Computer-Assisted Text Manipulation

A Look at

The Current State of the Art

And a Particular Example,

The Hypertext Editing System

David Edward Rice

Computer-Assisted Text Manipulation

A Look at
The Current State of the Art
And a Particular Example,
The Hypertext Editing System

by

David Edward Rice
A.B., Dartmouth College, 1967

Thesis
submitted in partial fulfillment of the requirements for the

degree of

Master of Science

in the Division of Applied Mathematics

at Brown University.

June, 1970

TABLE OF CONTENTS

Abstract

In the first part of this paper, an attempt is made to justify the use of computers in the processes of writing, editing and retrieving textual material. Existing text editing programs and computer-controlled typesetting are surveyed. The second part of the paper describes a multi-purpose text handling system which can be used for text editing and revision, information retrieval, typesetting (through IBM's TEXT360 program), and the presentation of non-sequential forms of writing, called hypertext. The Hypertext Editing System is fully operational on Brown University's IBM 360/50 with large disk file (2314), maintaining a single display (2250/I, III, or IV). The system is being generalized to maintain multiple displays, with varying degrees of power.

Acknowledgements

The research reported in Part II of this thesis was supported in part by a contract between Brown University and the International Business Machines Corporation, and in part by Brown University itself. I wish primarily to thank Andries van Dam, my thesis adviser, for his ideas and time expended in developing the Hypertext Editing System, and his contributions to this thesis; also Theodor Nelson for the concept of a hypertext and his numerous suggestions. Whereas I was responsible for the Format Phase and printing capabilities of the system, the Edit Phase was programmed principally by Steven Carmody and Walter Gross. Additionally, assorted programming efforts were contributed by Robert Batts, Martin Michel, Mark Pozefsky, Richard Schmidt, Kenneth Sloan, George Stabler, Daniel Stein and Robert Wallace. The staff of the Brown University Computing Laboratory is to be commended for their patience and understanding, especially during the "hard times" when the system was being debugged. Patricia Mitchell provided assistance in gathering source information for Part I and copy-edited the entire manuscript. Lastly, I wish to thank the Hypertext Editing System for its kindness and perseverence while this thesis was being composed.

PART I: COMPUTER-ASSISTED EDITING AND TYPESETTING

1. INTRODUCTION AND USES

The traditional method of manuscript composition and publishing is a slow, unsatisfactory process. There is not enough mechanization at either the input or the output end of the spectrum. Turnaround time and communications between authors and publishers are so poor that documents are often out-of-date before they are published.

An author's role in the process is perhaps the most old fashioned of all. His tools include a pencil (or possibly a typewriter), some pieces of paper, an eraser, a few reference works and his mind. For some "authors", like Harold Robbins, this is enough. They can write (type) several thousand words a day and never change it. But the fussy, professional (and probably imperfect) author faces a very real problem. He has to rewrite his manuscript many times, do research, and refer to previous writings. He cannot store all the information he needs in his own mind. He would like to try different versions and techniques before deciding on the final form. He would like to have his previous writings, and all his references, at his

fingertips. In other words, he needs the power and memory of a computer to assist him in his creative tasks. From the point of view of the computer scientist as author, it is especially aesthetically pleasing to utilize the power of a general-purpose computer in all aspects of his work.

The basic principle is that all his textual material may be stored in some form recognizable to the computer, and once in this form, need never be respecified. While remaining within the computer, the text may be perused on some interactive output device, modified with ease, and printed out when desired. When a powerful display console is added, the editing tasks become yet more facile.

The most obvious advantage offered by computer-assisted editing is the tremendous reduction in time required to produce a final, or alternate, document. Normal editing requires many cycles in which text must be read and reread, typed and retyped. Once a section of text has been finalized, it should never have to be manually retyped. No sane person enjoys typing intermediate and final drafts of a manuscript, and no one can do it perfectly. Secretaries were invented to "push-down" the problem another level, but their efforts could better be spent elsewhere. Typographical errors, aside from those occurring during initial input, should be completely eliminated. An author or editor should

not have to bother with correcting typographical errors while editing.

Using an online editing system, an author or editor receives cleaner-looking copy and he receives it more often: when he asks for it, not a day or two later. Of course, if the system were adapted to handle many users simultaneously, on a full time basis, he need never request any intermediate printed copy since all his editing would be done online, while he is creating and editing his manuscript at his computer-driven work station.

Since text is stored on disk files (and/or tape), the bulk storage necessary to maintain a group of active documents is reduced while the ease of access is increased. An editor does not get buried in mounds of rough drafts, a programmer is not lost in piles of program listings. Any desired document is readily accessible; all that must be done is to identify to the computer what is wanted.

When the writer and editor are completely satisfied as to the "correctness" of the final manuscript, the computer is told to produce a camera-ready flawless copy of the text. Thus, an author tends to become more particular about expressing an idea "just right" since he isn't hesistant about requesting another intermediate draft: utilizing an IBM 1403 Printer and the TEXT360 printing program (described

in section 3.4.2 below), a flawless upper and lower case, justified hard copy of approximately 100 pages is printed in about 10 to 15 minutes; "rough-draft" copies are considerably quicker.

An additional advantage, basic to the Hypertext Editing System but available in some form or another in most editing systems, is the capability of producing alternate versions of the same document without having to start from scratch. Only the sections that need changing have to be revised. An example of this feature is the manual for the Format Phase of the Hypertext Editing System [21], written by the author, for which there exists in one text file two different versions, one for use here at Brown University, and one for use at outside installations. The storage required to maintain working copies of the manual is not doubled, since only the changed sections are repeated, and the two versions can be readily modified to reflect any changes in the operation of the system.

In summary then, the advantages of working in this manner include:

1) Easy access.
2) Immediacy of response.
3) Ease of making hardcopy without intermediate stages of

typesetting, proofreading, resetting, reproofing, etc.

4) Reduced "turnaround" time for any type of file research and writing task.

5) Common access to the same data base. This is useful for a pool of researchers or documenters working in the same area, or for common access to updated (project) management information.

6) Easy modification of previously written materials for present purpcses (as in the writing of contracts, proposals or prospectuses).

7) Great simplification of document dissemination and storage; no hardcopy bulk, but any degree of archival protection desired.

8) Far greater flexibility for browsing and linking text fragments compared to manual methods with hardcopy.

9) Ability to design and define the format of a form (such as a tax form or an employment form), which is then entered as a standard into the file for subsequent filling out by the user community (this property is not common to all editing programs).

10) Relatively modest cost for all this increase in activity and efficiency, when compared with all aspects of present systems: writing delay, retyping, proofreading, typesetting, revisicn of galley and page proof, printing,

binding, distribution, storage (and subsequent inaccessibility due to distance, shelf space, poor indexing, borrowed or lost copies, etc.). The cost includes the machine time used (typically a 5% or less rate of CPU utilization/user) and the rental or purchase cost of the terminals employed.

1.1 AREAS OF USEFULNESS

Some of the areas in which computer text handling (not just editing) has been shown to be of tremendous value (with examples from our own experience) include:

Production and revision of technical manuals (all of the User Manuals for the Hypertext Editing System have been produced with the system, and have been updated several times in the past six months);

Program design and specification (the specifications of the next version of the system have been done online);

Proposal preparation (several major proposals by Brown University to the National Science Foundation are currently in the editing stages);

Report writing (all of the Quarterly Progress Reports necessitated by a contract between IBM and Brown University have been produced with the system);

Patent documents (a sample text file consisting of metallurgical patents with many cross references was prepared for the Great Lakes Carbon Corporation);

Library and journal abstracts (a text file consisting of many connected abstracts was prepared for representatives of the ACM Computer Reviews Journal);

Program documentation (the documentation for all of the projects under the direction of Professor van Dam has been done with the system, resulting in some twenty manuals); and lastly,

Thesis preparation (including this one, the master's thesis of Robert Sedgewick, and the doctoral dissertation of Charles Strauss).

