Xanalogical Structure, Needed Now More than Ever: Parallel Documents, Deep Links to Content, Deep Versioning, and Deep Re-Use

Theodor Holm Nelson

*Project Xanadu** *Web: http://xanadu.com/and*
Keio University *Web: http://www.sfc.keio.ac.jp/*
Web: http://www.sfc.keio.ac.jp/~ted/

* "Xanadu" is a registered trademark; "xanalogical structure" is intended as a related generic (non-trademark) for general use.

Summary

Project Xanadu, the original hypertext project, is often misunderstood as an attempt to create the World Wide Web.

It has always been much more ambitious, proposing an entire form of literature where links do not break as versions change; where documents may be closely compared side by side and closely annotated; where it is possible to see the origins of every quotation; and in which there is a valid copyright system -- a literary, legal and business arrangement -- for frictionless, non-negotiated quotation at any time and in any amount. The Web trivialized this original Xanadu model, vastly but incorrectly simplifying these problems to a world of fragile ever-breaking one-way links, with no recognition of change or copyright, and no support for multiple versions or principled re-use. Fonts and glitz, rather than content connective structure, prevail.

Serious electronic literature (for scholarship, detailed controversy and detailed collaboration) must support bidirectional and profuse links, which cannot be embedded; and must offer facilities for easily tracking re-use on a principled basis among versions and quotations.

Xanalogical literary structure is a unique symmetrical connective system for text (and other separable media elements), with two complementary forms of connection that achieve these functions -- survivable deep linkage (content links) and recognizable, visible re-use (transclusion). Both of these are easily implemented by a document model using content lists which reference stabilized media.

This system of literary structure offers uniquely integrated methods for version management, side-by-side comparison and visualizable re-use, which lead to a radically beneficial and principled copyright system (endorsed in principle by the ACM). Though dauntingly far from the standards which have presently caught on, this design is still valid and may yet find a place in the evolving Internet universe.

Xanadu History and Philosophy

I will try to summarize the intentions and designs of the Xanadu project over its forty years, stressing its main concepts. What follows necessary simplifies and omits, but I think it correctly represents the center of what the whole group has tried to do over all this time, although I am speaking for myself in particular (as chief designer and trademark holder until 1983, and again since 1992).

Project Xanadu [Nelson 1997], [Nelson 1980], [Nelson 1981], [Nelson 1987], [Nelson 1990], [Xanadu 1997] is an alternative paradigm for a computer universe, with its own alternative history of the computer field and alternative ideas of media, computer life, and the nature of connections. It has been misunderstood in as many different ways as there are computer ways of thinking. We have been frequently criticized for organizational and tactical errors and stylistic offenses, but there has been no general comprehension of our actual technical work. It is the objective of this article to clarify our alternative mental model with enough depth for others to understand; though there is room here for few of its many ramifications.

Like Xerox PARC, Project Xanadu has innovated broadly, with documentable wide influence, but has had difficulty delivering product (and much poorer public relations, perhaps due to our net negative funding).

This work began in 1960 with essentially the model to be described here, though in less detail. For forty years, Project Xanadu has had as its purpose to build a deep-reach electronic literary system for worldwide use and a differently-organized general system of data management. The point has been not to simplify the world of ideas and connection, or force others to simplify (as today's software and hypermedia do); the point has been to represent the world of ideas correctly and clearly, which is much harder -- replacing not just paper media, but conventional computer files and hierarchy, with finer-grained and wholly different families of structure.

Over these forty years, over one hundred people have been involved with the project, of whom about at least twenty-five* have been active at the software design level, creating a number of principal designs** (depending on what is counted); which have included a number of interesting technical ideas*** that there is no room to discuss here.

The Xanadu project has generally kept apart from mainstream discussion because we have agreed with little in the majority paradigms of the computer world, even as they evolved into today's global shallow hypertext. Both "computer science" and commercial software have had completely different agendas throughout these forty years. We have always had a completely different proposal for electronic literature, and indeed for the structure of all computing. We have always proposed a complete alternative computing and literary universe -- sweeping, simple and principled -- which has remained very different from the evolving computer world and its evolving traditions that masquerade as "technology".

Ours has been a tall agenda, quite specific from the beginning. It is easy for others to second-guess what we should have done; but in computerdom's perpetual standardization war and changing technical, political and business environments, we did what we thought was best under very difficult odds; perhaps changing designs too often but always seeking the same goals.

Project Xanadu was the original hypertext project and more. Our intention has been not merely to create an electronic literary structure, but to import literary concepts into a redesign of the rest of the software world. We sought to reduce the influence of hierarchical directories and conventional files (which we see as large lumps with stuck names in fixed places, with compulsory gratuitous naming [Nelson 1986] -- unsuited to overlap, interpenetration, rich connectivity, reasonable backtracking, and most human thinking and creative work [2000].

The World Wide Web was not what we were working toward, it was what we were trying to *prevent*. The Web displaced our principled model with something far more raw, chaotic and short-sighted. Its one-way breaking links glorified and fetishized as "websites" those very hierarchical directories from which we sought to free users, and discarded the ideas of stable publishing, annotation, two-way connection and trackable change.

This article is intended to explain our alternative model.

Parallel Documents and Transpointing Windows

Our parallel universe [Nelson 1998a] begins with parallel text. We hear now many theological arguments about the divinely intended form of electronic documents. Reluctantly skipping that level, let us directly consider a fundamental document type: *parallel documents* [Nelson 1998b].

Parallel documents are everywhere, but are not generally acknowledged. There are relatively few explicitly parallel documents (like Tom Stoppard's play "Rosencrantz and Guildenstern Are Dead", which is explicitly parallel to "Hamlet" -- showing events that occur offstage in "Hamlet", and vice versa). But implicitly parallel documents are everywhere -- the parallelism of commentaries, the parallelism of long and short versions of reports, the parallelism of translations, the parallelism of holy books [Nelson 1998]. It is vital that we be able to see this parallelism of documents and to intercompare and work with their side-by-side connection.

This leads to the basic interface of our model, parallel visualization. To clarify our model we will look at pictures.

Side-by-side connected comparison of parallel documents on the computer screen has always been Xanadu's fundamental visualization, first published in our 1965 paper [Nelson 1965].

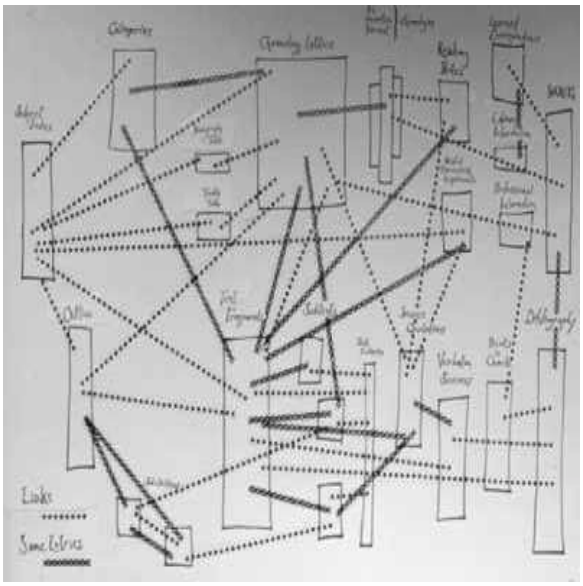

Figure 1. Side-by-side connection as schematically illustrated in 1965
(author's original diagram [Nelson 1965], later redrawn for publication).

Figure 1 shows what we believe is the first published visualization [Nelson 1965] of parallel screen documents, in abstract form. This was meant to represent side-by-side items in columns. Each rectangle indicates a sequence of paragraph-like content items (like today's "mailbox" files, to be viewed as a column on the screen). Braided lines in the illustration indicate that some of the items to be viewed in one column are to be shown as identical to items in the other column (transcluded), while dotted lines indicate that they are to be shown as only linked.

We published this proposal for side-by-side screen connection as a more detailed visualization in finer grain in 1972 [Nelson 1972] (Figures 2,3).

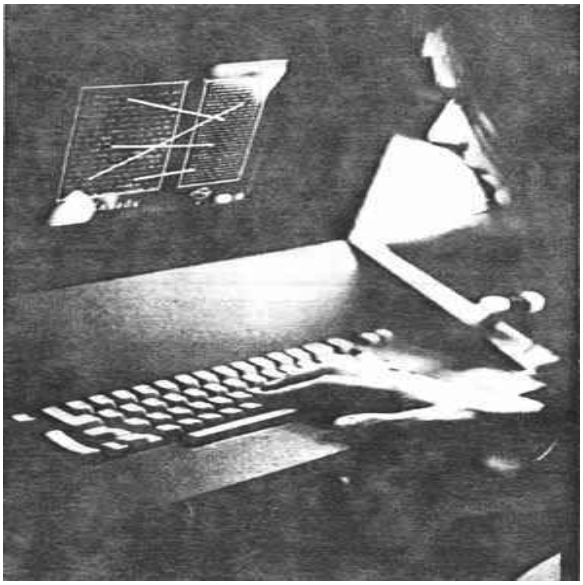

Figure 2. Mockup of transpointing windows, 1972.

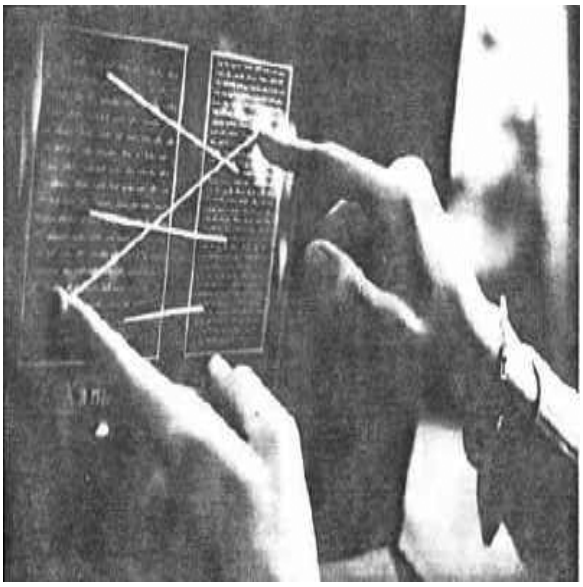

Figure 3. Same mockup. closeup view.

