Report No. UIUCDCS-R-82-1081

NOTESFILE REFERENCE MANUAL
(abridged)

by

Raymond B. Essick IV
Rob Kolstad

February 14, 1983
(Revised: October 20, 1985)
(Printed: August 9, 1987)

DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
1304 W. SPRINGFIELD AVENUE
URBANA, ILLINOIS 61801-2987

Supported in part by NASA Project NAS-1-138

1 Introduction.

Notesfiles support computer managed discussion forums. Discussions can have many different purposes and scopes: the notesfile system has been designed to be flexible enough to handle differing requirements.

Each notesfile discusses a single topic. The depth of discussion within a notesfile is ideally held constant. While some users may require a general discussion of personal workstations, a different group may desire detailed discussions about the I/O bus structure of the WICAT 68000 (a particular workstation). These discussions might well be separated into two different notesfiles.

Each notesfile contains a list of logically independent notes (called base notes). A note is a block of text with a comment or question intended to be seen by members of the notesfile community. The note display shows the text, its creation time, its title, the notesfile's title, the author's name (some notesfiles allow anonymous notes), the number of ''responses'', and optionally a ''director message''. Each base note can have a number of ''responses'': replies, retorts, further comments, criticism, or related questions concerning the base note. Thus, a notesfile contains an ordered list of ordered lists. This arrangement has historically been more convenient than other proposals (e.g., trees were studied on the PLATO (trademark of Control Data Corporation) system).

The concept of a notesfile was originally implemented at the University of Illinois, Urbana-Champaign, on the PLATO system. The UNIX (trademark of Bell Laboratoris) notesfile system includes these ideas with adaptations and enhancements made possible by the UNIX environment.

The UNIX notesfile system was designed and implemented by Ray Essick at the University of Illinois, Urbana-Champaign. It provides users with the abilities to read notes and responses, write notes and responses, forward note text to other users (via mail) or other notesfiles, save note text in their own files, and sequence through a set of notesfiles seeing just new text. Each notesfile has a set of ''directors'' who manage the notesfile: they delete old notes, compress the file when needed, grant and restrict access to the notesfile, and set different notesfile parameters (e.g., title, ''director message'', policy note, whether notes' authors can be anonymous). Some notesfiles contain correspondence from other computers. Like the UNIX ''USENET'', notes and responses are exchanged (often over phone lines) with remote machines. The notesfile system provides automatic exchange and updating of notes in an arbitrarily connected network.

This document details the use of notesfiles from invocation through intersystem notes exchanges. The last chapter summarizes the entire set of commands for easy reference. An appendix contains detailed checklists for the installation of a notesfile system.

2 Using Notesfiles.

The notesfile system is invoked with a single command line. Most notesfile commands require only a single character (like the vi editor). Those that require more than one character are terminated by a carriage return.

2.1 Invocation.

Invoke the notesfile system with:

notes [-sxi] [-a subsequencer] [-t termtype] [-f nfile] [topic1] [topic2 ...]

The topic list (e.g., topic1) specifies the notesfiles to read. Invoking the notes system with NO arguments yields a list of some available topics. When more than one topic is specified, the user encounters each topic sequentially (i.e., topic2 is entered upon completion of topic1).

The -s switch activates the ''notesfile sequencer'' which is discussed in section 2.8. Specify ''-x'' to use the extended sequencer. The ‘‘-i’’ flag selects yet another sequencing mode. The ‘‘-a’’ option specifies a particular subsequencer. This allows several users sharing a signon to maintain their own sequencing timestamp information.

The -t option directs the notesfile system to use ‘‘termtype’’ as the user’s terminal type, overriding the TERM shell variable.

The -f option directs the notesfile system to read the contents of the file ‘‘nfile’’ for a list of notesfiles to read. See section 2.3 (‘‘The -f Option’’) for more information on the format of this file.

2.2 Notesfile Names and Wildcards.

Notesfiles can be specified in several ways. The most common way is to merely give the name of the notesfile, such as ‘‘general’’. These notesfiles typically reside in the directory ‘‘/usr/spool/notes’’. Notesfiles may also be specified by their complete pathname; thus you could also refer to ‘‘general’’ by its full pathname ‘‘/usr/spool/notes/general’’. Using complete naming, notesfiles can be placed anywhere in the filesystem. This allows ‘‘private’’ notesfiles to be stored in personal directories.

