NCC 9-16

SEPEC Tutorial

hypermedia & information reconstruction

JOHNSON GRANT
IN-61-CR
332279
P.77

Structured Hypermedia

(C) OVERRIDE

J. Scott Johnson

December 3, 1990
Houston, Texas
South Shore Harbour Resort & Conference Center

Co-Sponsored by
University of Houston-Clear Lake
NASA/Johnson Space Center
Hypermedia Working Group

SEPEC
Software Engineering Professional Education Center
University of Houston-Clear Lake
2700 Bay Area Blvd., Box 258
Houston, Texas 77058
(713) 282-2223

Hypermedia '90
Structured Hypertext
Tutorial

J. Scott Johnson

NTERGAID

2490 Black Rock Turnpike

Suite 337

Fairfield, CT 06430

Table of Contents

Analytical Table of Contents

"HyperKnowledgeEngineer" Versus Chunker

IV. Analyzing Hypertext Documents for Structure

A. Organizational Tools of Existing Documents

- Tables of Contents
- Conceptual Maps
- Figure Lists
- Indices

B. Look at how users work.

C. Look at the raw information sources

D. Look at the method of organization in the existing work.

V. Designing Structured Hypertext Documents

Overview

A. Modular Document Structure

- Easier Maintainability
- Easier Document Construction
- Faster File Loads
- Freedom from Memory Constraints
- More Space for Reader Annotations

B. Design for Nonlinearity

C. Cards Versus Articles

D. Metaphors

E. Organization

F. Multiple Views

G. Brevity

H. Eliminate Preconceptions

I. Consider The Document's Audience

J. Designing User Interfaces

VI. Creating Structured Hypertext Applications

A. Guidelines for Creating Applications

No Magic Rules

Prototype early and often.

Easiest to create article based hypertext

Use quick "hacks" to test concepts

Evolved Applications

Domain Expertise Linking Versus Structural Linking

VII. Structuring Service and Repair Documents

VIII. Maintaining Structured Hypertext Documents

A. Document Design

B. Quality control

C. Maintenance Tools

D. Link Info

E. Document Security

F. Data Encryption

G. Guided Tours

H. Search and Replace

I. Detached Windows

IX. Structured Hypertext Conclusion

In reality, hypertext systems are often based on highly structured information such as:

- Technical Documentation
- User Guides
- Regulatory Information
- Service and Repair Guides

All of the above types of information tends to be rigidly, often hierarchically structured. The structure found in the above types of information does not vanish when they are converted to a hypertext format - rather, that structure can become even more explicit once this information has been converted to a hypertext format.

PRECEDING PAGE BLANK NOT FILMED

II. Hypertext Documents Have Structure

At their simplest level, hypertext documents are composed of two basic elements: nodes and links. These two elements can replicate the structure found in information before it is converted to a hypertext format.

Example 1: Structure in the FIP

Structural elements such as chapters, sections, and appendices all correspond to nodes - self contained information "units."

Connective elements such as tables of contents, figure lists and indices all correspond to links - dynamic relationships between two or more elements of information.

III. Structured Hypertext

A. Structured Hypertext Defined

Structured hypertext is the concept that any body of information has structure without content. It is almost equivalent to a book with blank pages and empty chapters.

The idea is to design a generic structure or template that can be universally applied throughout a sizable corpus of information

This template is called a "Kernel Document."

Example 1: Sample Kernel Document

Example 2: Real Kernel Document With Data

A characteristic of structured hypertext is that one or two kernel documents should be able to be applied to between 80 - 90 % of the information.

B. Where is Structured Hypertext Suited ?

Big document projects with (relatively) structured material. The most common example is that of technical documentation.

Structured Hypertext is suitable as a working philosophy/methodology for the design and creation of documents.

C. Modularity - Defined

Modularity is the view of hypertext not as huge, monolithic documents but instead as small, interlocking entities that are linked together.

Example 3: Elements of the Fip Document.

The idea of modularity is to create a "docuverse" or "meta document" from many, NOT few, files (documents).

D. Relationship between Structured Hypertext and Modularity

Structured Hypertexts are built from collections of small, modular documents.

E. How Structured Hypertext & Modularity Work

The creation of a structured hypertext document created as a series of modular elements can be broken down into three basic steps.

- Step 1: Design of the Template/Kernel Document
- Step 2: Chunking the Information Modularly with the Kernel Document.
- Step 3: Hand (Subjective) Linking (2nd Pass)

Step 1: Design of the Template/Kernel Document

The design of the kernel document is designing the basic "atomic element" of the docuverse. It is the "container" for the bulk of the information.