In Figures 2 and 3, the actual contents in one window are shown cross-connected to actual contents in another window. Naturally, these connections must remain connected to the contents no matter how the windows scroll or move around. We now call this visualization "transpointing windows" [Nelson 1995].

These early photographs show only a cardboard-celluloid mockup of transpointing windows.

We believe that the first functioning interactive screen implementation of transpointing windows was created by John Walker at Autodesk in 1988, supporting parallel screen connection and scrolling, served from our xu88 software. Unfortunately no pictures of Walker's system are available.

A different implementation of transpointing windows, a demo running under Windows 95, was produced by the author and Ian Heath in 1998 (Figure 4, [Transpointing 2000]). (Source material courtesy of *Caerdroia* magazine [Caerdroia 1999].)

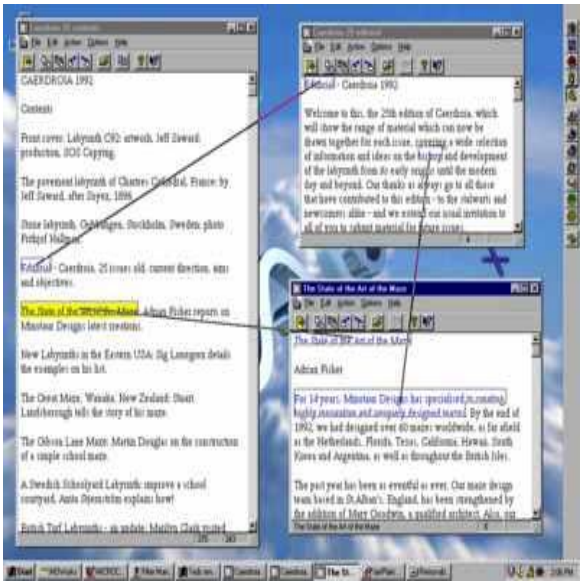

Figure 4. Transpointing windows:
screen shot of the 1998 working demo

People are amazed to see these windows move and scroll, with connections following the content as it moves on the screen. Unfortunately that is all that the demo does, since it has no support for editing or saving the connective structure.

A more recent interactive screen implementation of transpointing windows (screen shot Figure 5) was created by Ka-Ping Yee in 1999. This was served, like Walker's, from our xu88 server (rechristened as Udanax Green). Yee's PYXI is a skeleton front end that shows and edits text, links and transclusions, and may be extended to reach more of the server's functionality. Both Udanax Green and PYXI were released under an open source license in 1999 [Miller 1999], and are available at udanax.com.

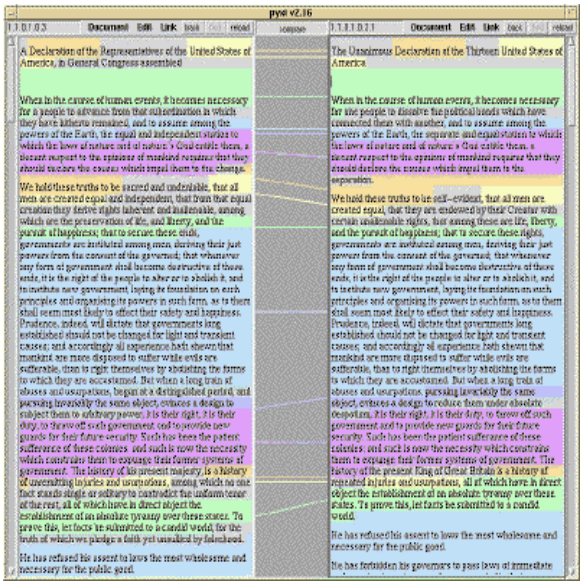

Figure 5. Screen shot of transpointing windows by Ka-Ping Yee, showing his PYXI viewer served from Udanax Green server (xu88 model). Only transclusions happen to be shown, though PYXI also handles content links.

In Figure 5 we see early and final drafts of the Declaration of Independence [Jefferson 1776], highlighting transclusions and differences with color.

Uses of This Visualization

We believe that everyone needs transpointing windows to support analysis and detailed understanding -- by parallel commentary, precise annotation and the explication of contents; by the facilitation of pinpoint controversy. These are needed not just in scholarship, but legislation, diplomacy, and anywhere that interrelated documents need to be seen. We need this visualization for side-by-side parallel documents (from holy books to legislation); for detailed explication, commentary or disagreement; for comparing successive versions of a document. We even need it for today's one-way hypertext: many users would be more comfortable seeing how they recently reached a document.

Transpointing windows are also badly needed in the writing and editing processes.

The real work of writing is rewriting; and especially in big projects, is principally the overview and control of large-scale rearrangement -- a rearrangement process that used to be called "cut and paste" until those terms were redefined by the Macintosh in 1984.

The writing process, and especially sophisticated rewriting, consists of repeatedly reorganizing evolving drafts. Thus what is most needed for writing and editing are ways to track the continuities among drafts, and compare alternative plans of organization and parallelism among different versions.

The Xanadu project has long intended to use transpointing windows for editing that shows origins, as illustrated in Figure 6.

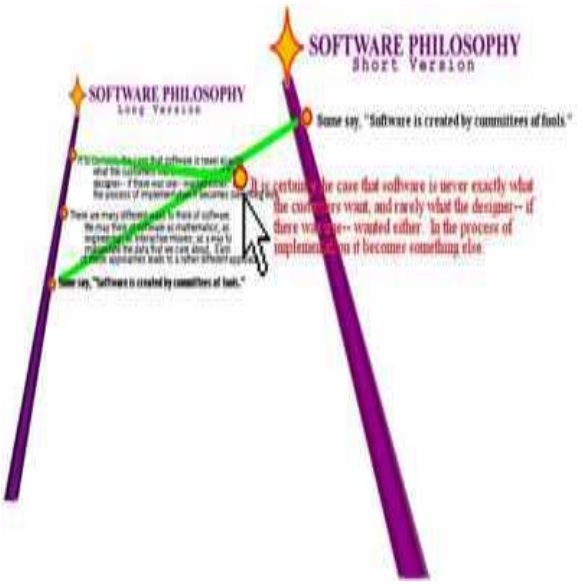

Figure 6. Pullacross editing is another use of transpointing windows (simulated graphic).

Instead of deleting content from one place and plugging it into another (today's distorted meanings of the venerable terms "cut" and "paste"), the author should be able to pull screen contents from old versions into new, seeing all points of origin and also seeing what contents have not yet been used.

In Figure 6, we see trails of origin, or origin beams, showing content transcluded from one version to another. Such beams, optionally visible by the author after content has been moved, should make clear where the content is from and what's been left out. (Shown at paragraph level for simplicity; in post-1965 models we have striven to take this to the level of individual characters.)

Such a mechanism of side-by-side comparison and work -- what we call pullacross editing -- should allow authors far easier revision, able to track visually the original content from previous drafts -- telling at a glance where things come from and what is missing.

Transpointing windows, as shown for editing, can be both a tool for the writer to perform the operation, and a tool for the reader to see versions and their differences. Note that version management and CASE (Computer-Aided Software Engineering) tools, as they have independently developed in various areas, often do the same things in more formalized and difficult style, and have typically not been generally intended for users as well as authors.

Making transpointing windows come true has always been a key milestone for the Xanadu Project. These self-explanatory pictures have been the simplest sharable tokens inside and outside the group. You do not have to understand computer technicalities (let alone ours, new or old) to understand this visualization, or see its importance. We are amazed that the goal of making it possible for people to work with transpointing windows appears not to have been embraced by anyone else in the computer field. (A search at Google.com for "transpointing windows" on 7 May 2000 found only ten references.)

A Literary Structure with Two Fundamentally Different Forms of Connection

Transpointing windows have long been a clear Xanadu objective, both to show links (as in Figures 2, 3, 4) and shared content (as in Figures 5, 6) -- or both (as in Figure 1). Now we must discuss these two forms of connection.

Documents are information packages with points of view; literature consists of persistent documents, maintaining the continuity of ideas. Literary boundaries are the boundaries between these packages.

Over the centuries, people only could save a few types of media as lump media objects (books, works), and connections had to be somehow be embedded in these lump objects. In those paper years, authors had to show connection and versioning in their lump objects through footnote, quotations, marginal glosses, Talmudic layout and edition numbers; but these were crude two-dimensional mappings of the true relationships. For over two computer decades we have imitated paper on screens (extremes being Adobe Acrobat and the HTML page). What we really need is the opposite -- to represent digitally the literary forms of connection which could not be represented before.

To Project Xanadu, that means enacting two types of connection: profuse and unbreakable *deep links* to embody the arbitrary connections that may be made by many authors throughout the world (content links); and *a system of visible, principled re-use*, showing the origins and context of quotations, excerpts and anthologized materials, and content transiting betwen versions (transclusions).

This may be simplified to: connections between things which are *different*, and connections between things which are *the same*. They must be implemented differently and orthogonally, in order that linked materials may be transcluded and vice versa. This double structure of abstracted literary connection -- *content links* and *transclusion* -- constitute xanalogical structure. They are literary connections because they deal with literary boundaries [Nelson 1995]. In the Xanadu designs we have always striven to represent both forms of connection.

Transclusion is what quotation, copying and cross-referencing merely attempt: they are ways that people have had to *imitate* transclusion, which is the true abstract relationship that paper cannot show. Transclusions are not copies and they are not instances, but *the same thing knowably and visibly in more than once place*. This is a simple point which is remarkably difficult to get across. While copies and cross-reference are workarounds in place of transclusion, aliases and caches are *forms* of transclusion.

