User Interfaces to Information Systems: Choices vs. Commands

V. J. Geller

M. E. Lesk

Bell Laboratories
Murray Hill, New Jersey 07974

ABSTRACT

Do users prefer selection from a menu or specification of keywords to retrieve documents? We tried two experiments, one using an on-line library catalog and the other an on-line news wire. In the first, library users could either issue keyword commands to see book catalog entries, or choose categories from a menu following the Dewey Decimal classification of the books. In the second, news wire users could read Associated Press news stories either by posting a keyword profile against which all stories were matched, or by selecting them from a menu of current news items. For the library users, keyword searches were clearly preferred, by votes of 3 and 4 to 1; for the news stories, retrieval by keyword search is 50% less common than menu choice.

We suggest that the difference is based on the degree of user foreknowledge of the data base and its organization. Menu-type interfaces tell the user what is available. If the user already knows, as in the library where a majority of the users have a particular book in mind, then the menu is merely time-consuming. But when the user does not know what is available (almost the definition of "news" is that it is new, and unpredictable), the menu is valuable because it displays the choice.

1. Introduction.

When, in a text information system, should users be asked to define a search strategy of keywords, and when should they merely be asked to choose from a menu? Traditionally, many believe that trained intermediaries are necessary for keyword-type searches,[1] and that novice users should only be asked to choose from menus.[2,3] But advances in data base systems are producing command languages that are easy for novices, and the relative advantages of menu-type and command-type interfaces need more consideration.[4,5]

Consider two ridiculous extremes. No one mails letters by going to the post office and choosing from a list of zip codes to which mail can be sent. Instead, we assume that we can name any place in the U.S.A. and the Postal Service will get the letters there correctly. On the other hand, no one approaches a railroad ticket window, picks a town, hour and minute, and expects to find a train going to that place at that time of day. What matters, we suggest, is how much the user knows about the available choices. The ecological niche of menus is when there is little choice and the user is not aware of what there is; the niche of command languages is when there are many choices, and the user is familiar with them. Thus, for example, the selection of programming actions on a time-sharing system is usually made by experienced users from a wide choice; it is normally done with named commands. Editors, used by less experienced people and with fewer choices, are sometimes command-based and sometimes menu-based (function keys). And data entry systems, which involve the least discretion and have the least trained users, are typically forms-filling.

This hypothesis arises from experiments with two retrieval systems at Bell Laboratories. In each case, users have two ways of getting information. In one access mode (menu) they merely respond to choices presented by the computer; in the other (keyword) they type a search request which the computer executes. In the "menu" systems, only a digit or two is typed to pick from displayed items, whereas the "keyword" systems require typing words not shown on the screen. In one experiment, an on-line public access catalog was installed in the library at Murray Hill in May 1981.[6,7] Either keyword access or menu access to book citations is possible. In the other experiment, an AP (Associated Press) wire was brought into one of our computers and users could either look at the twenty most recent news stories, or they could set up a profile to select stories of interest.[8] The next four sections of this paper describe the two systems: first how they work and then how they are used. The final section summarizes, giving general comments about keywords and menus.

© 1983 ACM 0-89791-107-5/83/006/0130 $00.75

2. An On-line Public Access Catalog: Description

The Bell Laboratories library has a collection of about 60,000 different books, with a catalog available in machine-readable form. We put two computer terminals in the library, connected to a pair of information retrieval programs. Although there are other on-line catalogs,[9,10,11] this system is unusual in offering the users a choice of methods. Users approaching the terminals are first asked to pick the menu or keyword system. During the experiment only a small card explained the systems: most instruction was given on-screen during the session. Both systems will now be explained briefly.