The notesfile system supports pattern matching for names in the same manner as the shell. By using the shell meta-characters ‘‘*’’, ‘‘?’’, ‘‘[’’ and ‘‘]’’, the user can specify a number of notesfiles with a single entry. To read all the notesfiles that pertain to unix, enter the following line (the quotes are required to protect the metacharacters from interpretation by the shell):

```
notes ‘‘*unix*’’
```

There are several ways to read the notesfiles test1, test2, test3 and test4:

```
notes test1 test2 test3 test4
notes ‘‘test?’’
notes ‘‘test[1234]’’
```

Entries can also be eliminated from the list of notesfiles to look at. By prefixing a notesfile name (possibly containing wildcard characters) with a ‘!’, the notesfiles are excluded from the list to be examined. If one wished to look at all of the ‘‘test’’ notesfiles except test3, one could specify:

```
notes ‘‘test?’’ !test3
```

If you use the c shell, you will have to escape the ‘!’, the history character:

```
notes ‘‘test?’’ \!test3
```

These features are available from the normal entry (notes) and the automatic sequencer entry (see section 2.8). Most notesfile programs recognize this format. Among those which do not are programs which must receive exactly one notesfile name.

2.3 The -f Option.

The ‘‘-f’’ option of the notesfile system specifies a file of notesfile names to read. The file consists of lines containing notesfile names:

```
nfgripes
net.unix-wizards
net.general
fa.telecom
```

The names start at the left margin; they are indented here for readability. Wildcard characters (‘‘*’’, ‘‘?’’, ‘‘[’’, and ‘‘]’’) are acceptable in this context. Full names such as ‘‘/usr/spool/notes/general’’ are also accepted. Notesfiles can be eliminated through the ‘‘!’’ feature as described in section 2.2. The sequencer mode can be changed (see section 2.8) by inserting a line of the form:

```
-s
```

Again, this starts at the left margin. The ‘‘s’’ can be any of: ‘‘s’’, ‘‘x’’, ‘‘i’’, or ‘‘n’’. When a line of this form is read from the file, the sequencer mode is set to the corresponding mode: The normal ‘‘s’’equencer, the e‘‘x’’tended sequencer, the ‘‘i’’ndex sequencer, and ‘‘n’’o sequencer.

To always enter nfgripes, micronotes, and bicycle while only entering the networked notesfiles ‘‘net.*’’ when new notes are present, one might use ‘‘notes -f myfile’’ with this ‘‘myfile’’:

```
-x
nfgripes
micronotes
bicycle
-s
net.*
```

2.4 General.

Almost all notesfile commands consist of exactly one character (no carriage return). Only commands that are longer than one character require a terminating carriage return (currently, choosing a note to read is the only non-single character command).

The commands were chosen to be easy to remember. Upper case forms of commands usually function like their lower case counterparts but with some additional feature or power (i.e., ‘‘w’’ writes a response, ‘‘W’’ includes the current displayed text in the response).

Some commands are available almost everywhere in the notesfile system. These include those for help, exiting, forking a shell, and making a comment for the suggestion box.

2.4.1 Help.

Typing ‘‘?’’ anywhere will list the available options in an abbreviated format.

2.4.2 Exiting.

Type ‘‘q’’ (‘‘quit’’) to leave the current notesfile. Capital ‘‘Q’’ leaves the current notesfile and refrains from entering your last entry time into the sequencer table (see section ‘‘The Sequencer’’). The notesfile system proceeds to the next topic in the invocation list. The ‘‘k’’ and ‘‘K’’ keys function exactly as ‘‘q’’ and ‘‘Q’’.

Use control-D (‘‘signoff’’) to leave the notesfile system completely (without updating entry time information). The ‘‘z’’ command (which functions only when reading notes or responses or when on the index page) behaves similarly to control-D: the user exits the notesfile system immediately, but unlike control-D, updates the entry time information for the current notesfile.

2.4.3 Shells.

Fork a shell at any time by typing ‘‘!’’ (just like many other Unix programs).

2.4.4 Comments & Suggestions.

Type capital ‘‘B’’ (‘‘suggestion Box’’) while on the index page or reading notes to make a comment or suggestion about the notesfile program. Your suggestion will be stored in another notesfile reviewed frequently by the notesfile system manager.