Any implementation of transclusion is necessarily a simulation or an enactment, whether accomplished by a live connection, a cache, or an alias on the desktop. But in these simulations and enactments we must make visible the knowable identity; which copies and instances do not.

Not distinguishing between links and transclusions is causing misery everywhere, for instance in lawsuits against having one's page brought into someone else's frame (a form of transclusion) which the lawyers refer to as "linking" -- hopelessly confounding a key difference.

Note also that the famous "trails" of Vannevar Bush's memex system [Bush 1945] were to be built from transclusions, not links.

The Xanalogical Model: Mechanics and Former Secrets

Thus far we have considered two levels of this proposed literary system: intended screen appearances, and the two complementary forms of xanalogical connection (content links and transclusion). Now we turn to its supporting system of indirect representation by version lists and connection lists.

What will be presented here is a fundamentally simple model for a large-scale hypertext literature built on xanalogical structure, extremely different from the World Wide Web, intended to support profuse fine-grained 2-way unbreakable links and principled re-use.

This underlying model is simple but generally not known, in part because our methods were under complex proprietary ownership, and thus trade secret, until the open source release of prior Xanadu code in August 1999. However, these proprietary methods (especially enfilade theory and enfilade specifics [Nelson 1999]) were really for efficiency in carrying out methods like those to be described below.

The central proprietary secret this all relied on -- which we considered whimsically obvious but never stated publicly -- was the freezing of content addresses into permanent universal IDs, below.

We will go over our fundamental model, resolved to specifics of the xu88 design (simplified for explanation). By being this specific we can hope to be tediously clear. While xanalogical structure is conceptually independent of this more detailed model, the xu88 design serves as an existence proof and an available client-server ensemble, now available at Udanax.com under an open source license with the names Udanax Green and PYXI, and thus constitutes (at the risk of some confusion) what is usually called a "reference version" of these concepts.

Use in Place: Referential Fluid Media

In this model, documents, versions and links are structures which point to content and use that content by reference. Operations on these structure lists do almost everything. This is the opposite of the prevailing, and extremely limited, embedded approach [Nelson 1997c]. What follows will be a detailed walkthrough of data and methods for implementing xanalogical structure with this referential data model.

Other people's text systems typically bring text into a software buffer and modify it -- thereby losing all information as to origins, other versions or other connections.

Our alternative is referential editing of referential documents. We give any text (including newly-input text strings) a permanent address, and thereafter use that text by pointing to it. Thus text is used "in place", accessible simultaneously to many possible applicative uses. The same method is used for other data consisting of countable, separable media elements. (Such reusable media content with permanent element addresses we now call "fluid media" [FloatingWorld 1999].)

Let us see how referential operations work for bodies of sequential text (Figure 7), then consider links and transclusions. Discussion of richer structures is for elsewhere.

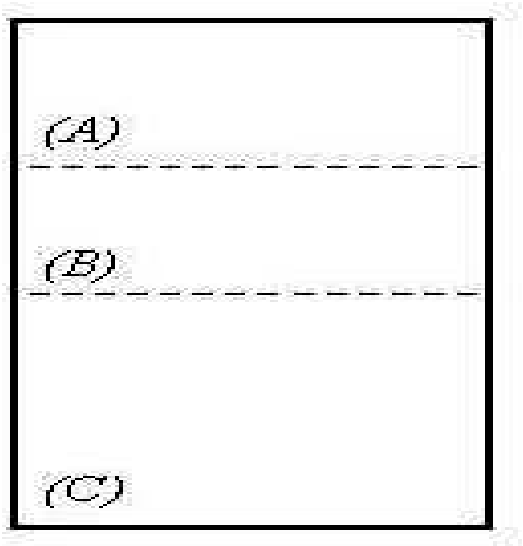

Figure 7. Sequential text as it often appears on a screen.
(Parenthetical labels and portion boundaries are only for matching to Figure 8.)

A sequential document or version is represented by a content list, which is the fundamental form of representing or transmitting a document. (Note that the operative unit of the xu88 design is the *version*, but the term "document" will be clearer to most readers.)

Though the content may appear on the screen as an ordinary block of text, structurally its chief representation is a list of contents -- a sequence of reference pointers. These pointers designate spans of characters (or other elements) (Figure 8). These spans, concatenated to make a virtual stream of the designated elements, thus represents the sequential text of Figure 7.

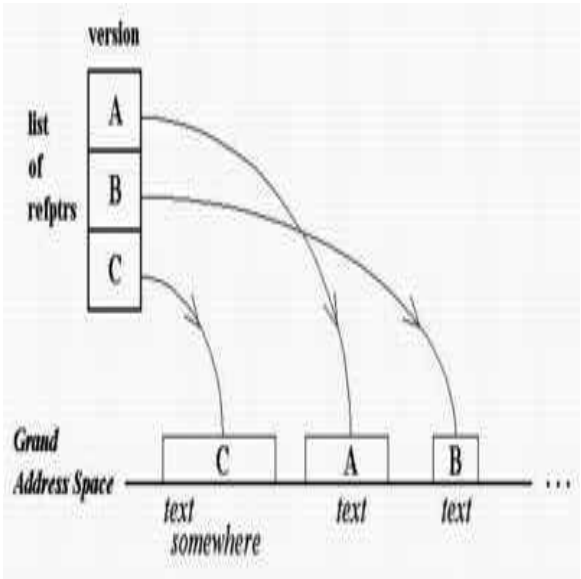

Figure 8. List of contents representing the text version of Figure 1. Proportions are not significant.

All editing operations change these content lists, which remain the fundamental form of representing or transmitting a document; and all search and editing operations are on these content lists. These operations are identically performed, regardless of who owns which parts of the content or where the content actually resides. Pulling up the actual contents for user viewing or editing is essentially cosmetic and informative to the user, since internal operations are on the version lists themselves.

These spans of content may be in separate locations anywhere in the address space (i.e., the universe). But content may be cached in many places, as long as it has the same permanent IDs. (However, this is a level of indirection separate from the basic model--a provision which onlookers have generally overlooked, to their great confusion).

Hollywood discovered this method separately. This is how movies are now edited; this is how the xanalogical model has always edited contents of any countable kind. What Hollywood calls an EDL, or Edit Decision List, we call a content list or version list.

Thus the delivery of a document is in two logical phases: the content list, then the fulfillment of the contents. Getting the actual contents is like going to retrieve the film negative itself from the vault -- conceptually a final stage. Having two stages may sound like unnecessary overhead, but it is similar in some ways to the many stages of protocol exchange in the delivery of a multipart Web page today. It may also be compared to the overhead of the Internet itself -- enormous in terms of program cycles and data packets, but an enormous simplifying benefit, rather than a burden, in its result.

Content Links, and How They Survive

The xanalogical content link is not embedded. It is *applicative* -- applying from outside to content which is already in place with stable addresses. Xanalogical links are effectively overlays superimposed on contents. Any number of links, comments, etc., created by anyone anywhere, may be applied to a given body of content. By this method it is possible to have thousands of overlapping links on the same body of content, created without coordination by many users around the world. (Today's embedded links, as in HTML -- a later and cruder design -- cannot be profuse, overlapping, bidirectional, or diversely created in this way [Nelson 1997c].

Each xanalogical link is implemented as a separate unit which connects spans of content. This in turn means that a link can apply to any of that content, wherever the content may be re-used.

First let's consider a simple document, then a comment link on it, and watch how that content link continues to adhere to the changing document. (We will assign toy addresses, simpler than those in the xu88 protocol, to make the concepts clear.)

Suppose author Adam creates document A, version 1, which he publishes on line as the first document in this format. He publishes all 300 characters at once, causing these characters to be given consecutive universal addresses. The text contents of document A v.1 are therefore registered as having consecutive addresses from 1 to 300.

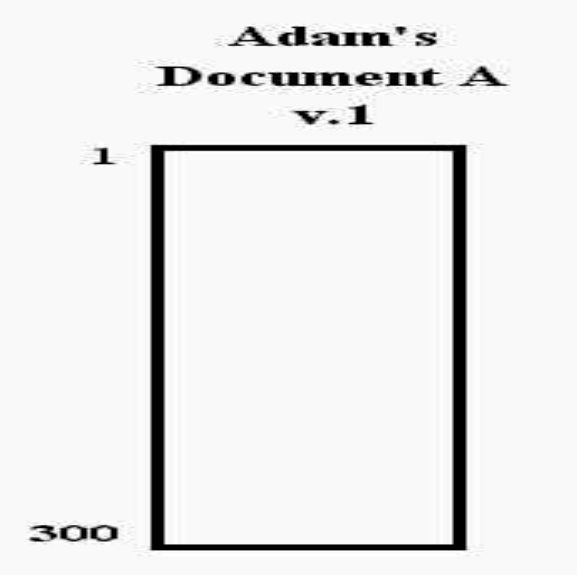

Figure 9. Screen visualization of Adam's document A, still an unbroken stretch of content bytes.
(Numbers on the side are for following further examples.)

This document is internally represented as a table of pieces. Since it so far only has one piece, it is saved as a single reference pointer to the consecutive characters 1 to 300.

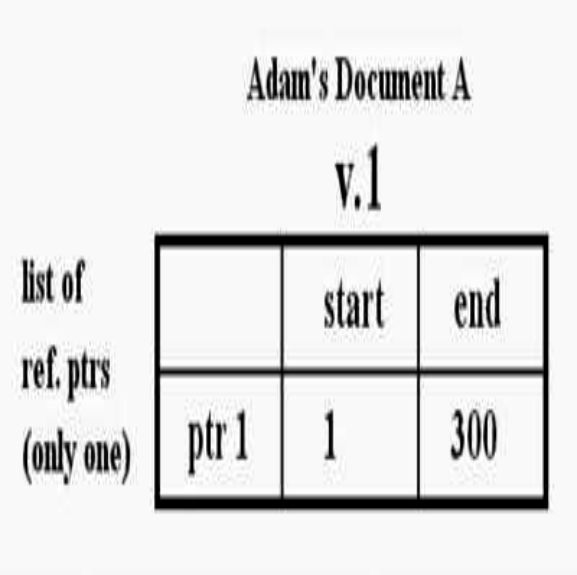

Figure 10. Internal representation of Adam's document:
a list consisting of one reference pointer, since it references a consecutive span of content.