The keyword system uses coordinate searching. The user types any number of words, which may be author, title, subject heading, or other terms (publisher, date. etc.). The system then finds books with as many terms as possible. Some examples follow. In these examples, what the user types is shown in italics. Thus in the first example the user typed *chocolate* and later responds to the prompt "one book" with *1*.

```
chocolate
1 items (1/1)
 1: Technology of chocolate.
Which book? 1
%T Technology of chocolate.
%A Kempf, N.W.
%I Oak Park, Ill., Mfg. Confectioner
%D 1964.
%P 118p.
%# 663.92/K32
%S Chocolate
%Z Copies at: PR
```

As mentioned, the system does not care whether words are title, author, or subject heading. Another example:

```
butter pecan
2 items (1/2)
 1: Peanut butter and jelly guide to computers.
 2: Neuropsychology; the study of brain and behavior.
Which book? 2
%T Neuropsychology; the study of brain and behavior.
%A Butter, C.M.
%I Brooks-Cole.
%D 1968.
%P 211p.
%# 152/B98
%S Nervous system
%S Psychology, Physiological
%Z Copies at: HO MH
```

Note that given a two word query in which only one of the words appeared in the Bell Labs catalog, the system automatically finds the items containing the useful word. If more than one word appears, it finds as many as possible:

```
salton kochen information retrieval
Postings: salton: 4; kochen: 6; information: 2237-C;
        retrieval: 347
6 items (3/4)
 1: Theory of indexing
 2: Principles of information retrieval
 3: SMART retrieval system: experiments in automatic d
 4: Automatic information organization and retrieval.
 5: Growth of knowledge; readings on organization and
 6: Some problems in information science.
```

In this case six books have three of the four terms, as indicated by the numbers in parentheses. The complete citation for any could be obtained as above, e.g.:

```
Which book? 1
%T Theory of indexing
%A Salton, G.
%I SIAM
%D 1975.
%P 56p.
%# 029.5/S17
%O {note1} An outgrowth of the Regional Conference
on Automatic Information Organization and Retrieval,
held July 1973, Columbia, Missouri
%S Automatic indexing
%S Information storage and retrieval systems
```

Other facilities of the keyword search system include:

1. A suffixing algorithm that automatically removes a few common suffixes, including *-s*, *-ing*, *-ed*, *-ly* and *-es*. This is done merely on the basis of the apparent form of the word, without a dictionary. A short table of final letter strings tells the program when to leave a final *-e*; for example, it leaves a final *e* on *danced* but not on *treated*. It gets *created* wrong; more seriously, it does not handle *-ation* (since we know no rules or short tables to explain that *determination* should become *determine* but *termination* should become *terminate*). In practice no one complains about the suffixing.
2. The ability to add words to an existing query. If you preface a query with **&** it is assumed to supplement a previous word list. In addition, if you merely give only the **&** with no word list, the program suggests words you might add. This is done by looking at the previously retrieved titles, and listing frequent words in those titles which appear to be information-bearing. Information-bearing words are those whose occurrences are unevenly distributed by Dewey classification. Typical words flunking this test are *McGraw* or *1980*. Syntactic function words, of course, have been removed earlier.
3. A date range may be searched: the keyword string 1973-1980 matches any year in the interval. This is not quite the same as 1973 1974 1975 ... 1980 since the written out query will count several matches for a book with several dates (e.g. a multi-volume work) whereas the 1973-1980 notation is only one search term.

The system is implemented on a PDP 11/70 running the UNIX† operating system.[12] It uses inverted files stored with B-trees,[13] and normally replies in less than five seconds.

The alternative menu system is based on the Dewey Decimal classification hierarchy. The first display is always:

```
0 000    Generalities, Computing (280 groups)
1 100    Philosophy (87 groups)
2 200    Religion (5 groups)
3 300    Social sciences (306 groups)
4 400    Language and languages (71 groups)
5 500    Science (1063 groups)
6 600    Technology (Applied Science) (1175 groups)
7 700    Arts (64 groups)
8 800    Literature (8 groups)
9 900    History and Geography (69 groups)
```

and the user chooses down which branch of the hierarchy to continue the search. The focus of the Bell Laboratories library is obvious from the number of groups in each main heading. Suppose the user now types *5*:

† UNIX is a Trademark of Bell Laboratories.

```
1 510       Mathematics (242 groups)
2 520       Astronomy (68 groups)
3 530       Physics (248 groups)
4 540       Chemistry (354 groups)
5 550       Geology (72 groups)
7 570       Nature (38 groups)
8 580       Botany (7 groups)
9 591.1     Sense organs (13 groups)
```