2.5 The Index Page.

When the notes system is invoked without the -s option, the user sees an index of the most recent notes. A sample page is shown below:

Workstation Discussion		2:03 pm Jan 4, 1982	
12/9/81	2 Stanford SUN	4	horton
	3*WICAT 68000		kolstad
	4 M68000	1	horton
	5 Dolphin	3	duke!johnson
12/10	6 CDC Standalone	1	smith
	8 IBM Personal Computer		henry
	9 Personal computers harmful?	8	Anonymous
	10 Ethernet interfaces 3 mhz?	23	essick
	11 Requirements for uiucdcs	10	botten
1/1/82	12 Happy New Year!	5	mjk

The upper left corner shows the notesfile’s title. In this example, the notesfile discusses personal workstations. The current time and date are displayed in the upper right corner. Approximately ten note titles are displayed (if available). More notes are displayed on longer screens (such as the Ann Arbor Ambassador). Each note is displayed with its date (if different from the previous date), note number, title, number of responses (if any), and author. The first note above was written by user ‘‘horton’’ on December 9th, is entitled ‘‘Stanford SUN’’ and has four responses. Note 7 has been deleted for some reason (by either its author or a notesfile director). Note 5 was written by user ‘‘johnson’’ whose signon resides on the ‘‘duke’’ system. Note 9 was written by an author who preferred to remain unidentified. Notes with director messages (sometimes denoting importance) are displayed with a ‘‘*’’ next to the note number (see note 3 above).

From the index page the user may:

- Scroll the index forward or backward.
- Read a note.
- Write a note.
- Go to the next unread note.
- Search for notes or responses after a specific date/time.
- Search for keywords within notes’ titles.
- Search for notes/responses by a specific author.
- Go to another notesfile.
- Consult the notesfile’s archive.
- Read the policy note.
- Check on anonymous and networked status.
- Register a complaint/suggestion about notesfiles.

- Fork a shell.
- Exit the notes program.
- Invoke notesfile director options (if the user is a director).

2.5.1 Scrolling the Index Page.

Scroll the index page by:

+, <return>, <space>	forward one page
*	forward to the most recent page (* is multiple +'s)
-	backward one page
=	backward all the way (= is multiple -'s)

2.5.2 Choosing Notes & Responses.

While on the index page, choose a note to read by typing its number followed by a carriage return. (This is the only command that requires a carriage return after it.) Usually the space bar is used to scan text. To skip to a particular note or response, use the features below.

While reading a note, ‘‘;’’ or ‘‘+’’ advances to the first response of the note. The next note is displayed if there are no responses. The number keys (‘‘1’’, ‘‘2’’, ... , ‘‘9’’) advance that many responses. If there are fewer responses, the last response is displayed. The return key skips the responses and goes to the next note. Press ‘‘-’’ or backspace to see the previous page of the current note; if the page currently displayed is the first, the notesfile program displays the first page of the previous note.

While a response is on the screen, the ‘‘;’’ and ‘‘+’’ keys display the next response. As with reading a note, if there are no further responses these keys advance to the next note. The number keys (‘‘1’’, ... , ‘‘9’’) will advance the appropriate number of responses. If there are fewer responses, the last response is displayed. The ‘‘-’’ or backspace keys display the previous page of the current response. If the current page is the first page of the response, these keys display the first page of the previous response. Enter ‘‘=’’ to see the base note of the current note string. Press the return key to proceed to the next note.

2.6 Notes & Responses.

2.6.1 Reading Notes.

After selecting a note from the index page (or entering the notesfile with your ‘‘sequencer’’ on), the note is displayed. A sample display is shown below:

```
Note 15             Workstation Discussion          2 responses
horton                WICAT 150          4:03 pm  Dec 11, 1981

Wicat System 150

8 MHz 68000, Mem. mgmt, Multibus architecture, 256k to 1.5 Mb RAM,16/32/64Kbyte EPROM,
10 ms interval timer, 2 RS232 (19.6k async, 56k sync), 16 bit parallel intelligent disk controller,
10 Mbyte winchester (5.25", 3600 rpm, access: 3 ms trk-trk, 70 avg, 150 max),
960Kb floppy (5.25", 300 rpm, access 10 ms trk-trk, 267 avg, 583 max)
Options: battery backed clock, graphics with touch panel, video disk control,
High Speed Serial Network Interface
Unix/V7 avail, Pascal, C, APL, ADA, Cobol, Fortran, Lisp, Basic, Asm
```

This is note number 15 in the ‘‘Workstation Discussion’’ file. User ‘‘horton’’ wrote this note at 4:03 pm on December 11th, 1981. Two responses have been written. The note’s title is ‘‘WICAT 150’’. If a director had written the note, the ‘‘director message’’ might have been displayed beneath the note’s title. Director’s notes sometimes contain important information or new policies.