Design of the kernel document should be done be a knowledge engineer or domain expert who knows not only the information being converted to a hypertext but also how the information will be accessed. Knowing how the information will be accessed helps to define the user interface of the kernel document.

The kernel document design tends to be specific to the project at hand and the destination hardware.

Example 4: FIP project and Touch Screens.

Step 2: Chunking the Information Modularly with the Kernel Document.

This is the manual process of taking long, sequential text and breaking it into a node and link structure. For now, it is a manual process. However, it is open to automation.

A real benefit to structured hypertext is that because everything is very well defined and universal, the chunking need not be done by a "HyperKnowledgeEngineer" -- once rules for chunking are defined.

The key to chunking the information is to look for natural breaking points (FIP Color Coding).

Example 5: Chunking of the FIP Document

Step 3: Hand (Subjective) Linking (2nd Pass)

The final step in structured hypertext is that of a second pass through the material by a domain expert who goes through and hand links important information. This second pass is the real "value-added" aspect of the hypertext document.

F. Benefits of Structured Hypertext & Modularity

The benefits of structured hypertext include:

- Quick and Easy to Create.

Structured hypertext is very quick and easy to create - there is minimal overhead on the part of the author.

Example 6: Chunking an entire FIP procedure
25-30K ASCII and chunk it in about 15 min.
Macros, Markers were used

- Non Domain Experts.

Structured hypertext can be created by Non Domain Experts / "HyperKnowledgeEngineers." The benefit here is that using a non domain expert is significantly less expensive.

Example 7: EPA's Reg-In-A-Box Project

- Large Documents.

Structured hypertext is very applicable to large documents - as a methodology it eliminates (to some extent) the effect of multiple authors on a single project. Every document has the same structure, allowing generic macros to operate on the structure of multiple documents.

The benefits of modularity include:

- Maintainability

Modular documents are small interlocking elements that can be easily replaced. When an element in a hypertext docuverse needs to be replaced, a modular document can be easily "slotted" in without disturbing other documents. This is an powerful maintenance advantage. Maintenance is covered further below.

- Freedom from Memory Constraints

Due to using small, interlocking elements, memory overhead is kept low. Although rarely considered, this is an important aspect for moving hypertext into a real world environment.

- Ease of Navigation

Small, modular documents are less complex in structure and thus easier to navigate. Navigational tools such as graphical mapping have more relevance in a modular environment than in a non-modular environment.

- Elimination of the "Hypermess" / Tangled Web of Links

Structured hypertext eliminates the "Hypermess" by removing the complex structures seen in Figure 1.

G. Issues

Does Structured Hypertext Work With Articles or Cards?

Structured hypertext tends to use a "Pseudo Card" metaphor - scrolling text with control panels. This appears like a card metaphor document but is actually an article metaphor document.

Example 7: A "Pseudo-Card" Metaphor

Article metaphor documents eliminate the huge quantities of chunking that accompany strictly card metaphor documents.

AutoLinking ?

Structured Hypertext is well suited to autoLinking due to use of consistency and "rules" for breaking information down.

"HyperKnowledgeEngineer" Versus Chunker

Structured hypertext can be created by (almost) anyone. It does NOT require the equivalent of a knowledge engineer for the chunking process. This lessens the skilled personnel needed

IV. Analyzing Hypertext Documents for Structure

In order to apply structured hypertext to a body of information, some sort of a structure must be distilled from it. Listed here are some basic rules for distilling structure.

A. Look at the organizational tools of the existing documents. Some sample organizational tools are:

- Tables of Contents
- Conceptual Maps
- Figure Lists
- Indices

Almost any organizational tool from a paper-based document has correspondence in a hypertext document. Don't be afraid to use a paper-based tool such as a figures list in a hypertext environment.

Example 8: FIP Organizational Tools

B. Look at both how users of the paper-based document are using the existing document and how they would LIKE to use the existing document. This is similar to knowledge engineering. With this information, new organizational tools can be devised.

C. Look at the raw information that goes into the document. The "granularity" or chunking level of the files may suggest a structure. If you can work with a structure developed directly from the existing files, your work may be significantly easier.

D. Look at the method of organization in the existing work. Numbering schemas such as "1.1.2.5" are perfectly usable in a hypertext environment.

Note: Whenever you can adapt an organizational tool from an existing document, this will lessen the learning curve faced by new hypertext users. It also often lessens the fear of moving to a hypertext system because users can draw analogies between the hypertext system and the old system.

V. Designing Structured Hypertext Documents

Listed below are guidelines for good document design in a structured hypertext environment.

Overview

The basic goal for designing structured hypertext is to move towards a hierarchy (or multiple hierarchies) interlinked with nonlinear references.

