Ben Shneiderman, Donald Byrd, and W. Bruce Croft

How to satisfy users whose searches return mountains of irrelevant documents—or nothing at all? Emphasize user-interface clarity and consistency.

Sorting Out Searching

A User-Interface Framework for Text Searches

INDIVIDUAL USER INTERFACES FOR TEXT SEARCHING ARE OFTEN CONFUSING; AS a group they are seriously inconsistent. We propose a four-phase framework for user-interface design that provides common structure and terminology for searching while preserving the distinct features of individual collections of documents and search mechanisms.

The ideal user interface is comprehensible, predictable, and controllable, but many current text-search interfaces—especially on the World-Wide Web—involve unnecessarily complex and obscure features. The result is confusion and frustration for advanced users as well as for beginners, scientists, and students [8].

Even when a user interface's design is improved, inconsistencies can cause mistaken assumptions and increase the likelihood of failure to find relevant documents as users move from one search system to another. For example, the search string "Hall effect" could produce various searches, including:

- Exact match for "Hall effect"
- Case-insensitive match for "hall effect"
- Best match for "Hall" and "effect"
- Boolean match for "Hall" and "effect"
- Boolean match for "Hall" or "effect"

Few systems spell out the interpretation they are using. Furthermore, systems often use surprising query transformations, unpredictable stemming algorithms, and mysterious weightings for fields. And in many systems, the results are displayed in a relevance ranking whose meaning is a mystery to many users (and sometimes a proprietary secret).

Our four-phase framework increases clarity and

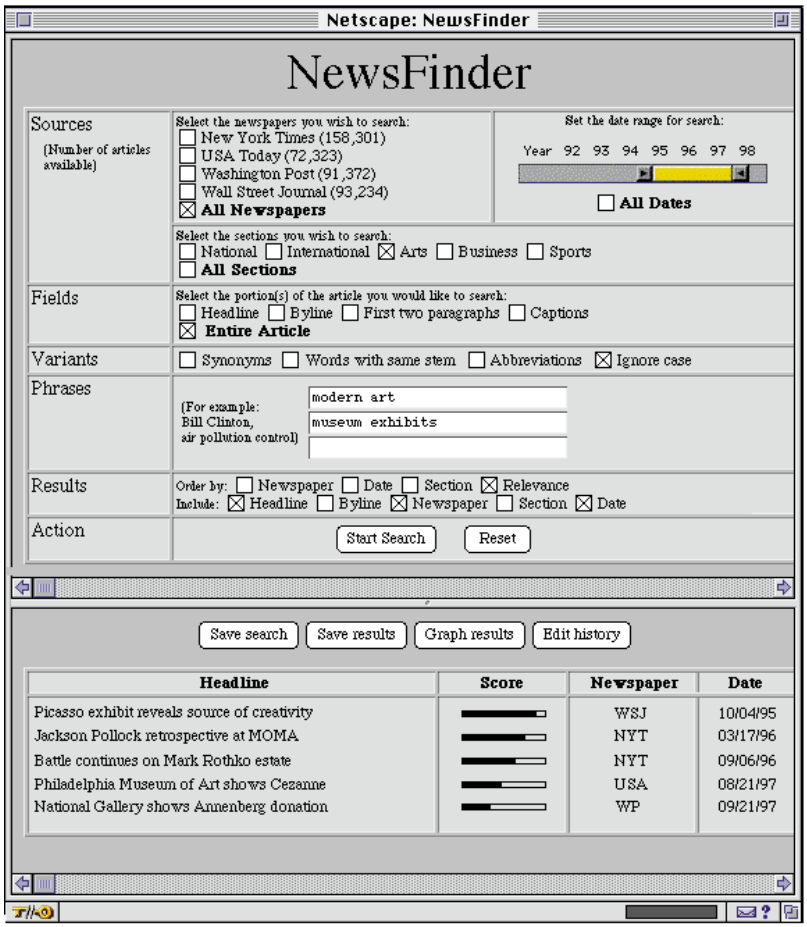

Figure 1. This NewsFinder prototype allows users to see and control all aspects of a search. The conventional interface here includes a button to start searching. But with dynamic queries, the results window (bottom) would be filled from the start and change whenever anything changes in the query window, eliminating the need for an explicit search button.

