Intermedia: Issues, Strategies, and Tactics in the Design of a Hypermedia Document System

L. Nancy Garrett, Karen E. Smith, and Norman Meyrowitz

Institute for Research in Information and Scholarship (IRIS)
Brown University
Box 1946
Providence, RI 02912

ABSTRACT

A hypermedia system provides a tool for cooperative work by allowing writers and designers to share a network of linked documents where they can create documents, link their own and others' documents together, and leave notes for one another. This paper discusses issues that designers need to address in the development of hypermedia systems. Major issues involve what kind of linking, contexts, and visual modeling the system provides. The composite of the answers to these issues determines the nature of the hypermedia system and how useful it is to those using it as a tool for cooperative work. The following presents a variety of solutions including those that developers at the Institute for Research in Information and Scholarship implemented in the creation of Intermedia.

1. INTRODUCTION

Annotating existing work and forging links between writings has long been an important undertaking of knowledge workers. Today, computer-based *hypermedia* systems model former handwritten margin notes, annotations in illuminated manuscripts, and conventional "see also" references in encylopedias [Enge68, Meyr85, Nels81, and Yank85]. Such systems provide an electronic equivalent of the cooperative processes of knowledge workers by allowing authors to create, annotate, and link together information from a variety of media such as text, graphics, timelines, video, audio, and spreadsheets.

Hypermedia systems differ from the traditional methods in several important ways. Where the traditional methods imply sequential access, hypermedia systems aim to allow multiple authors to add commentary to the same corpus at the same time. Where the traditional methods are limited to paper media, hypermedia systems are free to exploit the worlds of electronic information access. Where the traditional methods are associated with static textual documents and manually-created indices, hypermedia systems are able to present visualizations of a complex, changing, dynamic world.

The design and implementation of a system with such potential power, flexibility, and wide-ranging audience has many degrees of freedom. For a given audience, what is the best way to organize links? What is the best way to symbolize links? What is the best way to present visual renderings of the entire network of links?

This article catalogs many issues associated with building a hypermedia system. It draws on our experience in designing and building *Intermedia*, a framework and collection of tools that allows authors to link together the contents of text, timeline, and graphics documents over a network of high-powered workstations [Meyr85, Meyr86, Garr86, Haan86, Yank86]. First we describe Intermedia. Next, to present the basic problem domain, we discuss issues involved in the most elementary hypermedia system, one in which a single user is able to link together read-only documents. We then explore many of the same issues as they are complicated by the ability to edit documents. Finally, we examine these issues in a multiple user environment.

2. DESCRIPTION OF INTERMEDIA

Using Intermedia, readers can browse through linked information in a non-sequential but orderly manner. Individual applications in the Intermedia system include InterText, a text processor; InterDraw, a graphics editor; InterVal, a timeline editor that allows users to interactively organize information in date and time sequences; InterPix, a scanned-image viewer; and InterSpect, an application to view three-dimensional models. In the future, a music editor, a video clip editor and access to CD-ROM data will be added to the system.

Cooperative work is encouraged by the ease with which users can link information and navigate established links. To link information, the user indicates a source selection in a document, chooses the *Start Link* command, indicates a destination selection, and chooses the *Complete Relation* command. When the operation is finished there is a semantic tie – a navigational link – between the source and the destination. Whenever the user selects a link marker and issues the *Follow* command, the document containing the other end of the link is activated and the appropriate

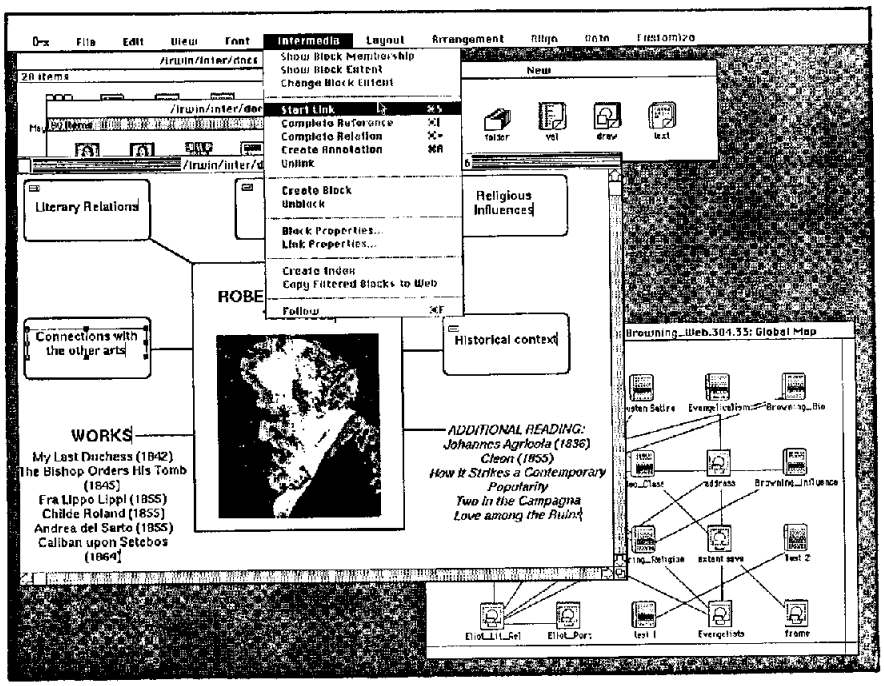

Figure 1a. *The user has selected "connections..." in the top window and is about to choose* ***Start Link*** *from the Intermedia menu.*

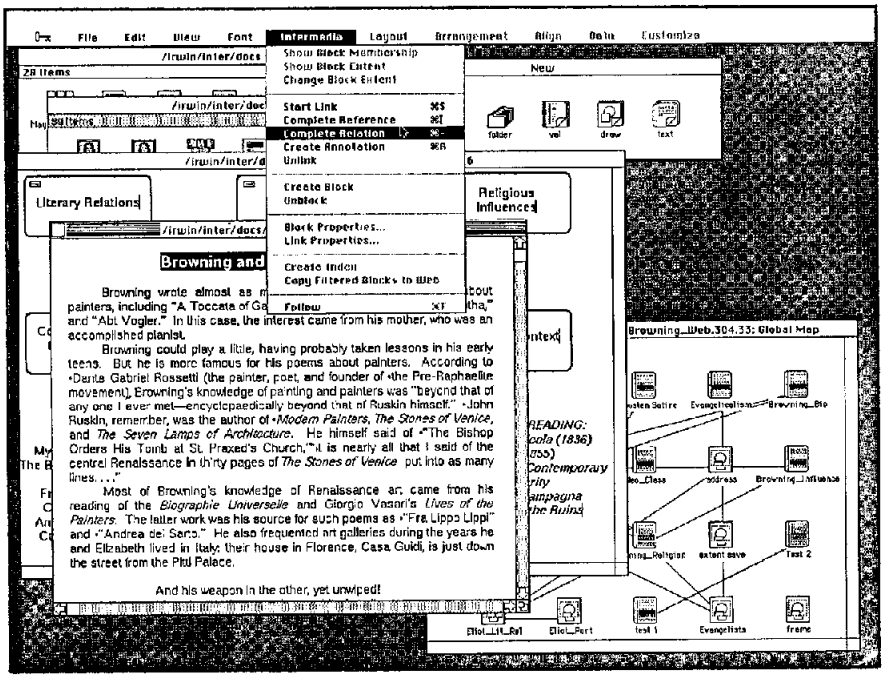

Figure 1b. *The user has opened another document, selected its title, and is about to choose* ***Complete Relation,*** *thereby completing the link.*

information is presented in another window on the screen [Figure 1]. *Webs* present users with contexts in which to collect and navigate a set of links. An individual user, a group of users, or even an entire campus can conceivably create a shared web of information. Visual representation of webs, via *maps*, is currently provided with enhancements to their functionality under implementation.