A. Modular Document Structure

Hypertext documents need not be huge and monolithic in nature. Rather, well constructed hypertext documents are modular in design with each separate "unit" of the document in its own file. This design eliminates memory constraints as well as makes documents more maintainable.

An example of modular document structure is creating a book in hypertext form and organizing it with each chapter in a separate file as opposed to putting each chapter as sub nodes of a single hypertext file.

The advantages of modular document design are:

- Easier Maintainability
- Easier Document Construction
- Faster File Loads
- Freedom from Memory Constraints

- More Space for Reader Annotations

B. Design for Nonlinearity

The most important characteristic of hypertext is that it is nonlinear.

Hypertext documents are not so much meant to be read as they are to be explored with the goal that exploring a document is more efficient than standard linear document reading.

Nonlinearity does not mean that the methods of paper organization cannot be used but that they should be used in a way consistent with hypertext's nonlinearity.

C. Cards Versus Articles

Articles and cards are the two basic methods of structuring hypertext nodes. With article based documents, there is scrolling, allowing linear material to be organized in a basically linear fashion. With card based documents, there is no scrolling - forcing linear material to be "chunked" across many separate cards. This forced chunking is the biggest drawback to card based documents as it tends to increase the amount of time to build a document by as much as 40%.

The advantages of article based documents are:

- Easier Construction
- More suited to long, linear material

The advantages of card based documents are:

- Easier for novice readers (there is always a next or previous destination)
- Tend to look more attractive due to onscreen control panels

A technique that works effectively is to merge both articles and cards. Use card organization for document organizational tools such as tables of contents and figure lists and use article organization for the body of the hypertext document. This approach creates easy to use organizational tools for novices to become familiar with hypertext on and eliminates the need for chunking linear material across cards.

D. Metaphors

Hypertext documents can and should use valid paper based metaphors such as tables of contents, figure lists, tables and so on. The use of familiar metaphors lessens the learning curve for new hypertext browsers.

A new metaphor is that of the control panel. Commonly found in card based hypertext documents, control panels provide readers with constant navigational tools on every card to aid readers.

E. Organization

Hypertext documents typically have two types of nodes: Organizational nodes and Content nodes. Organizational nodes hold tools such as tables of contents and indexes, the access methods for the document. Content nodes are the bulk of the document, the material the reader is looking for. A goal for hypertext authors should be to make access to the organizational nodes extremely easy.

When constructing hypertext documents, the author should come up with some consistent method for structuring his or her documents. By applying consistent structure and linking conventions, the learning curve for readers is lessened.

F. Multiple Views

One of the abilities of hypertext is to allow multiple views or entry points into the material. This should be facilitated with multiple tables of contents and indices. Overall, multiple views make it significantly easier for the reader to find the right entrance into the material that he is searching for.

G. Brevity

Brevity refers to the idea that hypertext documents need not be long winded in nature as nothing in a hypertext need be repeated - simply cross referenced by a link. The idea of brevity ties very strongly into one of the key tenets of document maintenance: Never Repeat Anything - Cross Reference It. This eliminates the need to ever update the same thing more than once.

H. Eliminate Preconceptions

One of the more limiting aspects of paper is that paper documents can visually frighten their readers due to their apparent size and density. Picture the typical repair manual for anything more complex than a ten speed bicycle. Hypertext can eliminate those preconceptions completely - there is no way to gauge the size of a hypertext document from simply accessing parts of it. This tends to encourage free browsing of information in a way that paper documents cannot.

In a hypertext document, visual cues such as Page 1 of 30 eliminate the advantage of hypertext in terms of preconceptions, discouraging browsing.

I. Consider The Document's Audience

Too many people, when designing hypertext documents, create the document to meet their own needs - rather than the needs of the audience for whom the document is created. Hypertext can be thought of as similar to an expert system -- it is of little use to its creator, but it can be of great use to a novice.

A related point is that hypertext documents should be created consistent with the abilities of the audience for the document. For example, don't create a document that relies on boolean searching if the audience doesn't understand boolean operators.

The document's author should design to and for the audience rather than to and for himself.

J. Designing User Interfaces

An oddity of hypertext as compared to other computer applications is that with hypertext, the hypermedia application software (browsing environment) is rarely seen as the user interface. Instead, the document itself is seen as the user interface.

The caveat with this is that the document author is (to some extent) thrown into the role of interface designer. Some knowledge of interface design is often helpful for building hypertext documents.

VI. Creating Structured Hypertext Applications

A. Guidelines for Creating Applications

Listed below are several guidelines that we at NTERGAID have developed over the course of three years of structured hypertext. They aren't absolute guidelines but they are good, basic rules of thumb to go by.

