Is annotation a form of hypertext linking?

P. J. Brown
University of Exeter
Harrison Building, North Park Road
Exeter EX4 4QF, UK
P.J.Brown@ex.ac.uk

Heather Brown
University of Exeter
Harrison Building, North Park Road
Exeter EX4 4QF, UK
H.Brown@ex.ac.uk

ABSTRACT
An annotation and a hypertext link share many common properties. We examine the similarities and differences. It often happens, however, that two ideas are conceptually similar, but the devil is in the detail, and it is in fact impractical to combine them at the implementation level. To see if this applies to the two ideas of annotation and hypertext linking, we conducted a practical experiment, where we tried to build an annotation facility by just using existing hypertext linking facilities, without any change.

Categories and Subject Descriptors
Information Systems Applications [**Hypertext/hypermedia**]: [architectures]; Document and Text Processing [**Document Preparation**]: [Hypertext/hypermedia]

General Terms
Documentation

Keywords
Annotation, link, hypertext

1. INTRODUCTION
Annotation is a fundamental need in document usage, as evidenced by its extremely common use on paper documents. The need is equally great for electronic documents. This was recognised, for hypertext documents, by Vannevar Bush [5], though annotation systems for hyperdocuments are still not widely used. For word processed documents, however, there is a more widespread use of electronic annotation, notably using the comprehensive facilities provided by Microsoft Word.

In this paper we look at annotation, and compare it with another fundamental facility, hypertext linking. The question is: are annotation and hypertext linking truly different, or is it better to regard them as instantiations of a single meta-facility? Having discussed this we then go from the theoretical to the practical, and report on whether annotation and hypertext linking can be combined at the implementation level.

When looking at hypertext, we consider hypertext systems in general, rather than just WWW and systems for augmenting WWW [2]. We concentrate on textual documents, though annotations can, of course, be useful in any media.

2. THE NATURE OF ANNOTATION
As a piece of terminology we will use the word 'system' to mean the software being used for reading and perhaps also writing documents; the system may, for example, be any hypertext system. We will use the term *original document* to describe the document that is being annotated. In any general system annotations can themselves be annotated: thus an original document may itself be generated by a process of annotation. Where there could be any doubt, we will use the term *very original document* to describe the first original document, before any annotations were added.

A fundamental property of an annotation is whether it can only be an addition to the original document or whether it can be a general edit, and therefore can involve both deletions and in-line additions to the original. We will use the terms *enhancement-annotation* and *edit-annotation* to describe the two cases, respectively. An example of an edit-annotation is where a user deletes several paragraphs of a displayed page, and replaces them by "... not of interest to me ...". A still more general form of an edit-annotation allows the user to change any element of the source of a document, for instance its mark-up as well as its content. Thus a user may wish to change the mark-up in order to enhance the display of paragraphs that interest them, and perhaps reduce the size of uninteresting paragraphs to 6-point type. Furthermore, as a WWW example, they may want to change the URL attached to an existing link, in order to make it lead to a document that is more relevant to them.

Most current annotation systems only allow enhancement-annotations, though Microsoft Word's facility for revisions is an exception to this. Here we cover edit-annotations too.

2.1 Informal definition of annotation
Before proceeding to further details we will give an informal definition of what we mean by an annotation. We will define the simplest form of annotation and relate it to the simplest form of (one-way) hypertext link.

An annotation is where the author causes an existing document, which may be owned by someone else, to link to a document of their own (i.e. the annotation).

This mirrors a possible definition of a simple hypertext link: a hypertext link is where the author causes a document of their own to link to an existing document, which may be owned by someone else.

The only difference between these is the ownership of one end of the link, or, to put it another way, whether the user's document is a source or destination of linking. Given this, it is not surprising that there are many similarities between annotations and hypertext links. The similarity continues to apply when more sophisticated hypertext links are considered, e.g. DeRose's [8] *intensional links* where, for example a link can be attached to all occurrences of a certain word within a document.

In practice there can be differences of scale between annotation and hypertext links: an annotation is often very short, perhaps a couple of words, so the new 'document' the author is attaching is therefore not a big one. This practical difference is, however, not fundamental.