Using Intermedia on a network of workstations with shared file systems, students can complete assignments that require linking together a variety of references. For example, a student preparing a report on the Venetian Cinquecento in an Art History course can include links to a chronological view of the political and cultural events of the period, biographies of noted artists, diagran illustrating compositional theories, and scanne reproductions of relevant works. The instructor ar possibly other students can read the report, examine t reference material by following links, and leave commen criticism, and suggestions for revision through annotati links. While revising the document, the student can s all of the instructor's comments and examine the sourc containing counter arguments.

3. A SYSTEM OF READ-ONLY DOCUMENTS

The simplest hypermedia system is one in which a sing user on a standalone workstation examines read-or

documents and creates and follows links among these documents. This section explores some of the concerns that appear in designing such an environment, including the issues of providing appropriate 1) anchors for individual links, 2) contexts for webs of links, and 3) visual representations of the linked corpuses.

3.1 Linking

3.1.1 The Link Anchors

Anything from entire documents to insertion points could serve as anchors for links, depending on the purpose of the system under development. Between these extremes lie other possibilities: a link might be anchored to any number of user-defined text segments or graphics objects or to only a single word or graphics object.

The designers may make the decision between these alternatives by balancing implementation and functional concerns. Limiting the user to link only to entire documents is easier to implement than the other options, but does not allow the user to reference specific contents. Insertion points are an improvement over linking to entire documents with manageable increase in implementation complexity. With insertion points, however, references may be ambiguous. Some authors might place the link at the beginning of a reference to signal that what is coming is important while others might place the link at the end as they are used to doing with

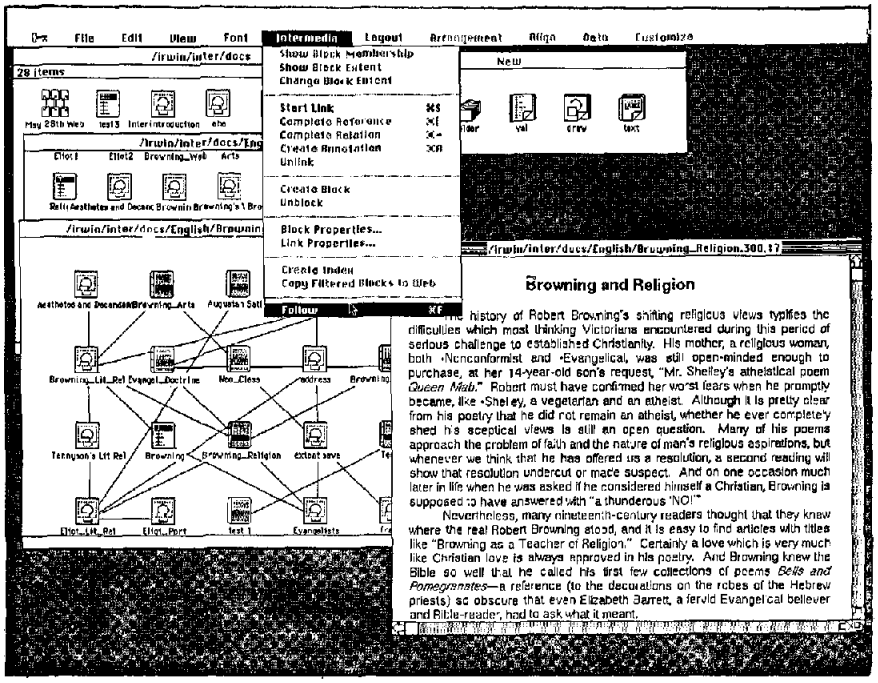

Figure 1c. *Once the link is established, Intermedia displays a marker icon at each end. Here the user has selected a marker above the title and is about to choose* **Follow**.

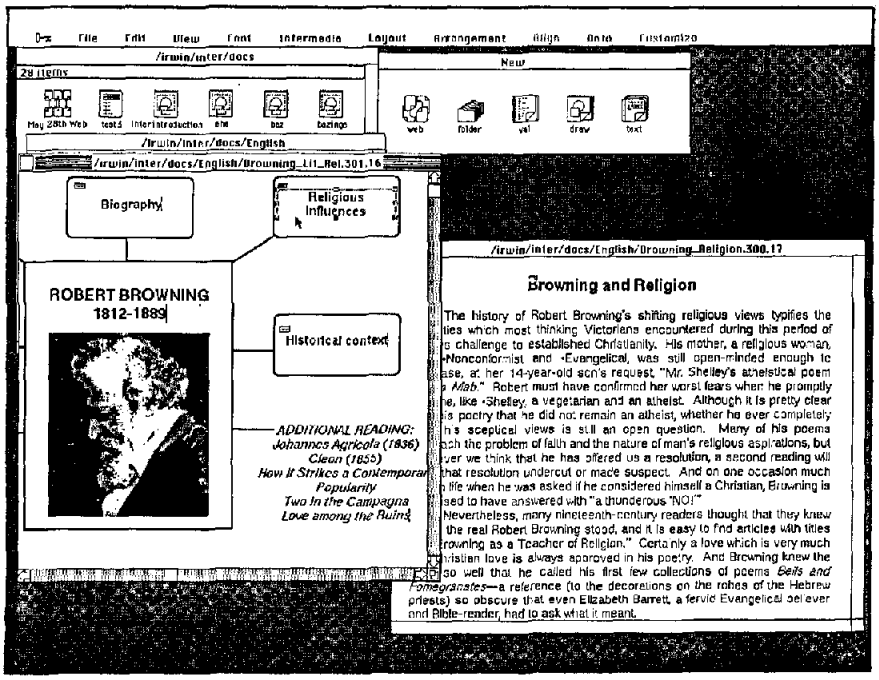

Figure 1d. *The document at the other end of the link opens with the other link marker selected and the destination, "Religious Influences", highlighted.*

conventional footnotes. If text segments or graphics objects could be the link anchor, the author would be able to indicate the reference clearly.