- There are no magic rules for structured hypertext -- it is a methodology, not a religion -- don't be afraid to throw it out the window when it doesn't fit.

- Prototype early and often.

- It is easiest to create article based hypertext -- which are more suited to structured hypertext.

- Don't be afraid to "hack" together a hypertext quickly to prove a concept or test a possible application -- it can always be redone later and, often, that "hacked" application can be used.

Evolved Applications

One of the characteristics of a structured hypertext project is that it tends to "evolve" as much as it is constructed. Often, when building structured hypertext, the original plans for the project change radically over the course of the project. This is to be expected. Hypertext tends to be malleable and rarely does a project follow its original specification.

The reason for this is that building a hypertext document tends to give you new insights into the material at hand -- altering the original project specification. As long as the original ideas behind structured hypertext are followed, an evolving specification is not a problem.

Domain Expertise Linking Versus Structural Linking

An unfortunate reality of hypertext is that, currently, there is a lot of "grunt" work in building a hypertext document. Tasks like text formatting and building structural links are basically repetitive and do not necessarily require a domain expert.

The role of the domain expert in a hypertext project should be the basic design, structure of the document as well as the subjective links that add the real value to the hypertext.

For example, in the EPA's Reg-In-A-Box project, although a skilled knowledge engineer (Bill Foskett) designed the project and built the majority of the subjective links, he did NOT build the whole application by any means. The majority of the application was built by relatively unskilled EPA secretaries that had hypertext explained to them in terms of an editor and were directed in the basic linking to be accomplished. This resulted in a tremendous savings in skilled labor versus (at least in this context) unskilled labor.

A second example, is that at NTERGAID, we have two separate people assigned to any one hypertext project. The first does the basic design and format of the project while the second actually builds the objective links structuring the material. Then, the first comes back to the project to add the subjective linking.

VII. Structuring Service and Repair Documents

Rather than covering this in paper format, an extensive service and repair document, the AIL FIP HyperManual, will be demonstrated.

VIII. Maintaining Structured Hypertext Documents

An important topic, not commonly recognized in terms of hypertext, is that of maintaining hypertext documents. Hypertext document maintenance consists of taking existing hypertext documents and bringing them up to date. There are three basic aspects of maintaining hypertext documents: document design, quality control and tools for maintaining documents. Each of these is covered below.

NOTE: Many of the maintenance techniques covered below are dependent on navigational tools and product features offered by HyperWriter. When using another hypertext system, comparable navigational tools or features should be substituted.

A. Document Design

The most important rule to follow when designing documents for maintenance is that of modularity. Documents should not be constructed as large monolithic entities but should be constructed as small, interlocking entities.

If you are creating documents that use a common structure in many different files, then the technique of kernel documents is probably applicable. With kernel documents, the author creates a hypertext "structure" that is used in many different files and then only adds to it the information specific to each file. This technique eliminates much of the time spent building objective links and document structure. A second benefit is that when a kernel document is used, macros can be create to operate on that structure -- without having to modify those macros for each particular hypertext file.

The final rule for designing documents for maintenance is to avoid duplicating data in more than one hypertext document. This eliminates having to correct, and possibly missing, more than one instance of any piece of information.

B. Quality control

Like document maintenance, quality control in a hypertext document, is a very poorly defined topic - if it is defined at all. Simply defined, hypertext quality control is making sure that every link works and that every link goes to its correct destination.

The simplest way to tackle quality control for a series of documents is to test every link in the document. This, however, is an exhaustive and lengthy process. Rather than test every link in a document, an alternative approach is to try and minimize the number of potential bad links.

Four ways to minimize the number of potential bad links are the use of the Mimic command, the use of word links, the use of macros, and authoring consistency.

The mimic link command duplicates an existing link created in a document. If you need to create more than one instance of a link to the same information, it is better to mimic the original link than to create a new link. First, mimicking an existing link is faster for the author. Second, provided that the first link was created correctly, the second link will also be created correctly as it is an exact duplicate of the original link. Finally, mimicking a link uses less storage and memory than does creating that from scratch.

HyperWriter's word link feature allows you to create implicit links based on user defined words or phrases. There are two advantages to word links from a quality control / maintenance standpoint. First, word links are global and can be changed globally. Changing an existing word link changes every instance of that word link.

This makes updating very easy. Second, all defined word links can be viewed with the View Word Link command. This aids you in easily maintaining a document.

Macros are a series of recorded keystrokes that follow an author's commands explicitly. The advantages to using macros as a document building tool is that provided they are recorded properly, they will create links and document structure faithfully without error.

