

Practical Experiences
with
Hypertext for Learning

John J. Leggett
John L. Schnase
Charles J. Kacmar

Hypermedia Research Lab

SEPTEMBER 1989

TAMU-HRL 89-016

Table of Contents

Abstract	1
Introduction	1
Case Studies	2
Hypertext Model and Taxonomy of Hypertext Systems	18
Discussion	22
Conclusions and Recommendations	23
Future Plans	24
Acknowledgements	25
References	25

Practical Experiences with Hypertext for Learning

John J. Leggett
John L. Schnase
Charles J. Kacmar

Abstract

This paper presents the results of three case studies in the use of hypertext for learning at the university level. An experiential evaluation of student learning from these hypertexts is given based upon responses from student questionnaires. A model and taxonomy of hypertext systems is proposed which helps to identify those characteristics necessary for particular styles of hypertext presentation. The main thesis of the paper is: the particular hypertext system, and more importantly the hypertext model, defines the bounds of what can be learned through the medium. The thesis is supported through practical examples from existing hypertext systems.

Introduction

Bush [1945], Engelbart [1962], and Nelson [1965] were among the first to propose systems that augment the human intellect by managing information stored in what we now call hypertext. These early proponents had a strong sense that the connections in hypertext were linked ideas, that associations could be easily and arbitrarily forged, and that the information could be personalized, freely annotated, freely viewed, and readily accessed. They proposed systems that offer a direct manipulation approach to information management and rely heavily on the user's increased use of visual cues, spatial reasoning, and associative thought. The revolutionary content of their ideas was,

and continues to be, the extent to which these systems engage the user as an active participant in interactions with information.

Hypertext systems for the educational community have recently become technologically feasible. The advent of workstations and local area networking is making hypertext use at the university level possible. Early hypertext systems built on these platforms are struggling to provide the vision of hypertext layed down by the pioneers. Existing systems provide minimal support for personalization, free annotation, and free viewing which are essential to providing a hypertext learning environment.

Research efforts on hypertext in education are now underway at several Universities [Crane 87; Jonassen 86; Yankelovich, Landow and Heywood 87]. The broader educational community has also embraced hypertext and hypertext systems for the delivery of educational media [Beck and Spicer 88; Por 87]. Although there is much activity, the literature is lacking in theoretical findings, empirical results, and practical experiences in using hypertext for learning.

This paper presents the results of three case studies in the use of hypertext for learning at the university level. An experiential evaluation of student learning from these hypertexts is given based upon responses from student questionnaires. A simple model and taxonomy of hypertext systems is proposed which helps to identify those characteristics necessary for particular styles of hypertext presentation. The main thesis of the paper is: the particular hypertext system, and more importantly the hypertext model, defines the bounds of what can be learned through the medium. The thesis is supported through practical examples from existing hypertext systems. Conclusions are drawn based on the thesis of the paper, and plans for future research on hypertext and learning are given.

Case Studies

Introduction

Three levels of students interacted with different hypertexts in order to study information presented at various levels of complexity. Group one consisted of undergraduate Sophomores learning a very

low-level, detailed lesson on constructing loops in an assembly language. The concepts presented in the hypertext were the same as those presented in the regular textbook for the course. The instructor did not lecture on the concepts. Group two consisted of undergraduate Seniors learning the programming language Modula-2. The concepts presented in the hypertext were not accompanied by a corresponding textbook. The instructor lectured only on advanced topics in Modula-2. Group three consisted of master and doctoral level graduate students learning about hypertext and the use of hypertext systems for design and collaboration. In this case, the medium itself was the object of learning. The concepts being learned experientially were also the subject of the readings and lecture of the class. Each case study is presented in four parts: introduction, design of the hypertext, the learning environment for using the hypertext, and the results.

Case Study One: Looping Constructs in an Assembly Language

Introduction

In this case study, a hypertext tutorial on the construction of looping mechanisms in assembly language is presented in the HyperTIES hypertext system [Shneiderman 86]. The tutorial is used in conjunction with a sophomore-level undergraduate class on assembly language programming. In general, the students are unfamiliar with assembly language programming and hypertext. The concepts presented in the tutorial are the same as those presented in the regular textbook for the course. The instructor does not lecture on these concepts. The concepts are necessary for student programming laboratory assignments which account for approximately ten percent of the course grade. The hypertext tutorial reduces the amount of class time necessary in developing loop construct skills and serves as a self-paced learning project for the class. The hypertext tutorial also allows the students to explore an alternative presentation of the material. The tutorial has been used for three semesters.

HyperTIES was chosen as the hypertext system for the following reasons:

- 1) it runs on easily accessible machines for this group of students
- 2) the browser is conceptually simple and easily learned in a short amount of time

Hypertext was chosen as the delivery vehicle for the following reasons:

- 1) to study the effectiveness of hypertext as an alternative medium for the presentation of textbook material
- 2) to study the effectiveness of hypertext in presenting very low-level, detailed lessons.

Design of the Hypertext

The material for the tutorial was derived from the regular textbook for the class; the author of the hypertext is the same as the author of the textbook [Kacmar 88]. The major sections of the hypertext match the major sections of the textbook, but the examples and supplementary definitions are different. The hypertext tutorial was constructed using the HyperTIES authoring tool in approximately 7 hours. The hypertext encyclopedia consists of 20 articles. The articles vary in size from 200 to 3400 characters with an average article size of approximately 1500 characters. The hypertext is strongly hierarchical with the first few levels as shown in Figure 1. Cross linking occasionally occurs between the various levels.

