DOCUMENT RESUME

ED 308 495 CS 211 870

AUTHOR Neuwirth, Christine M.; And Others
TITLE The Notes Program: A Hypertext Application for Writing from Source Texts. CECE Technical Report No. 1.
INSTITUTION Carnegie Mellon Univ., Pittsburgh, PA. Center for Educational Computing in English.
PUB DATE Jan 87
NOTE 34p.
PUB TYPE Reports - Descriptive (141) -- Reports - Research/Technical (143)

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS *Computer Software; *Computer Uses in Education; Decision Making; Experimental Programs; Higher Education; *Notetaking; Writing Improvement; *Writing Processes; Writing Research
IDENTIFIERS Computerized Techniques; Expository Teaching; *Hypertext; Knowledge Acquisition

ABSTRACT

This paper discusses "Notes," a hypertext application program which was designed and implemented to investigate the effects of computers on the writing process generally, and in particular to experiment with tools to support the decisions writers make while acquiring and structuring knowledge taken from source texts. The paper outlines the theoretical basis for the design of the "Notes" program, exploring typical writing activities in some detail. The benefits and limitations of conventional note cards are reviewed, as well as expected benefits of the use of computer-based note cards. The "Notes" program and its relation to relevant research is described, and a formative evaluation is presented based upon interviews with participants of the "Notes" program as it was used with five sections of college experimental writing courses for two semesters. Two problems that need to be addressed are discussed: (1) the first concerns the representation of notes; and (2) the second concerns support for the process of taking notes. (An appendix of interview questions, "Notes" program samples, and 26 references are attached.) (NH)

The Notes Program:

A Hypertext Application for Writing from Source Texts

Christine M. Neuwirth, David S. Kaufer, Rick Chimera and Terilyn Gillespie

Center for Educational Computing in English
English Department
Carnegie Mellon University
Pittsburgh, PA 15213

Center for Educational Computing in English
CECE Technical Report No.1
January 1987

The Notes Program:

A Hypertext Application for Writing from Source Texts

Christine Neuwirth, David Kaufer, Rick Chimera & Terilyn Gillespie

English Department
Carnegie Mellon University
Pittsburgh, PA 15213

ABSTRACT

Notes is a hypertext application developed to investigate the effects of computers on the writing process, in particular, on the processes of acquiring and structuring knowledge when writing from source texts. Notes is designed to help writers record their own ideas (e.g., reactions, inferences, plausibility assessments), recover the context for those ideas easily and view ideas from multiple perspectives. In this paper we outline the theoretical basis for the design of the Notes program. Then we briefly describe the program itself and its relation to relevant research. Finally we describe our experience with users.

INTRODUCTION

Writing an essay requires shaping a complex network of ideas, not all of which are present at the beginning of the writing process, into a coherent linear structure of sentences and paragraphs. For this linear structure to be successful, the writer must have constructed systematic conceptual groupings among ideas [Meye75]. When a writer knows a domain well, relatively simple reordering of available knowledge may be all that's necessary. However, when a domain is

new to the writer, or the writer is reconceptualizing a well-known domain, the writer may need to engage in extensive reorganization and elaboration of his or her own understanding.

We have designed and implemented a computer program, called Notes, to investigate the effects of computers on the writing process, in particular, to experiment with tools to support, not replace, the decisions writers make while acquiring and structuring knowledge. Notes is one component of a larger project to develop decision support systems for reading and writing [Neuw87].

The Notes program has an analog in earlier technology: 3x5 note cards. The following section, which outlines some key components in the writing process, lays the theoretical groundwork for exploring the benefits of note cards for writers, the limitations of conventional note cards, and the expected benefits of computer-based note cards.

The Writing Process

Theories of writing processes typically identify the following activities in writing: acquiring knowledge, viewing it from different perspectives to gain new insights, structuring knowledge according to those perspectives, selecting and possibly creating knowledge to meet goals for discourse and re-arranging it so that a reader with different perspectives will find it equally coherent [Youn71]. This section explores each of these activities in some detail and comments on the use of note cards as a technology that can aid a writer in carrying out these activities.

Acquiring Knowledge

Many of the ideas that we ultimately make use of in a text come to us while acquiring knowledge, that is, while exploring a problem and finding out more about it. Typically, ideas in a new domain do not come to us in an orderly fashion. Rather, they present a puzzle of seemingly unrelated concepts and unexplained connections. It is difficult to remember specific facts that we learn. We often restructure ideas to fit patterns that are already familiar or drop ideas that are difficult to assimilate to familiar patterns [Bart32].

As we read or find out new information, we are not simply recording it. We are constructing connections, drawing inferences, imagining scenarios and examples, commenting on plausibility, noting connections to other texts and knowledge as well as connections to our immediate goal and the problem we are investigating. These elaborations play an important role in acquiring new knowledge. Researchers postulate that elaborations play two vital functions: They form connections between what people already know and the new knowledge and they build multiple retrieval paths for the ideas [Rede79]. While it is important when reading to construct elaborations and inferences, it is equally important when writing to remember that they are elaborations and inferences, and not to confuse them with the original information.

Viewing Knowledge from Different Perspectives

The second activity usually included in writing, especially by a theory that includes invention, involves viewing knowledge from different perspectives.

Some inventional theories involve explicitly teaching writers a set of perspectives. For example, Aristotle's *topoi*, Young, Becker and Pike's tagmemic grid (particle, wave, field), Burke's pentad (act, scene, agency, purpose, etc.) or Nelson's system of synectics. Each of these techniques provides a system for exploring concepts, an activity essential to discovering new elaborations or relationships. Most such theories stress the importance of systematically varying perspectives, a way to overcome Burke's observation that "A way of seeing is also a way of not seeing." Indeed, studies which have examined creativity in writing have noted a direct relationship between the amount of examination of concepts from different perspectives and quality of writing and creativity [Youn73; Moor85].