The final aspect to quality control is that of authoring consistency. Try to establish a series of working guidelines for how you link and structure documents. Ultimately, this consistency should help in preventing you from making errors. Another benefit to authoring consistency is that should a new author have to maintain your documents, this consistency will help them come up to speed on your documents significantly faster.

C. Maintenance Tools

Some of the tools that HyperWriter offers for document maintenance are:

D. Link Info

HyperWriter's link info (<F2>) feature examines links in terms of their author, the date they were created and the keywords assigned to the link. This information is often useful when it comes to tracking changes in a document or documents.

E. Document Security

HyperWriter's document security features can be used to prevent updates to parts of the document by authors not qualified to make changes to the material.

F. Data Encryption

Although not directly related to document maintenance, HyperWriter's data encryption needs to be considered for maintenance. Data encryption prevents readers of the hypertext documents from modifying documents with disk editors such as the Norton Utilities. Data encryption can thus be viewed as a tool to maintain the validity of updated hypertext documents.

G. Guided Tours

Guided tours can be applied to document maintenance both for the document reader and the document author.

First, for the hypertext reader, a tour with the name of "Updated Material" or something similar could be created that offers the reader a tour of all updated material. If the tour was created as a "Query User" tour that prompts the reader at each step of the way, the reader can always exit the tour to browse the material in further depth. Then, when done browsing, the reader can resume the tour to view the next chunk of updated material. This approach of constructing tours through updated material eliminates a hypertext fear --- that of not being able to easily access ALL updates to a document.

If the document is being updated in many different areas, then several different tours should be constructed through the updated material. The first should be a meta tour that tours the reader through ALL the updated material. The additional tours that should be constructed should be created on a thematic basis. Each tour should cover one particular part of the updated material. This allows readers to only access the updates that pertain to them.

Document authors can also apply tours to document maintenance. When a hypertext document is being constructed (or any document, for that matter), it is often apparent what material will need to be updated. The author could construct guided tours through this material that is expected to require updates. Then, when it is time to update the document, the author has a record of the parts of the document or documents that may need to be updated.

H. Search and Replace

HyperWriter's search and replace functions can be used to locate information within documents for updating. Use search and replace for this in the same fashion that they are regularly used.

I. Detached Windows

The final tool for maintaining documents is that of detached windows. A detached window is simply a window that has been unlinked from a document and is stored in the unlink queue. Unlinked windows do not show up on the map, nor are they generally accessibly (except with the Search or Goto Window functions). Detached windows can be used store document relevant information such as notes, maintenance ideas, etc.

As an example of authoring consistency and detached windows in action, at NTERGAID, we have recently adopted a practice of making the first link created in a document a jump link to new text and then unlinking that window. This creates a detached window (Window #2) that we use to store our own notes on the building of the document as well as our maintenance notes for the document. This eliminates any need to maintain separate paper based notes for our documents.

By ensuring that this window is always window number 2, a general macro (say <F7> N for Notes) to access this window can be constructed and always used.

If you wish to prevent anyone from inadvertently gaining access to this window of the document, simply change its data access setting to Restricted.

IX. Structured Hypertext Conclusion

Structured Hypertext is a Methodology for creating hypertext more easily and quickly.

Structured Hypertext uses a kernel document approach.

Structured Hypertext scales up for large projects - it is aimed at large projects.

Structured hypertext lessens the work load of creating hypertext in three ways:

- Design work is done only once.
- Chunking can be done by a non-domain expert.
- In the future, documents are a known quantity for maintenance.

Appendix 1: How the Authoring Tool Affects Document Structure

Although not often recognized, the authoring tools used for a hypertext document greatly affect the nature of that document.

The classic example is that of HyperCard based document where every document, even documents not well suited for it, are all rendered as similar stack based documents. Similarly article based systems, such as HyperWriter or Guide, tend to inspire article based documents.

Appendix 2: About the Instructor

J. Scott Johnson is the President and Director of Marketing for NTERGAID. He has been actively involved with hypertext for the past 3 1/2 years since the founding of NTERGAID. Among his roles at NTERGAID have been systems designer, software engineer and documentation author. The different hypertext authoring projects he has been involved with include:

- Culture 1.0 / DOS
- The EPA's Reg-In-A-Box
- The AIL FIP HyperManual
- The ITCC Kiosk Program
- The Hypertext '87 Digest
- The Doctor Dobb's Journal June 1990 Hypertext Edition

Appendix 3: About NTERGAID

NTERGAID is a software developer and publisher specializing in the development of hypertext, hypermedia and multimedia authoring tools. Among the tools developed and published by NTERGAID are Black Magic, Help System II and HyperWriter. NTERGAID was founded in Spring of 1987.