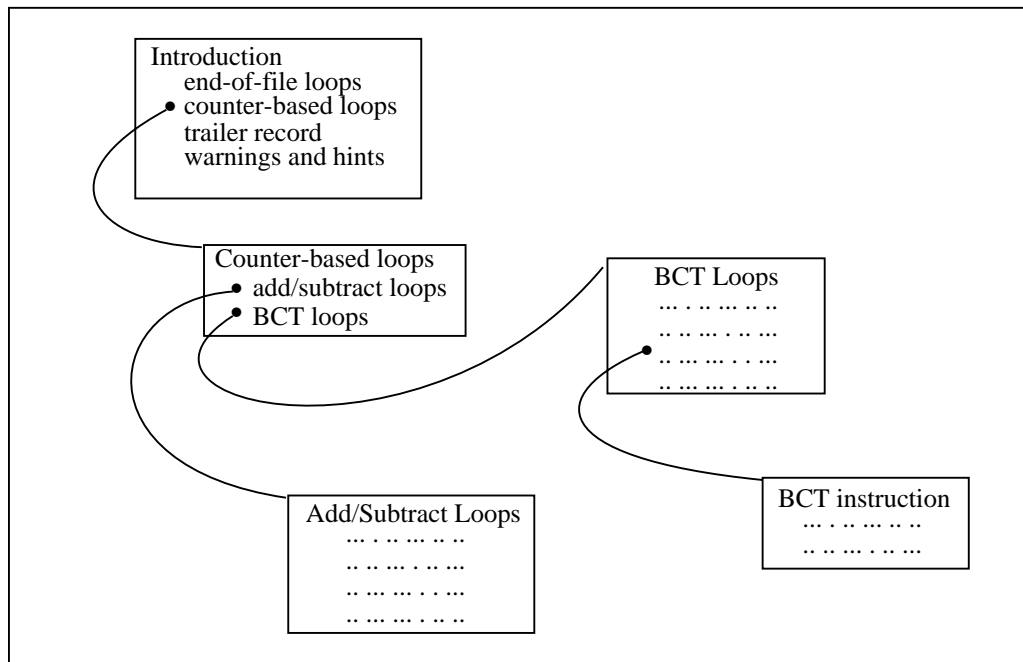


Figure 1. Looping constructs hypertext.

The Learning Environment

Access to the hypertext was provided in an IBM/PC departmental computer lab. The students could also check out a floppy disk version containing the HyperTIES browser and the hypertext encyclopedia to run on any accessible machine. The students were allowed to browse the tutorial at their convenience throughout the semester. After completing the lesson, students were required to complete a questionnaire.

Results

The questionnaire is shown in Figure 2. The results have been consistent across all three semesters.

In general:

- 1) the students found books easier to read and more accessible than the hypertext tutorial
- 2) they were divided on which method, books or hypertext, was more enjoyable
- 3) disorientation was a minor problem
- 4) accessing information by following links was well-liked
- 5) they felt that they understood the material presented in the hypertext tutorial
(The instructor feels that the students understood the simpler concepts well but not the more complex looping constructs. The instructor also feels that more personal instructor/student interaction is necessary for learning the more complex material.)

The top 5 comments have been:

- 1) "Want bookmark facility"
- 2) "Want to annotate"
- 3) "Want an overview diagram"
- 4) "Want trails through the information"
- 5) "Want an active system"
(so that examples could be modified and questions could be answered on-line)

Most comments related dissatisfaction with the hypertext system rather than the hypertext. Overall, the students enjoyed the tutorial and are quite positive about hypertext.

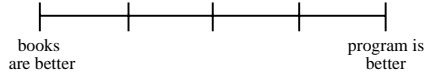
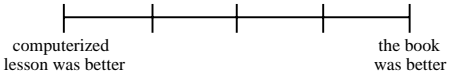
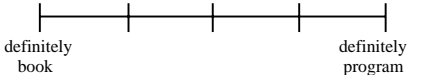
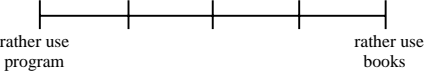
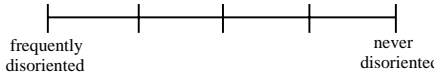
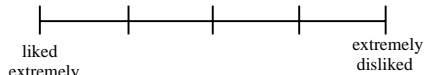
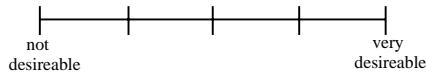
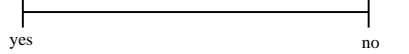
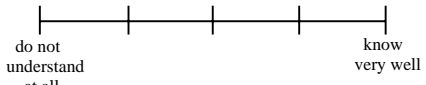
Compared to books, how did you like using the hypertext system for reading?	
Did you find the material in the hypertext tutorial as informative as material in texts?	
Which learning method (book or program) do you enjoy more?	
Suppose that books were replaced with this tool. How would you like reading if all material had to be learned this way?	
When using the hypertext tutorial, did you find yourself getting lost or disoriented, unable to remember what you had just read?	
How did you like the ability to jump from topic to topic by selecting the "anchor" words?	
The system did not allow you to create your own connections to material, and it did not allow you to add or change any information. How desirable would these capabilities have been?	
Did you read any outside sources prior to using the computerized hypertext tutorial?	
Estimate the depth of your understanding of the material that was covered in the computerized lesson.	

Figure 2. Questionnaire for case studies one and two.

Case Study Two: Modula-2

Introduction

In this case study, a hypertext tutorial on the programming language Modula-2 is presented in the HyperTIES hypertext system. The tutorial is used in conjunction with a senior-level undergraduate class on operating systems. Most of the students are familiar with the programming language Pascal

but are not familiar with Modula-2 or hypertext. The concepts presented in the tutorial are not covered in a corresponding textbook for the class, and the instructor lectures only on advanced topics in Modula-2. Twenty-five percent of the student's grade depends on Modula-2 programming laboratory assignments. The hypertext tutorial allows the instructor to focus on advanced concepts during class lecture periods and reduces the amount of class time required to develop Modula-2 skills. The tutorial also provides the students with low-cost access to high-quality Modula-2 information in their programming environment. The tutorial has been used for four semesters.

HyperTIES was chosen as the hypertext system for the following reasons:

- 1) it runs on the same machine as the student's Modula-2 programming environment (IBM/PC)
- 2) the browser may be distributed with the hypertext encyclopedia

Hypertext was chosen as the delivery vehicle for the following reasons:

- 1) to study the effectiveness of a hypertext in accessing this type of reference material in a programming environment
- 2) to study the effectiveness of a hypertext in quickly teaching advanced undergraduates a programming language

Design of the Hypertext

The material for the tutorial is composed of: the Report on the Programming Language Modula-2 [Wirth 1985] and the Definition Modules from the Logitech Modula-2 libraries [Logitech 1986]. The report on Modula-2 (26 pages) was scanned by a Kurzweil scanner to produce an ASCII text file. The Definitions Modules are supplied in ASCII format from Logitech. Approximately 50 hours was spent in constructing the hypertext tutorial by using the HyperTIES authoring tool. The hypertext encyclopedia consists of 102 articles. The articles vary in size from 100 to 5000 characters with an average article size of approximately 2000 characters. The hypertext is strongly hierarchical with the first few levels as shown in Figure 3. Cross linking does not occur between Logitech Definition Modules and the Report on Modula-2.