Other views and uses include [18]:

Central information depot

Records revision facility

Cross-reference and annotative retrieval system

Internal publication and memo library

Compcund and multilevel report generator

Text annotation and analysis work center

Fast document revision system

Super-secretary

2. A BRIEF SURVEY OF TEXT EDITORS

Having examined the general properties of online file editors, a look is now taken at currently existing editing programs. Those described below are intended to represent editors typical of the various categories and no claim of completeness of coverage is made. Commercial editing programs have only been available since about 1968, although experimental approaches began about 1966.

Current online editors have roughly fallen into two categories: those utilizing teletypes and typewriters as the interactive device, and those designed around Cathode Ray Tube (CRT) displays. The tremendous advantage in using the CRT display is that the "page" of text is presented at electronic speeds, thus enabling a user to look at more of his text while editing, since the lines of text are not mechanically printed. In fact, typewriter-like devices are so slow that they force the editor to work from a mocked-up printed copy, manually transcribing changes already made in the printed copy, and therefore doing the work twice. With a CRT display, the editor may think out and implement his changes at one and the same time.

From another point of view, editing programs have also been developed for two (usually) different purposes: the

editing of computer programs and the editing of general purpose textual material. While a few are equally suited for both purposes, the general correspondence is that teletype-typewriter-based systems are used for program editing and CRT-based systems are used for textual editing. There are, of course, exceptions to this trend and these will be discussed in detail below.

Other important distinctions between program and text editors are the types of editing operations they allow (which strongly influences how the text is stored internally), how the operations are specified, and the forms of output provided.

The first distinction is in the type of editing to be performed. In a program editor, one typically modifies "in place," substituting one small string of text, like an op-code or an operand address, for another, or inserting a label in a field (a portion of the line) which was previously blank. In this case, it is perfectly reasonable to store the text line by line. With a text editor, on the other hand, one wants to make insertions and substitutions of arbitrary sized character strings at arbitrary points in the manuscript. This implies that the data structure must be much more flexible in order to cope with these powerful types of insertions and deletions (overflowing or contract-

ing automatically from line to line, within a paragraph, for instance). Thus one typically finds the unit of storage, within which text may grow or shrink dynamically, to be a "super" line of several hundred characters, a paragraph, or even an entire page. (In the Hypertext Editing System, in fact, the storage unit is an arbitrarily large text "area", as large as a chapter in a book, or even the entire book, if desired.)

The second distinction is in the method of specifying operations. Any automatic text editing task requires two inputs to the computer: the task to be done, and the portion of text to which it applies. If a teletype is the input device, this information must be supplied by typing a command and identifying the text either by line number, by context, or by both. Context identification means that a user-specified string of characters (text) is searched for in the text to locate the desired position. Line numbers are somewhat arbitrary and bear little relationship to the text, while specifying context requires extra work for the user and is very prone to unintentional ambiguities. However, if a display unit is used, with a suitable input device (a lightpen, "joystick", "mouse" or data tablet), two improvements are made: text is rapidly displayed and identification can be made by "pointing". Thus, teletypes

require the user to locate the text in question, while CRT displays allocate this task to the computer.

The third distinction is in the forms of, and functions for, producing output. In a program editor, a line is displayed and printed as it is stored, line by line. In contrast, most text editors consider the displayed line and the printed line as two distinct, temporary units to be derived from the internal data structure. As text is edited, both the displayed and printed lines change. Related to the output criterion, are the text formatting capabilities offered by the editor. While writing and editing computer programs, the only functions required are a "new line" function and possibly a tabbing facility. Only upper case characters need by used. Text editing programs, however, should, and in fact do, provide more elaborate formatting functions, allowing for fairly intricate page layout to be performed online. In addition, a non-trivial routine is usually provided so that the text may be printed with changeable margins, paragraphs, upper and lower case, etc..

2.1 PROGRAM EDITORS

1. BRUTE (BRown University Terminal Editor)[9]

BRUTE is a good example of a basic program editor, designed originally for teletype terminals but currently operational using typewriters (IBM 2741 Communication Terminals) and small CRT displays (IBM 2260 alphanumeric consoles). It was modeled after the "Conversational Context-Directed Editor," described in section 2.3 below. Normal text can also be entered and edited with BRUTE, but complex text editing becomes very cumbersome, especially editing within a given line.

Since BRUTE was designed for program editing, the text is blocked (segmented) by line (80 characters/line). In the "input" mode, text is continually entered, a line at a time. In the "edit" mode, text can be inserted by line, one or more lines may be deleted, and a single line may be retyped. Via somewhat "unnatural" conventions, text within a line can be changed without retyping the entire line. However, only same length substitutions are allowed. Arbitrary insertions in the middle of a line are impossible. BRUTE's most restrictive feature is its traveling commands. One may scroll forwards or backward by line(s), but no commands

currently exist which search the text for context (e.g., go to a certain statement label).[1] The current location in the text is determined by a "line pointer" which changes as traveling and editing occur.

Even though BRUTE is operational on the 2260 console, it does not make full use of the CRT display, since the same commands and methods are used for the 2741; in particular, only one line of text is displayed for editing purposes at a time.

2. QED, Com-Share [8,20]

QED is a teletype-based editor which epitomizes what can be done with this type of system. It was developed with the express intention of providing maximum convenience for the user via a simple and mnemonic command language and line-independent access to the text. QED is internally line oriented. However, text is not stored as a fixed length record for each line; instead, the end of the line is delimited by a suitable marker. This seems to be the only reasonable orientation when working with teletypes. Text is still line addressable but the storage is more compact. A

---

1. As a result of some timely prodding, the "find" and "locate" commands of the "Conversational Context-Directed Editor," described below, have recently been added to BRUTE.

line can be up to 500 characters long, although only 80 are printed per teletype-line (i.e., it prints 80 character lines until it has exhausted the 500 characters).

In addition, QED has provisions for content addressing, providing a more natural means of locating sections of text. In this application of content analysis, text can be scanned for any occurrence of a label previously assigned (label addressing) or for a specified character string.

The editing commands include insert (one or more complete lines), delete (one or more complete lines), change (i.e., replace one or more complete lines), and substitute (replace only part of a line; this is not restricted to one-for-one character replacements but the replacement must be less than the teletype line). In addition, there are control characters which are used to make more complex (but still minor) editing changes within a line.

3. Interactive Programming Support System, System Development Corporation [3]

This editor has been designed for the IBM 2260 CRT display unit, making maximum use of its capabilities. As currently implemented, it is used to create and edit programs written in a procedure-oriented programming language (JOVIAL).

The editing repertoire includes insert, delete, replace, copy and move, all applying to one or more lines (but not within lines). The lines are numbered and same-length substitutions within a line can be made by specifying the line number, driving the "cursor" (a "typing bar") to the desired point with the "space bar" (compare with the "bucky buttons" of example 2 in section 2.2 below), and typing the correction in place. The line number is needed since an extensive correlation map between the data structure and the screen display is nonexistant. A continuous input mode (the "compose" mode) also exists during which consecutive lines may be inserted without having to specify the insert command for each.

The program tries to minimize user typing and, since it has been designed for a specific language, performs immediate syntactical checking, thus considerably reducing a programmer's debugging time.

Syntax checking is an important function of a program editor. However, any system having this capability is of necessity limited in the number of languages it can support. If the syntax checking were table-driven, however, all that would be required to add a new language to the system would be to add a new syntax table.

2.2 TEXT EDITORS

1. Magnetic Tape Selectric Typewriter (MTST), IBM Corporation [17], and ASTROTYPE, Information Control Systems [1]

These two "stand-alone" editing systems are similar in their design and intended users, and do not require the services of a large general purpose computer. As a result, they have a much broader potential market. MTST consists of a single IBM Selectric Typewriter connected to a small control/memory unit. The Astrotype system consists of up to four typewriters and memory units connected to one control unit.

In both systems the text is recorded on magnetic tape as it is typed in and can later be modified and printed in final form. Text can be typed in an "unchangeable" mode, in which case it is printed exactly as it was typed, or in an "adjustable" mode, in which case the control unit prints blocks of text (e.g., paragraphs) according to the current column width (which can vary). These systems were not designed for massive editing jobs. They are most useful for correcting minor errors in letters and in small, fairly

finalized reports. They are not suited for online composition or extensive editing.