There are two cases where annotations come particularly close to hypertext links: (a) when an author is annotating their own work, and thus the issue of separate ownership does not apply, and (b) in systems where all links are two-way.

Our informal definition of an annotation needs extension if we are to cover edit-annotations: by definition, an edit-annotation affects the content of the original document; thus an edit-annotation needs to be integrated with the original, not displayed entirely separately from it.

2.2 Pre-selection

The normal way annotations are displayed is, of course, different from hypertext links. Both have a similarity in that there is a *presence indicator* embedded within the display of the original document — this is the source anchor. We discuss anchors later. Then the behaviour becomes different. For a start, many annotation systems make all annotations appear automatically, without any need for the user to select them; we call these *pre-selected* annotations. Secondly, when an annotation appears, the original document does not generally disappear; instead the annotation appears alongside (see next Section). There is, however, no fundamental reason why an annotation should not be displayed in the same way as a hypertext link; in an extreme case the two might be indistinguishable to the user.

2.3 Display of annotations

We will now explore further the rendering and display of annotations. Typically systems that only allow enhancement-annotations will display annotations separately from the original document, e.g. as pop-up windows or as marginal notes. Sometimes the form of display will be fixed, but with more flexible systems it will be under user control. An example of the former is a very basic form of annotation: the footnote. Footnotes can be regarded as a type of annotation where the form of display is fixed (i.e. the footnotes always appear at the bottom of the page); authors rely on this behaviour when preparing documents. (A footnote is also a very basic example of annotation for another reason: it typically only involves one user, the author annotating their own work.)

One particularly well-developed style of display is based on Fluid Documents [14]. Fluid Documents use animated typographical changes and generally prevent the annotation occluding the source — an important property given that many annotations are designed to be read alongside the original document. This mirrors techniques used by folding editors, and by hypertext systems such as Guide [4] that display in-line the destination document of a hypertext link. This similarity is important in our practical experiments discussed later.

Systems that allow edit-annotations must of course integrate the display of the annotations with the rest of the document. The usual result will be that the user sees what appears to be a single document. There may be some defaults by which annotations are made to stand out visually from the original, e.g. added material is displayed in a different colour; moreover annotations that are deletions might still show the original text, perhaps as crossed-out material. Such default behaviour is useful when annotations are being used by a pair of authors jointly preparing a paper, or in critiquing applications where a tutor suggests corrections to a student's essay.

To summarise, there is a wide diversity in how current systems display annotations. In many cases the method of display determines what an annotation can be used for. Hypertext systems also have diverse methods for displaying the destinations of a link (though WWW, the dominant system, is limited). These diverse methods overlap considerably with those used for annotations.

2.4 Data types for annotations

In the simplest systems there is only one sort of annotation. In others, annotations can have an associated 'data type', which defines their behaviour. The same applies to hypertext systems. For example many hypertext systems for argumentation [6] have built-in data types for links (such as 'supporting-argument', 'counter-argument') and corresponding built-in behaviours. The data types help the system produce aids to the user such as a graphic summary linking all the supporting arguments and counter arguments.

Some annotation systems do not have built-in data types but allow the user to represent data types by tags attached to annotations (e.g. 'Vital', 'Suggested'). The user can often attach behaviours for each tag. Typically they are simple behaviours concerned with the display of the associated annotation, e.g. that Vital annotations be in red. In Annotea [9] *all* annotations are considered as metadata: one part of this metadata is interpreted as a data type.

Many systems support a form of filtering that allows the user to confine the view to certain data types, e.g. 'only show the Vital annotations' or 'only show me the original documents with an annotation of type Recommendation' (or even with a recommendation score above a certain threshold [12]).

In addition data types might be useful in applications where the user started by browsing annotations rather than the original documents, and only linked to the original document if the annotation was of a certain type. For example a user might browse a set of annotations representing a literature review, and only link to the original document when the annotation is of type Theoretical. The facilities offered by data types may be extended by the use of other metadata attached to an annotation, e.g. its author and its creation date.

A continual series of annotations applied on top of existing annotations, as might arise in a discussion group, can lead to an annotated document appearing as an incomprehensible mess. Data types (and metadata in general) are an aid to the implementation of *threads*, which make documents easier to view and more related to the user's interest.