user control while reducing inconsistencies in text-search user interfaces. Similarly, in the automobile, the driver/user interface is something we now take for granted, but it took decades to reach today's refinement and standardization [5]—and holdover inconsistencies, like left/right variations from country to country, still cause problems for drivers. Cooperation in interface design can spare text-search users millions of conceptual fatalities.

Four-Phase Search Framework

The four-phase framework allows designers of specific systems to offer a variety of features in an orderly and consistent way. Users begin the search process by considering their information needs and clarifying their search goals, after which they are ready to employ a computer-based system for the four phases: formulation (what happens before the user starts a search); action (starting the search); review of results (what the user sees resulting from the search); and refinement (what happens after review of results and before going back to formulation).

Formulation. The first phase is the most complex, in that it involves multiple levels of cognitive processing and decisions of several types, including the *sources* of the search, that is, where to search; which *fields* in which documents to search; which *text to search for*; and which *variants* of that text to accept. Some systems walk the user through these decisions, but the decisions themselves cannot always be made in a predetermined order; nor do they exhaust the query-formulation possibilities.

The first step in performing a search is usually deciding where to search (sources). The location is often a single physical database but increasingly is multiple and distributed databases, accessed across a network. Even if it were technically and economically feasible, searching all libraries or all collections in a library is often undesirable. When users know where the relevant material is, they generally prefer to limit their searches to that library, collection, or range of documents.

Each document in a collection may have multiple fields (sometimes called attributes, components, or tags). Users may wish to limit their search to specific fields or give a higher rank to documents whose titles contain search terms (see, for example, the elaborate weighting algorithm used by THOMAS, the Library of Congress's legislative-information system [1]). Searches may also be restricted by structured fields, such as year of publication, language, and publisher.

There are various ways to express what to search for in full text; the most important are probably unstructured text, text with embedded operators, and text with operators specified separately. Pure unstructured-text interfaces are unusual; most popular Web search services and other systems, such as the commercial system InQuery, accept either unstructured text or text with embedded operators. An example of the latter is the widely used syntax of ""city guide" +Boston" (the words "city" and "guide" must appear very near

each other, and the word "Boston" is required). Finally, the Web search services HotBot and OpenText offer text with separate operators. All three ways can be effective, but only if used properly.

In many situations, searches on meaningful phrases are far more effective than searches on the words in a phrase. For example, for someone searching for information on air pollution, the phrase "air pollution" is likely to find many fewer irrelevant documents (greater precision) than the pair of words "air" and "pollution"—though the search will tend to overlook relevant documents referring to "air quality" and "atmospheric pollution" (lower recall). It should be easy for users to specify that a series of words should be considered a phrase.

The unstructured-text approach, often called "natural language," can be confusing. For example, many systems treat "and" and "not" as stop words. In such systems, the query "bees and not honey" means the same thing as just "bees honey"; compared to the query "bees," "bees and not honey" is *more* likely to retrieve information about honey, not less likely. The traditional solution to such ambiguities is feedback to users on how the system interpreted their queries, but it is difficult to clarify such issues for novices.

In theory, the text-with-embedded-operators approach can be completely unambiguous and still be able to handle phrases and fields. However, our experience is that many users have trouble with this approach. One reason is lack of standardization—the syntax and meaning of embedded operators vary considerably from one system to another, so the user can easily get confused. Another problem is inadvertent activation, as in, say, innocently using text the user thinks of as unstructured but contains characters or strings that will be interpreted as embedded operators. For example, in the AltaVista Web search engine's advanced search option, "*" indicates a "wildcard" (matching anything); in Excite! and HotBot, "*" has no special meaning. These two problems are related in that lack of standardization can confuse users and lead to inadvertent activation. To experience these problems and others first hand, just use several Web search services to look for information on the electronic-commerce company called E*Trade.