Structuring Knowledge

Different perspectives also provide frameworks for structuring knowledge. Few studies have examined the process of writing while the writer is acquiring domain knowledge. Those few studies that do exist support the notion that structuring knowledge can be a significant task in writing in new domains. Newell's [Newe84] study, which examined the role of writing in learning, found that writing about a new domain required writers to move from relatively isolated and detached concepts to an integrated structure. Langer's [Lange84] study of the relationship between topic-specific knowledge and quality in expository writing suggests that the degree of organization of knowledge is directly related to a writer's success. Writers whose knowledge was highly organized, i.e., their knowledge base included superordinate concepts, precise meanings, analogies to other concepts, and explicit links among concepts, were most successful.

Selecting and Arranging

At some point in the process of writing, the writer must decide what knowledge, both acquired and original, is going to be suitable for communicating to a reader. Moreover, the writer must decide what linear order for the ideas--what juxtapositions and connections as well as oppositions--will result in best meeting the writer's goals. Exploration must, at least temporarily, come to an end.

The Benefits of Note Cards for Writers

There are many ways to "write" ideas down, to record the connections between them, to juxtapose ideas, perhaps discovering new connections: pencil and paper, 3x5 note cards, tape recorders, text-editors, etc. Some of these are better than others for aiding the processes of invention and arrangement just outlined. This section explores the benefits of note cards for carrying out some of these activities.

Writers use note cards for three primary reasons: First, note cards provide an external store for a large body of knowledge that as yet has no coherent linear structure. Second, note cards provide a convenient way for writers to record their own reactions, elaborations, and interpretations of texts while still maintaining a record of sources that the writer may want to return to or to acknowledge. Third, note cards provide a way of representing knowledge that makes some inventional activities easier.

The first benefit to writers using note cards is that they provide a convenient way to record ideas in a text. As noted above, recording concepts and propositions is particularly important when there might be a tendency to fit new knowledge to familiar but inappropriate patterns.

The second benefit to writers using note cards is that note cards give them a convenient way of recording their own reactions, elaborations, and interpretations of texts that they are reading while still maintaining a record of the source. By recording the source together with the elaboration, note cards make both available for review and reevaluation. The importance of review and reevaluation in learning a new domain has been cited by writing researchers as a reason that writing has a major role to play in learning [Emig71].

Various studies that are relevant to taking notes have explored the strategic significance of elaborations during reading. A study of elaborations during reading in which the elaborations are written down rather than unwritten (mental or verbalized) found that written responses led to better postest responses than unwritten [Mich61].

The third and most distinctive benefit for note cards is their power as a representational medium. A given network of ideas can be represented by a number of different structures, some of which are better than others for enabling a person to work. For example, numbers are usually better represented with Arabic than with Roman numerals. Likewise, the various structures that are encouraged by the use of note cards are better than an initial, relatively fixed,

linear structure when a person needs to seek out relationships among ideas. Note cards facilitate alternative representations for the linear structuring of concepts, allowing writers to experiment with tentative arrangements until the writer discovers or can impose a workable framework.

Limitations of Conventional Note Cards

The previous section argued that note cards provide a better representational system for writers working in new domains than linear structuring: note cards facilitate a writer's exploration for alternative structures of ideas. Despite this advantage, however, conventional note cards have disadvantages. Not infrequently, writers forget the context for the original note, and must return to the source material in order to make sense of the content of the card. A similar problem occurs with paraphrasing in notes: the writer introduces inaccuracies. Writers, especially inexperienced ones, tend to spend all their time writing down quotes from the source texts rather than recording paraphrases, elaborations, inferences, interpretations, etc.

The foremost problem with note cards arises when the writer is struggling to impose a workable framework on the material: although notes offer a more tractable medium for this activity than 8x11 paper, creating alternative frameworks nevertheless destroys the previous order. Writers have two alternatives to circumvent this problem. First, they can make duplicates of note cards, a time-consuming venture. Second, they can number note cards and then record the structuring by means of the numbers. Reconstructing the ordering is

then possible, but like the duplication solution, also time-consuming.

Expected Benefits of Computer-based Note Cards

When a writer is working with texts that are stored in the computer, the Notes program keeps a link between each note and the specific region in the source text from which it came. We reasoned that such a facility would free the writers (1) to paraphrase because they would always be able to easily recover the quotation, and (2) to record their own elaborations, reactions, inferences, etc., because they could easily recover the context for them.

Recovery of context is only easily accomplished when the texts are stored in the computer. Although this is possible in a writing course in which the number of readings is small, it will be a number of years before we see vast numbers of texts stored on computers. Thus, the primary benefit of computer-based notes in the near future will be its potential for helping writers create alternative organizational frameworks more easily. Unlike paper, the computer does not collapse the storage and display of information. Because of this feature, the computer can be easily programmed to allow writers to create and view alternative organizations of their notes. Creating new alternatives does not destroy previous organizations and the computer can easily keep track of the book-keeping involved.

Design Goals

We built the Notes program to explore the ideas just outlined. In the Notes program, we use an underlying database in order to maintain links from notes to sources and from sources to notes and to allow the user to view notes from multiple perspectives.

The following list represents other design goals for the Notes program, together with their rationale.

--Ease of learning and use. Writers typically come to a program like Notes wanting to get on with a task. The program must allow them to get started with useful work immediately and must be easy for them to learn as they go along. Student writers must be able to learn the system while engaging in useful writing activities; otherwise the system will be unattractive to their teachers who will see it as taking time from the teaching of writing. In a hypertext application, ease of learning and use appears to be intimately connected to the user's ability to negotiate links among text objects without getting lost.

--Quick access to notes. The time it takes to access a note must be comparable or better than the time it takes to do so from a traditional note card file. The program must exploit the searching and retrieval power of the computer with an easy to use search interface.

--Flexibility for online and offline work. It will be some years before

significant numbers of texts are online. The program must work well with off-line sources as well as online ones. Likewise, the notes must have a hard copy representation.