For some purposes yet another design might be appropriate. This might be the case if the system is for indexing or for organizing encyclopedia information. Such a system might provide automatic linking so that each link would be characterized by having a single keyword on one end and an entire document on the other.

As designed, an Intermedia anchor may typically be any legitimate selection in that application, such as a user-defined sequence of characters or a combination of graphics objects. With this design a textual anchor could be represented internally as pointers to the start point and end point in the underlying model. However, due to complications caused by providing text editing functionality (see Section 4) and due to time constraints in implementation, we were not able to implement this design fully in the first version of Intermedia. Instead, textual anchors are currently limited to insertion points and are represented internally by only one pointer. In Intermedia graphics applications, the user specifies an anchor by selecting any number of graphic objects, regardless of their proximity, and issuing the appropriate command. A graphics anchor is represented internally by a list of objects included in the anchor rather than the start and end points of a sequence.

3.1.2 Indicating Links

For users to be able to follow links, the system needs to indicate to the users where the links are anchored. The system can indicate the links in a number of ways, including marking the beginning, end, or extent of the anchor. The marking might consist of 1) an icon to mark each link, 2) a single icon at each anchor, where there may be more than one link per anchor, or 3) highlighting the extent of the anchor with font or color changes, outlining, or reverse video. Such indication could be done automatically by the system or upon the user's request. In deciding how to indicate links, designers must consider the purpose of the system, the desirablity of user control, and the potential problems of display clutter and ambiguity.

Icons are an appropriate way to mark links in some systems. Systems which require or allow anchoring links to insertion points would benefit from icons because they provide a tangible link anchor. Furthermore, icons can be designed to convey additional meaning such as whether they are the origin or destination of the link or whether the link leads to text or graphics. For example, imagine Penny Straker creating a hypermedia mail-order catalogue of traditional handknit sweaters. She wants to link from a picture of a sweater to information on price, ordering, size, yarn, and the tradition of the sweater. If all of the icons had the same design, the shopper might have to follow each link to find the desired information. If, however, the icons were designed to indicate graphically the various kinds of information, the shopper would know which to follow [Figure 2].

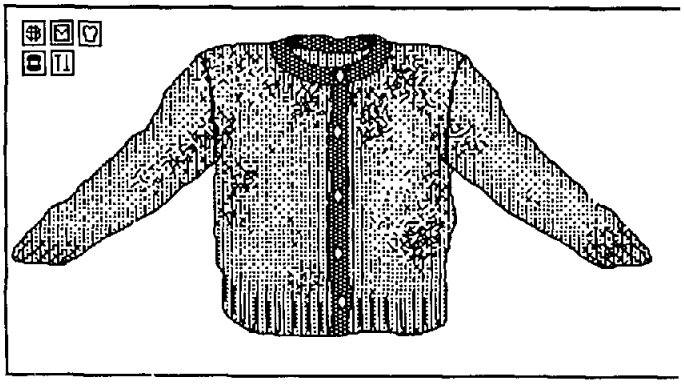

Figure 2. *Icons in a hypermedia mail order catalogue could convey what information is at the end of each link: price, ordering, size, yarn, and the tradition of the sweater.*

However, there may be some problems with icons. Readers might find the icons intrusive, especially in a document with many links. Another problem with icons is that with one icon per link, many icons might need to be displayed at the same location. For example, an article on Pennsylvania might mention Gettysburg. An author might have linked from Gettysburg to articles on Presidents Lincoln and Eisenhower, a description of the Battle of Gettysburg, and a copy of the Gettysburg address. Four similar icons would be at the same place, and the user would have no way to tell which one to follow, and even with the different icon design strategy mentioned above, the display could still become cluttered. Another strategy is to allow the user to control whether icons are displayed. The user could turn off their display to concentrate on the content of the document and subsequently turn them on to examine linked references. A third possibility is to limit each anchor to one icon, thereby uncluttering the display. However because information might be lost, the system would need to provide more information about the links that emanate from an anchor. Such additional information might include how many links exist and to what kind of document each one leads.

Icons are sufficient for marking the ends of links when anchors are just insertion points. However, if anchors can contain content, icons alone cannot give complete visual information about the anchor. A supplement to icons is the display of the anchor extent. Highlighting could consist of font or color changes, bounding boxes, or reverse video. Not only would such highlighting accurately show the extent of non-overlapping anchors, but it could also show the individual extents of overlapping anchors [Figure 3].

Although highlighting might provide some improvement over icons, it might still clutter the display. As with icons the user could control when the anchor highlighting is displayed by toggling the appropriate switch. In addition the system could provide a context-sensitive cursor that would invoke highlighting only when it moved into the extent of an anchor.

Intermedia uses a combination of icons and anchor highlighting to show links and anchors. Each anchor has one icon, regardless of the number of links attached to it. In graphics documents, icons mark the existence of anchors and highlighting indicates anchor extent. The user may

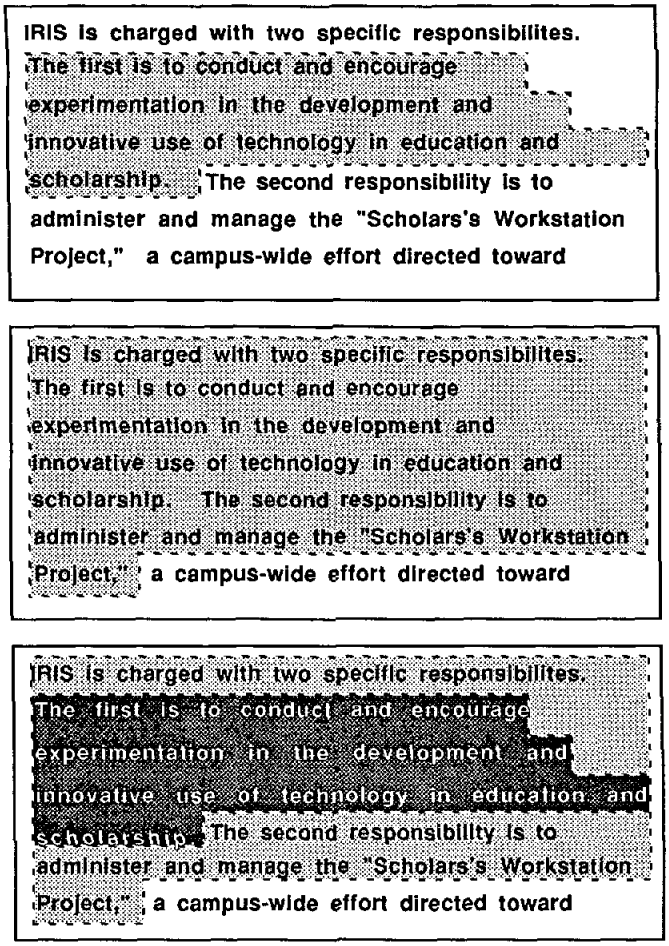

Figure 3. *Highlighting can show the extents of overlapping anchors.*

select any number of icons and ask that their extents be highlighted. Currently, text document anchors are limited to insertion points that are marked by icons. Because each icon may indicate more than one link, when the user selects an icon and gives the *Follow* command for an anchor with more than one link, Intermedia presents a list of the links that are attached to the anchor. The user can then choose which link to follow.