The Content Link

Now along comes a second author Barker, who publishes document B, version 1. The text contents of document B v.1 are registered as 217 consecutive bytes with permanent addresses 301 to 517, also an unbroken stretch of content bytes.

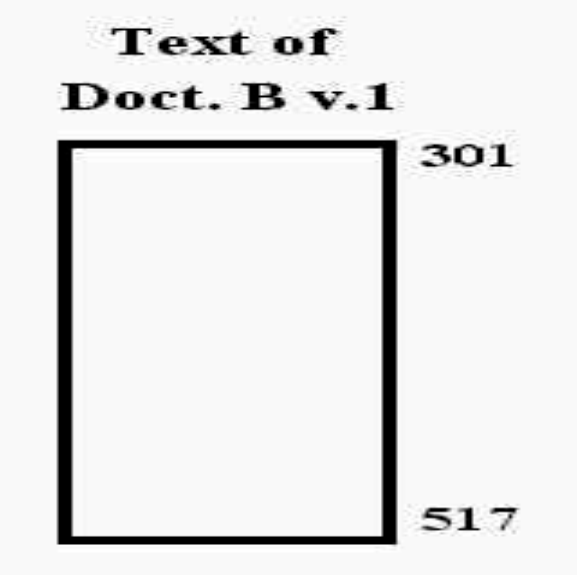

Figure 11. Screen visualization of text portion of document B v.1.
(Addresses shown only for clarification.)

Now for a link.

Suppose that in the text of Barker's document B, he has made a comment on Adam's document A. The text of the comment consists of characters 351 to 400; they comment on characters 101 to 200 in the middle part of A. While Barker's comment already resides in his text in written form, it gives no direct access to the material being commented on. So let us say that as the author, Barker chooses to make this textual relationship explicit, representing it as a comment link between those two spans (presumably using some comfortable visual editor to select the spans and specify the link type).

The link is from an entire span on the first side to an entire span on the second side. (Note that anyone from anywhere can make a content link to any published content.)

One way this might look on the screen: a stripe between both documents (Figure 12).

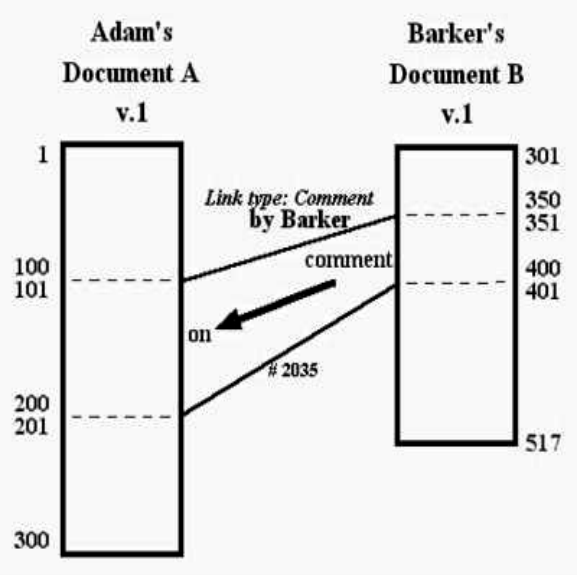

Figure 12. Possible screen visualization in transpointing windows of comment link (Barker's link, element 2053) connecting text from document B v.1 to document A v.1.
(Text addresses and dividers shown only for clarification.)

Content links are first-class, free-standing, and addressable. Even though the comment is a part of Barker's document B, the link representing it is a distinct entity. Let us say it is somehow assigned address 2053 to stress its utter independence of the text of document B.

The internal structure of the content link has three parts (Figure 13): a list specifying the first side, a list specifying the second side, and a type designator. (These are called in xu88 the 1-set, the 2-set, and the 3-set.)

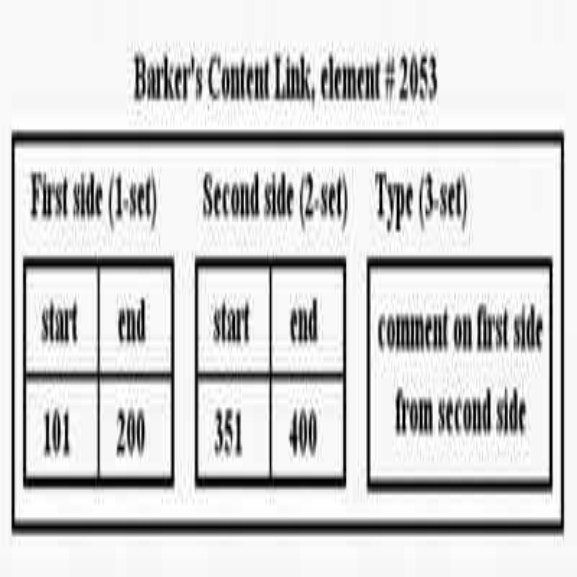

Figure 13. Internal representation of Barker's content link 2053 from document B v.1 to document A v.1. The content link connects all elements on the first side to all elements on the second side.

Making the link is one logical data step, including it in a document (Figure 14) is another (although the visual editor should usually combine these steps from the author's point of view).

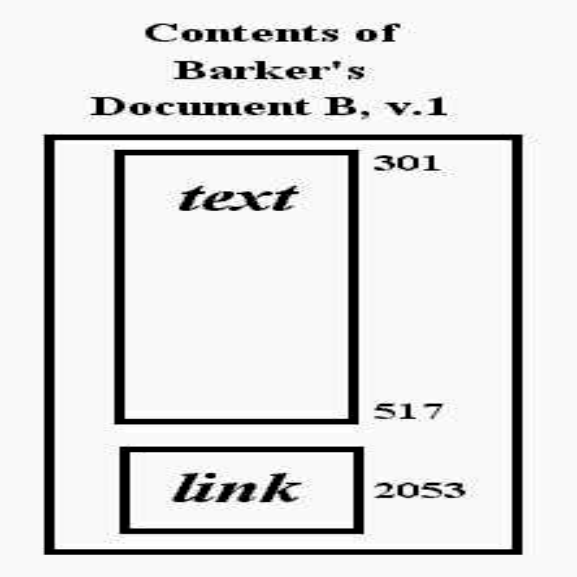

Figure 14. Virtual contents of Barker's document are the text and the link.

The link is put into the document as another pointer in the document list of B (Figure 15). Therefore the document is represented as a list pointing to B's characters *and to the link*, which is an enumerable element like the characters themselves.

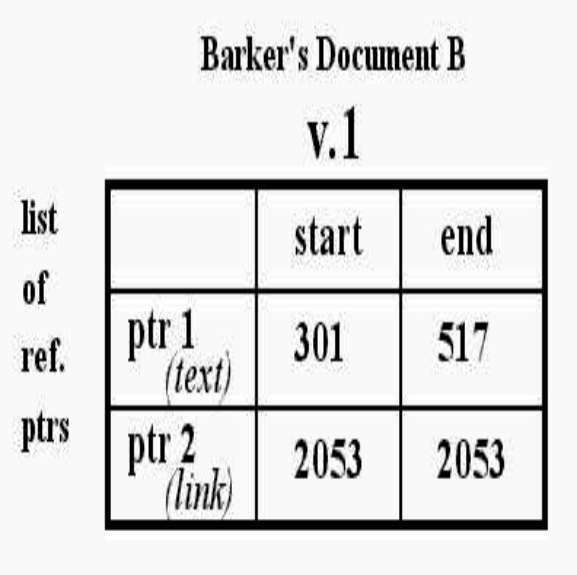

Figure 15. Content list of Barker's document B, version 1.

Although this link is a part of Barker's document (Figure 14), it is also externally addressable. (Thus the link may also be made part of any other document by transclusion -- like any other content, as will be seen.)

How Links Don't Break

Now the fun begins.

Suppose Adam now publishes version 2 of document A. To make version 2, Adam inserts 67 characters, which are assigned addresses 7784 to 7850. This breaks the original reference pointer in two, and splits the content list of document A into three reference pointers.

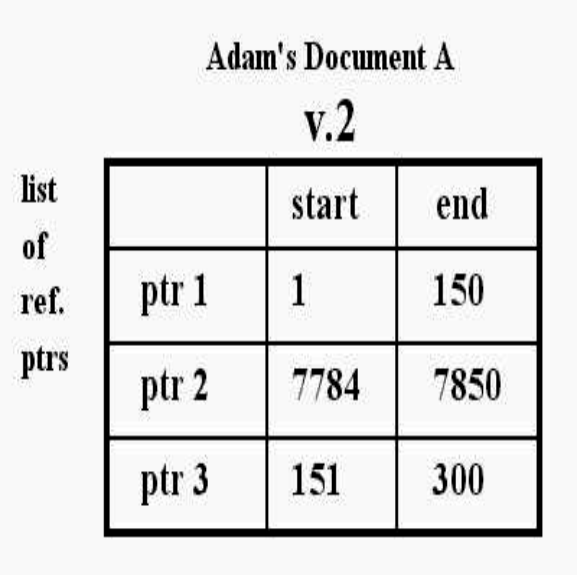

Figure 16. Internal representation of Adam's document A version 2: the first pointer has been split into two, with an insertion between.

Now what happens to Barker's link? The link itself has not changed, but the pattern of its connective positions on the new version has changed, so it will look different. In document A v.2, although an insertion has made into the middle of the previous text, all the characters from document A v.1 which participated in the link -- that is, were commented on -- remain. Thus the system can show the link to all these characters in their new positions. One possible visualization is seen in Figure 17.