NTERGAID and J. Scott Johnson can be contacted at:

NTERGAID, Inc.
2490 Black Rock Turnpike
Suite 337
Fairfield, CT 06430
Voice: (203) 368-0632
Fax: (203) 334-5658
BBS: The HyperBoard! (203) 366-5698
CIS: 75160,3357
BIX: s.johnson

Appendix 4: About HyperWriter

HyperWriter is an interactive, hypermedia and multimedia authoring systems that takes a "editor-based" approach to creating hypermedia and multimedia documents. HyperWriter's basic user interface is that of an editor that lets you mark blocks of information and indicate relationships between the information. Due to this editor based approach, no scripting is required.

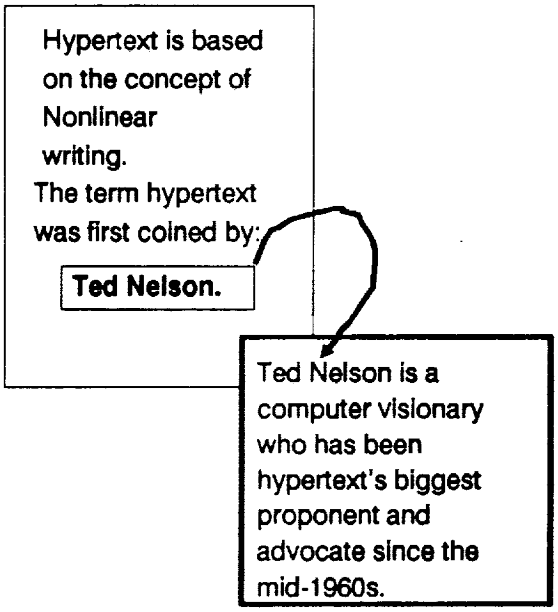
Pioneer and Sony laser disc players can be fully controlled by HyperWriter as a standard multimedia input device. Supported data types are text, graphics (640x480x256 colors), digital audio and digital video.

Appendix 5: Hypertext Concepts

I. Hypermedia Concepts

A. Hypertext

Hypertext is writing in a computer-based medium that supports nonlinear reading and writing.



Hypertext is based on the concept of Nonlinear writing.
The term hypertext was first coined by:

Ted Nelson.

Ted Nelson is a computer visionary who has been hypertext's biggest proponent and advocate since the mid-1960s.

Figure 2: A Hypertext Document with a Single Link

A characteristic of hypertext is that it gives documents a "three dimensional" nature.

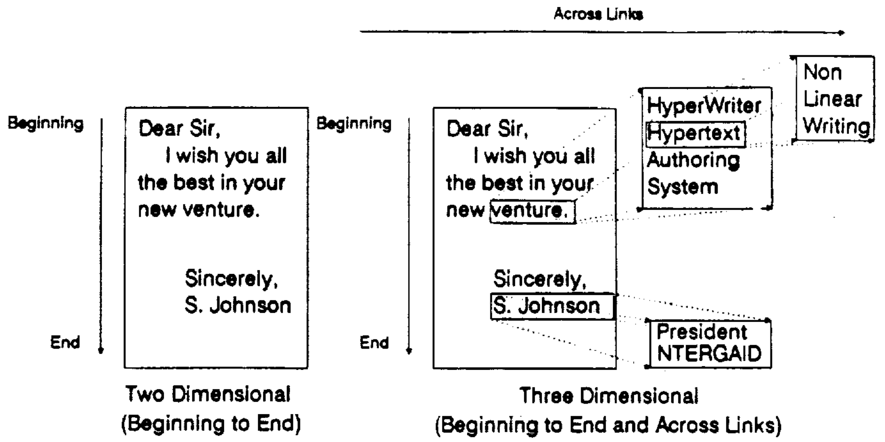


Figure 3: The Three-Dimensional Nature of Hypertext

B. Hypermedia

Hypermedia extends hypertext to allow nonlinear creation and retrieval of multiple media types such as text, graphics, sound and other media. Hypertext and hypermedia can be used interchangeably.

C. Multimedia

Multimedia is a computer technology that supports the use of several different media types on a single computer system and display. Media types such as text, graphics, animation, sound and video images are commonly used in multimedia systems.

D. Link

Links are the basic element in a hypertext system. Links form an instant relationship between two pieces of information.

E. Link Anchor

Link anchors are the selectable region that denotes the start of a link. This is also called the link source or the link icon. Link anchors can be characters, words, lines or even longer. A link anchor can even be a three dimensional "button" or simply any rectangular region over a graphic image.