Plans for future versions of Intermedia include allowing the user to control icon display. Text will mirror graphics by allowing the user to attach links to arbitrary selections rather than just insertion points.

3.1.3 Names, Keywords, and Keyword/Value Pairs

If a document has many links, the user would benefit from having additional information for each link before choosing which one to follow. Imagine a poem with links leading to the poet's biography, to the poem's location on the poet's timeline, to a document that describes the style of the poem, and to a related poem by an earlier poet. All of the links are marked by identical icons. If a reader who only wants to see related poems has a way to know which links are relevant, the process of choosing and following links would be more efficient.

The system could allow authors to attach a title, keywords, and arbitrary keyword/value pairs to their links. The title would be an arbitrary character string that the author would attach. The user might select simple keywords from an available list or might enter each keyword freely without matching against existing keywords. Keyword/value pairs allow a finer granularity for keyword searching; for example, a professor creating material for a poetry course might add a keyword/value pair *meter/iambic* to one link and *meter/dactylic* to another link. The professor could later ask the sytem to retrieve links only where *meter/iambic*.

For all links the system might also have common, pre-determined keyword/value pairs to which it assigns values automatically. Possible keywords are the kind of document at the other end of the link, the title of that document, and the creation date of the link. Such information can not only provide additional information on all of the links, but can be available for querying. A sample query might be, "Show all links created after September, 1986."

Currently, Intermedia incorporates three of these features: titles, keywords, and system-defined keyword/value pairs. The system fully implements titles by providing a default title that the user can edit. A menu command provides access to the title of any link. The functionality is partially developed for both keywords and automatically attached information. Automatically attached data include the date of creation and author's name (since in Intermedia, more than one author may have access to the same document). Authors can enter an arbitrary number of keywords for any link. Whereas titles and keywords may be edited, the automatic information is fixed.

3.1.4 Enhancing Link Functionality

What kinds of links are appropriate for a particular hypermedia system? An on-line encyclopedia might only have automatically-created links from keywords to other articles. There may be no need to have more than one link type. However, in a more general-purpose hypermedia system, users might create links for different purposes. One user might only create links to connect footnotes to their sources, while another user might want to link vocabulary words in a text to their definitions.

With the basic link type – a simple tie between two anchors – users can still specialize linking tasks. Users can link footnotes to their sources and link vocabulary words to their definitions. But, by specializing links to do a specific task, the system could make it nicer for the user to make certain kinds of links. For example, a footnote link type might be represented by a special automatically-numbered icon. A vocabulary link type might automatically find the definition to link to in a dictionary file. Depending on the designer's anticipation of how the system might be used, other link types could be included, such as user-defined links.

Additional link types could provide functionality that is not possible with just a simple tie between two anchors. For example, link destination could depend on such factors as the user's level of expertise and whether the user had previously taken the same path. These conditional links

would be useful in some educational and training situations.

The first implementation of Intermedia does not provide footnote, automatic or conditional links. Future plans include both footnote and automatic dictionary links as described above.

3.2 Providing Contexts

When an author links to a destination document containing links from previous sessions either all of the links of the destination document can be incorporated into the set of links on which the author is working, or the system could display the destination document as if it had no links at all.

In the first situation, the author would automatically inherit all of the connections anchored in the destination document; all of the links would exist in the same "plane" [Figure 4]. By making links, an author would thereby provide bridges to other worlds of links. This bridge is appealing in systems where the fundamental goal is to encourage the maximum interconnectedness of information. This would, however, prevent authors from presenting a limited corpus.

The second option solves this problem by enabling the user to partition links into private worlds, or separate contexts. Such contexts would be useful to a scholar who wants to organize material for a variety of research projects, each of which may reference the same document.

Intermedia designers opted for the second solution with construct called *webs*. Until the user opens a web is ope all opened documents appear without any links. To vie documents with the links that belong to a particular web, user opens a web and then opens one or more of i documents. Although other webs may also reference t documents, only the links which were made in the curre web are available. As a result the user does not have to s through the connections of many contexts.

3.3 Visualizing Connectivity

Additional tools could be useful in helping the us understand the information structure of the link network. One such tool might be a graphic representati or map, of a web of links.

The scope of such maps is an important consideration. *global map* could render the entire linked structure, simi to a highway map of the United States. However, glot maps become difficult to display in a way that is helpful the user both in systems without partitioned contexts a in systems with partitioned contexts but with a lar number of documents. These "flat" representations oft become tangles of lines. Three-dimensional global ma could untangle the lines somewhat to clarify the linki structure. However, users might be confused wh viewing such a map.

A subset of a global map could simplify the user's cogniti world model. A *local map* could show those links t emanate from the current document, like an inset of a ro

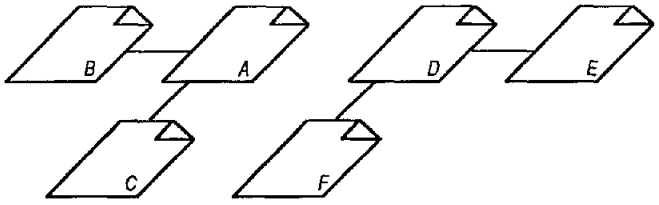

Figure 4a. *Two disjoint sets of links.*

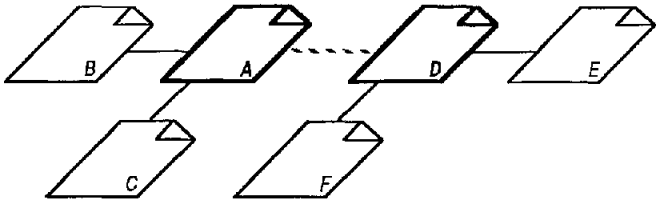

Figure 4b. *If a user makes a link between A and D, either...*

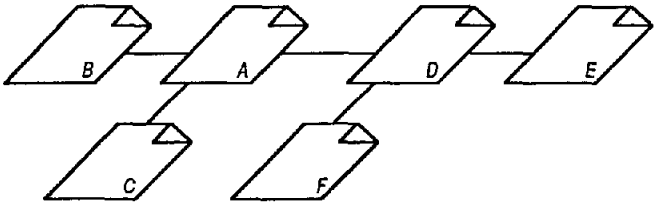

Figure 4c. *the two disjoint sets are automatically combined, or...*

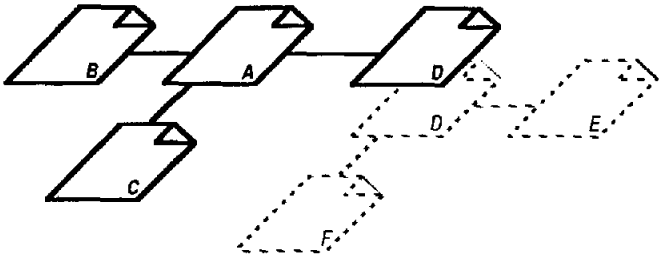

Figure 4d. *only the destination document, not the other documents to whi it is connected, in incorporated into the active set.*