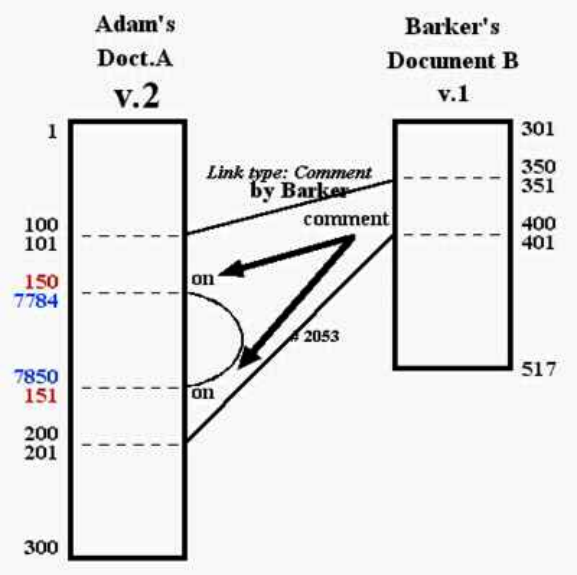

Figure 17. Screen visualization in transpointing windows of link 2053 still adhering to the same characters in document A as rearranged in Adam's version 2.

(Addresses and dividers shown only for clarification.)

Now again Adam edits Document A, creating a shortened version 3 by deleting characters 101 to 166. The resulting document content list is now:

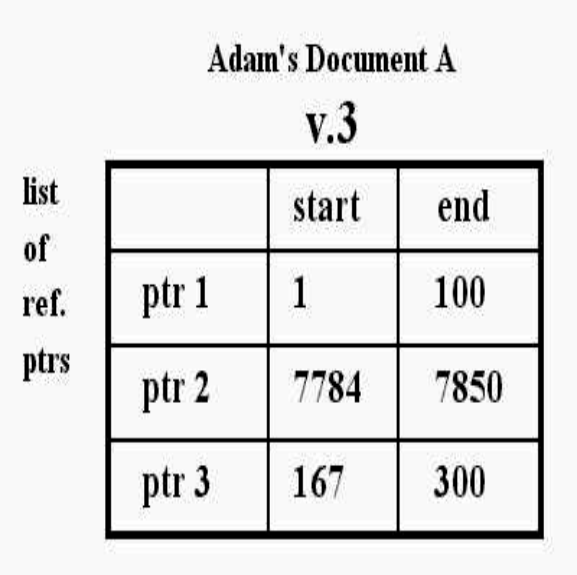

Figure 18. Internal representation of Adam's document A version 3: content has been deleted from version 2.

It happens that some of the characters that Barker originally commented on have been deleted. In this new version, fewer elements in the destination span still exist -- only characters 167 to 200. But link 2053 remains attached to all

of these surviving characters in any new version. One possible visualization is in Figure 20.

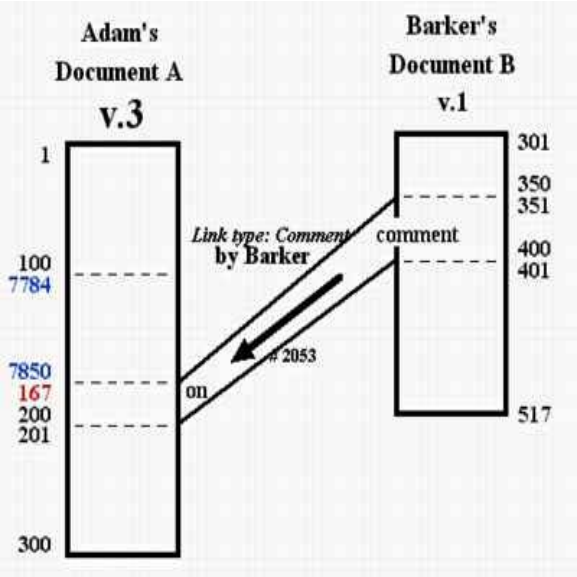

Figure 19. Screen visualization in transpointing windows of link 2053 still adhering to remaining linked characters of Adam's version 3. (Addresses and dividers shown only for clarification.)

Mechanics of Link Display

Let us clarify the mechanics of how the link is shown.

To make the screen presentation of link 2053 between two versions, the system compares content link 2053 against the content lists of whatever specific versions are being shown. The system finds those characters in each document on the screen which participate on either side of the link, and displays them on the screen with connection beams.

The generalized method of showing a content link, given two elements or versions chosen to be seen in transpointing windows is to find all the addresses of elements which are shared between the content link and the specific versions, by finding all the overlapping addresses of elements which are shared by the spans of the versions and the link; and show connection beams as appropriate. (If no shared elements are present on one side, and the user wishes, the user may also go back to some previous version that still had shared elements.)

Powers of the Content Link

It will be evident that this mechanism -- keeping a link attached between occurrences of surviving content through successive versions -- will retain structure that otherwise will be lost. Just as we cannot expect Adams to coordinate with Barker, we cannot expect any rich coordination among the ever-changing electronic documents of the world. So we represent the literary connections that move, or remain.

Obviously there can be different problems with this, depending on the particular revisions of different documents, documents' semantics. Still, this method manages the *presumptive* inheritance of meaning among versions; and the reader's intelligence, plus access to other versions, should take care of the rest.

Example: detailed marginal notes. These tend to lose their attached meaning as soon as new editions of original document come out; trying to maintain detailed marginal notes on the Web is virtually hopeless. But use of the

method for annotation, in future formats, will give some assurance that the annotations will have a much longer life expectancy.

This approach is extremely different from the 1-way links of HTML, each of which can have at most one successful destination (with tangled local and cached exceptions). The HTML link is attached not to a document but to a virtual place -- a file (or marker in a file) in a specific directory on a given server, not a document. The target is deemed to be whatever is at that address, if anything. It is like a shop-window -- we see it however it may be currently decorated, or boarded up.

(How to show many overlapping content links is another issue: color, translucency, reduction to lines are among the possibilities; but these are "interface issues".)

Global Link Following

But how, then, can a content link be followed into the blue, like an HTML link with unknown destination, when it can be satisfied in principle by any document or version having those specific content elements? The answer is for the user to pre-reduce the search -- making some contextual selection, such as what author's work or what collection to look at, whether to ask for the latest version or the original. Then, when the system says there are many possible hits (like a Web search engine), refine again (as with a Web search engine). This may present an interface challenge, but so does every rich capability.

Users are already accustomed to sorting for relevance among millions of Web pages; the prospect of sorting for relevance among millions of links should be no more difficult conceptually. We believe that *recommendations of links*, a new way of effectively creating moderated newsgroups of content, will become an important genre.

Finding Transclusions

Transclusions, or visible re-uses across literary boundaries, may be found straightforwardly with the referential model. We find them by address comparison.

Consider Adam's three versions of document A. What content do these versions have in common? We may find out easily by intercomparing the version lists of document A versions 1-3 (Figures 10, 16, 18), and displaying them with transclusion beams (Figure 20).

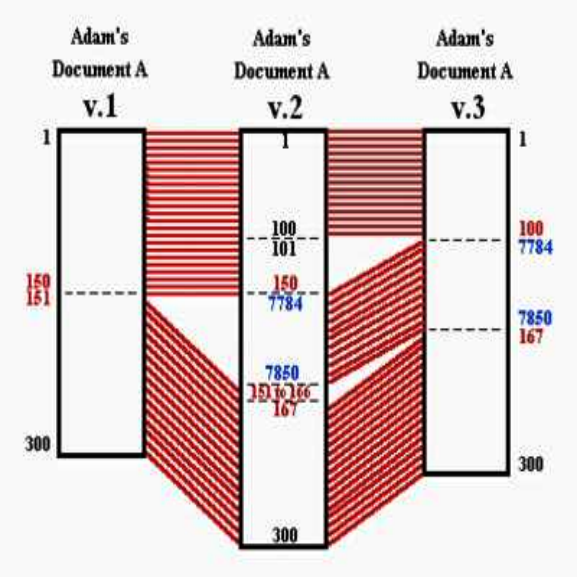

Figure 20. Transclusion between versions as it might be seen in transpointing windows, showing identity of content across literary boundaries. Note that fluid-media transclusions are always the same length.

Any transclusion may also in principle be made "clickable", so that a user viewing a quoted part of some document may pull up the original context in a transpointing window. The system does this by finding the quotation's position in the original version, opening the original version to that position, and displaying it with transclusion beams.

If this seems too abstract, consider a familiar example. In emails, we often quote previous emails -- often marked by angle brackets on the quoted portions -- but the context of each quotation is lost. A transclusive email viewer (assuming supporting mechanisms) would maintain the connection to the original context (Figure 21).

Figure 21. Simulated email viewer showing original context of a transclusion (rather than the usual disconnected quote). Customary inpointing angle brackets on the right indicate the transcluded material; outpointing on the left, they indicate the source.

Other straightforward examples of where readers might like to have the original context available from every excerpt: interviews and reviews. Another example is programming itself: it would make sense to follow the movement of code visually in big programming projects.

On a small scale, being able to step to the origins of things may not seem important; but when thousands of documents inter-reference one another, the confusion -- and need -- increase. As tomorrow's cosmic Niagara of digital media grows (which is guaranteed not to stay in predictable places), deep context-following, regardless of location of original, will become vital.

Managing the Global Space

There have been various proposals within the Xanadu project for ways to allocate and manage a distributed global address space. (The rest of the world is experiencing the difficulties of this problem now. Internet authorities manage two parallel address spaces centrally: the out-of-sight hierarchy of IP addresses, and the more public and emotive domain-naming system. Both of these are distributed systems with centralized root node management, and the resulting political problems are growing into strange forms.) By contrast, the rootless tumbler addressing system of xu88 [Nelson 1987], designed by Gregory and Miller, is particularly interesting for its breadth and parsimony.

Another key question about the universal address space is how to unify it for analysis; especially, within the model presented here, bringing together all the document and link lists to find and compare all the addresses (a serious combinatorial challenge). The Xanadu project pursued two different radical enfilade-based optimization strategies (built into xu88 and xu92). But newer approaches have arisen, such as the search-engine industry, which might perform such services; or the Internet Archive, which could be globally searched for commonalities of both transclusion and content link [Kahle 1999].