THE NOTES PROGRAM IN DETAIL

The Notes program consists of two basic objects, source texts and notes, and a single derived object, lists of notes. Source texts are those texts the user is reading and wants to take notes on. The source texts can be online or off, but the following discussion illustrates a user taking notes on an online text. Notes are those texts the user composes in order to record elaborations of the source texts, i.e., the user's record of his or her "writing" of the text. Notes are online.

In addition to the basic objects, the Notes program consists of a single derived object: lists of notes. In the current version of the notes program, the lists are automatically compiled by the Notes program. Lists have a linear order, alphabetically by the author of the source text and within sources, by the user-created name of each note.[1] The user can also create alternative lists, typically based on ordering principles that the Notes program cannot compute automatically. The alternative lists allows the user to impose a hierarchical structure on the notes as well.

Figure 1 illustrates how the screen might[2] appear to a user who is in the midst of reading on the topic of creativity. The user has taken notes on two source texts: one by Hayes and one by Perkins. The system maintains a list of all the notes a user has taken in the region labeled All Notes List. At this point and at any point,

the user can select from a range of activities: view the notes, create classes and classify the notes, form alternative organizations for the notes, etc. The user controls the order of these activities. Let's suppose that the user wants to take more notes on one of the source texts, Hayes, "What is a creative act?" To do so, the user opens as set of menus and uses a mouse to select Open from a Source Text menu card.

Taking Notes

To take a note, the user selects the region in the source text where he or she wants to take a note, moves the mouse cursor anywhere in the selected region, opens a menu, and chooses Take Note from the pop-up menu (see Figure 2).

Composing a Note

When a user chooses Take Note, a note region appears below the source text. The source text itself is recentered, if necessary, so that the selected region for the note remains visible on the screen. An icon appears in the source text. The icon looks like a footnote in a square and indicates that there is a link between the source text and the note (see Figure 3).

To compose a note, the user moves the mouse cursor inside the note region, clicks the left mouse button and begins composing ("Why is it important...?"). The Notes program uses the Andrew system base editor, so the user has the full functionality of an integrated text-editor/document-formatter to compose. In addition, the user can copy material from the source text or from other windows

on the screen and paste it into the note.

Although the note region approximates a 3 X 5 card, the text of the note can be as long as the user desires. If the text that the user composes exceeds the space allocated to a note region, the entire text will not be visible. However, the user can scroll the text to view different parts of it or enlarge the Notes program window so that more text is visible.

In addition to composing the text of the note, the user must also compose a name for each note ("Why Criteria?"). A name is a mnemonic for the contents of the card, and is used by the Notes program to display a list of notes that have been composed.

To take another note, the user selects a region of text and chooses Take Note again. The previous note is replaced by a blank note and except for the name of the note, which is put into the Notes listings, the previous note is "put away" from view.

Viewing Notes

After the user has taken a number of notes, perhaps in a different session, he or she may wish to review the notes. To view notes, the user positions the mouse cursor in the All Notes List, points at a note of interest and clicks the left mouse button. The note appears in the View Notes region (see Figure 4).

The user can display up to four notes at a time. In addition, the user can ask the program to expand the viewing region so that more notes can be viewed. When

the viewer has been viewing a series of notes and calls up a new note, the new note will appear in place of the note that has been dormant the longest.

Alternative Lists

In addition to viewing notes from the All Notes List or from the source text, the user can also create alternatively organized lists of notes, called alternative lists. Alternative lists support viewing notes from alternative perspectives. Users can create as many alternative arrangements as they need. They can cut and paste across different lists. In addition, they can display different alternatives on the screen and compare them.

Classifying Notes

Classes play an important role in the Notes program: classes allow users to group notes together. For example, while taking notes or after, a user may group notes according to classes that he or she creates. The classes might be related to the content or structure of the source texts, or to the nature of the elaborations that the user has composed. Figure 4 shows three classes: Original, Value, and Ability, located in the region labeled Classes at the top of the screen.

To create a class, the user displays the classes by means of a menu and chooses Add a New Class from the Edit Classes menu. There are also options to Delete a class or to Rename a class. Because deletion affects notes which might be in the specified class, the user is first informed of the number of notes which are in the class and asked to confirm or cancel the deletion. If the user

responds Confirm, the class is deleted; the notes in the class are not deleted, but only removed from the class.

To add a note to an already existing class, the user makes the note the current note and clicks on its class name. The class name highlights to indicate the current note is a member of the class. Notes can be added to as many classes as the user desires.

To delete a note from a class, the user makes the note the current note, and clicks on the class name. The class box is de-highlighted to indicate that the note has been deleted from the class.

Searching

The user can search for notes on the basis of content, classes, the author of the source text, the title of the source text, the date and time the note was created, and the date and time that the note was last modified. These facilities allow users to locate notes automatically. For example, if the user had taken notes on two source texts on creativity, one by Hayes and the other by Perkins, and classified several of the notes in a user-created class of definition, the user could search for all the notes by Hayes or Perkins that are in the class definition. The search results in a listing of those notes appearing on the screen. The user can view the contents of particular cards in the search result in the same way as any list of notes.

Implementation

The Notes program runs on advanced function workstations--IBM RTs, SUN2s & 3s, and VAXstations. It runs under Andrew, a window-management and base environment for UNIX 4.2 BSD [Morr86].

RELATED RESEARCH

Text Editors

Text-editors, one of the primary user interfaces, are closely tied to computer input/output hardware: Each generation of input/output hardware (keypunches, TTYs, CRTs, and bit-mapped displays) has brought a corresponding generation of editors (batch editors, line editors, screen-oriented editors, and integrated text-editor/document formatters).

Only recently, however, has the hardware been powerful and cost-effective enough so that attention could be turned from designing software that would run efficiently on the hardware to designing software tailored especially to the editing needs of users. New systems, such as integrated text-editors/document formatters, structure-based and network-based editors, have been developed in a resulting surge of research interest [Meyr82]. The Notes program has elements in common with each of these developments. The Notes program can be viewed as an integrated text-editor/document formatter that allows the user to take full advantage of the text-editing paradigm while providing constructs that *prima facie* will facilitate parts of the writing process. This section explores the recent

developments in structure-based editors and networked-based editors as they relate to Notes.