For example, consider the editing functions of ASTROTYPE. Substitutions within a line are made by typing the line number, the old character string, and the new string. Insertions and deletions within a line are degenerate forms of substitution. Thus, to insert within a line, one must specify context to the left or right of the insertion point as the old string and repeat this context plus the text for insertion as the new string; to delete within a line, the text to be deleted must be typed as the old string and null text specified as the new string. However, editing changes must be fairly local since the line number and old and new strings must be typed on the same typewriter line. Verification is provided by printing out the line before the change is actually made.

Printing is done at the typewriter at the rate of 150 words/minute and various fonts and type sizes may be used. The printing can also be programmed to stop in the middle, so that additional input may be entered, and then continue. This is especially useful when changing the header of a form letter, for example.

2. CALL/360 DATATEXT, Service Bureau Corporation [4], System/360 Administrative Terminal System, IBM Corporation [21], and VIPcom, VIP Systems [27]

DATATEXT, ATS and VIPcom are three almost identical commercial editing programs which utilize the IBM 2741 typewriters as the interactive device. They are quite complete systems, allowing various forms of input and formatted output, and providing a fair set of editing and formatting commands.

ATS is essentially DATATEXT with the added facility for sending and receiving messages between individual terminals; VIPcom is basically ATS with an additional output capability which allows a final document to be photocomposed on a Photon 713 (see section 3.2 below)., The editing commands and data base of all three systems are virtually identical.

Each time a line is typed in, an internal line is created and a line number assigned. The length of an internal line may vary from 0 to 130 characters, and a text file may contain up to 9999 internal lines. The line numbers are not absolute and may change dynamically as editing is performed. This can be confusing since the line numbers may not correspond to the most recent printed copy. The designers of ATS suggest that editing be performed from the bottom of the file to the top; this eliminates the problem of changing

line numbers but is not a natural way of working.

Deletions of one or more lines are possible, but to insert new lines in the middle of a file, the lines must first be typed at the end of the file and then moved to the desired position. Substitutions within a line are made by typing the line number, the incorrect character string (plus any additional context that might be required to uniquely identify the text), and the correct character string (including the additional context used above, if applicable). A delete within a line is a substitution without a replacement string. Substitution is also used for limited insertions within a line. A pseudo incorrect character string is typed to identify the insertion point and these characters are then repeated as part of the insertion. Arbitrary insertions within a line are impossible. Text can be moved around by lines (e.g., moving a paragraph of text), but not copied.

Text can be entered in "formatted" mode, in which case the text can be arranged by the program to satisfy the specified page format (e.g., line justification), or in "unformatted" mode, where text is saved and printed exactly as it was typed in (useful for rigidly formatted material such as tables). An online printout can be stopped in the middle, allowing the user to type in additional text (useful

for form letters).

3. TVEDIT, Stanford University [16,24]

TVEDIT is one of the earliest (1966) time-sharing CRT-based text editors. Its command structure is extremely simple but not very extensive. No pointer device is used; instead, a special set of keys (called "bucky buttons") are used to indicate positions in the text as shown by a pointer symbol displayed on the screen. Editing can be done either on a line basis, e.g., "delete the next 4 lines", or on a character basis, e.g., "delete the next 6 characters." Insertions may be arbitrarily long (up to a full screen's worth) and may be made within an existing line (the screen area below the insertion point is blanked to provide a buffer for the inserted text).

Text is segmented by "pages" which are established by the user and can be of variable length (editing cannot be done across page boundaries but the boundaries may be deleted). The traveling commands are: go to page "n", scroll back and forward by screen, and search the file for a specific character string.

4. A Tablet-Based Editor, Carnegie-Mellon University [6]

This experimental editor uses hand-drawn proofreader's symbols as the method of editing text displayed on a CRT. The symbols are drawn on a Rand tablet and are recognized by the program by passing various characteristics of a symbol through a decision tree ("discrimination net"). The editing actions recognized include insert, delete, substitute, interchange, move and scroll. Since this experimental system was not meant for production editing, there are no real paging and file handling mechanisms, nor are there any output facilities.

As an example of how this type of editing is performed, consider the substitute function. The user draws a line through the text to be deleted. The system deletes this line, blinks the indicated text for verification, separates the text by opening a blank line, and inserts a cursor enabling new text to be typed in from the keyboard.

The conclusions reached by this experiment were that vocal input of text is practical, even with high error rates, since any errors are easily corrected, using the tablet editing system, and that a self-sufficient editing terminal requiring computer intervention only to "fetch, format, and store text" could be constructed (since the

decision tree and primitive editing logic could be made part of the terminal's hardware). However, the cost both in CPU utilization for online recognition or in hardware development makes this form of editing as yet impractical.

5. The Hypertext Editing System, Brown University [5,21,25]

The Hypertext Editing System is the topic of Part II of this paper, but, for completeness, its characteristics will be given here. It is a flexible, CRT-based (IBM 2250), system allowing full editing and formatting capabilities. It is oriented towards typeset output (currently using a computer line printer) as well as flexible input and online editing and browsing. A lightpen and a set of "function keys", under program control, are used to indicate to the system the nature of the edit to be performed. The data structure is entirely line and page independent; text is organized into user-designated units called "text areas". Arbitrarily sized edits are theoretically allowed, although certain implementation restrictions have been imposed (e.g., a maximum of approximately 2500 characters may be deleted or rearranged at a time).

2.3 TEXT AND PROGRAM EDITORS

1. Syntext Editing System, Argonne National Laboratory [14]

The stated goal of the Syntext Editing System[2] is the minimal intrusion of the system on the work at hand. This precluded the use of typewriter-like terminals as the interactive medium; instead, a CRT display (IBM 2260) was chosen. Its basic editing features were borrowed from the TVEDIT program described above. The text file would be constructed of variable length lines (in this case, up to 255 characters per line).

Simple commands would be used to change what is seen on the screen, either by searching or by scrolling. Names (labels) would be assigned to sections of text and later retrieved by name. One or more lines could be inserted or deleted at a time. Somewhat cumbersome commands would allow modifications to be made within a line, but arbitrary insertions would not be possible. The system is also restrictive since it would not have a lightpen-like device for pointing, and thus a command language and context identification would be required.

The novel feature of the proposed Syntext Editing System

---

2. This system is about to be implemented.

is its "recursive region" structure (similar to the structure used at the Augmented Human Intellect Research Center, described below). A region is defined to be a piece of text or a sequence of subregions, where a subregion is just another region. This design follows the properties of Fortran's nested DO loops and PL/I's nested procedures. For textual material, regions can be thought of as chapters and can be referenced individually, while the sections of the chapter would be subregions. The overall organization of the document could then be viewed by displaying only the "skeletons" of the regions.

2. A Conversational Context-Directed Editor, IBM Cambridge Scientific Center [7]

This editor was the model for the BRUTE editor described previously and, as such, has basically the same data structure and all of BRUTE's facilities (and most of its limitations). It is used both as a program editor and a text editor. Replacements within a line do not have to be of the same length (but if the line length exceeds the value for the particular file type, the extra characters are truncated). There also exist two useful "jumping" commands which search for specified character strings occurring either in a fixed line position ("find") or free-form

("locate"). If a match is found, the line pointer is moved to the desired line. Commonly used command requests may be given a one-letter name (X or Y) and later executed, repeatedly if desired. A facility is also provided for printing special characters that are not on the standard keyboard (with IBM's 1403 line printer and the "TN" print chain).

Since this editor is also designed for program composition and editing, it provides flexible tabbing facilities which require less typing for the user when column-dependent languages such as FORTRAN are being used. A tabbing facility also makes a programmer more likely to indicate block structures and nesting levels by suitable indentations, a useful documentation technique. Depending on the language, default (internal) tab settings are automatically applied to the file. For example, if the program were being written in FORTRAN, the tab stops would be at positions 7,10,15,20,25 and 30.