Typing *3* now produces:

```
1 531       Mechanics (20 groups)
2 532       Fluid mechanics (15 groups)
3 533       Air flow (11 groups)
4 534       Acoustics (15 groups)
5 535       Optics (36 groups)
6 536       Heat transfer (23 groups)
7 537       Electricity (25 groups)
8 538       Magnetism (15 groups)
9 539       Atomic physics (50 groups)
```

At this point if the user descends one more level by typing *4* there are actually some books, as well as just further subdivisions:

```
b 534       Acoustics (124 books)
1 534.1     Sound (6 books)
2 534.2     Sound propagation (2 groups)
3 534.3     Music--Acoustics and physics (28 books)
4 534.4     Acoustics (2 books)
5 534.5     Ultrasonics--Applications (56 books)
6 534.6     Sound (1 book)
7 534.7     Phonetics (1 book)
8 534.8     Acoustical engineering (3 groups)
```

Among the features of the menu system are:

1. You can jump in at any point: if you know a particular Dewey number you can skip the introductory frames (almost no user of our library knows the system well enough to take much advantage of this).
2. Subdivision by date: If, when you reach a particular category, there are too many books in it (some categories have over 500 books) a menu by publication date is offered.
3. Locally determined category names: The labels on each classification number are not the "official" labels from the published classification, but the most frequent subject heading assigned to any book in that category by the catalogers at Bell Laboratories (a suggestion of Ian Ross of our library).

The catalog is very deep; there are over 3000 different Dewey categories with books in them at Bell Laboratories, and to reach the bottom involves ten decisions:

```
b 621.3815304 Transistor circuits (131 books)
1 621.381530401 Electronic circuits--Data processing (1 book)
2 621.381530402 Semiconductor circuits--Handbooks (2 books)
7 621.381530407 Transistor circuits--Programmed
                 instruction (4 books)
```

The average book is about five levels down; unfortunately that means that it may not be at all obvious on the top level where in the menu a particular subject will be found. For example, psychology is neither social science (300) nor science (500); it is under philosophy (one must remember that Dewey worked before William James, let alone Freud). Rearranging the categories is not likely to help, however; with 3000 categories and only 10 at top level, something must be buried.

Perhaps the most efficient way to do a subject search is to start with a keyword search to locate the correct category, and then browse through the classification. Suppose, for example, one wants to find Pascal manuals. Searching for *Pascal* and *manual* finds only four books with both keywords:

```
Postings: pascal: 72; manual: 1531-B; (B)
1: Stanford Pascal verifier user manual (2/2)
2: PASCAL language manual (2/2)
3: PASCAL user manual and report (2/2)
4: PASCAL: user manual and report (2/2)
5: Pascal programming structures for Motorola micropr (1/2)
6: Introduction to PASCAL (1/2)
7: Problem-solving principles: programming with PASCA (1/2)
8: UCSD PASCAL: a beginner's guide to programming (1/2)
9: Oh! Pascal! (1/2)
10: PASCAL: program development with ten instruction P (1/2)
```

Note that titles like *Introduction to Pascal* and *UCSD Pascal* do not contain the word *manual*. By printing one of the manuals, however, the classification number for these books can be found (here book 4 is shown):

```
%T PASCAL: user manual and report
%A Jensen, K.
%A Wirth, N.
%I Springer
%D 1974.
%P 170p.
%# 001.6424P27/J54
%S PASCAL (Computer program language)
%O {title added1} Lecture notes in computer science 18
%Z Copies at: PY
```

By typing *m* at this point, the user can switch to the menu system; the system chooses the category number of the book just displayed (001.6424P27) which yields a great many more books:

```
b 001.6424P27 PASCAL (Computer program language)
                      (52 books)
0 date 1982     (6 books)
1 date 1981     (16 books)
2 date 1980     (12 books)
3 date 1979     (5 books)
4 date 1978     (6 books)
5 date 1976     (4 books)
6 date 1975     (2 books)
7 date 1974     (1 book)
```