Since notes and responses can each be up to 3 Mbytes long, the display routine breaks text into pages automatically. For all but the last page of a long note or response, the lower right corner of the display shows the percentage of the note that has been shown. For all but the first page of long text, the message ‘‘[Continued]’’ appears in the upper left portion of the display. Use the space bar to see the next page of a long note or response. When the last page is displayed, the space key functions as the ‘‘;’’ key: it proceeds to the next response. The ‘‘-’’ and backspace keys back up the display to the previous page. Only the first 50 pages of text are managed this way; typing ‘‘-’’ from the fifty-second page will return to the fiftieth page. The ‘‘=’’ key returns to the first page of the note.

While reading a note, it is possible to:

- Display the next, previous, or first page of the note.
- Write a response to the displayed note.
- Read next note or previous note.
- Read next unread response or note.
- Return to the index page.
- Skip to a given response.
- Delete the note (if you are its author or a file director).
- Edit the note’s title (if it is yours).
- Edit the note (if it is yours and there are no responses).
- Copy the note to another notesfile.
- Save the note in your file space.
- Mail the note to someone.
- Talk (‘‘write’’) to the author of the note.
- Search for keywords in note titles.
- Search for notes/responses by a particular author.
- Toggle the director message (if privileged).
- Fork a shell.
- Go to another notesfile.
- Make a comment or suggestion about notesfiles.
- Exit the notesfile program.

2.6.2 Reading Responses.

Response displays are similar to those of main notes with the exception that ‘‘Response x of y’’ replaces the note’s title. The first response to note 15 is shown below:

```
Note 15            Workstation Discussion
koehler              Response 1 of 2        11:53 pm  Dec 11, 1981

Does anyone have any insight about the relative speeds of the Winchester disks available
on these systems?  The previous disk seems to have track to track response times commensurate
with reasonably fast 8" floppies.  I wonder if some of the manufacturers are using disks that
will not meet reasonable specifications for response time for these kinds of applications.

On the other hand, with intelligent layout of file sectors, the I/O system
could romp and stomp on often used files...
          ====================================
```

The commands for manipulating the text of a long response are the same as those for looking at long notes.

Typing space will move to the next page. Typing ‘‘-’’ or backspace will display the previous page, within the same limitations as for reading notes (only 50 pages are kept). Press ‘‘=’’ to go back to the first page of the text.

The options available while reading responses include:

- Display the next, previous, or first page of the response.
- Go to a different response (usually the next one).
- Go to the next unread note/response.
- Reread the base note.
- Reread the previous note.
- Return to the index page.
- Copy the response to another notesfile.
- Mail the response to someone.
- Save the response in your file space.
- Talk to the response’s author.
- Write another response to the note.
- Search for keywords in note titles.
- Search for notes/responses by particular authors.
- Delete the response (if you are its author or a file director).
- Edit the response (if it is yours and there are no later responses).
- Fork a shell
- Go to another notesfile.
- Register a suggestion or complaint about the notesfile program.
- Exit the notesfile program.

2.6.3 Writing Notes & Responses.

Write new base notes by hitting ‘‘w’’ while reading the index page. The notesfile system will then invoke an editor (‘‘ed’’ by default; use either of the shell variables NFED or EDITOR to change it). After the prompt, compose the text you wish to enter, then write the text to the disk and leave the editor. The system will prompt you for various options if they are available: anonymity, director message status, and the note’s title.

To write a response to a note type ‘‘w’’ while that note or any of its responses is displayed. The same steps used to write a base note should then be followed.

2.6.4 Mailing Notesfile Text.

Both notes and responses can be mailed to other users (with optional appended text). The capital ‘‘M’’ (‘‘mail’’) command gives you the opportunity to edit the text then send it to anyone. Its inferior counterpart, ‘‘m’’, allows you to mail a message to anyone. To mail to the author of the text, use capital ‘‘P’’ (‘‘Personal comment’’) to send the text and your comments; use ‘‘p’’ for a simple letter.

To use a specific mail program, set the environment variable MAILER. If this is not set, a standard mail program is used.

2.6.5 Forwarding Text To Other Notesfiles.

There are several methods for forwarding text from one notesfile to another. Single notes or responses can be copied with the ‘‘c’’ or ‘‘C’’ command while entire note strings can be forwarded with the ‘‘f’’ and ‘‘F’’ commands.

The ‘‘f’’ (‘‘forward’’) command is given when a base note is displayed on the screen. When given, the ‘‘f’’ command causes the base note and all of its responses to be copied to another notesfile. The user is prompted for the destination notesfile. The copied note and all of the copied responses contain header information detailing their origin. Where ‘‘f’’ copies the note string without change, the ‘‘F’’ command allows the user to edit the text of the note and each response before inserting it into the target notesfile.