Other than embedded operators, the only way we know to specify phrases unambiguously is text with operators specified separately; the program considers the contents of every text-entry box as a phrase and clearly says so on the screen. Then multiple entry boxes allow for multiple phrases. A text-entry box must also accept a single word. If users can choose among Boolean operations, proximity restrictions, and other strategies for combining the boxes, they should be able to express them. But regardless of whether any choices are available, users must be told what combining technique is being used. Ideally, users and service providers should have control over the contents of stop-word lists (such as common words and single letters), and users should at least be gently warned when they try to search for a stop word.

The basic issues are whether the program can interpret the query the way the user intended it and—even if it does—whether the user knows the program interprets it that way.

Users are often unsure of the exact value of the field they want. Indeed, there may be no single value that is appropriate. In such cases, users may want variants to be accepted. In structured fields of text databases, as in traditional databases, these acceptable variants may include a range on a numeric or date field. In unstructured text fields, interfaces may allow user control over:

- Capitalization (case sensitivity)
- Stemmed versions, so, for example, searching for "teacher" finds words like "teach" and "teaching"
- Partial matches, so, for example, searching for "biology" retrieves "sociobiology" and "astrobiology"
- Synonyms, so, for example, searching for "cancer" finds "malignant tumor"
- Abbreviations and acronyms, so, for example, searching for "Digital Equipment Corp." finds "DEC"
- Stop words, so, for example, common words, such as "the" and "to" are excluded

Other possibilities include phonetic variants, like those found by N-grams and Soundex-like methods, and broader or narrower terms, presumably from a thesaurus. In all cases, the user interface should make it clear to the user how variants are handled.

Action. Searches may be started explicitly or implicitly. The typical process today is for users to click on a Search button to initiate the search, then wait for the results. But an appealing alternative is "dynamic queries" in which there is no Search button, but the result set is continuously displayed and updated as the user changes the search. Such research prototypes as FilmFinder and HomeFinder [7] and such commercial systems as Folio Views apply this technique. The dynamic-queries technique requires adequate screen space and rapid processing, but the advantages are great, allowing a user to broaden, narrow, and refocus a search several times in as many seconds.

In situations in which it is not practical to re-run the query and continuously update results—for

example, when the database and the user are connected by a network with limited bandwidth—the "query preview" approach is worth considering [2]. With this approach, changes to a query simply update a display (perhaps just an estimate) of the number of hits. The query is not actually run until the user requests the full results, presumably when satisfied the number of hits is neither zero nor so high as to be cumbersome.

Review of results. Many information retrieval interfaces let users specify result set size (for example, a maximum of 100 documents), contents (which fields are displayed), sequencing of documents (such as alphabetical, chronological, or relevance ranked), and, occasionally, clustering (by field value and topics). All of these capabilities can be valuable in trying to make a list of documents easier to handle. But even with these features, a query against a large database can produce so many potentially useful hits that the result is overwhelming—say, several hundred or more. Fortunately, much more can be done to display results in a useful form.

Recent work in information retrieval interfaces, capitalizing on general information-visualization research, has dramatically expanded the palette of display techniques. For example, LyberWorld [3] displays document icons inside a circle, with terms around the circumference "pulling" the documents toward themselves; the terms can be moved and the strength of their pulls varied. Such techniques as tilebars, perspective walls, cone trees, and document lenses are described in [6].

Search interfaces should also provide helpful messages to support progressive refinement. For example, if a stop word or a misspelling is eliminated from a search input window, or if stemmed terms, partial matches, or variant capitalizations are included, users should be made aware of these changes to their query.