And now picking a date range (here 0) gives a selection of some of them:

```
 0 Introduction to PASCAL
 1 UCSD PASCAL: a beginner's guide to programming mic
 2 PASCAL: program development with ten instruction P
 3 BASIC programmer's guide to Pascal.
 4 Problem-solving principles: programming with PASCA
 5 Oh! Pascal!
Which book? 5
%T Oh! Pascal!
%A Cooper, D.
%A Clancy, M.
%I Norton.
%D 1982.
%P 476p.
%# 001.6424P27/C768o
%S PASCAL (Computer program language)
%Z Copies at: HO-2 MH PY WH WB CP MC
```

Once in the menu system, of course, one can use the usual commands to move up or down in the hierarchy and browse; for example, going up one level brings you to the next superordinate category (variants of Pascal) and another level up brings you to "programming languages whose names begin with P":

```
1 001.6424P15 PAL (Computer program language)
2 001.6424P27 PASCAL (Computer program language)
6 001.6424P69 PL/1 (Computer program language)
7 001.6424P70 PL/360 (Computer program language)
8 001.6424P84 PORTAL (Computer program language)
```

Most users, unfortunately, were not familiar with all of the ramifications of the systems. Since we wanted to measure the users' opinions of the two search schemes, not our description of them, we avoided long manuals and provided on-line instruction after the users had entered the search system. One thing we learned is that people often didn't read our on-line instructions. Fortunately, the search system was designed so that most simple things people tried worked; but this leaves them unaware of the more complex features.

3. An On-Line Public Access Catalog: Usage

For the first seven weeks the system ran, we watched users carefully. Full details are in our previously cited paper. During this period over 900 people engaged in over 3000 sessions with the system. About 80 sessions occurred each working day. We have both objective measures of preference, based on what the users did, and subjective measures, based on their answers to questions asked by the computer during their sessions. We dropped from our data all searches by the authors, by the library staff, and by users who did not identify themselves, plus all sessions in which no searches were done.

This left 1952 searches. Keyword searching was preferred: keyword searches were 79% of the total and menu searches 21%. Keyword searches were preferred initially (71% of first searches were keyword) and even more after experience (84% of searches after the first are keyword searches).

One would expect known-item searchers (59% of our sample) to prefer the keyword system, and browsers the menu system; but although browsers did fewer keyword searches (76% instead of 82%) the preference is small and gets smaller with experience. Table I shows the kinds of searches done, reporting the percentage of searches that were keyword searches for different kinds of users (browsing or known-item, first try or after experience with the system).

Table I

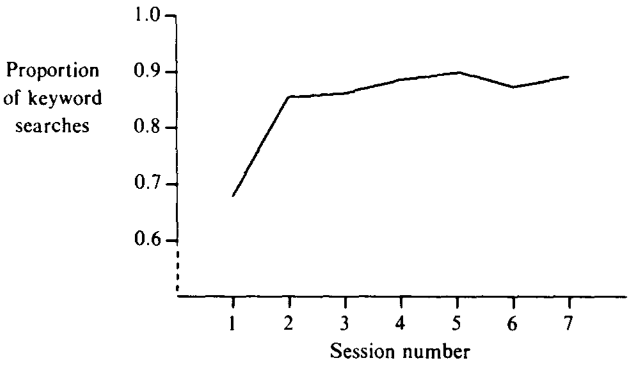
Percentage of keyword searches in sessions.

Was user looking for some particular book?	First Search	Later Sessions	All Sessions
Yes	75.7	86.1	82.2
No	64.6	82.0	76.4
All	71.3	84.4	79.8

Of the 208 people who did at least one keyword and one menu search, in their next session they do 84% keyword searches. As before, there is a preference (but smaller) for the known-item searchers to use keywords: 87% vs. 80%. Figure 1 shows the increase in keyword searching as people gain experience. It includes data from the 51 people who used the system at least seven times (in seven weeks). Their initial use is the same as the 1 or 2-time users; in Figure 1 we show their proportion of keyword searches vs. session number. All the increase comes after the first session; they find out quickly what the two systems are like.