Structure-based Editors

Structure based editors are editors whose user interface and functionality exploit the structural properties of the data that are being edited. Most were developed for editing programming languages; some can edit any general data structure, including graphic structures [Fras81]; a few have been developed specifically for editing English text [Walk81; Alle81]. The widespread distribution of personal computers has brought a number of structure-based editors for English text to the general public's attention (*e.g.*, ThinkTank).

Structure-based editors for English text usually provide two capabilities. First, they provide a diagram of the structure of the document--a hierarchical table of contents--to help readers and writers visualize the structure. Second, they provide a set of commands that exploit the structure; for example, a command to move the text cursor to the beginning of the next subsection; a command to exchange two sections; a command to show only sections and subsections, suppressing paragraph detail, etc.

The Notes program incorporates a structure-based editor for English text: Each Alternative List provides users with a structure-based editor in which they can impose structure on their notes by arranging them in a hierarchy, possibly creating new notes in the process. Unlike existing structure-based editors, which allow users to create only one hierarchical order, however, Notes allows users to

create multiple hierarchies. Although users of standard hierarchical editors can approximate this capability by copying the original file and creating an alternative order in the copy, changes in the contents of the notes in the original file will not be reflected in the copy; whereas in the Notes program, changes in the contents of the notes are reflected in each alternative list, regardless of whether the alternative list was created from an already existing one.

Network-based Editors

Network-based editors are editors whose user interface and functionality allow users to build networks of data by creating links among arbitrary pieces of structure. Most were developed to experiment with non-linear organizations for data that the computer medium makes possible. Users can typically use a network-based editor to exploit the structural properties of the data, but network-based editors leave it up to the user to impose the structure; the system does not provide it or enforce it as in a structure based editor.

The concept of a network-based editor is often traced to a paper by Bush [Bush45] who proposed creating a system that would allow users to build associative links through a set of documents. Early, partial implementations of linked data include NLS/AUGMENT [Enge68; Enge73] and Hypertext [Carm69]. Later developments have taken several directions. Xanadu, an outgrowth of Hypertext, is working toward a distributed, hypertext database that could support any number of user interfaces [Nels81]. Intermedia, also an outgrowth of Hypertext, is working toward linking pieces of data objects besides text, including

graphics and images [Yank85]. Textnet [Trig86] and NoteCards [Trig87] are experimenting with the effects of linked networks of data on human-computer interaction, both for individuals and for groups.

The Notes program also supports links, but of a much more restricted variety than these systems: The Notes program supports links between the source text and a note and between a bibliographic reference and a note. One way to view the Notes program is as an optimized interface for creating the links most useful for taking notes. A unit task analysis [Card83] illustrates the optimization. In a general network-based editor, taking a note and linking it to the original source text and to a bibliographical reference would require approximately 8-11 unit tasks: select a region, create a from-link, specify the type of the link as a note link, type some text of the note, select the text, create a to-link, select the text of the note again, create a from-link, specify the type of the link as a reference link, select the text of the bibliography, select a to-link. In the Notes program, taking a note requires 3 unit tasks: select a region, choose take note, type some text. The unit tasks savings comes about because Notes automatically selects the data to link to (i.e., a note) and creates two links (i.e., a note link and a reference link) in a single operation.

At least one of the general network-based editors, NoteCards, could probably be specialized to support the task-optimized linking of the Notes program. We plan to re-implement the underlying database for the Notes program so that it is based on a generalized networked database and subroutine library. A generalization of the database will allow us to make the Notes program compatible with other

specialized tools for writing that we are currently implementing.

FORMATIVE EVALUATION WITH USERS

Throughout its development, we have been conducting formative evaluations of the program, where by "formative evaluation" we mean a study that attempts to evaluate a program in order to improve it. We have operated Notes in five sections of experimental writing courses for two semesters.

Participants

The participants in the evaluation have been experienced and inexperienced computer users with no prior experience with the Notes program. Some had no prior experience with Andrew, the computer system on which Notes is implemented.

Methods

Each participant comes to two sessions. The first session is a training session. In the training session, we provide a one-on-one tutorial introduction to those parts of the Andrew system that participants need in order to work with the Notes program. The training time on Andrew averages about 30 minutes. Then we give participants a hard copy tutorial introduction to the Notes program and ask them to work through the tutorial at their own pace. The average time to work through the tutorial is about 45 minutes for experienced computer users, 90 minutes for inexperienced.

In session two, we ask the participants to read two short articles on an issue (controlling human behavior), and to write an essay that (1) synthesizes the issues from the other two essays as a springboard for developing a position on the issue and (2) lays out their position on the issue. To make the task demanding, we impose a time constraint: 45 minutes to read the essays and 45 minutes to write a draft. We ask participants to use the Notes program to take notes and write their essays. In both training and work sessions, we ask participants to think-aloud as they work and we record what they say [Eric84].

So far, all participants take the full 45 minutes to read the essays and, all participants but one have taken the full 45 minutes to write the draft of the essay.

Our observations of the errors participants make and the thinking-aloud protocol data give us a wealth of information about specific problems with the program, problems that were, for the most part, relatively easy to fix. But the most valuable information about the overall design of the program comes from interviews with participants after they have completed the reading and writing task. The interview questions, based in part on a set developed by Hidi and Klaiman [Hidi83], focus the users' attention on the process of taking notes and probe for the Notes program's effects on their usual note-taking processes (see Appendix I for a list of the interview questions).

Results of the Interview Questions

For the most part, we have incorporated solutions to participants' problems with the Notes program into the version described in this report. However, we have two outstanding problems that we urgently need to address. The first concerns the representation of notes; the second concerns support for the process of taking notes.