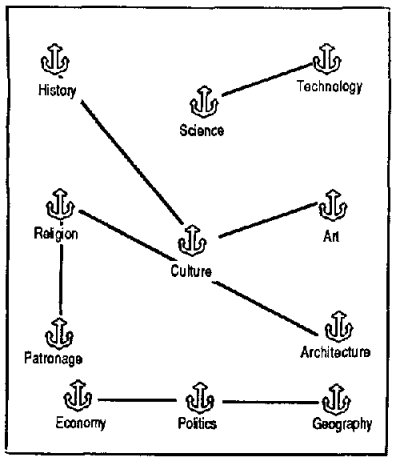

Figure 5a. *Global maps might show links between anchors,*

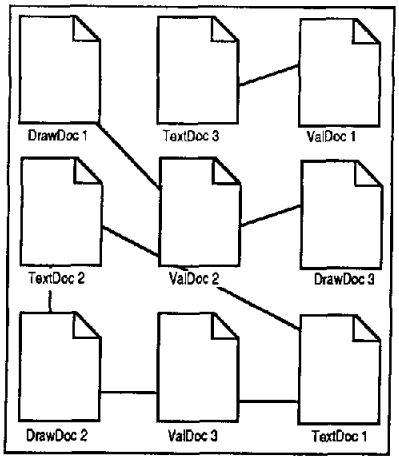

Figure 5b. *...links between documents,*

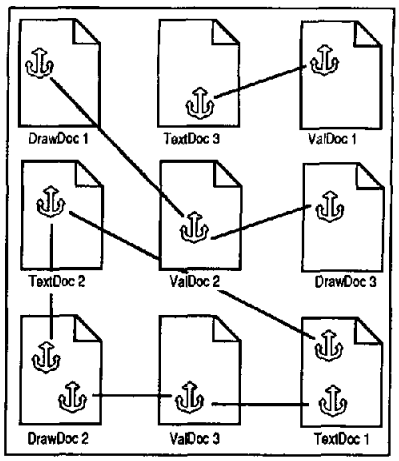

Figure 5c. *...or in both levels of detail.*

map that only has the roads that lead directly from Providence to other cities. A local map might exist as a static snapshot of the possible links from one anchor or document or as a tracking map that updates itself as the user moves from one document to the next, always keeping the current document as the central focus.

In a system where links can lead to anchors inside documents, system designers need to determine what level of detail it is appropriate to show in a map: a document to all of its related anchors or only to its related documents, an anchor to related anchors or just to related documents. Global and local maps have different potential levels of detail. A global map can show how anchors are linked together or how documents are connected. Furthermore, a global map could provide both levels of detail and illustrate how both anchors and documents are connected [Figure 5]. With local maps, there are more possibilities. As with global maps, local maps can show links between anchors, links between documents or a combination showing both levels of detail, but unlike the global maps, the connections would be shown only from the focus, the current document or anchor. In addition, because local maps have a focus they can show the documents that are linked to or from the current anchor, or the anchors that are linked to or from the current document [Figure 6].

Additional map features could include a label on each link and an indication of the direction of each link. If filtering is implemented the map could represent a subset of the web as processed through the active filters. The user could control these features, or the system could impose them.

Intermedia provides both global and local maps showing document interconnection. Future versions will provide additional map functionality.

4. A SYSTEM OF READ/WRITE DOCUMENTS

To be effective for cooperative work, a hypermedia system must provide authoring as well as browsing capability. Adding this capability to a hypermedia system increases its complexity. Designers need to determine how the system should respond to editing outside of the confines of anchors, changing the content of an anchor, and deleting anchor contents.

The effects of editing are influenced by the cognitive model of anchors that the designer decides to present to the user. An anchor can be either a list of distinct objects or a boundary that the user specifies at link creation. The former scheme means that you can clearly define the anchor, but the system has to store more information per anchor. The latter is more efficient as far as storage is concerned, but the potential for ambiguity is greater. Different media might benefit from different schemes for representing anchors. For example, anchors in a text editor are well represented by all of the text falling between two endpoints. In a structured graphics editor, on the other hand, anchors are better represented as a list of graphics objects. Even with a defined cognitive model, the designer must anticipate and resolve conflicts that can occur during normal editing. The remainder of this section discusses the details of these concerns.

4.1 Editing Content Within Anchors

With a pure boundary scheme, a text editor would represent an anchor made from the string, "green eggs and ham," as all of the characters between the "g" in "green" and the "m" in "ham," regardless of what might be changed between these two endpoints. When the user inserts the word, "fried," the phrase would read, "green fried eggs and ham," and the anchor would now reference the entire phrase. Therefore, by editing, the user could change the anchor contents. In many cases such simultaneous changes of document content and anchor content would be appropriate.

However, the integrity of a link could be at stake if the change is at the boundary of the anchor or if there are substantial changes to the anchor contents. If the user links, "eggs and ham," to a recipe, then later inserts, "green," so that the phrase now reads, "green eggs and ham," the link may no longer be appropriate. Similarly, if the user links "I do not like green eggs and ham" to a biography of Dr. Seuss and then substitutes "scrambled" for "green," the link might not be valid. Instead of

automatically updating the link anchor to include the editing changes, the system could eliminate the link on the assumption that any changes to the anchor would invalidate the link. A more flexible system could alert the user that there were changes to an anchor and allow him or her to determine if the link should remain.

The list scheme may solve some of these ambiguity problems, but it introduces new problems. With the pure list scheme, the user can select two wheels in a picture of a jeep and link them to a description of snow tires. If the user then adds an axle between the wheels, the axle would not automatically become part of the anchor; the system would appear to be correctly anticipating the user's intention. However, this scheme presents new design problems such as how the user can add objects to an existing anchor and how the system should respond to changes in any of the anchor's objects. For example, the user might want to add the two other wheels to the anchor. And, if the wheels were replaced by cement blocks, the author would probably not want the link to remain.

In Intermedia, anchors in graphics documents are objec lists and the addition of a graphics object does no automatically change existing anchors. Currently, anchor in text documents are limited to insertion points. Futur versions of Intermedia will implement the endpoin paradigm for anchors in text documents.