The ‘‘c’’ (‘‘copy’’) command prompts for a destination notesfile then copies the currently displayed note or response to the target notesfile. The user is allowed to choose between forwarding the note as a response or as a new base note. The ‘‘c’’ command does not give the user a chance to edit the text before inserting it in the new notesfile. The extended copying command ‘‘C’’ allows editing of the note text before it is copied to the other notesfile.

Both the ‘‘c’’ and ‘‘C’’ commands provide for the forwarded text to be entered as either a new note or as a response to an existing note. In the latter case, an index page is given to the user for choosing the appropriate note to which to respond.

2.6.6 Saving Text in Local Files.

The ‘‘s’’ (‘‘save’’) command appends the current displayed text to a file of your choice (which is created if not present). Notesfiles prompts for the file name; typing only a carriage return aborts the command -- no text is saved. Capital ‘‘S’’ appends the base note and all its responses. The number of lines saved and the name of the file written are printed when the command completes.

2.6.7 Deletion.

Capital ‘‘D’’ (‘‘delete’’) deletes a note or response if it is yours and has no subsequent responses. Notes already sent to the network can not be deleted by non-directors. Directors can delete any note or response with the ‘‘Z’’ (‘‘zap’’) command.

2.6.8 Online Communication.

Typing ‘‘t’’ (‘‘talk’’) attempts to page the author of the current displayed text. The Unix ‘‘write’’ command to him/her is issued if the author is local and non-anonymous. If the environment variable WRITE is defined, the program it specifies is used to write to the author.

2.6.9 Editing Note Titles.

While reading a base note, type ‘‘e’’ (‘‘edit’’) to change the note’s title (provided you are the author of the note or a notesfile director).

2.6.10 Editing Notes/Responses.

‘‘E’’ allows editing of the text of a note or response. It is not permitted to edit an article if it has subsequent responses or if it has been sent to the network. If the ‘‘later responses’’ are deleted, it is possible to edit the original text.

2.7 Other Commands.

2.7.1 Returning to the Index Page.

Type ‘‘i’’ (‘‘index’’) while reading notes or responses to return to the index page.

2.7.2 Searching Titles for Keywords.

While reading, you can search backwards for keywords appearing in note titles. Typing ‘‘x’’ (‘‘x is the unknown title’’) prompts for the substring to be found. Searching begins at the current note (or from the last note shown on the index page) and proceeds towards note 1. The search is insensitive to upper/lowercase distinctions. Use upper case ‘‘X’’ to continue the search. The search can be aborted by hitting the RUBOUT (or DELETE) key.

2.7.3 Searching for Authors.

The ‘‘a’’ command searches backwards for notes or responses written by a specific author. Notesfiles prompts for the author’s name. The ‘‘A’’ command continues the search backwards. The author name may be preceded by an optional ‘system!’. Abort the search by hitting the RUBOUT (or DELETE) key.

The entire name need not be specified when searching for articles by a particular author. Author searching uses substring searching. Searching for the author ‘‘john’’ will yield articles written by a local user ‘‘john’’, a remote user ‘‘somewhere!johnston’’, and any articles from the ‘‘uiucjohnny’’ machine. Author searching is case sensitive.

2.7.4 Stacking Notesfiles.

Sometimes it is useful to be able to glance at another notesfile while reading notes. Using ‘‘n’’, the user can save (stack) his current place and peruse another notesfile.

When on the index page or while reading notes/responses, type ‘‘n’’ (‘‘nest’’) to read another notesfile. Notesfiles prompts for the notesfile to read. If the notesfile exists, the place is marked in the old notesfile and the new one’s index is displayed.

Type any of the standard keys to leave the nested notesfile. Both ‘‘q’’ and ‘‘Q’’ leave the nested notesfile and return to the previously stacked notesfile. Control-d (‘‘signoff’’) causes the notesfile program to exit regardless of the depth of nesting.

Sequencing is turned off in the new notesfile regardless of its state in the old notesfile. The depth of the stack of notesfiles is limited only by the amount of memory available to the user.

2.7.5 Accessing Archives.

As notesfiles grow, it becomes impractical to keep every discussion. In some cases, the old discussions are deleted; other cases require these old discussions to be saved somewhere. Each active notesfile can have an archive notesfile. An archive notesfile contains the old discussions from the active notesfile.

The archive of an active notesfile is accessed by explicitly naming the notesfile (/usr/spool/oldnotes/micronotes for example) or through the ‘‘N’’ command from the active notesfile.

2.7.6 Policy Note.

A notesfile director can write an optional policy note to describe the purpose of a notesfile. Read the policy note by typing ‘‘p’’ (‘‘policy’’) from the index page.