3. Augmented Human Intellect Research Center, Stanford Research Institute [11,12]

The work being done by the AHI group at Stanford University is most impressive. Their system embodies much more than just an editor. It is a new way of thinking and

working: utilizing the power of the computer in all aspects of one's work. However, only the editing subsystem is described here.

The work stations are standard television monitors, driven by small CRT's, equiped with a "mouse" (a hand-held X-Y transducer usable on any flat surface) for pointing to text and a one-hand five-key handset for specifying commands to the system. The handset can also be used to input text, although a standard keyboard is provided.

The text files are arranged into explicit hierarchical structures. The basic unit (a terminal node in the tree) is the "statement". Statements are grouped to form a section, sections form a chapter, etc. References may be made among any elements in the hierarchy. Various ways of viewing the text exist (determined by "viewspecs", e.g., displaying only the first "n" levels of the tree) and elaborate methods for jumping around in the text are available (by content searching, by the tree structure, etc.).

The editing commands are extensive and very much specialized, e.g., insert character, insert word, and insert statement are all separate commands. Since the text is hierarchically arranged, rearrangements of the structure are easily and economically made (only the associated pointers are changed, the text remains intact).

3. COMPUTERIZED TYPESETTING

3.1 BACKGROUND

While editing programs have speeded up the input and editing of manuscripts, work has also been done to improve the output end of the publishing process. The art of printing is 12,000 years old, but only recently has the original method of relief on blocks or plates been replaced by the photo-offset method and the electron beam. In conjunction, computer programs have been developed to perform necessary tasks of page layout: hyphenation, line and column justification, etc. First, a brief look will be taken at the history of printing and the development of photographic and electronic typesetting,[3] and then two computer programs for typesetting will be discussed.

Printing originated in China in the eighth century, using clay plates and wooden blocks in relief. In the eleventh century, the Chinese invented movable clay type, which was the only real innovation in printing until 1948, when

---

3. A more complete summary of typesetting is given in reference [28]; for a brief tongue-in-cheek history, see "The First Page, Followed by Others," subtitled "The Complete History of the Graphic Arts, except for printing, which has its own problems" [13].

photo-offsetting was introduced. In between, printing was brought to Europe by Gutenberg in the fifteenth century, lead became the relief medium allowing movable type to be reused, and mechanical typesetting devices were constructed.

The first mechanical typesetter was introduced in 1849 by Soerensen, although it never really worked. The pieces of type were notched to identify each character, brass rods were used to select and guide the transport of the type to the line holder, and the type was returned to the rack when done.

The first successful mechanical typesetter was designed by Mergenthaler in 1885. It was called Linotype and the design principle is still used today. It is operated from a keyboard; thus, when a key is struck, the corresponding character is delivered from an overhead rack to the line-collector. The operator justifies the line by inserting expandable wedges within the line. After the line is filled, the character matrices are locked in place and transported to a chamber where molten metal is introduced under pressure. Thus the line is cast in one piece, and, after cooling, the cast line is ejected into a tray. The relief matrices are returned to the overhead rack via a sorting machine. While one line is being cast, the operator is assembling the next line with the keyboard.

The Linotype system is still used today to typeset many newspapers, which are constructed of regular columns and simple text.

However, Linotype is not very well suited for complicated, irregular formats such as textbooks, and in addition is quite noisy, hot and cumbersome. These shortcomings were removed to a certain extent with the introduction of Monotype by Lanston. This system was based on Hollerith's use of punched cards for the 1890 census. Using Monotype, the operator punches the characters into paper tape. The machine adds up the width units and gives what keys to strike for word spaces to justify the line. When the line is completed, the punched tape is inserted into a casting machine, which selects the character matrices one at a time, pushes them into a mold where molten metal forms the line, after which the cast line is deposited onto the galley.

3.2 PHOTOGRAPHIC TYPESETTING

Monotype eliminated the noise and heat problems but did not greatly increase the speed of typesetting. The first major breakthrough, increasing the speed by an order of magnitude of 10, was photographic typesetting, introduced in 1948.

Several different varieties of machines have been developed, but all use the same principle. A matrix, containing all the characters in photographic images, is scanned with a stroboscopic lamp that flashes on the images of the selected characters. The character image produced is focused through a series of lenses onto a photo-sensitive plate. Depending on the system used, a line or a page is accumulated before the film is advanced. After the plate is developed, only the character image remains. Ink is deposited on the images, after which the plate is rubbed on a rubber blanket that picks up the images and serves as the printing surface.

The lens makes magnification possible from the master character matrix, thus providing various type sizes. In addition, the matrix can contain various typefaces which can be selected by dialing the desired fcnt. The entire process can be operated either from a keyboard or by computer, using operator-produced tape to drive the typesetter. Companies

currently marketing photocomposers include the Mergenthaler Linotype Co. (the Linotron Photocomposer) and Photon, Inc. (Photon 200 to Photon ZIP 901).

Typesetting speeds of up to 500 characters per second can be achieved using the photographic method. However, the speed is limited by the speed of the operator. A good human operator can only set approximately one and a half characters a second, while a computer can realize the maximum speed. This fact has led to the development of electronic typesetting.

3.3 ELECTRONIC TYPESETTING

Electronic typesetting is a recent innovation in the printing industry and several companies are now actively involved in its development. The companies include RCA (the VideoComp) [26], the Mergenthaler Linotype Co., the Intertype Co., and IBM (in cooperation with Alphanumeric Incorporated) [10].

Various methods are used in electronic typesetting. In some, the character is painted on a cathode ray tube with vertical strokes, while others use horizontal strokes. The screen is an optically flat disk coated with aluminum

phosphor. With each stroke, the electron beam produces an extremely fine line (less than 1/1000 inch wide) on the screen. The stroke, produced by moving the beam for a brief instant, is very short. The machine records the characters on film. For projecting to the film, some manufacturers use a stationary lens, others a moving lens, to pick up the characters as they are painted; some typesetters use continuously moving film, some photocompose one entire page at a time on stationary film, and still others advance the film for each line.

The virtue of the electronic system is that the beam can paint any graphic material, not only type characters. It can easily generate tabular or columnar displays, line drawings and even halftone pictures,

Electronic typesetting requires a computer-type memory in which to hold the fully detailed instructions for the painting of each character. Entire fonts in various styles are stored and selected as the text is composed. The operator only need type as if the text would fit on one line. The computer justifies the line (and in many systems even provides hyphenation) and writes the justified line onto magnetic tape.

Once the computer was introduced into the printing process, it was only natural for it to assume greater

responsibility. As a by-product of electronic typesetting, computers can compile indexes, bibliographies and various lists by noting the key terms and where they occur. One good example of increased productivity as a result of maintaining a computer-stored data base is a telephone directory where, after arranging the items in alphabetical order, the computer can also produce additional directories arranged by districts, street numbers or telephone numbers.

The affects on the publishing industry have been substantial. The throughput time for books has been reduced from a year to less than three months. A newspaper page can be set in less than 5 minutes. It would even be possible to publish an international newspaper since the completely typeset paper could be sent world wide by telephone or wireless to local publishers who could electronically reproduce the newspaper, adding any local stories they wished.

3.4 COMPUTER TYPESETTING PROGRAMS

Another application of computers in the field of typesetting is the creation of programs which produce as output "typeset" documents via the computer's line printer. Two of these, TEXT360 [23] and FORMAT [2], both developed by the

IBM Corporation, are discussed here. These programs have proven to be highly effective for documents which do not require the highest quality printing and are economical for installations which wish to publish their own documents "inhouse", without requiring the services of a commercial printer.

3.4.1 FORMAT

FORMAT is a fast, one-pass text processing program written in Fortran IV. It performs line justification ("on-the-fly"), can format up to eight columns per page, and has an automatic capitalization facility. Input consists of text and Command Words, while the layout for the document is determined by appropriate "control cards" in the input stream. The input is completely free-form.

FORMAT has no provisions for hyphenating words at the end of a line. This was a design decision, producing a smaller, faster processor.