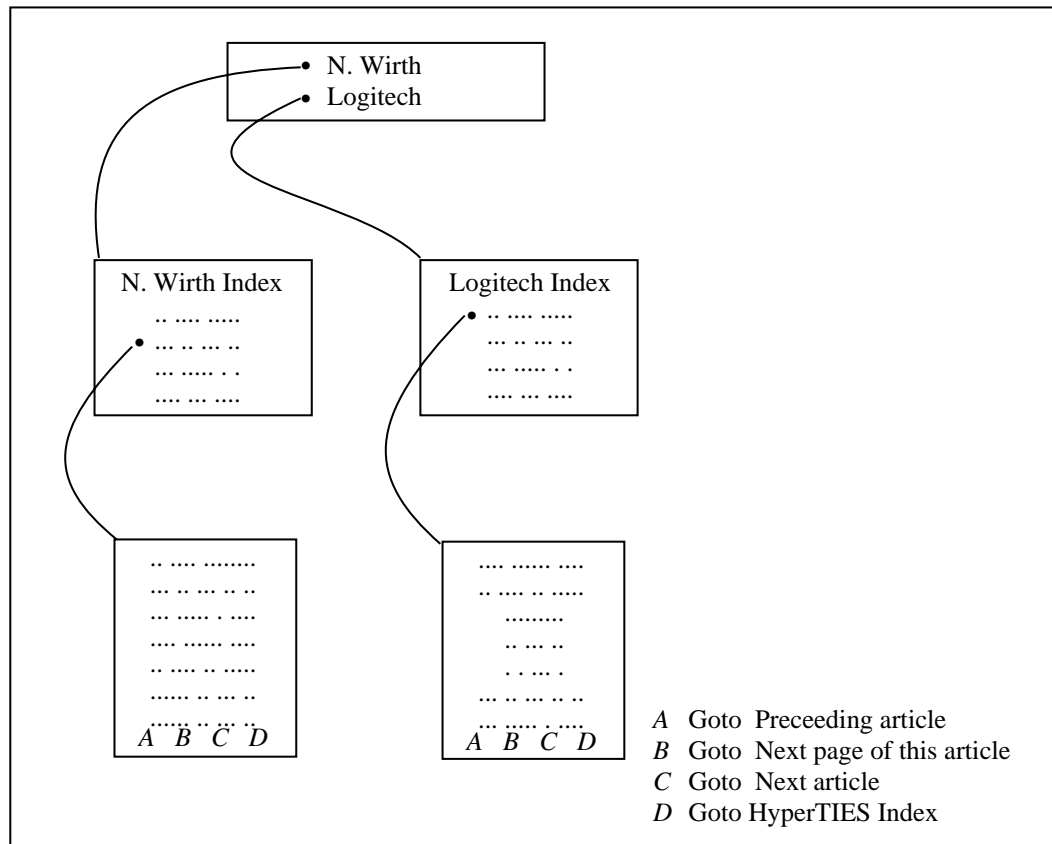


Figure 3. Modula-2 hypertext.

The Learning Environment

The students are given a floppy disk containing the HyperTIES browser, the Modula-2 hypertext encyclopedia and an example Modula-2 program to be used in a programming laboratory assignment. The students may then browse the Modula-2 tutorial at their convenience. They may also refer to the tutorial when programming their laboratory assignments. The students are free to use the tutorial on any accessible machine. The students are required to return the diskette and a completed questionnaire at the end of the semester.

Results

The questionnaire is shown in Figure 2. The results have been consistent across all four semesters.

In general:

- 1) the students found books easier to read and more accessible than the hypertext tutorial
- 2) they were divided on which method, books or hypertext, was more enjoyable
- 3) disorientation is not a problem
- 4) accessing information by following links is well-liked
- 5) they felt that they understood the material presented in the hypertext tutorial
(this is substantiated by their performance on programming laboratory assignments)

The top 4 comments have been:

- 1) "Want to annotate"
- 2) "Want mouse selection"
(HyperTIES supports mouse selection, but keyboard selection was used to ensure the tutorial could be used on all IBM PC's)
- 3) "Want bookmark facility"
- 4) "Want the tutorial integrated directly into the programming environment"
(so programming examples in the tutorial could be executed)

Most of the student complaints relate to the hypertext system's shortcomings and not to the hypertext. The hypertext concept itself is very appealing to the students.

Case Study Three: Design and Collaboration

Introduction

This case study reports on experiences in teaching hypertext at the graduate level in computer science and, as such, is quite different from the previous two case studies. The students consist of master and doctoral level graduate students learning about hypertext and the use of hypertext systems for design and collaboration [Foster and Stefik 86; Kraemer and King 88; Trigg, Suchman and Halasz 86]. The medium itself is the object of learning and the subject of readings and lectures in the class. In general, the students are unfamiliar with hypertext and have never used any collaborative computer system. Twenty-five percent of the student's grade depends on participating effectively on design projects in a collaborative hypertext environment. The class consists of reading the literature (about 1000 pages), surveying existing hypertext systems, authoring and

browsing hypertexts in at least five different hypertext systems, and design and collaboration projects in the KMS hypertext system [Akscyn, McCracken and Yoder 88; Yoder, Akscyn and McCracken 89]. The class has been taught twice.

KMS was chosen as the hypertext system for the following reasons:

- 1) it runs in the distributed local area network environment available to the students (Suns)
- 2) it provides support for annotation (which is necessary for critical review)
- 3) it provides at least minimal support for collaboration

Design of the Hypertext

The students are asked to form 2-3 person design teams. Each team is given the same hypertext design project and required to work entirely in the hypertext system. Teams have access to each other's designs, collaborations, and deliverables.

For each design project, the teams must deliver:

- 1) a statement of the issues in the design (with all deliberations)
- 2) a statement of the design decisions that were made (including argumentation and rationale)
- 3) a functional implementation of their design

The projects are browsed at irregular intervals throughout the duration of the project by the instructor. The instructor comments, annotates, links, and otherwise directs the student projects through the hypertext medium. All grading is done *in situ* in the hypertext.