It is now well-established that the right representation can significantly influence the ease of problem-solving [Simo81]. Our interviews with users suggests that we would do better to provide them with not one representation for notes but a variety of representations, with each representation providing a better match to a particular sub-task. For example, some users would like to cluster their notes in a graphical network of notes before deciding on any linear order for them whatsoever. Others would like their notes to represent a path through an issue.

The second problem concerns the support for the process of taking notes. At the present time, it is easy for writers to move from notes to prose. But, not too surprisingly, writers requested the ability to move from prose to notes. Writing actual prose represents a bottom-up planning procedure. As in other complex tasks, writers engage in a combination of top-down and bottom-up planning.

We will be working to provide these additional capabilities and testing whether they provide the useful decision-support for the complex task of writing an original paper from sources.

REFERENCES

[Alle81] Allen, E., Nix, R., & Perlis, A. PEN: A hierarchical document editor. *Proceedings of the ACM SIGPLAN/SIGOA Conference on Text Manipulation*, (Portland, Oregon, June 8-10, 1981) ACM, New York, 1981, 44-81.

[Bart32] Bartlett, F.C. *Remembering.* Cambridge: Cambridge University Press, 1932.

[Card83] Card, S.K., Moran, T.P., & Newell, A. The Psychology of Human-Computer Interaction. Lawrence Erlbaum Associates, Hillsdale, N.J., 1983.

[Carm69] Carmody, S., Gross, W., Nelson, T.H., Rice, D., & van Dam, A. "A hypertext editing system for the /360", in *Pertinent Concepts in Computer Graphics*, M. Faiman & J. Nievergelt, Eds. University of Illinois Press, Urbana, Ill., 1969, 291-330.

[Enge68] Engelbart, D.C. & English, W.K. "A research center for augmenting human intellect", in *Proceedings of the Fall Joint Computer Conference*, vol. 33. AFIPS Press, Reston, Va. (1968), 395-410.

[Enge73] Engelbart, D.C., Watson, R.W. & Norton, J.C. "The augmented knowledge workshop", in *Proceedings of the National Computer Conference*, vol. 42. AFIPS Press, Reston, Va. (1973), 9-21.

[Eric84] Ericsson, K.A. & Simon, H.A. *Protocol analysis: Verbal reports as data.* MIT Press, 1984.

[Fras81] Fraser, C.W. Syntax-directed editing of general data structures. *Proceedings of the ACM SIGPLAN/SIGOA Conference on Text Manipulation*, Portland, Oregon, June, 1981, 17-21.

[Emig77] Emig, J. "Writing as a mode of learning," *College Composition and Communication, 28*, 1977, 122-128.

[Hidi83] Hidi, S. & Klaiman, R. "Notetaking by experts and novices: An attempt to identify teachable strategies," *Curriculum Inquiry*, 13, 4 (1983), 377-395.

[Lang84] Langer, J.A. "The effects of available information on responses to school writing tasks," *Research in the Teaching of English*, 18, 1 (Feb. 1984), 27-44.

[Meye75] Meyer, B.J.F. *The organization of prose and its effects on memory.* Amsterdam, The Netherlands: North-Holland, 1975.

[Meyr82] Meyrowitz, N. & van Dam, A. "Interactive editing systems," *ACM Computing Surveys*, 14, 3 (Sept. 1982), 321-415.

[Mich61] Michael, D.N., & Maccoby, N. "Factors influencing the effects of student participation on verbal learning from films," in A.A. Lumsdaine (Ed.), *Student response in programmed instruction*. Washington, DC: National Academy of Sciences, 1961.

[Moor85] Moore, M. T. "The relationship between the originality of essays and variables in the problem-discovery process: A study of creative and non-creative middle school students," *Research in the Teaching in English*, 19, 1 (Feb. 1985), 84-95.

[Morr86] Morris, J.H. et al., "Andrew: A distributed personal computing environment," *Communications of the ACM*, 29, 3 (March, 1986), 184-201.

[Nels81] Nelson, T. Literary Machines. T. Nelson, P.O. Box 118, Swarthmore, PA 19081, 1981.

[Neuw87] Neuwirth, C.M., & Kaufer, D. Decision support systems for reading and writing. *Technical Report CMU-CECE-TR-3,* Pittsburgh: Carnegie-Mellon University, Center for Educational Computing in English, English Department, 1987.

[Newe84] Newell, G.E. " Learning from writing in two content areas: A case study of protocol analysis.," *Research in the Teaching of English*, 18, 3 (Oct. 1984), 265-287.

[Rede79] Reder, L.M. "The role of elaborations in memory for prose," *Cognitive Psychology*, 1979, *11*, 221-234.

[Trig86] Trigg, R. & Weiser, M. "TEXTNET: A network-based approach to text handling," *ACM Transactions on Office Information Systems*, 4, 1 (Jan., 1986), 1-23.

[Trig87] Trigg, R., Suchman, L., & Halasz, F. "Supporting collaboration in NoteCards," In Proceedings of the Conference on Computer-Supported Cooperative Work (Austin, Tex., DEC.). ACM, New York. 1987.

[Walk81] Walker, J. "The document editor: A support environment for preparing technical documents," *Proceedings of the ACM SIGPLAN/SIGOA Conference on Text Manipulation*, Portland, Oregon, June, 1981, 44-50.

[Yank85] Yankelovich, N., Meyrowitz, N. & van Dam, A. "Reading and writing the electronic book," *Computer*, 18, 10 (Oct., 1985), 15-30.

[Youn71] Young, R. E., Becker, A., Pike, K. *Rhetoric: Discovery & Change.* Harcourt, Brace & Yovanovich, San Francisco, 1971.

[Youn73] Young, R. E., & Koen, F.M. The tagmemic discovery procedure: An evaluation of its uses in the teaching of rhetoric. *Final Research Report to the National Endowment for the Humanities*, Grant number EO-5238-71-116, University of Michigan, 1973.

