JR, Lagoze C (2000) NCSTRL: design and deployment of a globally distributed digital library. J Am Soc Inf Sci 51(3):273–280
5. Dillon A (2002) Technologies of information: HCI and the digital library. In: Carroll J (ed) Human–computer interaction in the new millennium. ACM Press/Addison-Wesley, New York, pp 457–474
6. Dorward J, Reinke D, Recker M (2002) Preserving, securing, and assessing digital libraries: an evaluation model for a digital library services tool. In: Proceedings of the second ACM/IEEE–CS joint conference on digital libraries (JCDL '02). ACM Press, New York, pp 322–323
7. France RA (2001) Personal communication
8. Futrelle J, Chen S-S, Chang KC (2001) A CIS framework for NSDL. In: Proceedings of the first ACM/IEEE–CS joint conference on digital libraries (JCDL '01). ACM Press, New York, pp 124–125
9. Geisler G, Giersch S, McArthur D, McClelland M (2002) Creating virtual collections in digital libraries: benefits and implementation issues. In: Proceedings of the second ACM/IEEE–CS joint conference on digital libraries (JCDL '02). ACM Press, New York, pp 210–218
10. Glover EJ, Lawrence S, Gordon MD, Birmingham WP, Giles CL (2001) Web search – your way. Commun ACM 44(12):97–102
11. Gupta A, Ludsher B, Moore RW (2002) Ontology services for curriculum development in NSDL. In: Proceedings of the second ACM/IEEE–CS joint conference on digital libraries (JCDL '02). ACM Press, New York, pp 219–220
12. Hackman GS, Biers DW (1992) Team usability testing: are two heads better than one? In: Proceedings of the Human Factors Society 36th annual meeting. Human Factors and Ergonomics Society, Atlanta, GA, pp 1205–1209
13. Hartson HR (2003) Cognitive, physical, sensory, and functional affordances in interaction design. Accepted for publication in Behav Inf Technol 22(5):315–338
14. Hartson HR, Castillo JC (1998) Remote evaluation for post-deployment usability improvement. In: Proceedings of AVI'98 (advanced visual interfaces). ACM, L'Aquila, Italy, pp 22–29
15. Hartson HR, Hix D (2003) Developing user interfaces – ensuring usability through product and process: short course in the usability engineering process. Unpublished Tutorial Notes
16. Hartson HR, Shivakumar P (2002) Usability inspection report of NCSTRL (TR-02-08). Virginia Tech, Blacksburg, VA
17. Hix D, Hartson HR (1993) Developing user interfaces: ensuring usability through product & process. Wiley, New York
18. Janee G, Frew J (2002) Digital libraries for spatial data: the ADEPT digital library architecture. In: Proceedings of the second ACM/IEEE–CS joint conference on digital libraries (JCDL '02). ACM Press, New York, pp 342–350
19. Klaus C, Andrew K (2001) An atmospheric visualization collection for the NSDL. In: Proceedings of the first ACM/IEEE–CS joint conference on digital libraries (JCDL '01). ACM Press, New York, p 463
20. Lagoze C, Van de Sompel H (2001) The Open Archives Initiative: building a low-barrier interoperability framework. In: Proceedings of the first ACM/IEEE–CS joint conference on digital libraries (JCDL '01). ACM Press, New York, pp 54–62
21. Lagoze C, Hoehn W, Millman D, Arms W, Gan S, Hillmann D, Ingram C, Krafft D, Marisa R, Phipps J, Saylor J, Terizzi C, Allan J, Guzman-Lara S, Kalt T (2002) Core services in the architecture of the national digital library for science education (NSDL). In: Proceedings of the second ACM/IEEE–CS joint conference on Digital libraries, Portland, Oregon, USA, pp 201–209
22. Lagoze C, Terizzi C, Hoehn W, Millman D, Allan J, Guzman-Lara S, Kalt T, Arms W, Gan S, Hillman D, Ingram C, Krafft D, Marisa R, Phipps J, Saylor J (2002) Core services in the architecture of the National Science Digital Library (NSDL). In: Proceedings of the second ACM/IEEE–CS joint conference on digital libraries (JCDL '02). ACM Press, New York, pp 201–209
23. Laleuf JR, Spalter AM (2001) A component repository for learning objects: a progress report. In: Proceedings of the first ACM/IEEE–CS joint conference on digital libraries (JCDL '01). ACM Press, New York, pp 33–40
24. Landauer TK (1995) The trouble with computers: usefulness, usability, and productivity. MIT Press, Cambridge, MA
25. Laurel B, Oren T, Don A (1990) Issues in multimedia interface design: media integration and interface agents. In: Proceedings of the CHI conference on human factors in computing systems. ACM Press, New York, pp 133–139
26. Mayhew DJ (1999) The usability engineering lifecycle. Morgan Kaufmann, San Francisco
27. Nanard M, Nanard J (2001) Cumulating and sharing end users' knowledge to improve video indexing in a video digital library. In: Proceedings of the first ACM/IEEE–CS joint conference on digital libraries (JCDL '01). ACM Press, New York, pp 282–289
28. Nielsen J (1993) Usability engineering. Academic, Boston, MA
29. Nielsen J, Mack RL (eds) (1994) Usability inspection methods. Wiley, New York
30. Pomerantz J, Lankes RD (2002) Integrating expertise into the NSDL: putting a human face on the digital library. In: Proceedings of the second ACM/IEEE–CS joint conference on digital libraries (JCDL '02). ACM Press, New York, p 405

31. Powell AL, French JC (2000) Growth and server availability of the NCSTRL digital library. In: Proceedings of the fifth ACM conference on digital libraries. ACM Press, New York, pp 264–265
32. Theng YL (1999) Lostness and digital libraries. In: Proceedings of the fourth ACM conference on digital libraries. ACM Press, New York, pp 250–251
33. Van House NA (1995) User needs assessment and evaluation for the UC Berkeley electronic environmental library project. In: Proceedings of Digital Libraries '95: the second international conference on the theory and practice of digital libraries, pp 71–76. URL: http://www.csdl.tamu.edu/DL95/papers/vanhouse/vanhouse.html
34. Van House NA, Butler MH, Ogle V, Schiff L (1996) User-centered iterative design for digital libraries. D-lib Magazine. Retrieved 17 January 2003 from the World Wide Web: http://www.dlib.org/dlib/february96/02vanhouse.html
35. Yaron D, Milton DJ, Freeland R (2001) Linked active content: a service for digital libraries for education. In: Proceedings of the first ACM/IEEE–CS joint conference on digital libraries (JCDL '01). ACM Press, New York, pp 25–32