F. Link End

The Link End is the destination of a link; the data that the link anchor accesses when the link is activated. A link end can be a window of text. It can be a specific paragraph within a window. It can also be a graphic image. A special link end, called a link to action, does not have a physical destination, but instead its link end executes a physical action.

G. Window or Node

A window or node is the basic sub-unit or element of a hypertext document. It can also be called a card or a page. Hypertext documents generally have a number of nodes connected by nonlinear links. A single hypertext file can consist of one or more nodes although generally a single file has many nodes.

H. Text Stream

Text streams are linear piece of text within a window or node of a hypertext document.

I. Articles Versus Cards

Two basic ways to structure a hypertext document. Article based documents allow scrolling of text. Card based documents utilize fixed-length chunks or "cards" of information that do not scroll.

Overall, article based documents are easier and faster to create than card based documents due to the elimination of "chunking" linear material across cards.

Card based documents are well suited to a presentation medium.

J. Unidirectional Links

Unidirectional links go from link anchor to link end but NOT from link end to link anchor. Bidirectional links go from link anchor to link end and from link end to link anchor.

HyperWriter uses unidirectional links.

K. Point to Point Linking

There are two basic types of linking - point to point links and node to node linking. Point to point linking consists of linking from one specific piece of information inside a node to another piece of information within a node. Node to node linking, on the other hand, is linking from one node to another node. A variant on point to point linking is called point to node. Point to node linking links information from specific link anchors in a document to whole nodes - not to specific points within a node.

HyperWriter supports both point to point and node to node although it is more oriented to point to point linking.

L. Objective and Subjective Links

Objective links form the structure of a document - they are the links from the table of contents to the table of contents entries. Subjective links are links created by editors that offer insights into the material.

Overall, objective links make hypertext documents possible but subjective links add the real value to a document.

M. History Trail

A history trail is the path of links activated and nodes visited that a user has followed throughout reading a document or documents. HyperWriter supports an infinite history trail (limited only by available memory).

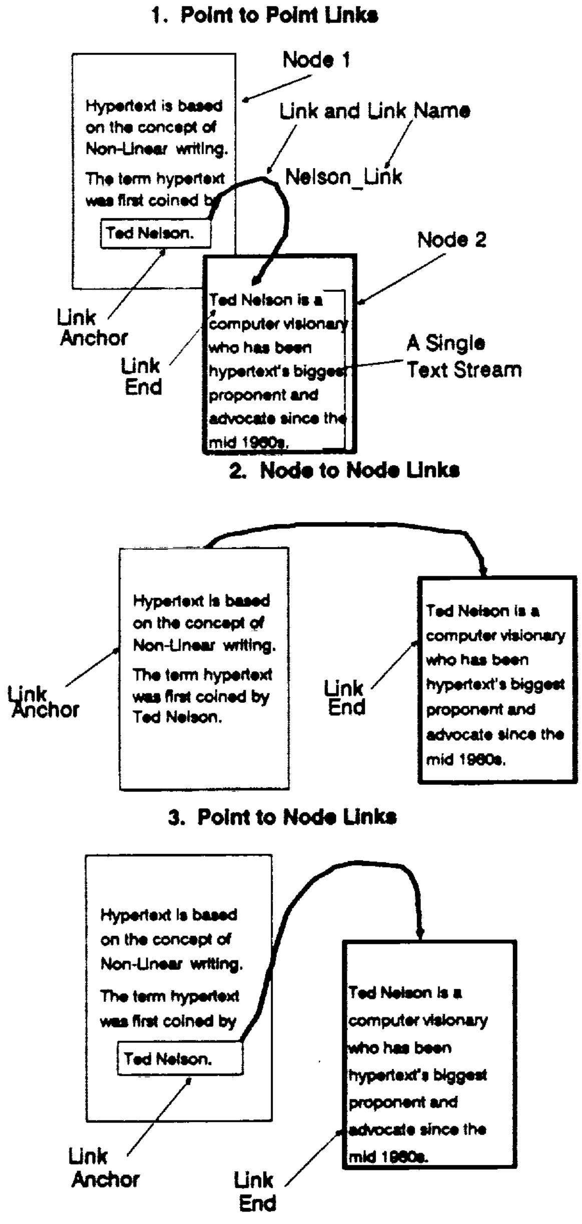

1. Point to Point Links
Node 1
Link and Link Name
Nelson_Link
Hypertext is based on the concept of Non-Linear writing.
The term hypertext was first coined by
Ted Nelson.
Node 2
Ted Nelson is a computer visionary who has been hypertext's biggest proponent and advocate since the mid 1960s.
Link Anchor
Link End
A Single Text Stream
2. Node to Node Links
Hypertext is based on the concept of Non-Linear writing.
The term hypertext was first coined by Ted Nelson.
Ted Nelson is a computer visionary who has been hypertext's biggest proponent and advocate since the mid 1960s.
Link Anchor
Link End
3. Point to Node Links
Hypertext is based on the concept of Non-Linear writing.
The term hypertext was first coined by
Ted Nelson.
Ted Nelson is a computer visionary who has been hypertext's biggest proponent and advocate since the mid 1960s.
Link Anchor
Link End

Figure 4: Anatomy of a Hypertext Document

Comment

In **1.**, a Point to Point link is shown linking the reference to Ted Nelson to a node that contains a description of who Ted Nelson is. Additional information could come either before or after the description.