Programmed hyphenation algorithms usually are a combination of logic and table lookup. The logic separates suffixes and prefixes, looks for double consonants, etc. However, the logic, of necessity, is dependent on the language used and this is quite restrictive. Table lookup

is needed for proper names and exceptions to the logic rules. No hyphenation algorithm can be perfect: how does one hyphenate "PRESENT"? The context must be known.

In spite of these difficulties, good algorithms can obtain accuracy rates of 99%, a better rate of success than most humans are able to obtain (although when the algorithm fails, it does so badly, e.g., line 7 in the first paragraph of section 3.4.2 below). However, these routines are very large, especially if table lookup is needed. A hard look should be taken to decide just how necessary hyphenation actually is. It adds nothing to the meaningfulness of a document (i.e., no ambiguity is introduced if hyphenation is absent). Its sole purpose is to provide fewer "holes" on the printed copy, a problem only with narrow columns. For these reasons, the lack of hyphenation routines is not considered by the author to be a serious drawback.

Additionally, the FORMAT program has an editing facility with which typographical errors can be corrected and revisions to a document made. It also provides two useful indexing tools. The first is an alphabetized list of words in a document longer than three characters (with certain exceptions) and the number of times each occurs. An indexer would use this listing to decide which words were to be included in the index. He would then use the "locate"

facility to produce a listing of the page numbers on which each specified word occurs.

3.4.2 TEXT360

TEXT360 is a similar text processing program written in Programming Language/I (PL/1). It contains more elaborate formatting options and automatic hyphenation but at the expense of being much slower (requiring five passes to produce the final document). It has been used as part of the Hypertext Editing System at Brown for the past year and has proven to be quite useful (with only occasional "blowups"). Its major drawbacks from the production point of view are the fairly high cost in computer time and its quite large space requirements.

TEXT360's input is, moreover, free-form, containing text intermixed with format codes. The "edit" codes are analogous to FORMAT's Command Words, while the "alter" codes correspond to "control cards". TEXT360's format codes are in general more mnemonic than those of FORMAT; for example, the code to center a line of text is C for TEXT360 and M for FORMAT. However, TEXT360 has more characters reserved to denote the formatting codes, thus requiring the user to remember more conventions for their literal representations.

TEXT360 has updating and indexing facilities similar to those of the FORMAT program and, in addition, produces an automatic table of contents on request.

4. CONCLUSION

Recent advances in two phases of manuscript production, editing and typesetting, have been examined with respect to increasing human productivity. However, initial input of text and final printing lag far behind in speed.

Initial input is still predominantly manual (by keyboard), although various electronic scanning devices are now being developed which convert printed text directly into machine-readable form. The scanning devices are at present extremely expensive and prone to error. At the same time, they are not capable of "reading" normal (i.e., concatenated, cursive) handwritten text and, consequently, the manuscript must initially be keyboarded. However, these devices are extremely useful for converting already existing manuscripts into machine readable form for storage and revisions.

A possible input procedure which has only recently been investigated is the recognition and translation of vocal

input. This is an extremely complex problem which, when solved, will result in a very powerful and fast input device. It should be noted that the input stage of manuscript production is not the major bottleneck in the process since input devices (typewriters, keypunches, time-shared terminals, etc.) are far more prevalent and less expensive than suitable editing, typesetting and printing procedures and equipment.

Since editing and typesetting can now be done electronically, there is a greater need for new printing (i.e., imprinting) techniques which can keep up with the increased amount of information being generated, and do so electronically, so that the typeset document can be immediately printed. The vast amount of printing is still being done with the rotary press but when new techniques are developed, the complete publishing cycle will be handled electronically. Electrostatic printing devices are now being developed which will fill this gap.

A recent example (probably unique in the industry) of how these processes fit together is the American Heritage Dictionary, recently published (September, 1969) jointly by American Heritage and Houghton Mifflin, under the guidance of Inforonics, Inc. [19]. The dictionary, consisting of 1600 pages, 155,000 entries, approximately 4000 illustra-

tions and 11,000 distinct characters, was edited using a single CRT display on a PDP9 computer and an editing program developed by Inforonics, and typeset using a Photon 560 photocomposer. The editing was still done by first indicating the changes on hard copy and then having the changes repeated at the display console. It was a four million dollar, five year effort, and the dictionary was delivered two weeks early. Although many crises had to be met and several innovations made (e.g., a new type of paper had to be developed), this project illustrated that electronics can become an integral and viable part of the publishing industry.

In summary, much progress has been made in reducing the time necessary to process the immense amount of information being produced today. However, much more attention must be given to making an author's tasks as pleasant and productive as possible. No commercially viable multi-display console system for the flexible compcstion and editing of textual material currently exists. This is considered by the author to be a most severe problem.

5. BIBLIOGRAPHY, PART I

1. ASTROTYPE, Form No.30, Automatic Office Division, Information Control Systems, Inc., Ann Arbor, Michigan.

2. Berns, Gerald M.: A Description of FORMAT, a Text-Processing Program, Communications of the ACM, Volume 12, No.3, March, 1969, p. 141-146.

3. Bratman, Harvey, Hiram G. Martin, and Ellen C. Perstein: Program Composition and Editing with an Online Display, Proceedings of the 1968 Fall Joint Computer Conference (Volume 33, Part 2), p. 1349-1360.

4. CALL/360: DATATEXT Introduction, Form No.65-2259, Service Bureau Corporation, New York, 1969.

5. Carmody, Steven, Walter Gross, Theodor H. Nelson, David Rice, Andries van Dam: A Hypertext Editing System for the /360, Proceedings of the 2nd Annual Conference on Computer Graphics, University of Illinois, March, 1969.

6. Coleman, Michael L.: Text Editing on a Graphical Display Device Using Hand-Drawn Proofreader's Symbols, Proceedings of the 2nd Annual Conference on Computer Graphics, University of Illinois, March, 1969.

7. "A Conversational Context-Directed Editor," IBM Cambridge Scientific Center Report, Form No. 320-2041, March, 1969.

8. Deutsch, L. Peter and Butler W. Lampson: An Online Editor, Communications of the ACM, Volume 10, No.12, December, 1967, p. 793-799.

9. Dorin, Robert and Donald Smith: "The Brown University Terminal Editor (BRUTE)," The Computing Laboratory, Brown University, October, 1968.

10. "Electronic Composition in Printing, Proceedings of a Symposium," National Bureau of Standards Special Publication 295, February, 1968.

11. Engelbart, Douglas C.: "Study for the Development of Human Intellect Augmentation Techniques," Stanford Research Institute (preliminary draft), July, 1968.

12. Engelbart, Douglas C. and William K. English: A Research Center for Augmenting Human Intellect, Proceedings of the 1968 Fall Joint Computer Conference (Volume 33, Part 1), p. 395-410.

13. "The First Page, Followed by Others," RCA Graphic Systems Division, Dayton (N.J.), 1969.

14. Hansen, Wilfred: "Syntext Editing System," preliminary draft, not yet published.

15. Lessing, Lawrence,: The Printed Word Goes Electronic, Fortune Magazine, September, 1969, p. 116-119,188-190.

16. McCarthy, John, Dow Brian, Gary Feldman and John Allen: THOR -- a Display Based Time Sharing System, Proceedings of the 1967 Spring Joint Computer Conference (Volume 30), p. 623-633.

17. Magnetic Tape Selectric Typewriter, Form Nos. 543-0510-1, 543-0515, 549-0204, and 549-0700, IBM Corporation, New York.

18. Nelson, Theodor H., in an informal communication, February, 1969.

19. Producing a $4 Million Book, Publishers' Weekly, September 1, 1969, p. 56-58.

20. QED Reference Manual, Com-Share, Ann Arbor Michigan, Ref.No.9004-4, January, 1967.

21. Rice, David E.: "A Manual for the Format Phase of the Hypertext Editing System," Center for Computer & Information Sciences, Brcwn University, Providence, File No. HPS360-1, May, 1969.