4.2 Editing Content Outside of Anchors

A system with read/write documents needs to insure tha editing changes outside of anchors do not change what i referenced by the anchors [Figure 7]. This is only a proble in editors of linear data, such as text, that use the boundar scheme. In such editors the insertion of new data "bumps existing data. Therefore, the system might need to adjust it internal anchor representation to insure that the prope part of document is presented as an anchor to the user. B contrast, under the list scheme, editing changes outside a an anchor would not affect the identification of the object to which a link is anchored.

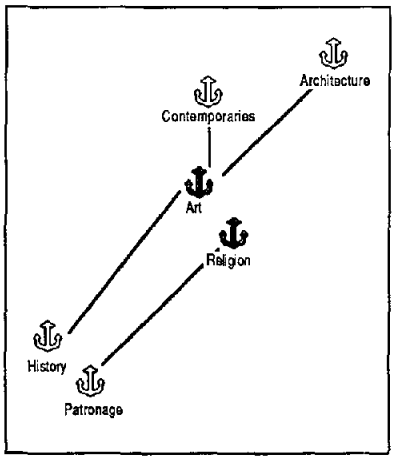

Figure 6a. *Local maps might show links between the anchors in the current document to other anchors,*

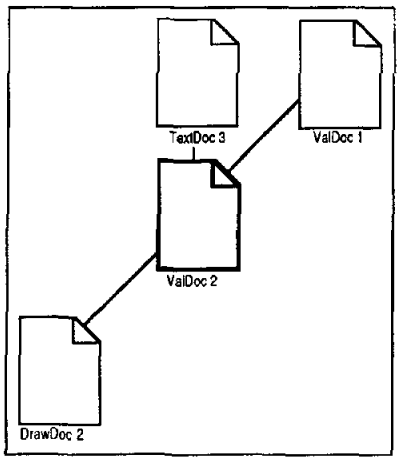

Figure 6b. *...the current document to other documents,*

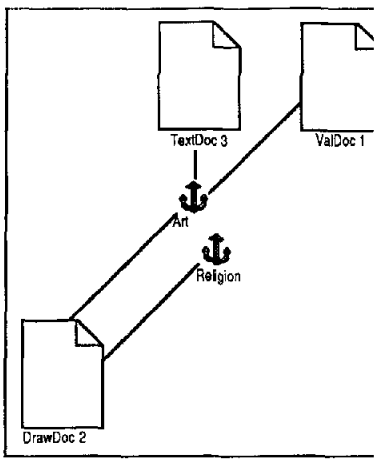

Figure 6c. *...the anchors in the current document to other documents,*

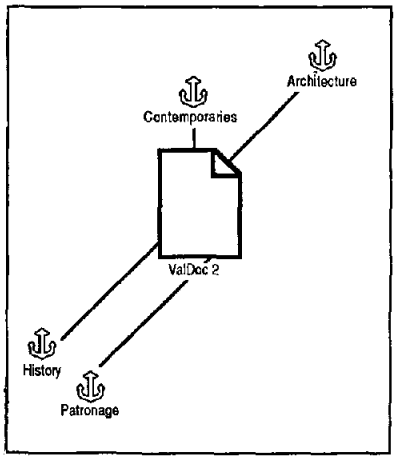

Figure 6d. *...the current document to other anchors,*

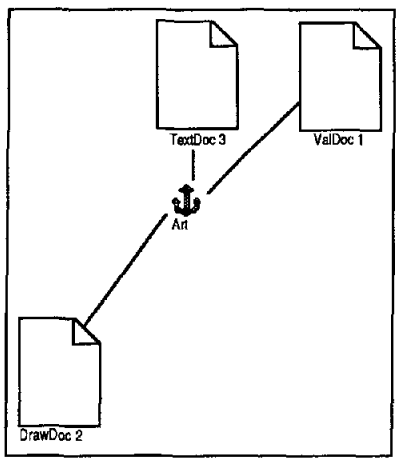

Figure 6e. *...the current anchor to documents,*

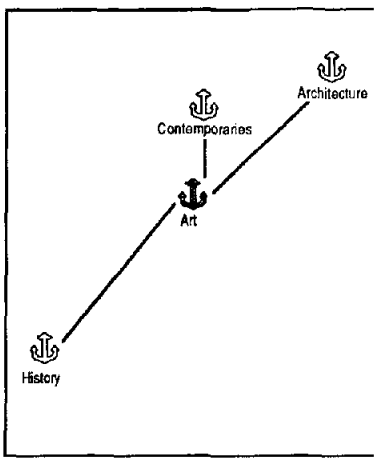

Figure 6f. *... or the current anchor to othe anchors.*

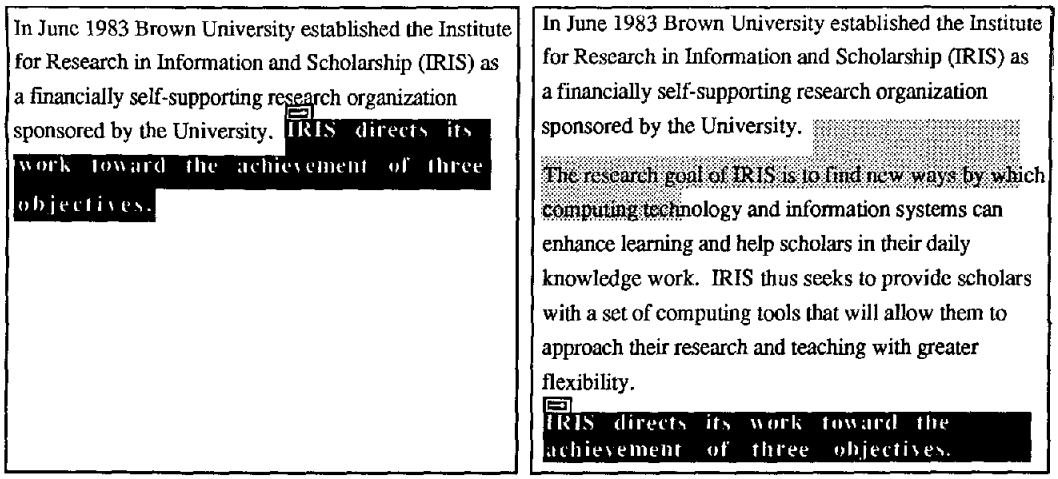

Figure 7. *(a) The user has established a link anchor. When (b) the user inserts a new paragraph, the system needs to adjust the location of the anchor highlighting.*

4.3 Deleting the Entire Contents of an Anchor

When the user deletes the entire contents of an anchor, should the system also delete the attached links? Although it would be possible to implement such a scheme by leaving the link information attached to an insertion point anchor, it might not be appropriate to maintain a reference to something that no longer exists. For example, if a reference to Dylan Thomas's, "A Child's Christmas in Wales," linked to a copy of the poem itself, an author would probably not want to maintain the link if the reference were eliminated. Alternatively, if the user wanted to cut the reference only to replace it with more accurate information, it might be appropriate for the system to attach the link to the new content. There is no way to predict whether the link would still be relevant. A modification to the above alternatives of the system arbitrarily maintaining or eliminating link information is leaving the decision to the user on a case by case basis.