Figures 4-9 show some of the resulting hypertext from a calendar design project. The hollow bullets indicate links. The solid bullets indicate action which may include linking. Instructor comments are enclosed in double boxes. Figures 4-7 show two student's collaboration on issues in the design of the calendar. Most students were not as formal in their collaboration. Figure 8 shows a more typical dialogue with links and actions becoming part of the natural communication. Figure 9 shows student awareness of other team designs.

Final Version of the Issues

1. Space.

This is concerned with how much room needs to be available for writing and annotation. Should this space be fixed or user-expandable?

2. Abstractions to help you find information quickly.

For a calendar to be effective there needs to be some way of making events stand out and make it easy for you to find information. On a conventional desk calendar you can use colored pens and draw arrows, for example. This is especially important for the electronic analog where you may have a much more limited use of "space".

3. Implementation of daemons or reminders for important events.

If we want our calendar to be better than the conventional kind it should provide some support for some kind of daemons or reminders to be used to flag upcoming and important events. For example, you should be able to set a daemon to remind you of some upcoming important event some user-specified number of days before it is to occur.

4. Navigational model.

How to incorporate features of a conventional calendar model to aid in orienting the user while still taking advantage of the features of KMS to improve over the paper calendar.

5. Design model.

What type of conventional calendar model should be used (if any)? There are several different types of conventional calendar models that we could choose from: desk calendar, pocket calendar, event calendar, etc. This will affect the overall view, that is, is it a daily, weekly or monthly calendar?

6. Inclusion of extra features.

What and how many additional features should be added. For example, address books, maps, to-do lists, etc.

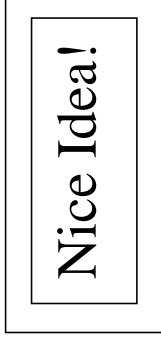
7. The user.

What audience is the calendar being designed for? Is this a personal business calendar, or what?

The Scratchpad

John's Ideas

Mary Ann's Ideas



About this frame:

The right side of the frame contains links to issues frames that we composed individually. After annotating and discussing each others thoughts on the issues, we composed a final list of primary issues which appears on the left side of the frame.

Figure 4. Issues and scratchpad.

John's Ideas

Issues

1. Space for writing information.

This is concerned with how much room needs to be available for writing and annotation. For example, how much space needs to be set aside to list the events of a single day? Should this space be fixed or user-expandable?

2. Abstractions to help you navigate easily and find information quickly.

For a calendar to be effective there needs to be some way of making events stand out and make it easy for you to find information. On a conventional desk calendar you can use colored pens and draw arrows, for example. This is especially important for the electronic analog where you may have a much more limited use of "space".

3. Implementation of daemons or reminders for important events.

If we want our calendar to be better than the conventional kind it should provide some support for some kind of daemons or reminders to be used to flag upcoming and important events. For example, you should be able to set a daemon to remind you of some upcoming important event some user-specified number of days before it is to occur. You may want a reminder every day for a week before the event is to take place (especially if you are the procrastinating type).

4. Use of innovative features of KMS versus conventional calendar models.

In reference to the item above and other innovations that are made possible by the computer, we need to decide how much the KMS calendar should differ from the conventional model. If it works no better than a conventional calendar, then we really have not accomplished much. On the other hand, if there is no intuitive model then the calendar will probably be hard to learn to use.

5. Schemes for abstracting elements of conventional calendars.

This is related to items 2 and 4 above. The real issue is to decide how to design abstractions for conventional calendar elements (days, weeks, months, etc.) in a way that makes sense and effectively uses the features of KMS.

6. Type of conventional calendar model to use (if any).

Just as most hypertext systems have an underlying model, we would most likely benefit from use some type of conventional calendar as a foundation. There are several different types of conventional calendar models that we could choose from: desk calendar, pocket calendar, event calendar, etc. We need to look at the calendar models and what features our calendar will have to see if any of the models would be appropriate.

Mary Ann's Comments on John's Ideas

1. Space

User-expandable would be nice but difficult. I think if we have a hierarchy of month to week to day, that this addresses the problem to a degree. Space is sort of expandable, to a degree. Unlimited space may cause lost in space problems. But I agree this is an issue.

2. YES!!!! Something I had not considered whatsoever.

← I think this should be integrated with my issue of priority lists.

The hierarchy idea seems to bring together types of calendars that in the paper model cannot be done very well. But the daemon idea would do this, as well.

← Pictorially? That is, make the parts look like a conventional calendar.

Figure 5. John's ideas and Mary Ann's comments.