22. System/360 Administrative Terminal System, IBM Corporation Application Program, Manuals: "Application Description Manual", No. H20-0297-2; "Terminal Operations Manual", No. H20-0589-0; "Program Description Manual," No. H20-0582; "Console Operations Manual," No. H20-0590-0.

23. TEXT360 Reference Manual and Operating Guide, IBM Corporation, New York, 1968. Program Number 360D-29.4.001.

24. Tollivar, Brian: TVEDIT, Appendix I of the THOR Reference Manual, Stanford Computation Center, August, 1966, p. 104-107.

25. van Dam, Andries: "A Manual for the Edit Phase of the Hypertext Editing System," Center for Computer & Information Sciences, Brown University, Providence File No. HES360-1, May, 1969.

26. VideoComp System 70/830 Series Reference Manual (73-06-001-P), RCA Graphic Systems Division, Dayton (N.J.), August, 1969.

27. VIPcom User's Guide, VIP Systems, Washington, D.C., August, 1969.

28. Walter, Gerard O.: Typesetting, Scientific American, May 1969, Vol. 220, No 5, p. 60-69.

PART II: THE HYPERTEXT EDITING SYSTEM

1. INTRODUCTION

The Brown University Hypertext Editing System is an interactive man-machine computer system for text manipulation, display, and typesetting (via the computer's line printer). It is both a sophisticated system for the composition and editing of manuscripts, and a reading machine on which to browse and query written materials having arbitrary complex structure.[4]

The author has been primarily responsible for the typesetting aspects of the system, in particular, the formatting and printing capabilities. These are described from a macro viewpoint in section 4 below. A detailed description of how the capabilities work in practice is contained in a user's manual written by the author.[4]

In the Hypertext Editing System, any text structures may be interconnected (linked) in arbitrary ways, and a user may proceed through a text along connections in this linkage structure. This linkage structure is involved in all uses

---

4. Much of the following material was adapted from [1].

of the system, since the conventional forms of text division, in particular, lines, pages, and page numbers, are completely absent.

Non-sequential texts with complex linkages among their text fragments are called "hypertexts" ("the combination of natural language text with the computer's capacities for interactive, branching, or dynamic display... a nonlinear text... which cannot be printed conveniently... on a conventional page..."[3] so as to distinguish them from the more conventional "linear" manuscript form found in books or manuals. Because the user may view and revise ordinary written texts at the screen, as well as create, edit, rearrange and print hypertexts, the system is called the Hypertext Editing System-- a name meant to embrace all of these activities.

The system uses a standard computer and CRT display (IBM 360/50 and 2250 display) and has been designed for economical use in a multi-programming environment where its demands on the main computer are comparatively light (the present system averages 5% CPU utilization).

The user sits facing the 12" by 12" screen and browses in a variety of ways through (portions of) arbitrarily sized texts. He has access to data sets of up to many hundreds of disk tracks in size (twenty 7200-byte tracks, a normal data

set size, is equivalent to about forty pages of ordinary text). The text is taken from disk at the beginning of a session, copied to a work data set, brought into core as required, and returned to the archival data set, if specified, at the end of a working session. Original text input may be entered into the system via the display's alphanumeric keyboard, or through any device producing standard character string input.

The user controls the system by pressing keys on a programmable function keyboard, by pointing at the text with a lightpen, and by entering text via the alphanumeric keyboard, as required for specified operations. "Prompting" messages keep the user oriented as to the status of the system and to the actions available to him at any instant.

2. AN OVERVIEW OF THE SYSTEM

2.1 The Hypertext Aspects of the System

The aim of the Hypertext Editing System to create a general-purpose text handling and editing system for use by writers, editors, and on-line readers. The system was designed to be easy to use, to run on standard equipment,

and to fulfill usefully all the complex needs of the editor or creative writer.

In a hypertext system, writings are stored in a central computer pool, where they may be easily reached, viewed, changed, and automatically printed; they may be arranged in conventional manuscript forms such as articles, manuals, or books, or they may be assembled in more complex arrangements, i.e., hypertexts.

An author might use the hypertext form, for instance, to write sections of his manuscript in the order in which they occur to him, or he might work on several sections "at the same time", linking them in final form only at the moment that he has to go to press.

Another instance of the hypertext form is an interlinked group of patent documents constructed in such a way that cross-references between the various patents can be immediately followed.

Similarly, an online reader is not constrained to read the hypertext in any particular order and may follow optional trails (trains of associated ideas and expositions) through the manuscript, at his choosing, as well as at the author's suggestion (choices may be indicated with brief annotations). It is very easy for an author to construct different sequences of text constituting 'alternative ver-

sions' of the same material, all of which may exist simultaneously, and are to be selected by the author or another reader according to his mocd or his background. Thus the author has many more trial and error capabilities available to him than is the case with conventional modes of writing and publishing.

2.2 The Writing and Editing Aspects of the System

Considering just the editing capabilities of the Hypertext Editing System, there is a difference in kind between this approach and the standard computer approaches to editing.

Most of the computer editing systems previously described are designed for use with minimal equipment, such as the teletypewriter. In contrast it was thought desirable to experiment with a system utilizing the full capabilities of a high-powered computer display. An editing system which might suffice for programmers will not fulfill the needs of writers. Most computer editing systems, especially the so-called "line" or "context" editors, (e.g., BRUTE, QED, etc.), are primarily used with formalized or stylized texts-- such as computer programs themselves-- where line

numbers, keywords, or labels must be used as sure guides to content. From a writer's or editor's point of view, it is not deemed desirable to inflict line numbers on the user, or to make him "program" little changes in his data (as by typing in substitution operators). Such systems require great patience and saintliness on the part of the user, and run less by computer power than by the power of positive thinking. The Hypertext Editing System instead tries to maximize the ability to effect changes by minimal, rudimentary, and, hopefully, natural actions.

An attempt has also been made to allow the user to make reference to his work conceptually, by sections or by context of ideas, however he feels is natural.

The philosophical position is essentially that the writer is engaged in very complicated pursuits, and that this work legitimately has a freewheeling character that should not be encumbered with irrelevant restrictions on size and structure of text or operations-- ideally, "anything goes" as long as it is well defined.

Therefore it was decided to provide the user with unrestricted "spatial" options, and not bother him with arbitrary concerns that have no meaning in terms of the work being performed. The entities he would be concerned with would correspond to the content of conventional writing:

words, sentences, paragraphs, sections, and also non-structured, arbitrary fragments of text to be rearranged and spliced into appropriate combinations. He would not encounter line numbers, page numbers or footnote numbers, all of which are extraneous artifacts of conventional writing "hardware", that is, paper.

His activities, too, would correspond to the operations ordinarily performed upon text by writers and editors. He would be able to perform manipulations directly upon pieces of text: correcting, moving, linking and copying, etc. Such actions would correspond directly to the "scissors-and-paste" operations of rearranging manuscripts. In addition, the writer would be able to do various other things which are usually very costly in time and/or money, or downright impossible: file previous drafts, spin off alternative versions for separate tinkering, and communicate between separate versions, lifting or replacing sections as desired.

In terms of scope, the only other system known which attempts to accommodate the writer and editor to such an

extent is that of the AHI group.[2][5]

---

5. One of the differences between HES and the AHI system is that the AHI system imposes a hierarchical tree structure of individual statements upon all textual contents, in contrast to the unstructured "string" of text approach of HES. While many benefits accrue from this automatic structuring, it represents an arguable theory about human thought, and constitutes a preconstraint upon the user.

3. THE SYSTEM IN BRIEF

3.1 Structures

The system does not store text in numerical pages or divisions known to the user, except as he may deliberately number his headings, etc., as has been done here. It thus allows the user to divide and link his text in ways entirely of his own choosing. In this way he may create, or represent, any maze of interconnections among ideas and items. Rather than filing these materials by page number (as in books) or formal code name (as is usual in computers), they are stored either as fragments ("strings") or as logical units (such as chapters, or sections) with logical connections or pathways linking them together. Such paths may lead from anywhere to anywhere else, and permit the user to arrange the information according to the structure he sees in it. Finally, the system permits cross-accessing among materials contained in it, i.e., immediate access among cross-referenced writing.