Intermedia leaves the link even when all of the anchor's contents are cut to avoid being destructive. The user can then explicitly remove the link if necessary. This solution allows the user to control what links are left in each instance and recognizes that actions on content and actions on link information are different.

5. A MULTI-USER ENVIRONMENT

If a hypermedia system is to be useful for cooperative work it must provide ways for multiple authors to read, link to and from, and even edit the same set of documents, while also allowing authors to protect their work from unauthorized access or changes. At its simplest this functionality could be implemented in a system where authors have sequential access to documents. However, it would be more useful for cooperative work within a fully networked environment where multiple authors would have simultaneous access to documents.

The possibility of multiple users examining and editing documents simultaneously introduces complexity to any system. In hypermedia systems the complexity is increased because the user might not need to edit a document but might want to link to or from the document contents. Interactive linking adds another dimension to the traditional read/write acess rights paradigm. The issues involve access control to documents and webs, data storage design for both document and link data, contention management, and update schemes for propagating editing changes.

5.1 Access Control

In a system that promotes shared, cooperative work, it is important to control access to document contents and link information. Appropriate access control assures the integrity of information, allows individuals working together to have full access to their shared data, and prevents those without permission from viewing or modifying such data.

5.1.1 Document Access

Typically, systems provide two categories of privileges to documents. *Read access* allows the user to view data, while *write access* allows the user to modify data. These categories are still important in the hypermedia world, but they are complicated by the hypermedia notion of *annotate access*. Such a category would be appropriate for systems that encourage users not only to read documents and follow links, but also to create new links as they browse. With annotate access, users could create links without having the right to make changes to the content of the document . Presumably, these access rights are additive, meaning that write access encompasses annotate and read access, and annotate access includes read access.

The annotate privilege could be 1) limited to writers of documents, 2) assigned to all those with read access, or 3) be a separately designated category. The first case is most appropriate when it is necessary to prevent readers from changing the system in any way, like in delivering training materials. The case where everyone with read access can make links is appropriate in a system that encourages maximum connectivity of information, as in a university course that emphasizes discovery of meaningful relationships. The third provides the flexibility and protection necessary for effective cooperative work.

Intermedia provides all three document access categories. *Readers* may examine the content of a document and follow links. *Annotators*, in addition to having readers' rights, can add links and can modify their links, but cannot alter the content of the document. *Writers* have all of the capabilities of annotators and can also modify the content of a document.

5.1.2 Web Access

A system with webs, or separate link contexts, could also control access to the webs. The web access paradigm might follow the document access categories: read, annotate, and write. Users would need read access to navigate the links associated with a web. As in the document access paradigm, users would need annotate access to add links to a web. *Writers* would have the previous rights plus those to delete or rename the web. Furthermore, to determine if the user could access or change any document and follow or establish any link, the system would need to match the web access rights against the particular document access rights of the user. For example, to browse a web, not only would the user need read access to the web but would also need at least read access to each document that he wanted to view.

5.1.3 Assigning Access Rights

After designers specify the categories of access rights, they need to determine how these privileges might be assigned. One possibility is that the creator of the document or web is the only person with rights to modify any contents or links, and all other users would have only read access. Another possibility is to designate a document as either private or public; once a document is launched into the public domain, all users would have all rights to it. Alternatively, users could have only "constructive" rights rather than "destructive" rights, allowing them to add information but not to remove any information from the system.

Intermedia designers solved the problem by designating one *owner* for each document and web and allowing the owner to designate individuals as having read, annotate, or write access to documents and to webs. Such users may be referred to individually, by specified lists of users, by groups, or by default as the remainder of the *universe*. When a document opens, its access rights are combined with the rights of the currently open web to create an access rights *capability* for that document. For example, if the document access rights allow annotation but the web access rights only allow read access to the web, the document capability would be set to only allow read access. This allows individual owners to specify the rights they desire for their document, while allowing the owner of the web to set limits for that web.

5.2 Storing Document Content and Anchor Information

Between sessions, anchor information could be stored with document content or separately. Because the anchor information is closely related to the content, it seems natural to store the two kinds of data together. In a system where only one user at a time can annotate or edit a document, this approach might be appropriate. But if the hypermedia system allows 1) multiple simultaneous annotators, 2) different webs imposed on the same document at the same time, or 3) static documents such as those available on compact disks, system designer needs to consider a different approach.

If the system allows multiple simultaneous annotators, it would be unwise to use the one-file system; allowing multiple users to write to the same file simultaneously presents potential concurrency problems that are not handled by most operating systems. Rather, the system should provide a way for multiple individuals to access and update the anchor information without fear of write collisions. Furthermore, in the one-file scheme, the document would have to store anchor information for all webs. Each application would need to filter the link information on a per web basis.

The approach that we have taken is to store the link data independently from the document content that they reference. Such data is stored by a database management system on a per web basis. When a document opens, the database is queried for all of the document's link information in the currently opened web; the links are then dynamically added to the document's internal data structure. The database management system provides concurrency management at the record locking level over a network of workstations, so that no two individuals can update the same link information at the same time. Updates to the database are not made at the time of link creation/deletion, but are batched together using a transaction-based scheme and are added to the database when the web is saved. Because the link information is stored independently from a document, User A can have Web A opened on a set of read-only documents, User B can have Web B opened on the same set of documents, and both can add or delete links and save their respective webs without modifying the document files at all. Additionally, we anticipate that such a scheme will facilitate adding hypermedia structure to large-scale CD-ROM document sets.

5.3 Contention Management

If the system does not limit the number of users with write access at the document level, it must provide a scheme to manage simultaneous users with equal rights at run time to assure data integrity. Plans for contention management, which are well-known for conventional read-write access situations, are complicated by the addition of annotate access to the system.

writer	annot	reader	s = single, m = multipl
s	s	s	
s	s	m	
s	m	s	
s	m	m	
m	s	s	
m	s	m	
m	m	s	
m	m	m	

For each document there are eight possible combinations of current status where sss denotes a single user accessing with write, annotate and read capabilities, and ssm implies single user accessing for write and annotate access but multiple-user access for reading.

The hypermedia system designer needs to determine which combinations are acceptable. Some are far more difficult than others to implement. In particular, the last four include multiple writer privilege and require complex chalk-passing and synchronization protocols. Of the remaining four, *sss* and *ssm* are the easiest to build. Similarly, *sms* and *smm* are significantly more difficult to implement than the previous

two because of the concurrency problems involved with multiple simultaneous annotations. They require that the access to information in the web stay synchronized while multiple users add and delete links from the web and another user simultaneously makes changes to the document contents.