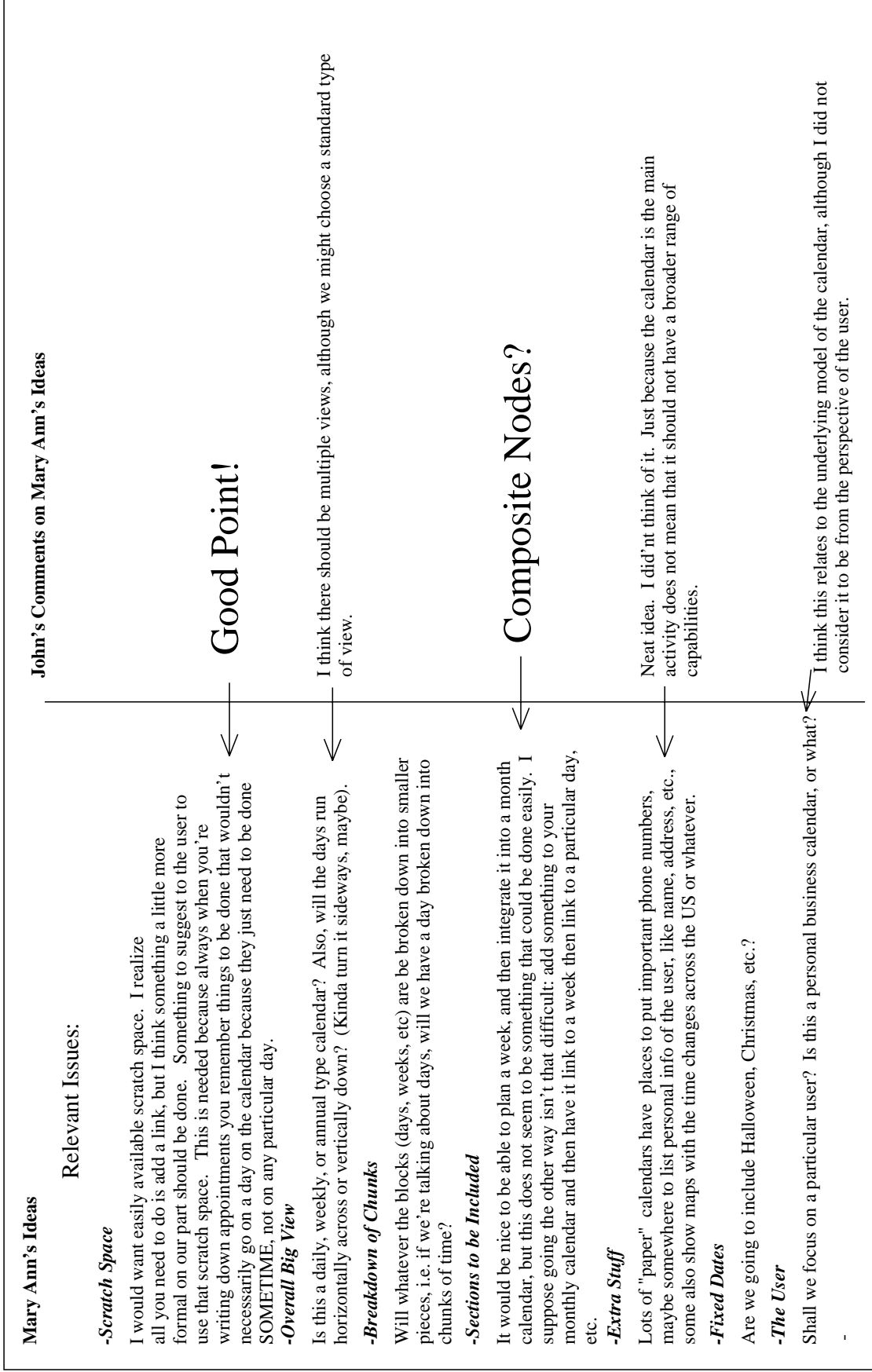


Figure 6. Mary Ann's ideas and John's comments.

Description of Final Version of Project

The project as it stands is only a prototype. There were some features that we tried to show in our example calendar that are not actually implemented. For example, we wanted the user to be able to type in his schedule for a day only on a "day" frame. This information would be placed automatically in the "week" frame and the "month" frame (at least as much as would fit). Then the day would become boldface on the miniature calendars around the month calendar, as well as on the month calendar, to indicate that information for that day had been entered by the user.

Example (see February 18)

Annotations were added on some frames to highlight links.

Good idea. This is one of those painful choices...
FORCE the user to fill in the day frame and do something neat with the info **OR** let the user have complete freedom and not help them out as much...

Figure 7. Description of final version of the project.

Fenton-Hicks

temporary address book

The temporary address book shows only one column of the address book. My goal is to have each column of the address book have a prompt "New name:" beside which you can enter the name to be added to the address book. Then you click on the action and have the new card brought up for your entry. The action frame will transfer the new name to the card and place it in the column of the address book. Then it will create a link from the entry in the column to the card and move the prompt "New name:" down the column. I have witnessed that it works intermittently??? KHF 11-29-88
Killough found the fix. Thanks.
KHF 11-30-88

Kay,

I'm going to do some strange things that will hopefully be useful for the term project.

First, I hope to try some animation. Before you invoke this action make sure that this text is in the left window.

- look at blinker

Here's the code  blinky

Next, I hope to be able to capture some input off the keyboard and use the input for some action.

- pick a figure

Here's the code  show me

I got caught in a wicked loop by trying to create an interactive program which would terminate when the viewer placed a special character in a certain frame. KMS' action language is an "all or nothing" environment to try to do something in. It took me about an hour to finally break the infinite loop, and I was only able to do it by remote logging in from another sun an killing the main KMS process on the original sun. What a drag!

yea, thanks Ronny!! JL

a prompt in the message window would have been nice...

Figure 8. Hypertext dialogue.

Group Members: Kerm & Evan

Issues faced: The screen is too small for us to effectively present a calendar which follows the traditional paper model. Thus, we were forced to make a trade off between the amount displayed at any given time and the level of detail offered.

Design Decisions: Our first design decision involved how to present our information. We have decided to offer the user a hierarchy of detail, ranging from the full year down to hourly information. Thus, our calendar system is presented as follows: The user is first shown a screen which contains the entire year at a glance. On this, the user is able to "scribble" short reminder memos to himself for each month. From the yearly calendar, the user can link to an exploded view of a selected month, where once again they are able to scribble important events on a daily basis. From this monthly view, the user can receive a daily break down of information given on an hourly basis. Finally, the user has the option to link to an hourly frame in which they can display important and detailed information (e.g., the agenda for a meeting). One other important feature of our calendar is that the user is always able to go directly back to the yearly view by merely clicking on the "Goto Year" link always located in the upper left hand of a frame. Below is a graphical representation of our calendar system:

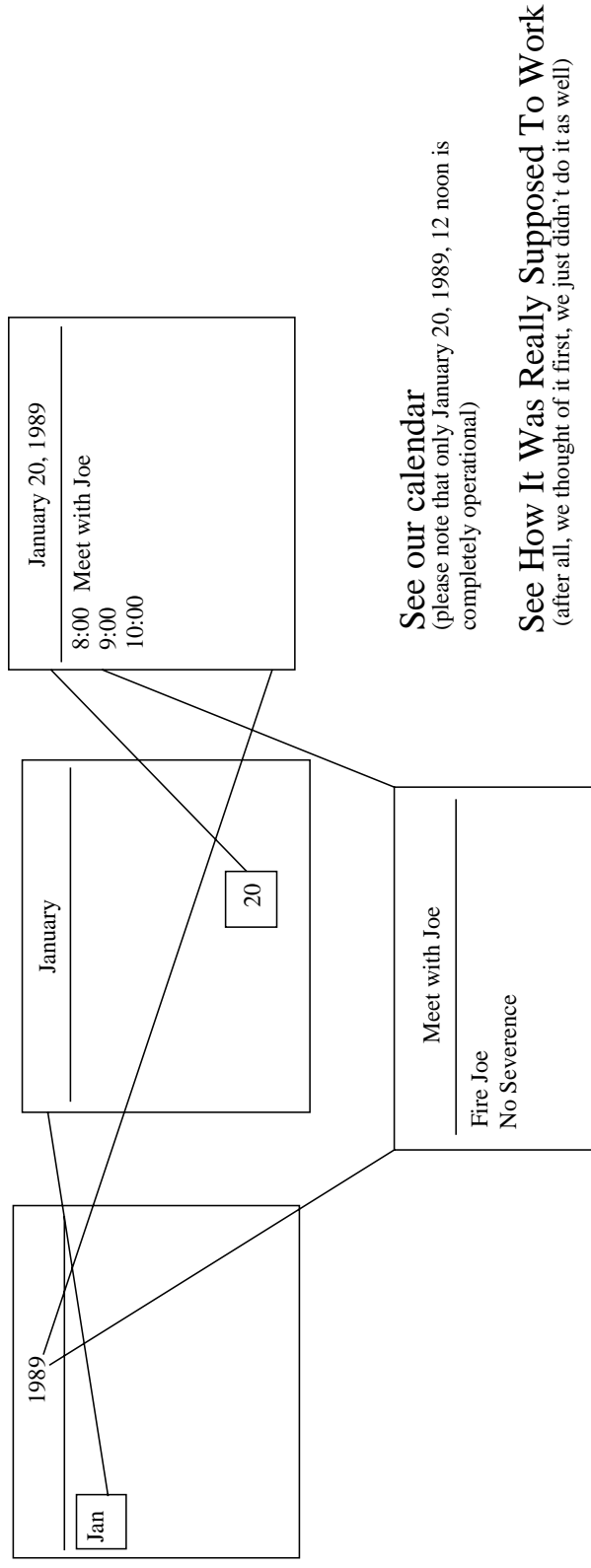


Figure 9. Reference to another team's work.

The Learning Environment

The students had access to four Sun workstations running KMS in a local area network of approximately 30 workstations; the instructor had KMS access from his office. The hypertext was distributed across the network. This allowed the team members to work in various parts of the hypertext at the same time and to work asynchronously in time. Since all parts of the hypertext were shared, the students could browse each other's work and make links from their work (arguments, collaborations, etc.) to the work of other students across team boundaries. The students could also communicate with the instructor through the hypertext medium. This communication usually consisted of a link being anchored at a particularly conspicuous place in the instructor's workspace. When followed, this link would take the instructor into the student's workspace to a particular place of concern to the student.

Results

In this case study a questionnaire was not returned by the students. The results are a compendium of comments and experiences shared by the students throughout the course of the semester.

In general,

- 1) the students like being able to annotate each other's work at precisely the place of their choice
- 2) they liked the capability of collateral display
- 3) they liked having "space" to jot things down while browsing
- 4) disorientation was not a problem
- 5) navigational access to information was well-liked
- 6) they felt that they understood the issues of computer-supported collaboration

The top 5 comments have been:

- 1) "Want better collaboration support when working in the same area of the hypertext"
(for example, locks on particular frames)
- 2) "Want a communication facility within KMS that is easy to use when working at distributed sites"
- 3) "Want to be able to link inside a frame"

- 4) "Want to personalize KMS functionality"
- 5) "Want an easy bookmarking facility"

The students were very positive about the future of hypertext and computer-supported cooperative work. Their criticisms focused mainly on the hypertext system.

Hypertext Model and Taxonomy of Hypertext Systems

Conceptual Model and Hypertext System Definitions

In this section we present a conceptual model (Figure 10) and set of definitions for hypertext. The set of definitions provide a framework for conceptualizing hypertext architectures.

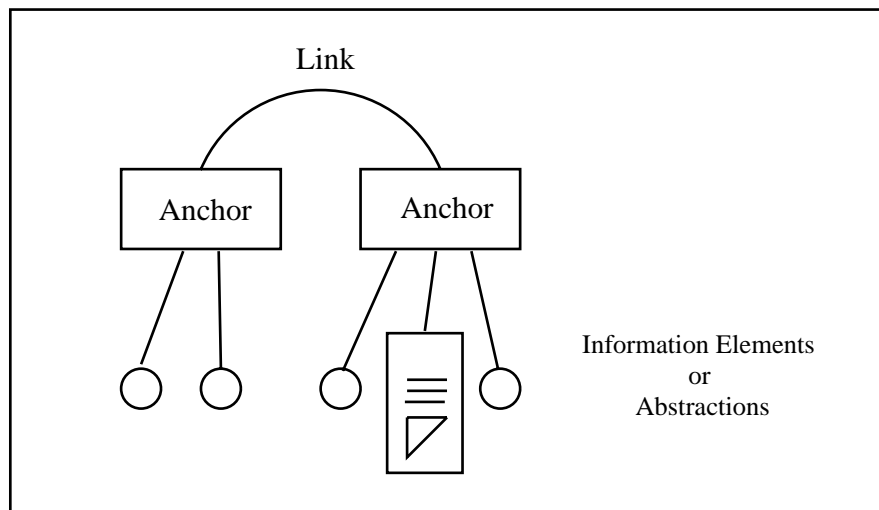


Figure 10. A simple conceptual model of hypertext.

Information element. Information elements are the objects stored and manipulated by the user. These include thoughts, diagrams, ideas, discussion, lesson plans, and arguments. Their physical manifestations are such things as text, bit-mapped images, graphics, sound clips, and animations. Information elements may be typed and named.

Abstraction. An abstraction is an object into which other objects (including abstractions) may be placed, structured, or grouped. Abstractions may or may not be user-manipulable objects. These include such things as containers, sets, and composites. Some example abstractions from existing hypertext systems include: cards and fileboxes (NoteCards); frames and framesets (KMS); articles and encyclopedias (HyperTIES); and, documents and journals (NLS/Augment). Abstractions may be typed and named.

Anchor. An anchor designates information elements or abstractions that may be the source or destination of a link. Anchors may be typed and named.

Link. A link is a connector among anchors. Links may be typed and named.

Hypertext model. A hypertext model is a set of abstractions which provide a conceptual framework for creating, storing, and retrieving information in a hypertext.

Hypertext system. A hypertext system is a functionally related set of computer hardware and software components that implement a hypertext model.