2.8 The Sequencer.

Most users prefer to scan notesfiles and see only those notes written since their last reading. The notesfile ‘‘sequencer’’ provides this capability. It is activated by the ‘‘-s’’ option (‘‘sequencer’’) on the command line. When the sequencer is activated, the notesfile system automatically remembers the last time the user read notes in each notesfile. Subsequent entries to the notesfile can use the ‘‘last time’’ information to show only new notes and responses. If there is nothing new in a notesfile, the sequencer proceeds to the next notesfile specified in the command line.

The normal sequencer does not give the user a chance to read the notesfile if there are no new notes or responses; sometimes it is desirable to be able to do so. Use the ‘‘-x’’ option to enable the sequencer and enter the notesfile even if there are no new notes.

No keys need be pressed if there are no new notes in the entire list and the normal (‘‘-s’’) sequencer mode is selected. With the extended (‘‘-x’’) sequencer, the user must type ‘‘q’’, ‘‘Q’’, or control-d for each notesfile regardless of whether there are new notes.

The ‘‘-i’’ mode of sequencing is similar to the ‘‘-s’’ mode. Using the ‘‘-i’’ mode, notesfiles without new entries are passed over. The user starts reading on the index page of notesfiles which contain new notes.

2.8.1 Seeing New Notes and Responses.

The sequencer always shows the base note of a modified note string, whether or not is has been shown before, in order to establish the context of the new response(s). The ‘‘j’’ command skips to the next modified text (note or response).

If the rest of a particular note string seems uninteresting, skip to the next modified note string with the ‘‘J’’ (‘‘big Jump’’) command. This skips any new responses on the current note string. It is common to follow closely only a few note strings, skipping others using the ‘‘J’’ command.

The ‘‘last time’’ information is kept in a special file for each user. When the sequencer is enabled, the time for the notesfile is loaded into a variable and used to specify which notes and responses are new. If the sequencer is not enabled, this variable is initialized to January 1, 1970. The ‘‘j’’ and ‘‘J’’ keys use this variable to determine which notes and responses are ‘‘new’’.

If the sequencer is enabled, after exiting a notesfile the ‘‘last time’’ information is updated to the time that the user entered this notesfile. The entry time is used rather than the exit time to ensure that all notes are seen, including ones written during the just completed session. If the sequencer is disabled, the ‘‘last time’’ information is not modified. The ‘‘last time’’ information for a particular notesfile is updated as that notesfile is exited; using ‘‘Q’’ or control-D later will have no effect on the sequencer information for notesfiles already read.

The ‘‘o’’ and ‘‘O’’ commands allow the user to modify the variable used to determine whether notes and responses are ‘‘new’’. The ‘‘o’’ command allows the user to set this variable to any date he wishes. Use the ‘‘O’’ command to set this variable to show only notes and responses written that day. The ‘‘last time’’ file kept for each user is never modified by the ‘‘o’’ and ‘‘O’’ commands.

When no more new notes or responses exist, both the ‘‘j’’ and ‘‘J’’ commands will take the user to the index page. To exit the notesfile, use the ‘‘q’’ command. Exiting with ‘‘q’’ will update the user’s ‘‘last entry’’ time. Exiting with capital ‘‘Q’’ will NOT modify the ‘‘last entry’’ time for that notesfile (neither will control-D).

The ‘‘l’’ and ‘‘L’’ command behave similarly to ‘‘j’’ and ‘‘J’’. The difference is that while ‘‘j’’ and ’’J’ take the user to the last index page when no more new notes or responses exist, the ‘‘l’’ and ‘‘L’’ commands will leave the notesfile as if a ‘‘q’’ had been typed. Thus when no more new notes exist, the ‘‘l’’ command is like typing ‘‘jq’’.

2.8.2 Alternate Sequencers.

If several people share a login account, it is convenient for each to have a set of sequencing timestamps. This is accomplished through the use of the subsequencer option of notesfiles.

Specifying the -a option and a subsequencer name causes notes to use a different sequencing timestamp file. Many different subsequencer names can be used with each login account.

The main sequencer file for a given account is distinct from each of its subsequencer files. Each of the subsequencer files is normally distinct. If the subsequencer names are not unique in their first 6 characters, subsequencer files may collide.