Figure 1

Proportion of keyword searches vs. session number for frequent users (≥7 sessions)



Why were keyword searches preferred? Well, they worked better: twice as many keyword searchers found what they wanted. This is true for both the "known-item" searchers (users who claim they have a particular book in mind) and "browsers" (users who say they are just looking for information on a subject). 65% of the known-item keyword searchers found their book, while only 30% of menu users found it. Among the browsers, keyword users said they found books 69% of the time, and menu users reported only 36% success. We can confirm these subjective answers by noticing that 24% of keyword users failed to look at any full citations (implying that they didn't find any promising titles) while 55% of the menu users read no citations.

Finally, let's look at the responses to questions on preference. We asked users to compare the computer system they had used to the old printed catalog. Keyword users reported the computer as easier to use, more complete, and less time-consuming. (There may be some wishful thinking here; the average session time is 4 minutes. Presumably they are getting more done than they used to). All of these ratings for keyword users indicate more satisfaction than the menu users, although the menu users do think the computer is easier and faster than the old way (but not as complete). These results are shown in Table II. The questions asked for a three-way choice: in the table "more" is scored as 1, "same" as 2, and "less" as 3. The *F* tests are for the first two columns of the table (keyword against menu).

Table II
Subjective evaluations compared to manual search

	Keyword system (n=959)	Menu system (n=147)	Both in one session (n=214)	*F* tests *p* levels
How easy?	1.6	1.9	1.7	8.89, $p<.001$
How complete?	1.7	2.2	1.6	27.18, $p<.001$
How time consuming?	2.4	2.2	2.3	6.63, $p=.001$

Answers for people who used only keyword and people who used both methods in one session are not significantly different.

4. News Wire Access: Description

Every day the Associated Press sends us about 200,000 words of news service. We have again provided different ways to access this data, again unlike most news retrieval systems.[14] First, logged-in users can examine a display of recent news items, doled out twenty stories at a time, most recent first. Secondly, keyword profiles or keyword searching can be used. As stories arrive, roughly every three hours the computer matches the profiles against the stories and processes the stories as requested by the users (typically mailing the stories to the requestor). Again, we wished to compare the preferences of the users.

The typical display from the interactive menu system looks as follows:

```
 1: (a0533) ChryslerStrike       11: (a0523) NewsAdvisory
 2: (a0532) FleshEaters          12: (a0522) Focus-Byrne
 3: (a0531) RevengeKilling       13: (a0521) FocusAdvisory
 4: (a0530) GarfieldBooks        14: (a0520) China-Soviets
 5: (a0529) PurringGift          15: (a0519) OldCottonwood
 6: (a0528) MenonObit            16: (a0518) OldCottonwood
 7: (a0527) PastPages            17: (a0517) Pentagon-Israel
 8: (a0526) LateNewsAdvisory     18: (a0516) PollFacts
 9: (a0525) TehranAirport        19: (a0515) SocialSecurityPoll
10: (a0524) Bush                 20: (a0514) Brites
```

At this point one may type, for example *5* to see the "PurringGift" story:

```
r a AM-PurringGift    11-21 0287
^AM-Purring Gift,300<
^Injured Youngsters Get Unusual Present<
     IOWA CITY, Iowa (AP) _ Two brothers critically
injured in a house explosion were worried about the
fate of the two family cats, but a tape recording brought
by the state police and Red Cross calmed the kids down
right away.
     It was one of the cats, purring contentedly.
     "I guess they were tickled to death," American
Red Cross volunteer George Wheeler said. The boys also
got to look at their pets _ Wheeler's wife, Linda, took
some pictures.
```