In Intermedia, when a user opens a web or opens a document from within a web, the system computes a capability for that user for that session for that object. For this computation, the system matches the user's access rights to the object (document or web) against the system's current activity level. It then determines whether the object should open with read, annotate, or write rights for this user at this time. For example, when a user with write access opens a document that another user with write access currently has open, the system will prevent write and annotate access for the second user and open the document for read only. Similarly, more than one user with annotate rights can have the same object open simultaneously only if no one with write privileges has the document open. Then in situations such as saving where access rights are important, the system checks the user's capability and allows only the level of activity proper for that capability. When a user opens a document from the desktop, the document opens read-only. With the appropriate access rights, the *Annotate* or *Write* menu items will allow the user to change the status to enable linking or editing.

5.4 Annotating/Editing Effects on Documents

Notification of updates is another issue to consider in a multi-user environment. There are four possible ways the system could respond to a user's changes: immediate update, immediate notification, passive notification, or no notification. The complexities of these options differ depending on whether the underlying documents are read-only or read/write.

The basis for immediate update is that a user should always have current information. Every new link could be broadcast across the network and would appear immediately in all open instances of that document. In a read-only environment this would cause little disruption for the user. The appropriate indication of the new link would appear which the user could then access. However, in a read/write environment this option would pose not only severe implementation problems, but also several conceptual problems. Presumably, if the system were to update links immediately, it should also update the document contents immediately to preserve the integrity of the information. It may be extremely distracting to a reader if the content of a document changes automatically because another user is currently editing that same document.

Immediate notification differs slightly from immediate update in that changes to an open document would result in sending a message to let all current readers of the open document know that someone had changed the document. Users could then request to have their version of the document updated. This would give the reader more control over the changes presented, thereby preventing the problem of unexpected display changes.

Passive notification would inform users, via a facility such as electronic mail, that there had been changes to documents since they last opened them. Such notification would be particularly helpful to users who had made links to these documents; they could then determine whether or not the links should remain given the changes in the document.

The simplest solution for implementation is not to provide any notification of changes. Open documents would remain as opened, but the next time the document opened it would reflect the changes. These last two alternatives, passive notification and no notification, imply that most link information is not so timely that it needs to be known immediately.

Intermedia currently provides the no notification scheme as an interim solution. We are investigating ways of allowing users to determine the level of notification they would prefer.

6. FUTURE RESEARCH

There are other issues in hypermedia that we are anxious to explore in future versions of Intermedia. One such issue pertains to sophisticated navigational tools. Such tools could show what links the reader followed to get to the current document, could allow the author to define paths for readers to follow, and could let the user leave a "bookmark" in the corpus so that later the reader could easily "jump" to that place. We also want to implement filtering so that the user could concentrate on links of particular individual interest in the current document and perhaps throughout an entire web. Furthermore, we are planning on looking into "hot links" that would force an instance of the information at one end of the link into the document at the other end. A hypermedia system that would automatically keep track of versions of documents [Nels81, Deli85] is also an area for future exploration.

7. CONCLUSION

Hypermedia is a powerful concept for cooperative work. The goal of presenting a "seamless" environment, one in which a reader can become an annotator or an author, presents interesting user interface problems. The goal of presenting that same seamless environment over a large corpus of documents that can be read, annotated, and edited simultaneously by multiple users over a communications network presents comparably difficult systems implementation issues. And the goal of presenting a new paradigm for sharing information in such an integrated fashion poses similarly difficult questions of social interaction and the management of such systems. With Intermedia, we have made a first attempt at resolving some of these issues, and we hope that this paper will allow us and others to continue to look at the complicated but fascinating unanswered questions of hypermedia.

8. ACKNOWLEDGEMENTS

Thanks are due to Helen DeAndrade for designing the figures for this paper and to Nicole Yankelovich for critiquing early drafts.

The development of Intermedia has been the culmination of over a year of intense, coordinated effort by a large team, all

of whom equally share the credit for the determination and resolution of the issues explored above. In particular, Nicole Yankelovich coordinated the end-users of the system, developed and refined much of the Intermedia user interface and developed extensive end-user documentation. Charlie Evett developed a large part of the Intermedia architecture, the graphics building block and the InterDraw application. Ed Grossman, Matt Evett, and Tom Stambaugh developed the text building block and the InterText application. Steve Drucker and Bern Haan developed the Intermedia framework. Page Elmore developed the web database and the web map capability. Helen DeAndrade created the graphics for the system.

In addition, Mike Braca, Dave Bundy, Dan Stone, and Scott Bates provided the base operating system, the database, and the operating systems extensions that enabled Intermedia to run. Brian Chapin and Sam Fulcomer configured the systems countless times. Eric Wolf developed the program for decompressing scanned images. George Landow, David Cody, Glenn Everett, and Rob Sullivan produced an enormous corpus of material using Intermedia and heroically subjected themselves to the ordeal of teaching an English literature course with the system in its infancy. Andy van Dam, Marty Michel, Bill Shipp, and Don Wolfe provided advice and support throughout the project. Kate Archambault translated much gibberish into English.

This work was sponsored in part through a grant by the Annenberg/CPB Project and a joint study contract with International Business Machines Corporation. In addition, we greatly acknowledge Larry Tesler of Apple Computer, Inc., Jeff Singer and Stan Fleischman of Cadmus Computer, Inc., and Mark Nodine of Bolt Beranek and Newman for their assistance in making available key software that made Intermedia viable.

9. REFERENCES

[Deli85] N. Delisle and M. Schwartz, "Neptune: a Hyperte System for CAD Applications," Tektroni Laboratories Technical Report No. CR-85-5(Beaverton, OR, 1985.

[Enge68] D. Engelbart and W. English, "A Research Cent for Augmenting Human Intellect," *Proceedings c FJCC*, Vol. 33, No. 1, AFIPS Press, Montvale, NJ, Fa 1968, pp. 395-410.

[Garr86] L. Garrett and K. Smith, "Building a Timeli Editor from Prefab Parts: The Architecture of a Object-Oriented Editor," *Proceedings of OOPSL* Portland, OR, September, 1986.

[Haan86] B. Haan, S. Drucker, and N. Yankelovich, "A Object-Oriented Approach to Developing Consister Integrated Applications," Submitted to SIGCHI '87.

[Meyr85] N. Meyrowitz, et al, "The Intermedia Syster Requirements," IRIS Report, Institute for Research Information and Scholarship, Providence, R September, 1985.

[Meyr86] N. Meyrowitz, "Intermedia: The Architecture ar Construction of an Object-Oriente Hypertext/Hypermedia System and Applicatio Framework," Portland, OR, *Proceedings of OOPSL* September, 1986.

[Nels81] T. Nelson, *Literary Machines*, Swarthmore, P. 1981. Available from author.

[Yank85] N. Yankelovich, N. Meyrowitz, and A. van Da "Reading and Writing the Electronic Book," *IEE Computer*, Vol. 18, No. 10, October, 1985, pp. 15-30.

[Yank86] N. Yankelovich, "Intermedia: A System f Linking Multimedia Documents," Submitted SIGCHI '87.