Transcopyright, Permission to Transclude Publicly

In the system outlined here, a document is basically a list of contents. Thus inclusion of contents in a document is quite independent of ownership of that document, since anyone can list anything.

Then why not allow content to be transcluded freely, by these virtual listings? And allow delivery to be arranged as a separate step? Suppose we separate the issues of who owns the contents, and who may include them?

This leads to a very simple copyright solution. But first let us consider the problem it may solve.

The Two Parody Views of Copyright

Copyright thinking has been polarizing into two nasty and crazed views: the "we'll steal it all" school, anticipating the destruction of all copyright and copyright law, and the "we'll nail you for it and lock everything down" school, with pay-per-view methods and client display software that clangs shut when the user's money runs out. These two factions -- copyright hawks and anti-copyright hawks -- are currently engaged in legal and illegal maneuvers throughout the world.

Regrettably, many seem to think these polarized, harsh extremes embody the only possible copyright views.

It is of course conceivable that copyright will be overthrown, as many youngsters hope. But my assumption is that the copyright wars will get nastier and nastier, and the polarization is likely to worsen. As bandwidth and storage increase, the kids will steal and swap more, and crackdowns and unpleasantness will become more frequent and more likely.

The flagrant copyright violators are counting on public sentiment to keep them out of jail. But as with marijuana possesion and sale, which huge numbers of people consider no crime, copyright violators may well start getting prison sentences en masse.

What Better Alternative?

I would gladly live in a world without copyright, but I don't think that is going to happen. Therefore the best objective, most beneficial to all parties -- and a Xanadu proposal now for forty years -- is to find some way to make copyright less painful, and to facilitate well-intentioned uses of content.

As a rule, finding a principled basis for such a political solution can be very difficult. But by luck, Xanalogical structure directly provides such a principled basis for a win-win copyright method.

A Copyright Approach for Frictionless Republication

There exists an alternative, legal copyright system, different from both extremes, which is demonstrably fair to all parties. It is an anomalous system with no close relatives. (We have recently adapted to the Internet the original Xanadu copyright model, formerly centralized [Nelson 1994], but now basing it on the same transclusion and payment concept in a new decentralized version.)

Transcopyright [Nelson 1997a], [Nelson 1997b] is a unique alternative copyright system -- an honest system granting users more power, but wholly compatible with existing law and ownership. It takes away nobody's rights. It allows unlimited virtual re-use of contents: all parties may republish, reorganize, anthologize without negotiation, and place excerpted materials into texts, maps, diagrams, time-lines, overlays and so on. And readers may go from every quote to its original context.

Other digital copyright proposals seem principally concerned with sharpening and fortifying a sort of Maginot Line between the two parody views; while transcopyright grants new kinds of re-use to the public -- whose important interest in flexible re-use has not been represented.

The idea is simple. Since the distributed unit is the *content list*, and the delivery of the document is in two phases -- first, the content list; then the content -- why not control the content separately? Why not simply let anyone include any content on their lists, and allow the rightsholder to control the delivery of the content?

We call the legal doctrine *transcopyright*; what it makes possible we call *transpublishing* [HTS 1999], [Nelson 1999a], [Nelson 1999d], [Nelson 1999i], or virtual republication of excerpts without limit from the contents of participating rightsholders.

Transcopyright is a unique bargain, which any rightsholder is free to accept or reject: in return for each excerpt being tethered to its original context (connected transclusively), and therefore no quote being out of context (by inquiry), the rightsholder gives permission in advance for all transclusion in any amount, in published on-line documents, and agrees to furnish any content portions requested. For those crass enough to want to be paid -- and who have some hope of massive downloading of excerpts -- a gateway micropayment may be added.

These two deals represent two levels of compromise and accommodation. The first level of acceptance means you give permission for the re-use in return for the original context being available. The second level means you want the original context available, but you also want to get paid for each downloaded excerpt.

However, the implementation for these two deals is the same at the server level. To get the payment, a gateway micropayment system (such as Hypercoin(tm) [HyperCoin 1999], [Nelson 1999c], and [Nelson 2000a]) is interposed by the rightsholder.

Many have misunderstood this to mean that transpublishing can only work if a live connection to each unique original is in place; this is a misunderstanding. The same content may be made available from many sources in parallel, provided that they work within the system. Rightsholders are free to operate the servers dishing out their own content, or to delegate that job.

A key factor is that payment should be *proportional* to an excerpt, and small. The gateway micropayment client should have a settable threshold that frees the user from seeing payments less than a chosen amount: "If I click and it costs one cent or less, just buy it."

The example that seems to make the transpublishing idea clearest to people is the following: think of all the movie reviews you've seen on TV where you've wanted to see more -- but not necessarily the whole movie. Imagine movies published on line under transcopyright, from which anyone is free to include favorite scenes virtually -- and any viewer of these new compositions is free to click to see more beyond the favorite scene, jumping to its context in the original.

Consider that this arrangement turns all participating material into easily-reusable boilerplate, which you can add to any on-line document without discussion. If the idea of microsale seems far-fetched for text, is it far-fetched for streaming video? Or for high-resolution movies? Or the use of unedited footage which is posted for this purpose? What about diagrams? What about historical footage? Surely there will be a niche for this method.

Permission Formats

We have developed specific formats for transcopyright permission [KeioPermission 1999] that lawyers consider to be universally legal (since such permission is a license and does not affect existing copyright law). Refining these permission licenses for simplicity and ease of understanding is a continuing endeavor.

Independence of Xanalogical Structure

The transcopyright method can be used without the linkage part of xanalogical structure, as long as some transclusive delivery method is employed. Indeed, transclusion formats are possible for this using nearly-ordinary HTML [Nelson 1999j]; unfortunately, these formats have had to be tweaked for variations between Netscape and Explorer, and the results look somewhat different in each). At Keio University we have produced an experimental server and editor for transpublishing with these formats [HTS 1999], under a grant from the Japanese government.

Caching, Speed and Reliability

It is true that transpublishing requires some kind of network; but it not necessarily perfect connectivity; it should be possible for transpublishing to work efficiently on imperfect networks through caching systems designed for the purpose.

- "But that means if the network goes down I can't get it" -- true, but that's the same as for any other downloads.
- "It's too complicated." As compared to what other system that allows you to include copyrighted material?
- "It will all get stolen." Right. Unemployed napsters and napstresses are going to cache millions of hours of video. And then what will they do with it?

ACM's Indirect Endorsement of Transcopyright

The transcopyright system has been endorsed in principle by the Association for Computing Machinery in its current "Interim Copyright Policy" by Peter J. Denning, presently at the ACM site [Denning 1998]*. To quote:

> "Transcopyright permission for electronic dissemination. ACM incorporates a principle similar to one Ted Nelson called "transcopyright". ACM will hold its copyrighted works on its servers and will give free and unlimited permission to create and copy links to those works or their components. So that readers can locate the context from which an excerpt was drawn, ACM will provide a way of linking a component to its parent work."

Laudably, this is intended to maintain connection to the original context, as few Web links do. Unfortunately, the ACM statement is silent on the central tenet of transcopyright, which is the permission freely to recomposite and republish virtually by some transclusion mechanism. Still, the ACM's approval-in-principle is very heartening.

Summary of Transcopyright

Transcopyright is a copyright system which is open, liberal, benign, fair, frictionless, honest, cheap, and win-win. It is intended to facilitate ease of access to partial documents; to facilitate ease of recomposition and republishing as if contents were in the public domain; and to do what paper cannot: provide access to the original context of any excerpt.

It was always a premise of the Xanadu project that in the future people are going to read and view fewer and fewer finished works as the avalanche of media increases; and so paying for only what you want (and getting to keep it), but knowing you can get the rest, is what we really need.

Conclusion

This has been a brief statement of a deeply-held philosophical position and a few of its corresponding methods. We have presented a referential system of data representation that allows overlaying applicative structure; and which implements in particular two forms of visible cross-connection between parallel documents and versions: re-use (trackable automatically) and user-created links.

This view and its techniques have many other tenets and ramifications that we have no room to discuss here. I have stayed as far away as possible from particulars, except to be clear in the hypothetical examples, which suffice only to explain the basic feasibility of our referential structures and methods.

Referential structure of media has great power. Besides clean support for unbreaking links and transclusions, separating structure from content has many other benefits. Data may be used in place. All content is additive. All structure is additive and applicative, rather than tangled inside (as in the SGML model). This permits many structural variations on the same particular documents and their contents -- variations whose cross-connections may in turn be viewed.

If the list of content is made the fundamental unit, many things become possible and principled: nondestructive, additive editing; branching versions, all accessible and re-branchable; profuse unbreaking links; principled and visible re-use (transclusion); deep intercomparison along both links and transclusions; and transpublishing under transcopyright.

This proposes a vision of a very different world of media: a network literature of a totally different form and nature from anything that can now be seen. It will allow completely new ways of organizing material--

- where a fundamental building block is the excerpt directly connected to its origin;
- which does not require closed packaging of content in segregated files or directories;
- where you may recomposite anything, and may build layered and overlapping structures where needed;
- which allows detailed annotation and detailed intercomparison -- showing and following commonalities, showing and following content links; and
- which allows open recompositing and open republication, based on the right of open referential re-use,
- with copyright handled automatically.

I believe this all fits together cleanly. I think the simple technical examples given above will make clear that this is an entirely feasible approach -- and, once you understand it, obvious. And I think it is altogether possible that many people will want to visit and work in, and help build, such a subuniverse.

There is no question that these visualizations and methods are a long way from the prevailing protocols, browsers, and "standards". But that would not be the important question anyway. (Standardization wars will continue forever.) The important question is whether these ideas can enable a principled and feasible alternative universe, beneficial even if implemented only in part, a potential zone of clarity in the chaos of the Web, desirable enough to be worth the uphill effort.