Or the user can type *m* to see another batch of stories (slightly older; a story typically arrives every 3-4 minutes, so twenty stories cover about an hour of news).

```
 1: (a0513) PopeAdvisory         11: (a0501) NationalBriefs
 2: (a0512) People               12: (a9998) NewsDigest
 3: (a0511) RestaurantFire       13: (a9999) NewsDigest
 4: (a0510) Nimmo                14: (a0500) ForeignBriefs
 5: (a0509) Poland               15: (a9501) GlobalTemps
 6: (a0508) TyreExplosion        16: (a0662) Advisory
 7: (a0505) Soviets-US           17: (a0661) Iran-Iraq
 8: (a0504) China-Cambodia       18: (a0660) Iran-Iraq
 9: (a0503) Pope-Sicily          19: (a0658) Bomb
10: (a0502) Turkey-Labor         20: (a0656) ShortObit
```

At each menu, *m* provides another twenty stories; a number prints the corresponding story; and *t* goes back to the top. It is also possible to search for stories with particular words in them while logged in. Typing, for example, *s dog* will print a similar display but only of stories containing the word *dog*. However, keyword searching is much more commonly done through our SDI facility. A particular file, if present in the user's home directory, is taken as a search strategy and the commands in it are matched against stories several times a day. For example, here is a profile to match items of interest to Bell Laboratories:

```
FCC
bell laboratories
bell system
telephone , industry
telephone . regulation
telephone. deregulation
western electric
american telephone
communication . regulation
communication . deregulation
```

The syntax is simple. Each line is a matching request; if any matches occur the story is selected. A single word (e.g. FCC) matches if it appears. An unpunctuated phrase (e.g. *bell system*) must appear with the words adjacent and in that order. Words separated by commas must appear in the same sentence; words separated by periods may appear anywhere in the story (but all must be found). Preceding a word with ! negates it (e.g. *!chocolate chip* to try for stories on integrated circuits).

The user can specify a command line to say what is to be done to a matching story. The most common entries are either (a) squirrel away in a directory or (b) mail it to a user. Most other entries turn out, on examination, to be a sophisticated version of one of these. If the stories are stored in a directory, the same scanning command that runs on the normal full wire can be used to examine that directory. Since we routinely discard stories after 24 hours, anyone wishing to read over a subject at leisure must save the stories himself. There is also software to print little booklets of accumulated stories, but it is not in demand; the saying that yesterday's news is like yesterday's fish really seems to be true.

5. News Wire Access: Usage

We find here that about the same number of stories are read per day from each source. Each day (except Sunday) the AP sends an average of 460 stories. During a typical workday forty different people read 840 stories interactively. There are about 50 keyword profiles, and 40 hit at least one story during a normal day. A total of 550 stories are sent by the SDI system each day. Thus here, even assuming every story mailed out is really read, menu-distributed stories run ahead of keyword-picked stories by 50%.

The reader may immediately object: but the menu system is interactive and the command system is run in background. However, keyword searching is possible in the interactive mode: it is almost unused there. During October 1982 an average of 839 stories a day were read (excluding weekends), with people asking to read the news 93 times per day. There were only 25 searches done per day; in other words only about 1 time in 4 that someone was reading the news did they do even one keyword search.

The users of this system are not, as before, drawn from the whole Murray Hill staff; access to the system receiving the wire is limited, and the user population is between 100 and 150 people. Thus the 90 uses a day of the AP wire, compared with the 50 uses a day of the library terminals, is really a factor of 40 higher usage (since the full technical population on site is between 2000 and 3000 people). Of course, even in our highly technical population, the AP wire is still more interesting on average than the library catalog (and also changes more). The average user spends 10 to 15 minutes at a time reading the news, and reads about 8 stories. There is little agreement among people as to what stories are read; in fact we have been frustrated in attempts to cluster the stories on the basis of readership. The statistics have been fairly steady over time; people are not drifting to or from the menu system.

6. Discussion.

Returning to the original question, when are menu interfaces appropriate and when are keyword or command interfaces appropriate? There are, of course, many differences between our two retrieval systems.