To summarise, data types were introduced in the early days of programming languages in order to add discipline, functionality and expressive power: the same applies to annotations.

2.5 Annotation without changing the original

In many cases when a user wants to make annotations on a document this can be achieved by making a copy of the original and embedding the annotations in this copy. There are, however, many cases where it is desirable to store the annotation separately from the original document. Some possible reasons for this are:

- the original document may change, and it may be desired to apply the annotations to the new version. Several systems have embodied heuristics for achieving this, one of the most developed being ComMentor [12]. Inevitably, however, if change is too great the heuristics will fail and the annotations will become orphans. We did not implement change-resistant heuristics in our experiment, since this was outside its aims, but such heuristics could be added if necessary.
- group annotation: several people may wish to apply annotation to the same original document, and, moreover, may wish to see all the annotations, not just their own. The obvious way to support this is to keep annotations separate from documents.
- the original may be part of an integrated set of documents, and making a copy may lose the advantages of such integration.
- the original may be huge, thus making a copy expensive in storage.

In a parallel manner, hypertext links often lead to existing documents that are not owned or controlled by the author of the link. In such cases it is rarely sensible for the author of the link to make their own copy of the destination document, and to link to this copy rather than to the original.

3. THE EXPERIMENT

The above discussion has shown many similarities between hypertext links and annotations. However, mechanisms can be similar in theory, but may differ so much in small practical details that it is impractical to combine the concepts. This part of the paper describes a small experiment to test the detailed similarity of annotations and hypertext links.

The aim of the experiment was therefore to make a practical test of the hypothesis that annotation and hypertext links are similar. The experiment was focussed on the source form of a document; we were not interested in the rather different issues of annotating, say, an image derived from the Postscript form of a document. Our requirements for the experiment constrained it to use an existing hypertext system to effect annotation, thus aiming to show that hypertext linking covers annotation too. Obviously using WWW itself was infeasible: WWW has a limited model of hypertext linking, which is unsuitable for annotation; that is why so much effort has been devoted to creating separate annotation systems for WWW documents. The existing hypertext system needed to be more flexible and general than WWW. Moreover it needed to offer built-in authorship: the user, when reading a document in the normal way, wants to change to become an author, in order to create an annotation. This facility was relatively common among the hypertext systems that proliferated in the eighties and early nineties, before the success of the WWW knocked most of them out of the water. We chose to use the UNIX implementation of Guide, partly, of course, because of our own association with it, but partly because one of its principles is 'the reader is the author and the author is the reader', i.e. readership and authorship are integrated. Guide — we use this term henceforth to mean the UNIX implementation of Guide rather than the separate PC product created by OWL — is now rarely used, being one of the victims of the web's success. It has not been changed since 1992, and is based more on the founding principles of hypertext than on recent extensions to the principles.

Important cornerstones of the experiment were:

1. the hypertext system must be used 'as is', without any extensions being added to make annotation easier.
2. the experimental focus was on functionality rather than HCI, i.e. the question was 'can it reasonably be done?' rather than 'can it be done in a very pleasant-to-use way?'.
3. it should be possible to apply annotation to an existing read-only document, without making a copy — see previous discussion; thus the annotations should be stored separately from the underlying document.
4. it should cover edit-annotations as well as enhancement annotations.
5. the annotations should be first-class citizens, e.g. they themselves might be subject to annotations, hypertext links, etc.

Guide offers two facilities that are important in achieving the above:

1. it offers a facility, called 'contexts', to mark any material (text and/or graphics) as being of a different logical type. The author can create their own contexts. Attributes are attached to contexts to distinguish them: many of these attributes are concerned with appearance, such as font and colour. Thus in many ways the <SPAN> element of HTML is similar to contexts. Guide contexts can be nested, and inherit attributes from the containing context.

2. it offers a facility to save mark-up, together with extra text/graphics, separately from the underlying document. This was inspired by the thinking behind Microcosm [7]. The facility caters for many different layers, one on top of the other, thus, for our experiment, allowing annotations of annotations. The original intended application of the facility was to add hypertext mark-up to read-only documents, e.g. documents from CD-ROMs. The upper layers refer to the lower layers by their document name. The very lowest layer is often an existing read-only document. (Other layers can be read-only documents too, if the user is annotating existing annotations made by someone else.) In retrospect the implementation of the layering facility was based too closely on the CD-ROM application, there being little flexibility in choosing what belongs to what layer. Contexts, however, provide a partial way round this inflexibility, since an attribute of a context is the layer it belongs to.