To underpin this, we will need some alternative infrastructure -- protocols, servers, permanent publishing methods, compliant browser plug-ins and editors, transpublishing caches, and a very different payment method from those currently popular.

Is it too late? Would it mean just too much overhead? We believe that like the hidden overhead of the Internet itself, this will pay for itself manyfold.

We can see no other important agenda for hypertext.

Footnotes

* Principal Designers

Principal designers involved in various versions of the Xanadu designs have included (in alphabetical order)--

Paul Baclace, William Barus, Yoshihide Chubachi, Cal Daniels, K. Eric Drexler, Stuart Greene, Roger Gregory, Edward Harter, Chris Hibbert, Eric Hill, Hugh Hoover, Rob Jellinghaus, Roland King, Tuomas J. Lukka, Marlene Mallicoat, Michael McClary, Mark S. Miller, Theodor Holm Nelson, Kiyoki Ookubo, Andrew Pam, Ravi Pandya, Bill Richard, Jonathan V.E. Ridgway, Jonathan Shapiro, Marc Stiegler, Johan Strandberg, Dean Tribble, Ken'ichi Unnai, and Steve

---

** Principal Designs

The principal designs that I know of are the following. (However, other members of the project may remember other significant designs.) For the sake of discussion here, I have given coordinated nicknames to these principal designs. Unfortunately documentation is spotty and being attempted retroactively. At least ten of these designs (marked with asterisks below) have been implemented to some state of demonstration or delivery. Dates selected are those of significant closure.

xu60

-- 1960 preliminary design (currently undocumented)

**** xu65**

--1965 zipper lists [Nelson 1965]; implemented 1996 by Kiyoki Ookubo and others, using Yuzuru Tanaka's IntelligentPad system [IntelligentPad 1995]

xu66

-- 1966 SNP design (Sexus-Nexus-Plexus) separated data into content, referential lists and links (currently undocumented)

xu70

-- 1970 parallel markup, ring buffers [Xanadu 1970]

**** xu72**

-- 1972 first enfilade (model T), enfiladic referential editing using word-processor interface [Xanadu 1972]

xu76

-- 1973-6 designs by William Barus (currently undocumented)

****

-- core design of "ent" versioning enfilade system by K.Eric Drexler [Nelson 1999]

**** zz86**

-- multidimensional list system with cursor-centric views; now called ZigZag(tm) [ZigZag 1999], [Nelson 1998c]; prototype implemented 1997 by Andrew Pam and variously downloadable [ZigZag 1999a], instructions at [ZigZag 1999b]; extensible with explorable genealogy demo and genealogy demo instructions [Geneology 2000a]; Java version now being developed under open source ("Gzigzag") at Sourceforge.net under stewardship of Tuomas J. Lukka (recent version downloadable [Nelson 1999], instructions are in [Nelson 2000]

**** xu88**

(formerly called Xanadu 88.1 and now called "Udanax Green") -- distributed client-server system programmed in C, principally designed and implemented by Roger Gregory and Mark S. Miller, partly based on the general enfilade theory of Mark S. Miller, Stuart Greene and Roger Gregory; for fluid media registry, version and link editing, version delivery, link following and transclusion following across a distributed network [Nelson 1980], [Nelson 1987]; now released under open source at udanax.com), with extensive protocol documentation [FEBE 1988], [FEBE 1999]; previously-secret internals are now being published [Nelson 1999]; remaining work to be done [Udanax 1999]

**** xu92**

(formerly called Xanadu 92.1) -- 1988-92 design [Nelson 1990]; by Miller, Tribble and Pandya based on K.Eric Drexler's "ent" data system invention [Nelson 1999] and programmed in a small common subset of Smalltalk and C; now released as Udanax Gold under open source at udanax.com

**** hcoin96**

n HyperCoin(tm) gateway micropayment system for Internet microsale of transpublished content portions [HyperCoin 1999], [Nelson 1999c]; implemented as working prototype 1996 by Andrew Pam; patented 2000 [Nelson 2000a]

**** osmic97**

-- demonstration of branching referential versioning [Nelson 1999b], [Nelson 1996]; designed 1996 [Nelson 1999e], [Nelson 1999f], implemented 1997 by Ken'ichi Unnai in Perl (server) and elisp (client), downloadable [Nelson 1999g], [Nelson 1999h]; implemented again 1999 again under the direction of Yoshihide Chubachi (OSMIC renamed INLUV)

**** tpw98**

-- downloadable and self-installing functioning articulated demonstration of transpointing windows [Transpointing 2000], to run under Microsoft Windows; implemented by Ian Heath, based on U.Southampton Microcosm engine

**** hts99**

-- transpublishing server prototype [HTS 1999] using transclusion formats for ordinary browsers [HTS 1999], running from document database; implemented 1999 under direction of Yoshihide Chubachi; design [Nelson 1999j]; current implementation front page [HTS 1999]

xu99

-- Floating World design [Nelson 2000c], large preliminary design document [FloatingWorld 1999] (then called ZX)): structures for interactive multidimensional graphics with discrete ZigZag backbone and xu88 fluid-media model

xuArc

-- Possible functions to be performed across the Internet archive [Kahle 1999]

**Unusual Design Ideas

Unusual design ideas have included (keyed to the above list**, with best recollection as to dates), and no attempt to sort out credit:

- hypertext xu60 (term introduced 1965)
- hypertext linksxu60
- referential editingxu60
- transclusionxu60; item-level transclusion with parallel display, xu65; character-level transclusion in a universal address space, xu88
- branching versions1960
- copyright solution by referential transclusionxu60
- transpointing windows1965 (term introduced 1994)
- parallel markupxu70
- ring buffers for text, parallel ring buffers for parallel markup and editing, multidimensional ring buffers, xu70

- threading a display processor through the main data structure (rather than generating a separate display list), xu70 and xu99
- enfiladesthe model T enfilade (a data management tree for indexing linear text and editing it by reference), xu72; tightly coupled enfilades, xu76 and xu88; two-dimensional enfilades, xu88
- general enfilade theoryca. 1980 (generating data management trees with an upwardly propagating search property and simultaneously a downwardly imposable structural property)
- fluid mediawith stabilized universal addresses, xu88 (term "fluid media" introduced 1999)
- generalized xanalogical protocol based on span lists, xu88
- tumbler addressing (multi-segment universal addresses), tumbler span arithmetic based on transfinite arithmetic
- version management by tumbler span permutation matrices, xu88
- search of entire docuverse for overlapping spans by 2D enfilade, xu88
- the ent (a singular enfiladic structure with built-in versioning), xu92
- multidimensional list structures with cursor-centric viewing in rows and columns zz86 (ZigZag(tm) structure), zz86
- transcopyright doctrine separate from software, transco95
- transpublishing within conventional HTML, 1996
- transclusion tag and protocol specifications to work in ordinary Web browsers from special transpublishing servers, hts99
- 3 1/2D graphicsxu99
- multidimensional computer structures and graphics with a ZigZag spine xu99
- ND mapped to 3 1/2D for high-performance use of 3D graphics boards, with OpenGL threading through 3D views of the ND structure, xu99
- profuse interpenetrating structures as an alternative to "applications", xu99
- high-performance 3 1/2D as a potential alternative to the "desktop", xu99

Bibliography

[Bush 1945] Vannevar Bush. "As We May Think" in The Atlantic Monthly, 176(1),101-108, July 1945. {Online: http://www.theatlantic.com/unbound/flashbks/computer/bushf.htm}

[Caerdroia 1999] Caerdroia, the Journal of Mazes and Labyrinths, published by Jeff Saward, caerdroia@dial.pipex.com; produced by Labyrinthos, the labyrinth research center, photo library and archive, 53 Thundersley Grove, Thundersley, Benfleet, Essex, SS7 3EB, England, 1999. {Online: http://www.labyrinthos.net}

[Denning 1998] Peter J. Denning. "ACM INTERIM COPYRIGHT POLICY" Version 3, 12/18/98, 1998. {Online: http://www.acm.org/pubs/copyright_policy/}

[FEBE 1988] FEBE. Cover page of xu88 FEBE Protocol Manual when the software was still called "Xanadu 88.1". The 'confidential' notice is now inoperative, 1988. {Online: http://www.xanadu.com/XOCDOC-cover.tif}

[FEBE 1999] xu88 FEBE Protocol Manual chapters: Front End-Back End (now called in the industry "server-client) interface protocol of xu88 design (now Udanax Green), 1999. {Online: http://www.udanax.com/green/febe/}

- Original cover page when it was still called Xanadu 88.1 [FEBE 1988]
- Front page, www.udanax.com/green/febe/
- Introduction, www.udanax.com/green/febe/intro.html
- Philosophy and Motivation, www.udanax.com/green/febe/philosophy.html
- Implementation, www.udanax.com/green/febe/implementation.html
- Designing Frontends, www.udanax.com/green/febe/frontends.html
- FeBe Example, www.udanax.com/green/febe/example.html
- Technical Overview, www.udanax.com/green/febe/overview.html
- Addressing, www.udanax.com/green/febe/addressing.html
- Tumbler Arithmetic, www.udanax.com/green/febe/tumblers.html
- Links and Link Types, www.udanax.com/green/febe/links.html
- Versions, www.udanax.com/green/febe/versions.html
- Fe vs. Be, www.udanax.com/green/febe/fe-vs-be.html
- FeBe Protocol, www.udanax.com/green/febe/protocol.html
- Bibliography, www.udanax.com/green/febe/bibliography.html
- Appendix A - FeBe Protocol Syntax, www.udanax.com/green/febe/syntax.html
- Appendix B - Useful Tables, www.udanax.com/green/febe/tables.html
- Appendix C - Manual Pages, www.udanax.com/green/febe/man-pages.html
- Appendix D - Glossary, www.udanax.com/green/febe/glossary.html

[FloatingWorld 1999] Floating World specs of November 1999 (then called ZX, 1999. {Online: http://www.xanadu.com/FW99/}

[Geneology 2000] Genealogy demo download, 2000. {Online: http://www.xanadu.com/zigzag/FAMDEMO}

[Geneology 2000a] Genealogy demo instructions, 2000. {Online: http://www.xanadu.com/HolmFamilyDemo.html}

[Gzigzag 1999] Downloadable of Gzigzag from, 1999. {Online: http://xanadu.com/gzigzag-tarball.zip}

[HTS 1999] "HTS" page, 1999. {Online: http://green.iprs.sfc.keio.ac.jp/~hts/}

[HyperCoin 1999] "HyperCoin Executive Summary", 1999. {Online: http://www.xanadu.com/hcoin/HcoinSum9906.html}

[IntelligentPad 1995] "IntelligentPad: Next Generation Visual Programming Paradigm", 1995. {Online: http://www.fujitsu.co.jp/hypertext/InfoPro/opsp/intelpad.html}

[Jefferson 1776] Thomas Jefferson. "The Unanimous Declaration of the Thirteen United States of America" Philadelphia, 1776. Jefferson's earlier draft, 1776. {Online: http://odur.let.rug.nl/~usa/D/1776-1800/independence/doi.htm}

[Kahle 1999] Brewster Kahle, personal communication.