Areas

An area is a space containing a segment of text. It may be of any length, expanding and contracting automatically to accommodate however much text is put into it. Areas may be connected in two ways: by links and/or by branches. A link goes from one point (its point of departure signified by an asterisk) to an entrance point in another, or the same, area. A branch goes from a branch-point at the end of an area to an entrance-point in another, or the same, area. Branches terminate areas and force a choice of "next segment" on the user; links are optional paths embedded in the text. A point within an area may also be given a name (label), and later summoned by name to the screen.

The user goes from area to area by "travelling" ("jumping") via a link or a branch, or by "getting" a label. The text is repositioned on the screen, so that an entrance-point is always at the upper left corner of the screen, with as many text lines following it as will fit on the screen (on the 2250/I, a maximim of 38 lines of up to 74 characters each).

Annotations

The second type of structure is called an annotation. Annctations are short pieces of text attached to points in the text or to connections, i.e., to links and branches.

1) Tags:

An annotation to text is called a tag. Although tags may be specified at any time, in the current implementation of the system, they have been chosen not to be printed.

2) Explainers:

An annotation to a link cr branch is called an explainer. It currently is specified only when the link or branch is created, and is not printable in the current implementation of the system.

Tags and explainers are completely free as to contents, but currently dc not have the general hypertext properties. This is the reason for not including them in the printed text. They are thought of more as margin notes, as guides for the online reader and editor.

3.2 Operations

There are four types of operations open to the user.

1) He may "travel", going from place to place in his text structure. This means, of course, that the text being shown on the screen is replaced by the text to which the user has travelled. Travelling can occur within a single area or between areas. In addition to the various methods for scrolling (i.e., turning pages within an area) there are four jumping functions:

LINK, which allows the user to jump along a link;

BRANCH, which enables the user to select a branch from those at the end of an area;

GETLABEL, which jumps to an area entrance-point whose previously defined name the user selects from a menu on the screen;

RETURN, a special command, which takes the user back along the remembered trail of LINK, BRANCH or GETLABEL operations to areas seen previously.

2) He may change the structure of previously written material, or create new structures, using "structure commands." These create links, branches and labels.

3) He may "edit" the text, perfcrming insertions, deletions, and their various combinations having different

names: copy, rearrange, substitute. The editing operations are carried out as the user executes them, with the ability to approve them (say, for appearance, or phrasing) before they are finalized. The edits are not restricted as to size, except for certain implementation constraints (e.g., only one screen may be rearranged or deleted at a time).

4) He may "format" the text for ease of reading on the screen and for final output, specifying margins, number of columns, paragraphs, indentations, types of headings, underscoring, etc. This aspect of the system, programmed by the author, will be discussed in greater detail below.

3.3 Prompting

So that the user will know what he may do next, his alternatives are continually displayed in a prompting area at the bottom of the screen. Emphasis was placed in the design on good "human factors" techniques -- in this case having a controlled system directing the user's every move without loss of effectiveness. (Even a non-programmer novice is able to use the system productively after 10 minutes of instruction).

3.4 Original Input

Text material may be entered into a hypertext in a variety of ways. For instance, the user may enter text directly from a console keyboard, or he may copy selected portions from previously established files. He may also have his rough draft copy keypunched (with or without detailed format codes) and read into the system. Finally, at Brown University, he may use BRUTE, described previously, to produce raw text files for subsequent input to the Hypertext Editing System.

4. FORMATTING AND PRINTING A HYPERTEXT

4.1 Formatting

The user cannot format text elaborately for display within the current Hypertext Editing System since the 2250/I display unit has only upper case letters[6] and since the aspect ratio and center separation of the characters differ so widely from those of the line printer's printchain. In addition, many of the options would be impratical to implement (e.g., performing line justifcation and hyphenation online) and a few are impossible (e.g., displaying a single column of text 80 spaces wide on a display unit whose width is only 74 characters). Thus, only a subset of the available formats are shown on the display, including indents (both regular and hanging), paragraphs, lines skipped, and left justification (the display is formatted as ragged right without word truncation). This choice has proved satisfactory and no major complaints have been raised in the nine months the system has been operational.

However, the user can format text for elegant printout

---

6. The system is now operational on a 2250 Model 4, having software upper and lower case character generation. Thus the upper case restriction is really only a function of what display device is available.

utilizing IBM's TEXT360 Program. The TEXT360 Program provides printout with such formating as upper and lower case, indentations, underscoring, special characters, line centering, line and column justification, hyphenation, page-numbering and an automatic table of contents.

The formatting phase of the system permits the user to assign the varicus formats to characters and strings within the text. The assigned format codes are not distinguished one-from-another on the screen. This would require complex display conventions which could only result in a grotesque appearance. Primarily, there are not as many readily discernable markers as format options. In addition, if 50 different marker types were used, it would be quite difficult to distinguish between markers and text. And, of course, a typical user could not be bothered to learn what marker stands for what option.

It was decided instead to represent the formats with a couple of markers, and allow the user to proofread formatting specifications when necessary by inquiring with an INQUIRE button as to the individual meaning of each.

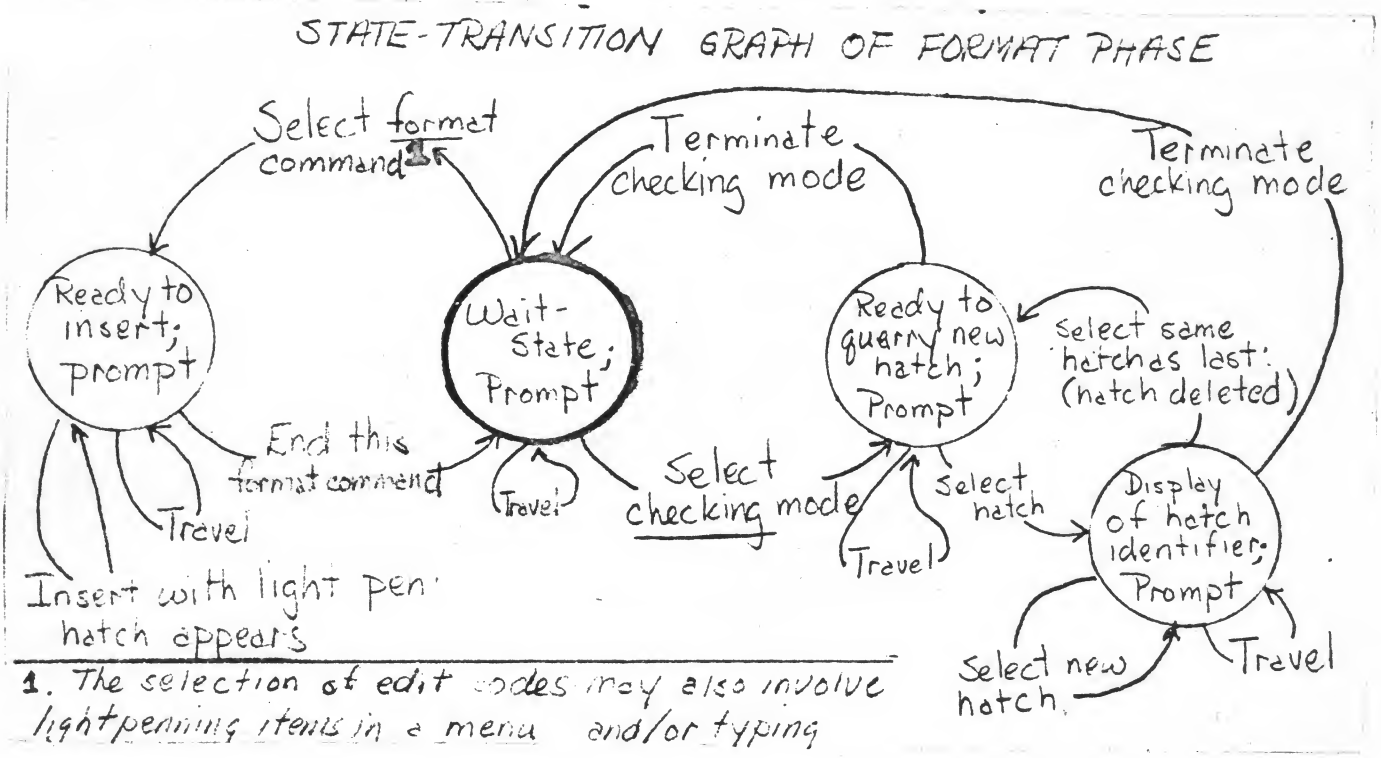
In the format insertion mode, the user selects the format code to be assigned (such as capitalization of first letter, all caps, underlining, indentation) and inserts it by lightpenning a point in the text. A marker appears each

time he points, indicating that the format has been added to the data structure. Upon hitting the desired function key only once, he may insert the same code repeatedly by pointing at successive locations. After each penpointing a new marker appears.