2.8.3 Automatic Sequencing.

An alternate entry to the notes program allows the user to invoke notes with the sequencer enabled and a list of notesfiles to be scanned with a single, simple command. The ‘‘autoseq’’ command is invoked by typing

```
autoseq
```

and reads the environment variable ‘‘NFSEQ’’ to find the names of all notesfiles to be scanned. On some systems, the ‘‘autoseq’’ command may be known as ‘‘readnotes’’, ‘‘autonotes’’ or some similar variant; substitute the appropriate name in the following paragraphs. The ‘‘NFSEQ’’ variable should be defined in .profile for Bourne shell users as follows:

```
NFSEQ=‘‘pbnotes,micronotes,helpnotes,works’’
export NFSEQ
```

For users of the C shell, the following line should be added to the .login file:

```
setenv  NFSEQ   ‘‘pbnotes,micronotes,helpnotes,works’’
```

With NFSEQ assigned this value, a call to autoseq will process the notesfiles ‘‘pbnotes’’, ‘‘micronotes’’, ‘‘helpnotes’’, and ‘‘works’’ with the sequencer turned on.

The full naming conventions, pattern matching capabilities, and ‘!’ exclusion described in section 2.2 (‘‘Notesfile Names and Wildcards’’) are available in autoseq. To read all notesfiles with ‘‘unix’’ in their names, and the four test notesfiles (‘‘test1’’ though ‘‘test4’’), the NFSEQ variable might be defined as:

```
NFSEQ=‘‘*unix*,test[1234]’’
```

If the first character of an entry in the NFSEQ list is ‘‘:’’, the notesfile system reads the file name following for a list of notesfiles. To have the automatic sequencer read the file ‘‘/usr/essick/.nfseq’’ for a list of notesfiles to scan, define NFSEQ as:

```
NFSEQ=‘‘:/usr/essick/.nfseq’’
```

For this feature to work, the file must have group read privileges. The notesfile program runs ‘‘set-uid’’ and can not read files which are readable only by the owner.

The following definitions are also valid. The first one reads the notesfiles specified in the file ‘‘/usr/essick/.nfseq’’ and then reads the notesfiles pbnotes and micronotes. The second definition will read the notesfile pbnotes, those specified in ‘‘/usr/essick/.nfseq’’, micronotes and the ones specified in ‘‘/usr/essick/.other’’. If the notesfile program is unable to read the file specified, it skips to the next entry. For a description of the format of these files, see the section 2.3, ‘‘The -f Option’’.

```
NFSEQ=‘‘:/usr/essick/.nfseq,pbnotes,micronotes’’

NFSEQ=‘‘pbnotes,:/usr/essick/.nfseq,micronotes,:/usr/essick/.other’’
```

The automatic sequencer uses the ‘‘-s’’ mode of sequencing. The user does not enter notesfiles which have no new text. By specifying ‘‘-x’’ or ‘‘-i’’ on the command line, the user can use the appropriate sequencer mode.

The subsequencer option of notes is available from the autoseq program by specifying ‘‘-a name’’ on the command line, and has identical semantics with use of this option when invoking notes.

2.9 Environment Variables.

The notesfile program reads several environment variables to tailor the system to the user's preferences. Below is a list of the variables, their purpose, and their default values. These defaults are for UNIX 4.xBSD and may be slightly different for other versions of UNIX.

- ''NFED'' specifies which editor will be invoked when the user writes a note or response. If this variable is not specified, the notesfile system looks for the environment variable ''EDITOR'' (which many other programs use). If neither ''NFED'' nor ''EDITOR'' are defined, a default editor is used (/bin/ed).
- ''NFSEQ'' is a list of notesfiles that the user wishes to scan using the automatic sequencing entry to notesfiles. The use of this variable is described in the section on sequencing. If unspecified, the system uses a standard set which usually includes ''general'' and ''net.general''.
- ''PAGER'' is the paging program (''more'', ''pg'') which is used for scrolling the help files. The default paging program is /usr/ucb/more.
- ''MAILER'' determines the mail program to use. This defaults to /usr/ucb/mail.
- ''WRITE'' is used to specify the program for communication between users. If undefined, the Unix program ''write'' is used.
- ''TERM'' determines the type of terminal in use. This must be set for notes to know what screen handling conventions to use. In most cases the value will be correctly initialized by the system at login time.
- ''SHELL'' specifies which shell the user is running. This will almost always be set by the operating system.

3 Other Notesfile Utilities.

The notesfile distribution includes utility programs to provide hard copy output, additional interfaces to user programs, and statistics. They are described below.

3.1 Hard Copy Output.

The program ''nfprint'' sends to standard output a nicely formatted listing of the notesfile in its command line. Its format is:

nfprint [-lnn] [-p] [-t] topic [note#] [note#-note#] [...]