[KeioPermission 1999] Specifics of current Keio permission formats may be found at "Transquotation Demo", 1999. {Online: http://www.sfc.keio.ac.jp/~ted/TPUB/TQdemo.html}

[Miller 1999] Mark S. Miller. "Xanadu Secrets Become Udanax Open-Source", 1999. {Online: http://www.udanax.com/}

[Nelson 1965] Theodor H. Nelson. "A File Structure for the Complex, the Changing and the indeterminate" in Proceedings of the ACM National Conference, 1965.

[Nelson 1972] Theodor H. Nelson. "As We Will Think" in Proceedings of Online 72 Conference, Brunel University, Uxbridge, England, 1972.

[Nelson 1980] Theodor H. Nelson. "Replacing the Printed Word" in S.H. Lavington, editor, Information Processing 80 (Proceedings of IFIP 80 World Computer Conference), North-Holland Publishing Co., 1013-1023, 1980.

[Nelson 1981] Theodor H. Nelson. Literary Machines, early editions (1981 to 1986). Published by the author, 1981.

[Nelson 1986] Theodor H. Nelson. "The Tyranny of the File" Datamation, 15 December 1986.

[Nelson 1987] Theodor H. Nelson. Literary Machines, recent editions (1987-current). This contains numerous technical details about the xu88 functional concepts and addressing system. Mindful Press. (Translated also into Japanese and Italian.), 1987.

[Nelson 1990] Theodor H. Nelson. "Virtual World Without End" in Proceedings of Cyber Arts International Conference, September 1990.

[Nelson 1994] Theodor H. Nelson. "Xanadu: Document Interconnection Enabling Re-Use with Automatic Author Credit and Royalty Accounting" in Information Services and Use 14:4, 255-265, 1994.

[Nelson 1995] Theodor H. Nelson. "The Heart of Connection: Hypermedia Unified by Transclusion" in Communications of the ACM, 38:8, 31-33, August 1995.

[Nelson 1996] Theodor H. Nelson. "OSMIC: Open Standard for Media InterConnection", 1996 {Online: http://www.sfc.keio.ac.jp/~ted/OSMIC/OSMICd1m.html}. The original OSMIC defining document.

[Nelson 1997] Theodor H. Nelson. "Literature to Last: Design for a Universal Digital Medium" in Ute Hagel (editor), Labile Ordnungen: Netze Denken, Kunst Verkehren, Verbindlichkeiten. Hans-Bredow Institut, Hamburg, 98-102, ISBN 3-87296-0849, 1997.

[Nelson 1997a] Theodor H. Nelson. "Transcopyright: Dealing with the Dilemma of Digital Copyright" in Educom Review 32(1), 32-5, January/February 1997. {Online: http://www.educause.edu/pub/er/review/reviewArticles/32132.html}

[Nelson 1997b] Theodor H. Nelson. "Transcopyright: A simple legal arrangement for sharing, re-use and republication of copyrighted material on the Net" in Takashi Masuda, Yoshifumi Masunaga and Michiharu Tsukamoto (Eds.), Worldwide Computing and Its Applications. Springer-Verlag, Berlin, 7-14, ISBN 3-540-63343-X, 1997. Original paper. "Transcopyright: Pre-Permission for Virtual Republishing", {Online: http:///www.sfc.keio.ac.jp/~ted/transcopyright/transcopy.html}

[Nelson 1997c] Theodor H. Nelson. "Embedded Markup Considered Harmful" in XML: Principles, Tools, and Techniques, World Wide Web Journal 2(4), Fall 1997. Table of contents, {Online: http://www.w3j.com/xml/}

[Nelson 1998] Theodor H. Nelson. "Examples of Parallel Documents", 1998. {Online: http://www.sfc.keio.ac.jp/~ted/TN/PARALUNE/parexamples.html}

[Nelson 1998a] Theodor H. Nelson. "My Parallel Universe", 1998. {Online: http:// http://www.sfc.keio.ac.jp/~ted/TN/PARALUNE/paraluniverse.html}

[Nelson 1998b] Theodor H. Nelson. "Documents are Parallel by Nature", 1998. {Online: http://www.sfc.keio.ac.jp/~ted/TN/PARALUNE/paradoxx.html}

[Nelson 1998c] Theodor H. Nelson. "What's On My Mind", 1998. {Online: http://www.sfc.keio.ac.jp/~ted/zigzag/xybrap.html}

[Nelson 1999] Theodor H. Nelson. "Xanadu Technologies -- An Introduction", 1999. {Online: http://www.xanadu.net/TECH/xuTech.html}

[Nelson 1999a] Theodor H. Nelson. "Transpublishing, a Simple Concept", 1999. {Online: http://www.sfc.keio.ac.jp/~ted/TPUB/TPUBsum.html}

[Nelson 1999b] Theodor H. Nelson. "Virtual Editing and Microversions: OSMIC", 1999. {Online: http://www.sfc.keio.ac.jp/~ted/OSMIC/OSMICpage.html}

[Nelson 1999c] Theodor H. Nelson. "Transpayment Methods", 1999. {Online: http://www.sfc.keio.ac.jp/~ted/TPUB/TransPayMethods.html}

[Nelson 1999d] Transpublishing "poster" page, 1999. {Online: http://www.sfc.keio.ac.jp/~ted/TPUB/TranspubPoster.html}

[Nelson 1999e] Theodor H. Nelson. "Models of Time, Backtrack and Groupware", 1999. {Online: http://www.sfc.keio.ac.jp/~ted/OSMIC/osmicTime.html}

[Nelson 1999f] Theodor H. Nelson. "The Two Spools of OSMIC", 1999. {Online: http://www.sfc.keio.ac.jp/~ted/OSMIC/osmicTwoSpools.html}

[Nelson 1999g] Theodor H. Nelson. "Installing the OSMIC Prototype", 1999. {Online: http://www.sfc.keio.ac.jp/~ted/OSMIC/osmicInstall.html}

[Nelson 1999h] Theodor H. Nelson. "Using the OSMIC Prototype", 1999. {Online: http://www.sfc.keio.ac.jp/~ted/OSMIC/osmicUse.html}

[Nelson 1999i] Theodor H. Nelson. "Transpublishing: A Simple Concept", 1999. {Online: http://www.sfc.keio.ac.jp/~ted/TPUB/TPUBsum.html}

[Nelson 1999j] Theodor H. Nelson. "Transpublishing for Today's Web: Our Overall Design and Why It Is Simple", 1999. {Online: http://www.sfc.keio.ac.jp/~ted/TPUB/TQdesign99.html}

[Nelson 2000] Theodor H. Nelson. "ZigZag Tech Notes", 2000. {Online: http://www.xanadu.com/zigzag/zzTech.html}

[Nelson 2000a] Theodor H. Nelson, U.S. patent 6,058,381: "Many-to-many payments system for network content materials", 2000.

[Nelson 2000b] Theodor H. Nelson. "Clearing the Boulders from the Catacombs: Principled Alternatives to the Tradition of Named Stuck Files in Hierarchical Directories", 2000. {Online: http://www.xanadu.com/TYRF/BoulCat.html}

[Nelson 2000c] Theodor H. Nelson. "Floating World: a New User Environment", 2000. {Online: http://www.xanadu.com/FW/fwIntro.html}

[Transpointing 2000] Download page for transpointing windows, 2000. {Online: http://www.xanadu.com/TPWdemo/}

[Udanax 1999] "Current Status of the Udanax Green Code", 1999. {Online: http://www.udanax.com/green/status.html}

[Xanadu 1970] 1970 Xanadu design document, 1970. {Online: http://www.xanadu.com/xu70des.tif}

[Xanadu 1972] 1972 Xanadu design document, 1972. {Online: http://www.xanadu.com/xu72des.tif}

[Xanadu 1997] The clearest and most accurate summary of Project Xanadu's work and views, which I do not believe was written by anyone associated with the project, is an anonymously-written Web page published by Ziff-Davis. "Ted Nelson, Hypertext Pioneer", 1997. {Online: http://www.zdnet.com/zdtv/screensavers_story/0,3656,2127396-2102293,00.html}

ZigZag 1999] ZigZag page at Xanadu.com, 1999. {Online: http://www.xanadu.com/zigzag/}

[ZigZag 1999a] The ZigZag prototype, in more than one version, may be downloaded from the page "Xanadu ZigZag Hyperstructure Kit", 1999. {Online: http://www.xanadu.net/zigzag/zigzag.html}. Especially recommended is the boot floppy downloadable file, with comes with installation instructions but not directions for use, {Online: http://www.xanadu.com.au/zigzag/zzdemo.zip}

[ZigZag 1999b] Directions for use of the ZigZag prototype are to be found, 1999. {Online: http://www.xanadu.net/zigzag/zzDirexCondensed.html}