Node. Node is a generic term for an abstraction in a hypertext model. Some example nodes from existing hypertext systems include: cards (NoteCards); frames (KMS); articles (HyperTIES); and documents (NLS/Augment).

In the simplest case, links may connect anchors that are attached to a single information element or abstraction. Four basic associations are thus possible as shown in Figure 11.

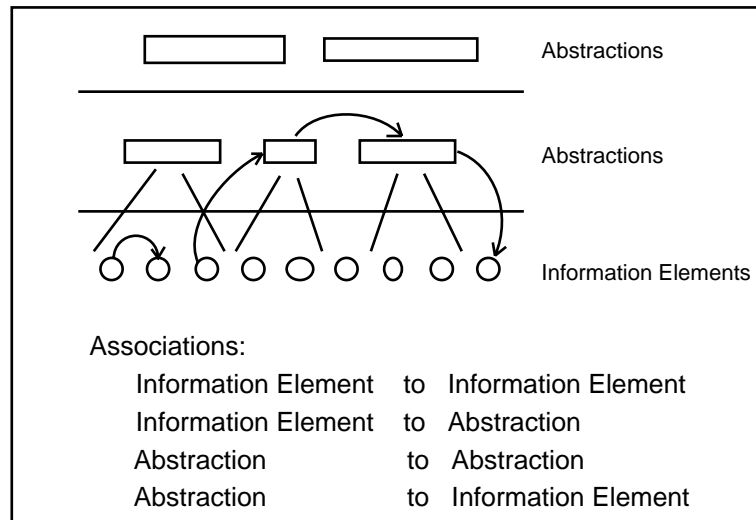


Figure11. Basic associations in hypertext.

An example of an existing hypertext system in this model is given in Figure 12 for KMS.

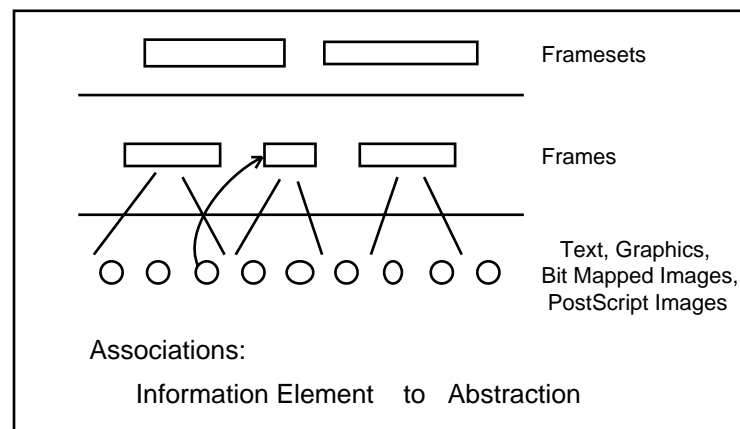


Figure 12. The KMS model.

Taxonomy of Hypertext Systems

An application-oriented taxonomy of hypertext systems is provided to characterize existing and future systems. Any taxonomy of hypertext systems is somewhat arbitrary since a particular system may be used (or misused) for many purposes. The taxonomy presented here consists of the following five classes of hypertext systems: Literary, Structural, Presentational, Collaborative, and Explorative. Each class is described by giving its main characteristics and listing example

applications and systems. The Collaborative and Explorative classes are not represented well by any particular system, and in these cases, hypertext systems with a significant (yet limited) amount of functionality have been listed.

Literary Hypertext Systems

In the Literary class, the links are relatively more important than the nodes. That is, the associations among the information elements are very important while the structuring of the information elements is not as important. The links have a smaller granularity of referent, and these systems have information element to information element associations. Literary hypertext systems allow free annotation.

Example applications: Reading, writing, publishing, critiquing, education, scholarship

Example systems: MEMEX, NLS/Augment, Xanadu, Textnet, Intermedia, IGD, Guide

Structural Hypertext Systems

In the Structural class, the nodes are relatively more important than the links. That is, the associations in the hypertext are not as important as the structuring of the information elements. The links have a larger granularity of referent, and these systems do not have information element to information element associations. Structural hypertext systems have limited or no annotation capabilities.

Example applications: Information management, argumentation, idea processing

Example systems: KMS, gIBIS, NoteCards, HyperCard

Presentational Hypertext Systems

The Presentational class has the same characteristics as the structural class with the additional restriction of a separate author and browser.

Example applications: Information kiosks, reference material, documentation

Example systems: HyperTIES, Concordia/Document Examiner

Collaborative Hypertext Systems

In the Collaborative class, links and nodes are equally important. That is, both the associations among information elements and the structuring of information elements is important. The links have a small granularity of referent, and these systems have information element to information element associations. Collaborative hypertext systems allow free annotation, are distributed, and provide security.

Example applications: Software engineering environments, organizational information management

Example systems (Limited): Neptune/HAM, NLS/Augment, Intermedia, KMS, NoteCards

Explorative Hypertext Systems

In the Explorative class, links and nodes are equally important. That is, both the associations among information elements and the structuring of information elements is important. The links have a small granularity of referent, and these systems have information element to information element associations. Explorative hypertext systems allow free annotation and have a spatial metaphor in their front-end. The concept of "space" allows the display screen to be used like a painter's canvas for recording and manipulating information elements and abstractions.

Example applications: Early activities of writing and thinking, problem formulation and exploration

Example systems (Limited): KMS, NoteCards

Discussion

The same Presentational class hypertext system was used in case studies one and two. In the first case, it was used to teach a small, detailed lesson to less mature students. In the second, a larger, more general lesson was taught to more mature students. The results were as expected by the instructors. Both groups found books easier to read and more accessible than a computer screen,

enjoyed hypertext and navigational access to information, and felt that they understood the material presented in the hypertext. The one difference was that the less mature group became disoriented as they browsed the tutorial, while the more mature group did not.

More importantly, the comments from both groups are similar. Both groups would like the capability to personalize the material in the tutorial through annotation, a bookmarking facility to allow quick review of disparate parts of the information space, and the hypertext system to be integrated into their normal computing environment. Additionally, the less mature group wanted an overview facility and trails (paths) to help with their disorientation. The comments of both groups reflect a frustration with the capabilities of the hypertext system, not with the hypertext.