ACKNOWLEDGEMENTS

We thank the many people involved with the development of the Notes program, especially the teachers and students who have used the program and given us valuable feedback. The work described in this report represents contributions by the following people:

System design: Chris Neuwirth, Rick Chimera, David Kaufer

System programming: Rick Chimera, Gary Keim, Dale Miller, Keith Evans, Aaron Oppenheimer

User testing: Chris Neuwirth, Terilyn Gillespie

Documentation: Terilyn Gillespie, Tom Gomoll

System design consultant: Thom Peters

The design, development and testing of the Notes program has been supported, in part, by the Fund for the Improvement of Post-Secondary Education (FIPSE) as part of the Warrant Project, Preston Covey, Cheryl Geisler, David Kaufer and Chris Neuwirth, Principal Investigators, the Alfred P. Sloan Foundation, the Pew Memorial Trust, and the A.W. Mellon Foundation. We would also like to thank Carnegie Mellon's Center for the Development of Educational Computing for programming support, Carnegie Mellon's Information Technology Center for

system support and IBM for an equipment grant. Finally, thanks to Davida Charney, Erwin Steinberg and Richard Young for their comments on earlier drafts.

NOTES

[1]A planned extension will allow users to see note cards by other program-generated orders, such as grouped by classes that the user has put the notes in.

[2]Because the system that Notes is implemented on allows multiple windows, the entire screen may not be devoted to the Notes program. If a user has more than one window on the screen, Notes occupies the portion of the screen that the user has allocated for it, but the Notes window itself would still appear as described. In addition, the Notes window itself can take on different appearances. For example, the user can hide various regions of the Notes program's window from view and expose regions to view. For example, if the user is primarily engaged in taking notes, he or she may not want the listing and viewing regions exposed to view and there is an option to Hide/Expose the All Notes List and Viewing regions.

[3]From Hayes, J. R., [1982] *The Complete Problem Solver.* The Franklin Institute Press, Philadelphia, pp. 197-8.

APPENDIX I: INTERVIEW QUESTIONS

1. Do you usually take notes?

2. When you do take notes, do you take them as you just did or was this particular session unusual? If unusual, how?

3. When you do take notes do you have a particular style, format or procedure that you use? Does this depend on the reason you are taking notes? Did the Notes program interfere with or enhance your style, format or procedure?

4. Do you usually have some specific ideas in mind before you begin to take notes or does the text suggest ideas to you? Did this process feel any different when working with the Notes program?

5. Do you usually reword or select particular ideas to take notes on, or do you often copy parts of the text verbatim? Does this depend on your purpose? The material? Did your rewording or copying practices change or stay the same as a result using the Notes program?

6. How do you select the ideas you take notes on? Did taking notes with the Notes program affect your selection of ideas?