The ''-l'' option specifies an alternate page size (the default is 66). The optional note number list specifies that only certain notes of the notesfile are to be printed. The list can specify individual notes and ranges. The notes are printed in the order specified.

The -p option specifies that each notestring is to begin on a new page. The -t option signifies that only a table of contents is to be generated.

3.2 Piped Insertion of Notes.

The nfpipe program enters text from the standard input into a notesfile:

nfpipe topic [-t title] [-d] [-a]

The -t option allows specification of a title. The -d and -a options specify the director and anonymous flags respectively (if available). If no title is specified, one is manufactured from the first line of the note.

3.3 User Subroutines.

3.3.1 Nfcomment.

The nfcomment subroutine is callable from a user's C program. It allows any user program to enter text into a notesfile:

nfcomment (nfname, text, title, dirflag, anonflag)

The parameters are:

```
char *nfname;     /* name of notesfile */
char *text;       /* null terminated text to be entered */
char *title;      /* if non-null, title of note */
int dirflag;      /* != 0 -> director flag on (if allowed) */
int anonflag;     /* != 0 -> anonymous note (if allowed) */
```

If the text pointer is NULL, the text of the note will be read from standard input. If no title is specified the subroutine will manufacture a title from the first line of the note. This routine is useful for error reports, user comments about programs, and automatic logging of statistics or internal states.

This routine can be loaded with a C program by specifying '-lnfcom' on the 'cc' command line.

3.3.2 Nfabort.

Nfabort allows users to generate core images of their process, save the core image in a ''known'' place, and log that fact in a notesfile. This proves useful for intermittent failures; The programmer regularly scans the notesfile and can examine the core dump at leisure. Some of the problems of recreating conditions which cause errors are eliminated by this approach.

Nfabort is callable from the user program. It accepts the following parameters:

nfabort (nfname, message, title, cname, exitcode)

The parameters are:

```
char *nfname;     /* name of notesfile */
char *message;    /* text string to insert */
char *title;      /* title of the message */
char *cname;      /* prefix for core image destination */
int exitcode;     /* code for exit() */
```

The core image is placed in the file specified by concatenating the ''cname'' argument and a unique integer (the process id of the current process). The notesfile specified by the ''nfname'' parameter receives a note whose body consists of the text pointed to by ''message'' and a line telling the complete pathname of the core image. The title of the note is specified by the ''title'' parameter. After the core image is generated and the note has been written, nfabort terminates with the exit code specified by the ''exitcode'' parameter.

Nfabort generates default values for each of the string parameters if NULL pointers are passed. This routine can be loaded with a C program by specifying '-lnfcom' on the 'cc' command line.

3.4 Statistics.

The notesfile system keeps statistics on where notes and responses originate, the number of network accesses, duplications and orphaned responses. Combined with the use of the log maintained by the notesfile networking software, monitoring notesfile traffic is quite easy.

The -s option specifies that only a summary is to be produced, skipping the individual reports. Wildcard constructs with '*', '?', '[', and ']' are recognized by nfstats. Invoke the statistics program with:

```
nfstats [ -s ] topic1 [ ... ]
```

Typical output is:

```
        rbenotes on uiucdcs at  6:24 pm  May  7, 1982
                        NOTES       RESPS       TOTALS
Local Reads             359         115         474
Local Written           53          55          108
Networked in            0           0           0
Networked out           0           0           0
Network Dropped         0           0           0
Network Transmissions: 0   Network Receptions: 0
Orphaned Responses Received: 0    Entries into notesfile: 109
Total time in notesfile:   66.57 minutes  Average Time/entry:   0.61 minutes
Created at 10:04 pm  May  5, 1982, Used on 3 days
```

A combined set of statistics is produced at the end of listings of more than one notesfile. The statistics are largely self explanatory.

3.5 Checking for New Notes.

The checknotes program checks the notesfiles specified by the NFSEQ environment variable to determine if there are new notes. The exit code is arranged to make the program useful in shell scripts: 0 (TRUE) is there are new notes, 1 (FALSE) otherwise.

Use the ‘‘-q’’ option to receive a message

There are new notes

if one or more of the notesfiles have notes/responses written since the user's last entry time into that notesfile.

The ‘‘-n’’ option is similar to the ‘‘-q’’ option, with the exception that it yields output when there are no new notes. The output of checknotes with the ‘‘-n’’ option is:

There are no new notes

Use ‘‘-v’’ to print the name of each notesfile with new notes/responses. The ‘‘-s’’ option is suitable for use in conditional expressions in shell scripts; no output is generated by this option.

#

TABLE OF CONTENTS