In **2.**, a Node to Node link is shown linking the node on hypertext to the node on Ted Nelson.

In **3.**, a Point to Node link is shown linking the reference to Ted Nelson to the node describing him.

Appendix 6: Obtaining the Hypertext Version

A hypertext version is available free of charge directly from NTERGAID. Simply contact Scott either at the conference or via phone or mail after the conference and a DOS based version will be sent to you.

Appendix 7: Navigation In HyperWriter

A. Link Activation & Return

Links are represented by ASCII delimiters, three dimensional buttons, or areas on a graphic image.

ASCII link delimiters can have one of three different colors - Green for a Link to Text, Blue for a Link to Graphics and Red for a Link to an Action.

Links from graphic images are represented by Areas on the graphic image - not by explicit link tokens. A command key, <Ctrl> S can be used to highlight all links on a graphic image.

Links are activated with the <F1> key, the Left Mouse Button, or by using the Menu System.

Return From Links with the <Esc> key, the Right Mouse Button, or by using the Menu System.

HyperWriter maintains an infinite trail of everywhere that you've been in a document. This trail can always be backtracked through with the <Esc> key.

B. The History Function

The History function displays a pop-up list of the windows visited in the document along with the time spent in each window. This is useful for quick browsing and recall of your path. It can be used to jump back directly to any of those windows by highlighting the window and pressing <Enter>.

C. Text Search

HyperWriter has a basic in-memory string search activated by <Ctrl> F. Searches are repeated with <Ctrl> A. Search Options are set with <Ctrl> O.

D. Graphic Link Map

HyperWriter's graphic link map can be used to look at the structure of a document visually. To access the map, press <Ctrl> <F2>. This maps the document out and scrolls the map down to your current location in the document. The <ArrowKeys> or the Mouse can be used to scroll the map up and down. The Left Mouse Button or <F1>

key can be used to jump to any node on the map.

E. Tag Window/Collapse to Tag & Home

Tag Window marks the current window as one to be later returned to from deeper within the document. Tag Window is activated with the <Ctrl> <F1> sequence. This tags the window and displays the message "Current Window Tagged." To return to that window from deeper in the document, press <Alt> <F1> to collapse back to the tagged window.

The Home command is used to return to the first window of the first hypertext document accessed during a HyperWriter session. The Home command is accessed from the Navigate menu.

F. Annotation

Annotation allows authors or readers to add their own notes to a document. To annotate a document, choose the Add Readers Note command from the Utility menu.

Readers Notes are entered into a vertical text entry window that appears at the right hand edge of the screen. Full editing control including block manipulation is available. Readers Notes can be of unlimited length and do allow scrolling.

After readers notes are added to a document, a Check Mark (See Figure II-4, top) appears at the top of the screen. To read the notes, the check mark can be activated by selecting it with the Right Mouse Button, or by issuing the Read Notes command from the Navigate menu.

G. Guided Tours

Guided tours allow an author or a reader to create a directed path through a document or documents. When a tour is run, the user is sequenced through a series of nodes of the document or documents either being prompted at each step of the way or on a timed basis. When the tour prompts the user to continue, he or she has the option to exit the tour and browse the material or continue to the next stage of the tour.

Some of the uses for guided tours include slide shows and links from one link anchor to many different destinations.

To create a tour, press <Shift> <F8>. Select the Create option and enter a name for the tour. After the name is entered, navigate to each node to be on the tour and press <Shift> <Enter>. This adds that node to the tour. After all nodes are added to the tour, select the Save option from the <Shift> <Enter> dialog box. This saves the tour.

To run a tour, press <Shift> <F8> and highlight the desired tour in the Tours dialog box. Then, select Run to run the tour.

H. Bookmarks

Bookmarks are tiny (6 byte) files written, at the user's preference, when exiting a HyperWriter document. Bookmarks have the filename FILENAME.BKM where

FILENAME is the name of the original HyperWriter document being exited from. Whether or not bookmarks are used can be set on the General Configuration screen accessed from the Options menu.