1. The library users are more likely to be seriously interested in something; the news wire readers are more likely to be idly curious. Personal inspection of the news requests suggests that the SDI keyword system is more likely to be used for serious purposes and the menu system is more likely to be read frivolously.
2. The library data base is bigger: it contains about 60,000 book citations, and totals about 18 megabytes. Per day, the AP wire has about 500 stories and amounts to 1 megabyte.
3. The library data base is static, and the AP wire is constantly changing.
4. The readers of the AP wire are more likely to be computer programmers than the users of the library catalog.
5. The library has a much more complex structure than the AP wire; the menu is deep with buried items. No user is familiar with the whole structure. On the wire there are no subordinated menus; the stories are simply presented as is.
6. There is much more foreknowledge of the library collection than there is of the news service; many people are quite familiar with what is in the library and are merely using the catalog to find an exact shelf location. It is less common for people to read the news looking for a specific story (although it does happen; the days that Reagan and Sadat were shot both had many people logging in to find out the latest facts).

We suspect that the last two items are the most important. The point of a news service is that you don't know what is coming next (although the existence of stories on particular subjects, e.g. election returns or trips by politicians, may be predictable). The computer must tell you something about what it has for you to know what to ask for. By contrast, in the library, our users know that there are no books on art or religion; there is little point in the computer telling you this.

One can consider that there is a bargaining process going on: the computer has some data, you have some needs, and you have to get together. It would not be efficient for either side to do a complete memory dump at the other. The fastest way to narrow down the options is for the side which is least known to the other to describe its information. This corresponds to menu-type systems when users know little about their choices, and command-type systems when users are quite familiar with the data base. And if the data base is very complex, a menu is inefficient at getting information from a user, since each step provides only a 1 in 10 or 20 choice. An English word can be one of thousands. Thus the more information needed from the user, the greater the advantage of words.

In conclusion, we have compared keyword and menu access to two textual data bases. We suggest that keyword access is preferable when the data base is complex and predictable, and that menu access is preferable when the data base is simple and unpredictable. A simple and predictable data base is probably boring, and a complex and unpredictable data base is probably unmanageable.

References

1. D. M. Krentz, "On-line searching: specialist required," *J. Chem. Inf. Comp. Sci.* **18**(1), pp. 4-9 (1978).
2. B. Shneiderman, "Improving the human factors aspect of database interactions," *ACM Trans. on Database Systems* **3**(4), pp. 417-439 (Dec. 1978).
3. B. Shneiderman, *Software Psychology,* Winthrop, Cambridge, Mass. (1980). See page 241.
4. P. Reisner, "Human factors of database query languages," *Computing Surveys* **13**(2), pp. 13-32 (1981).
5. L. A. Miller and J. C. Thomas, "Behavioral issues in the use of interactive systems," *Int. J. Man-Machine Studies* **9**, pp. 509-536 (1977).
6. V. J. Geller and M. E. Lesk, *How Users Search: A Comparison of Menu and Attribute Retrieval Systems on a Library Catalog*, Bell Laboratories internal memorandum (1981).
7. V. J. Geller and M. E. Lesk, "An On-Line Library Catalog Offering Menu and Keyword User Interfaces," *Proc. National On-Line Meeting*, New York, (to appear) (1983).
8. M. E. Lesk, "Combining Data Bases," *Proc. AFIPS Office Automation Conference*, San Francisco, pp. 415-426 (1982)
9. Christine Borgmann, "Monitoring of Users of the Ohio State Library Online Catalog," *Proc. ASIS Annual Meeting*, Columbus, Ohio **19**, pp. 35-36 (1982).
10. V. Veneziano, "Library automation: data for processing and processing for data," pp. 109-145 in *Annual Review of Information Science and Technology* **15**, ed. M. Williams, Knowledge Industry Publications (for the American Society for Information Science), Washington (1980).
11. L. Arret, "New catalog/new librarian? Reference at LC," *RQ* **19**(1), pp. 44-51 (Fall 1979).
12. "Special Issue on the UNIX System," *Bell System Technical Journal* **57**(6) (July-August, 1978).
13. P. J. Weinberger, *UNIX B-trees*, Bell Laboratories internal memorandum (1981).
14. Thomas H. Hagan, "News Retrieval Services — Growing, But Where Are They Heading?," *Online Review* **3**(3), pp. 247-252 (1979).