Refinement. Relevance feedback is one of the most important ways current information retrieval technology supports progressive refinement. An empirical study confirmed that users produce superior searches and are more satisfied when they can see and manipulate the words relevance feedback adds to their query [4]. Another aspect of refinement is support for successive queries. As searches are made, the system should keep track in a history buffer, allowing review, alteration, and resubmission of earlier searches.

Conclusions

The sample Web interface in Figure 1 requires nothing more advanced than HTML tables and forms, but in some cases, applying our four-phase framework successfully over the Web may require additional tools, such as Java. On the other hand, while we have concentrated on text situations, we suspect the framework will prove appropriate for multimedia as well as for some traditional database management applications.

Finding common ground for search interfaces is difficult, but not finding it would be tragic. While early adopters of technology are willing to push ahead to overcome difficulties, the middle and late adopters are less tolerant. In particular, the future of the Web as a universal tool may depend on interface developers' ability to reduce frustration and confusion for the masses of users while enabling them to reliably find what they need. Some progress has been made; much more remains to be done. C

REFERENCES

1. Croft, W., Cook, R., and Wilder, D. Providing government information on the Internet: Experiences with THOMAS. In *Proceedings of Digital Libraries '95* (Austin, June 11–13), ACM Press, New York, 1995, pp. 19–24.
2. Doan, K., Plaisant, C., and Shneiderman, B. Query previews for networked information systems. In *Proceedings of the 3d Forum on Research and Technology Advances in Digital Libraries (ADL) '96* (Washington, D.C., May 13–15), IEEE Computer Science Press, Los Alamitos, Calif., 1996, pp. 120–129.
3. Hemmje, M., Kunkel, C., and Willett, A. LyberWorld—A visualization user interface supporting full-text retrieval. In *Proceedings of the 17th Annual International Conference on Research and Development in Information Retrieval (SIGIR'94)* (Dublin, July 3–6), W. Croft and C. van Rijsbergen, Eds. Springer-Verlag, London, 1994, pp. 249–257.
4. Koenemann, J., and Belkin, N. A case for interaction: A study of interactive information retrieval behavior and effectiveness. In *Proceedings of CHI '96, Human Factors in Computing Systems* (Vancouver, B.C., Apr. 13–18), ACM Press, New York, 1996, pp. 205–212.
5. Oliver, S., and Berkebile, D. *The Smithsonian Collection of Automobiles and Motorcycles*. Smithsonian Institution Press, Washington, D.C., 1968.
6. Rao, R., Pedersen, J., Hearst, M., Mackinlay, J., Card, S., Masinter, L., Halvorsen, P.-K., and Robertson, G. Rich interaction in the digital library. *Commun. ACM 38,* 4 (Apr. 1995), 29–39.
7. Shneiderman, B. Dynamic queries for visual information seeking. *IEEE Software 11,* 6 (Nov.–Dec. 1994), 70–77.
8. Somerson, Paul. Web coma. *PC Comput.* (Aug. 1996), 57.

BEN SHNEIDERMAN (ben@cs.umd.edu) is a professor in the Department of Computer Science and director of the Human-Computer Interaction Laboratory at the University of Maryland, College Park.

DONALD BYRD (dbyrd@cs.umass.edu) is a researcher in the Center for Intelligent Information Retrieval in the Computer Science Department of the University of Massachusetts at Amherst and president of Advanced Music Notation Systems, Inc., in Williamsburg, Mass.

W. BRUCE CROFT (croft@cs.umass.edu) is a professor in the Department of Computer Science and director of the Center for Intelligent Information Retrieval, University of Massachusetts at Amherst.

This work is supported in part by the National Science Foundation, Library of Congress, and Department of Commerce under cooperative agreement EEC-9209623; by NRaD Contract Number N66001-94-D-6054; by NASA contract NAG-528-95; and by NSF contract IRI-96-15534. Any opinions, findings, conclusions, and recommendations expressed here are those of the authors and may not reflect those of the sponsors.