In the checking (inquiring) mode, he may inquire as to a hatch's (#) meaning by pointing at it. A statement of its meaning will then be displayed in the prompting area. Pointing twice in succession at the same hatch will delete it, both from the screen and (as a format code) from the data structure.

The display of hatches may be suppressed with a function key. Also, the user may flip back and forth between the editing and formatting phases using a single function key. Thus formatting is in actuality a special case of insertion in which the format codes are inserted into (and deleted from) the data structure.

A state diagram of the Format Phase is:



4.2 Printing

Upon command, the "printing" program compiles the corresponding TEXT360 input source string from our data structure, and transmits this augmented text to the TEXT360 program which does the actual printing.

The hypertext Printer is in actuality a translator which transforms all the text and formatting of a hypertext into a lineal string of characters recognizable by the TEXT360 program. This eliminates the need for learning the somewhat

mysterious TEXT360 format codes, and reduces the probability of typographical errors. Thus, a user does not have to know anything about the TEXT360 program or text formatting to produce reasonably "pretty" hard copies; and secondly, if he desires quite elaborate printout, the only concepts required are those of column widths, margins, tabs, and running heads and foots, or precisely those concepts (hopefully) understood by most competent secretaries.

The hypertext Printer reinitializes itself after completing a printout (signified by an audible "fweeeep" at the scope), so that in the same session several different printouts may be obtained.

Due to the non-lineality of a hypertext, certain conventions have been established in the current implementation so that the text can be printed lineally. As a default option, the first branch at every branch point is followed while printing. However, the user can designate a different branch level to be taken by the Printer. In this way, alternate versions can be printed with no extra work required by the user.

At print time (though not necessarily at any other time), a linked section of a hypertext is assumed to be a footnote and is generally printed as such, with the convention that the footnote is terminated when another link, a branch, or

the END of the hypertext fragment (area) is reached.

Normally, if the user desires an expanded footnote in his Hypertext, but only wishes the first few lines printed in hard copy, he would signify the end of the text for the printed footnote by inserting, at the end, a link to any other point on the scope page. Thus the complete link would be shown on the display, but only the designated portion would be printed as a footnote.

If, however, he wished for the entire linked section to be printed (in-line), he would have to make a branch to the linked section and another branch back to the main-line text from the end of the linked section.

Future schemes will permit the printing of arbitrary hypertexts, by including link-identifiers and branch-identifiers in a directory of links and branches, and thence back to the proper entry-points in the overall printed document (cross-reference table).

5. PAGING AND DATA STRUCTURE

The system uses a list processor with automatic paging control. Currently a given area of a hypertext consists of one or more pairs of pages, each pair consisting of a command page and a text page. The paging subsystem retrieves from disk such pairs of pages corresponding to an area of the hypertext selected by the user. Pages not currently being viewed by the user may have to be stored back on disk as new pages are requested and brought into core. Pages are stored on disk only when there is not enough space in core to bring in a desired page (or when the user has finished). To make better use of storage (both in core and on disk) the size of individual pages varies as the user inserts and deletes text. In addition, free space on disk is collected and reused.

The command (orders) page of each pair for a given area contains command codes with pointers to text data in the text page, while the text (data) page contains the actual text data, probably somewhat scrambled. In the current system the commands are ordered sequentially and thus reflect the sequential structure of each text area, while the text is ordered as it was typed in.

A pointer consists of a displacement (from the top of the

text page) to a string of text, and the length of that string. As text is inserted, the appropriate command code and a pointer to the text in the text page are inserted in their sequential place in the command page, while the text itself is inserted at the end of the text page.

When either the page of pointers or the corresponding page of text becomes too large to be efficiently manipulated, an additional pair of pages is created and the area is now spread across both the old and new pairs. As this spreading occurs, unused space is eliminated, the pointers are compacted, and the pieces of text are reordered (i.e., garbage-collection is performed).

6. CONCLUSION

The Hypertext Editing System, an experimental prototype, has been used extensively in the last year to produce "inhouse" approximately one hundred and fifty documents ranging in size from several pages to several hundred pages. The major drawback of the system is its inability to support more than one terminal. Therefore, a multi-console version of the system, called FRESS (File Retrieval and Editing SyStem), is currently (Fall, 1969) being implemented.

This system will support a variety of terminals, including: IBM 2741's (primarily for original input), IBM 2260's (for light editing and browsing), and the IBM 2250 Model IV (for major editing and restructuring), as well as a storage tube display console. Optimal use is made of the characteristics of the various terminals. Many new capabilities have been added, in particular the hierarchical (tree) manipulations of Engelbart's AHI system. However, the free form "string" approach of HES has been retained, distinguishing FRESS from all the other approaches described in Part I.

7. BIBLIOGRAPHY, PART II

1. Carmody, Steven, Walter Gross, Theodor H. Nelson, David Rice, Andries van Dam: A Hypertext Editing System for the /360, Proceedings of the 2nd Annual Conference on Computer Graphics, University of Illinois, March, 1969.

2. Engelbart, Douglas C. and William K. English: A Research Center for Augmenting Human Intellect, Proceedings of the 1968 Fall Joint Computer Conference (Volume 33, Part 1), p. 395-410.

3. Nelson, Theodor H.: Getting It Out of Our System, in George Schecter (Ed.), Information Retrieval: A Critical View, Washington, D.C., Thompson Books, 1967.

4. Rice, David E.: "A Manual for the Format Phase of the Hypertext Editing System for the IBM System/360 Model 50," Center for Computer and Information Sciences, Brown University, Providence, Rhode Island, File No. HPS360-1, 1969.

5. van Dam, Andries: "A Manual for the Edit Phase of the Hypertext Editing System," Center for Computer & Information Sciences, Brown University, Providence, File No. HES360-1, May, 1969.

DISTRIBUTION LIST

David Berry
RCA Graphic Systems Division

Dr. Robert Brown
Arcata National

Steven Carmody
Brown University

Dr. Douglas Englebart
Stanford Research Institute

Prof. Walter Freiberger
Brown University

Terry Gross
San Francisco

Dr. Fred Haney
Scientific Data Systems

Wilfred Hansen
Argonne National Laboratory

Sheila Howard
IBM New York Scientific Center

Craig Jchnson
IBM Cambridge Scientific Center

Prof. Joseph LaSalle
Brown University

Prof. J.C. Licklider
Massachusetts Institute of Technology

Robert Loomis
IBM New York Scientific Center

Dr. V.A. Lvov
Institute for Mathematics, USSR Academy of Sciences

Dr. Paul Maeder
Brown University

Martin Michel
University of Illinois

Prof. William Miller
Stanford University

Theodor Nelson
New York

Don Rully
IBM New York Scientific Center

John Stallard
National Military System Support Center

Prof. Andries van Dam
Brown University

COMPLIMENTS OF THE

HYPERTEXT EDITING SYSTEM

CENTER FOR
COMPUTER & INFORMATION SCIENCES
BROWN UNIVERSITY
PROVIDENCE, RHODE ISLAND

13 OCTOBER, 1969