The KMS hypertext system was used in case study three. KMS is a Structural class hypertext system with features of the Collaborative and Explorative classes. KMS is distributed, has a spatial metaphor in the user-interface which allows annotation, provides collateral display of two disparate parts of the information space, and has one kind of link - information element to abstraction.

The students in case study three (like those in case studies one and two) enjoyed hypertext and navigational access to information and felt they understood the lesson to be learned from the hypertext. These students did not become disoriented. They particularly liked the spatial metaphor of the front-end.

The students in case study three (like those in case studies one and two) wanted a bookmarking facility [Bernstein 88]. Additional comments concerned the request for links of finer granularity, better support for distributed collaboration and communication, and a desire to personalize the hypertext system as well as the hypertext. The comments focus mainly on the hypertext system, not the hypertext.

Conclusions and Recommendations

Obviously, this is not a controlled study, and we all know that thoughtful students will complain no matter what characteristics your hypertext system has or does not have. On the other hand, practical

experience is valuable, and several common threads are apparent in the remarks of students in these case studies.

First, the capability to personalize information through annotation is a paramount requirement for use of hypertext systems in learning at this level. Second, a bookmarking facility is very desirable even when the hypertext system allows collateral display. Third, the hypertext system should be integrated into the typical working environment of the user. Fourth, for collaboration, information element to information element links are highly desirable along with supporting mechanisms for communication and security. Finally, advanced users will demand that the hypertext system be tailorable and extensible [Halasz 88].

The implication for the use of hypertext in learning environments is that we should at least use a Literary class hypertext system for the delivery of educational material. Also, since scholarship (reading, writing, critiquing, learning) is both collaborative and explorative, we should look toward Collaborative and Explorative class hypertext systems for future learning environments.

Using a Structural or Presentational class hypertext system for learning is like using a slide show. The slide show would be non-linear and under user control, but it is a slide show just the same. The students should be able to personalize the information in order to deeply integrate it with their existing knowledge. We should allow scribbling in the information space, "bending" of node corners, copying of pertinent passages, restructuring of the hypertext, and integration of existing work environments. We should not give the learner less with this medium, but much, much more.

Future Plans

We are continuing the use of hypertext in the courses described in this paper, and we are introducing other hypertext tutorials into the curriculum. One project involves a hypertext learning environment for the C programming language. We hope to transparently integrate this tutorial into the Unix and C programming environments. Undergraduate sophomores will use hypertext as the user-interface paradigm while learning to program in C.

Our other main interest in hypertext for learning concerns the use of hypertext for traditional scholarship. We are experimenting with hypertexts that are deeply integrated into our customary work environment. All reading, writing, critiquing, publishing, simulating, and so forth is done in one integrated hypertext. In one case, the hypertext contains: a linear, camera-ready journal article which is generated from the hypertext; a non-linear hypertext version of the article; the simulation program which generates several figures and tables for the article; and the experimental data used as input to the simulation. The hypertext may be mailed electronically to a peer for review. The reviewer may annotate *in situ* and return the hypertext electronically.

Acknowledgements

The authors would like to thank those students who have participated in the case studies. A special thanks to John Desoi, Mary Ann Branch, Kay Fenton, Jaye Hicks, Ronnie Killough, Kermith Harrington, and Evan McGinnis for allowing us to view their collaborative hypertext.

References

- Akscyn, R., McCracken D., and Yoder, E. 1988. KMS: A distributed hypermedia system for managing knowledge in organizations. *Commun. ACM*, 31, 7, (July), 820-835.
- Beck, J. R., and Spicer, D. Z. 1988. Hypermedia in academia. *Academic Compt.*, (February), 22.
- Bernstein, M. 1988. The bookmark and the compass: Orientation tools for hypertext users. *ACM SIGOIS Bulletin*, 9, 4, (October), 34-45.
- Bush, V. 1945. As We May Think. *Atlantic Monthly*, 176, (July), 101-108.
- Crane, G. 1987. From the old to the new: Integrating hypertext into traditional scholarship. *Proceedings of the Hypertext '87 Conference*, (Chapel Hill, NC, November), pp. 51-56.
- Engelbart, D. 1962. Augmenting human intellect: A conceptual framework. SRI Technical Report AFOSR-3223, Contract AF 49(638)-1024, (October).

- Foster, G., and Stefik, M. 1986. Cognoter, theory and practice of a collaborative tool. *Proceedings of the CSCW '86 Conference*, (Austin, TX, December), pp. 7-15.
- Halasz, F. 1988. Reflections on NoteCards: Seven issues for the next generation of hypermedia systems. *Commun. ACM*, 31, 7, (July), 836-852.
- Jonassen, D. H. 1986. Hypertext principles for text and courseware design. *Educational Psychologist*, 21, 269-292.
- Kacmar, C. 1988. IBM 370 Assembly Language with Assist - Structured Concepts and Advanced Topics. Prentice Hall, *Englewood Cliffs, New Jersey*.
- Kraemer, K., and King, J. 1988. Computer-based systems for cooperative work and group decision making. *ACM Compt. Surv.*, 20, 2, (June), 115-146.
- Logitech. 1986. System and library modules. In Modula-2/86 software development system user's manual, 3rd edition, Logitech, Inc., Redwood City, CA.
- Nelson, T. H. 1965. A file structure for the complex, the changing, and the indeterminate. *Proceedings of the 20th National ACM Conference*, pp. 84-100.
- Por, G. 1987. Hypermedia and higher education. *Computer Currents*, (August), 14-16.
- Shneiderman, B., and Morariu, J. 1986. The Interactive Encyclopedia System (TIES). University of Maryland, College Park, MD.
- Trigg, R., Suchman, L., and Halasz, F. 1986. Supporting collaboration in NoteCards. *Proceedings of the CSCW '86 Conference*, (Austin, TX, December), pp. 153-162.
- Yankelovich, N., Landow, G., and Heywood, P. 1987. Designing hypermedia "ideabases" – The Intermedia experience. Brown University, IRIS Technical Report 87-4, Providence, RI.
- Yoder, E., Akscyn, R., and McCracken, D. 1989. Collaboration in KMS, A shared hypermedia system. *Proceedings of the CHI'89 Conference*, (Austin, Tx., May), pp. 37-42.
- Wirth, N. 1985. Report on the programming language Modula-2. In Programming in Modula-2, 3rd corrected edition, Springer-Verlag, New York.