7. What features did you like about the Notes program?

8. What features did you dislike?

9. What would you change? What would you change it to?

10. If you had to name just one additional feature that the Notes program should have, what would it be?

11. Would you use the Notes program again if you had the opportunity? Why or why not?

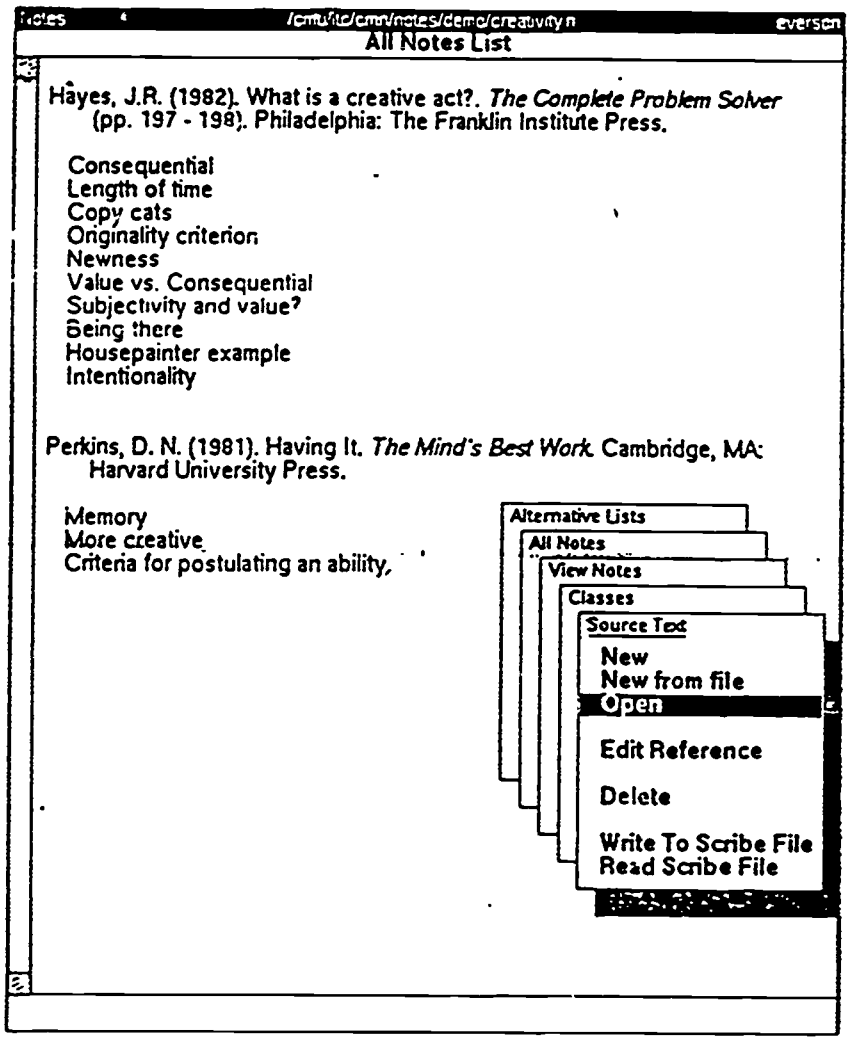

Figure 1. The Notes Program Window

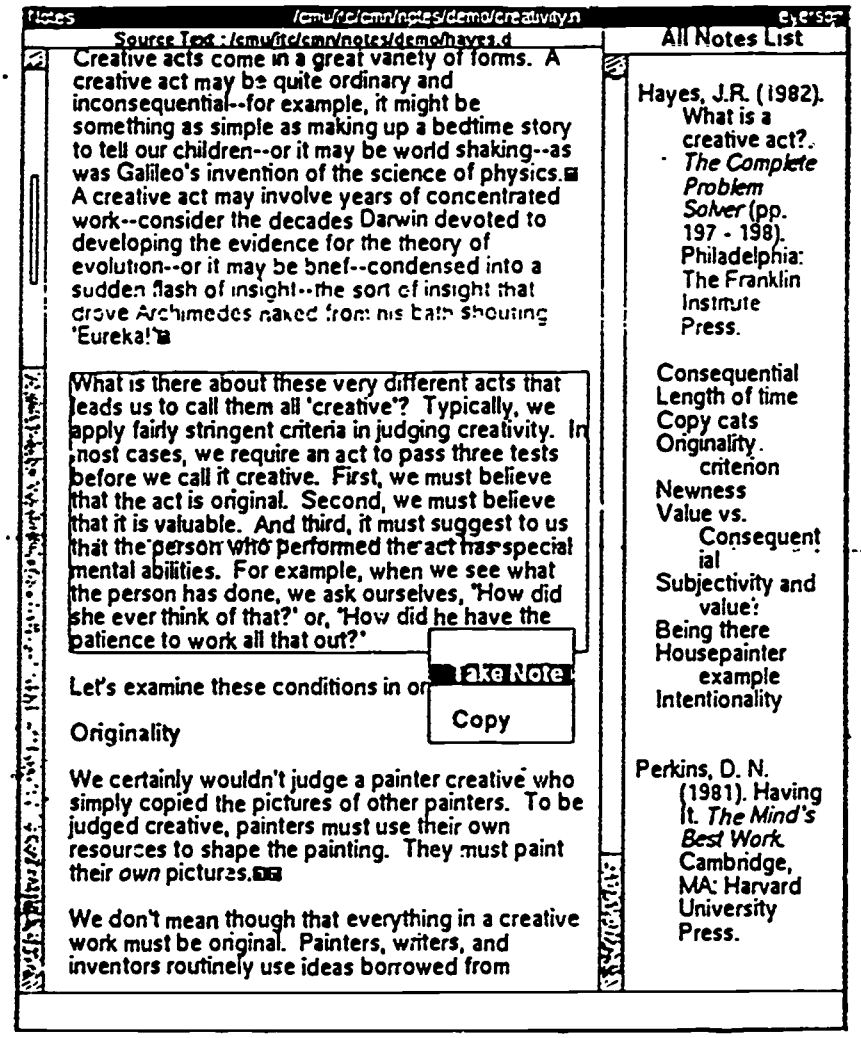

Figure 2. Taking a Note

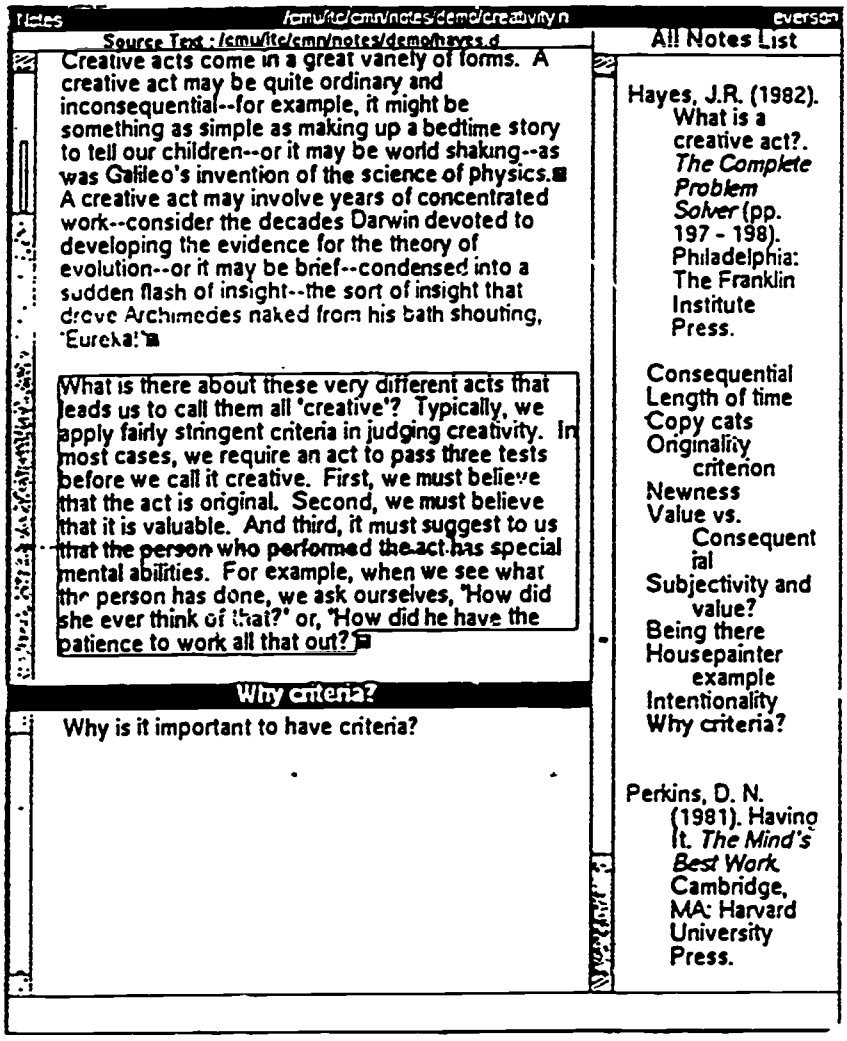

Figure 3. Composing a Note

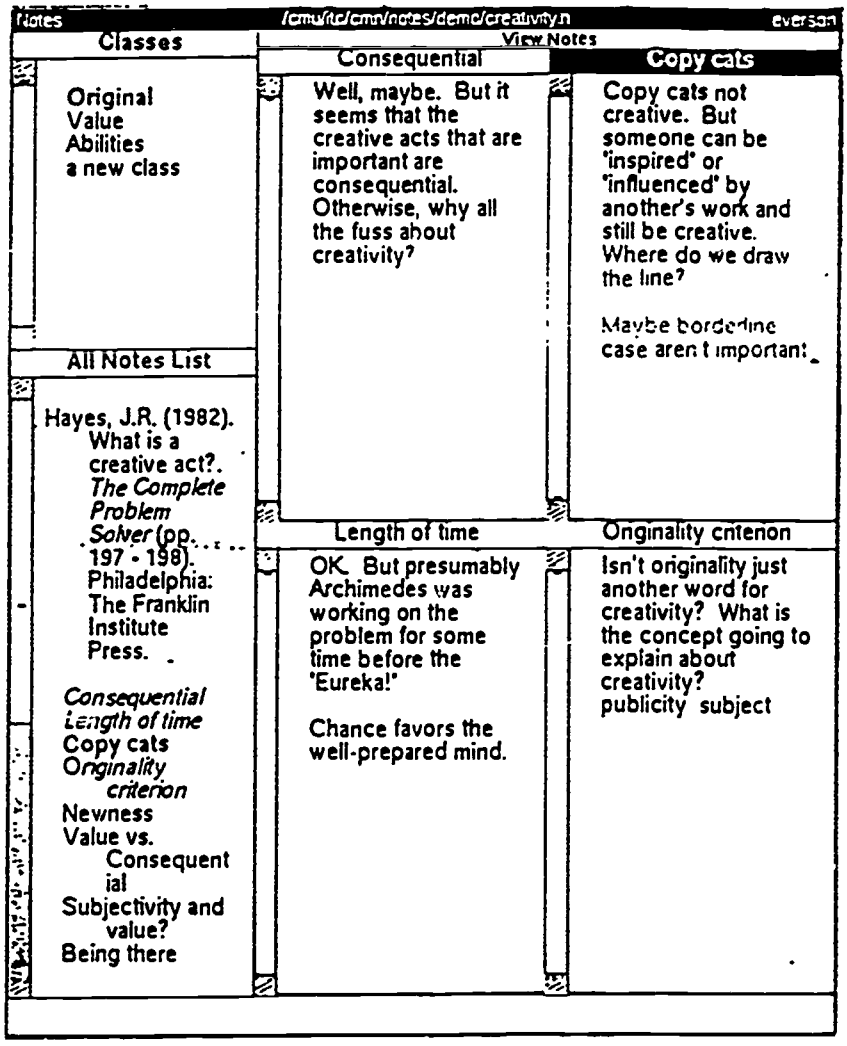

Figure 4. Viewing and Classifying Notes

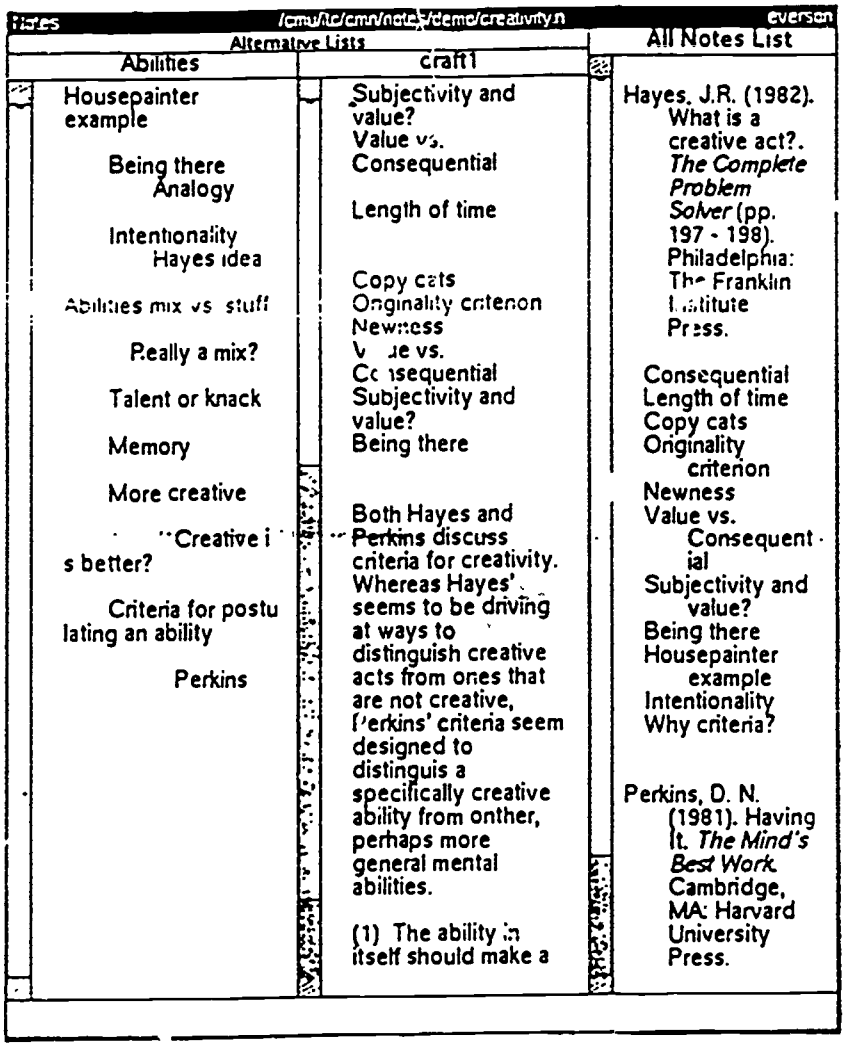

Figure 5. Alternative Lists