We used both the above facilities. In the case of the layering facility, the underlying document is not aware of the layers placed above it — in the same way that a hypertext destination document is unaware of the links leading to it, at least if the system is based on one-way links. In our case, if the user wants to see both a document and its annotations, they have to start by selecting the annotation file, not the underlying document.

3.1 The anchor and the attached document

Each annotation consists of:

- an anchor: this is the fragment of the original document to which the annotation is attached. For some annotations, e.g. an insertion, the anchor may be a point, i.e. a null fragment. Ideally it should be possible to create an annotation with multiple anchors, for example the annotation 'these two sentences contradict', where the anchors are the two offending sentences. We did not, however, cover multiple anchors in our experiment, and Guide is inadequate for supporting them.

- an attached document: this is the annotation itself. Although we call it a document (reflecting its status as a first-class citizen), it often consists of only a few words. We took a simplified view of destination anchors: we always assumed the destination anchor was a point at the start of the attached document.

The above is similar to a hypertext anchor and a hypertext destination document; thus it presents no problem to a hypertext system. Indeed the anchoring process is almost identical in hypertext systems and in annotation systems. As an aid to our combined approach, we will use the generic term *association* to cover both the attached document of an annotation and the destination of a hypertext link.

3.2 Conducting the experiments

The experiments were performed by ourselves. We re-emphasise that they were tests of what functionality can be achieved in practice. We did not aim to perform usability testing, where a panel of typical users tried to use the functionality.

The experiment consisted of two sub-experiments, both corresponding to tasks commonly performed by researchers:

- an experiment with edit-annotations. Here the annotations were a set of comments on a paper, many of them suggesting edits that should be made. The experiment was a real one in the sense that we took the full set of annotations that had previously been made, by hand on paper, on a version of a paper. The experiment was to create and use these electronically rather than on paper. We assumed the annotator did not have the authority to make revisions to the original paper, e.g. to use the equivalent of Microsoft Word's revision marks.

- an experiment with research papers. Here the annotator wished to mark those parts of the paper that were of special interest.

The experiments built on proof-of-context work done nearly ten years ago by Wusteman and H. Brown [13].

3.3 Experimental details

Our first need was to analyse the types of annotation used in each application, and to design a different Guide context corresponding to each such data type. The name of the context should be chosen to capture the nature of the annotation. Possible names might be 'Relevant-to-my-thesis', 'Replacement', 'Suggested-improvement', 'Required-improvement', etc. We have found that using a name to capture the type of an annotation is adequate for many applications, though it is inadequate for numerical values (e.g. levels of importance) or for types with composite properties.

Our second need was to set the parameters for the Guide user interface so that it provided the necessary facilities. Guide allows its menu to be enhanced by adding new buttons; these buttons essentially represent macros that run a script consisting of a sequence of Guide actions. For our experiments we used a number of new menu-buttons, e.g. menu-buttons called $+\mathbf{X}$, where X is the type of the annotation — in Guide terms the name of a context. These menu-buttons create annotations of the appropriate type. In detail, each menu-button runs a script which, *inter alia*, creates a new Guide button and asks the user (the annotater) to type the association of the button.

We then needed to decide how to effect an annotation. Our first choice, in deciding what $+\mathbf{X}$ would do, was to decide what type of Guide button would best act as an annotation

mechanism. Buttons are Guide's mechanism for effecting a hypertext link; the button itself is a presence indicator and provides the source anchor of the link; the association (in Guide terminology the replacement) of the button, which may be static or created dynamically, provides the destination of the link. Guide offers three types of button:

1. *action-buttons*, which just perform an action. Typically the action is to run a script — a UNIX shell script and/or a script of Guide actions — which is attached to the action-button. The script can do anything: in particular it can create a pop-up window and place information in it — an approach commonly used by annotation systems. The original window is unchanged, though it might be partially obscured if new pop-up windows are created. An action-button can readily be used to provide an annotation facility.
2. *replace-buttons* — of which there are several sub-types. The replace-button is replaced in-line by the association of the button, in a manner akin to a folding editor. Thus the display of the original document changes: usually activating a replace-button makes the document, as displayed on the screen, grow. On option the original button-name can remain, so that the user sees both the original button-name and its association. This is ideal for use as an annotation facility. Indeed, as we have said, it approximates the approach of the Fluid Links system.
3. *glossary-buttons*. When a glossary-button is selected, an extra sub-frame appears within the Guide window. This sub-frame contains the association of the glossary-button. As its name implies, the default behaviour of a glossary-button is to search for the association in a dictionary that acts as a glossary of terms. However the association can be created in other ways too, in particular by dynamically running a UNIX shell-script. Like the other button types, a glossary-button is a possible foundation for implementing annotations.

It is interesting that any of the three Guide button types could potentially be used to provide an annotation facility. This gives further evidence that hypertext linking can be similar to annotation not only in theory but also at the level of implementation detail.

For the experiment we decided to use just one button type: the glossary-button. This was mainly for the practical reason that this offered the best route for keeping annotations separate from the original document. The design of the glossary-button envisaged the author selecting a word or phrase in an original document (e.g. a specialised word or phrase that the user might need explanation for) and turning it into a glossary-button by adding extra mark-up. As we have said already, Guide allows this mark-up to be stored separately from the underlying document.

In fact Guide offers some (rather baroque) facilities for pre-selecting buttons, and thus for pre-selecting annotations. Taking the parallel with a folding editor, Guide allows its documents to be presented with pre-defined folds and unfolds. Although pre-selected annotations are a facility some users would value, we chose not to use them in our experiment. Instead the user had to select the anchor of an annotation in order to cause the association of the anchor to appear.

3.4 Separating commentary and content

In our experiment with edit-annotations, the association of an annotation often consisted of a mixture of the following:

- *commentary*: this is at the meta-level, i.e. it is *about* the original document.
- *augmentation*: this is at the same level of the document, and might, for example, cover suggested rewordings. (Where unambiguous, we call this 'content'.)

Some examples of the above are:

- 'I disagree' is an example of a commentary.
- 'shall' (as an annotation on the anchor 'will') is an example of an augmentation — in this case a suggested change.
- 'Apo3' (as an annotation on the anchor 'DR3' — these two words are synonyms, being names for the same protein) is another example of an augmentation. The reader may have made the annotation because 'Apo3' is the term familiar to them.
- 'shall — Fowler recommends this in preference to will' is a composite of augmentation and commentary.

In practice, when we make annotations — either on paper or electronically – we are often careless in distinguishing augmentation from commentary. There are conventions, e.g. when proof-reading we might encircle material that is commentary, but we often forget these conventions. Overall, annotations are often something done quickly, and thus carelessly. In our experiments, however, we tried to be careful and reap the benefits of this care, but inevitably failed sometimes. To give ourselves a chance of success we defined a data type called *Commentary*, used to delimit commentary — see below. This is a rather different use of data types from their use to mark the separate types of annotation.

3.5 Approaches used

Each annotation is represented by a Guide glossary-button, and every Guide glossary-button is an annotation. Thus glossary-button is effectively a data type to mark an annotation. In the experiment, the user (actually ourselves) created a number of Guide contexts to describe the different data types used in annotation. Each such context was given a unique name (this name then automatically appears in the Guide menu, and provided a means of creating an instance of the context), and a number of properties — in our case these were concerned with appearance (fonts, etc.) and layering. In the first sub-experiment, the one concerned with suggested changes in a paper, these contexts were generic: they would be usable by anyone making annotations on any paper. The contexts were:

- *Deletion*: this is used to mark the anchor of an annotation that is a suggested deletion, i.e. the anchor is the material to be deleted.
- *Annotation*: this represents all annotations that are not deletions. In practice we took this as the default, and rarely used the *Annotation* context.
- *Commentary*: this delimits, within the association of an annotation anchor, the parts that are commentary rather than augmentation. If material is not within the Commentary context it is augmentation. The replacement of an insertion or replacement is often just augmentation, but could involve commentary. The association of a deletion, if not null, is typically a commentary describing the reason for the suggested deletion.

Figure 1 illustrates the first sub-experiment.

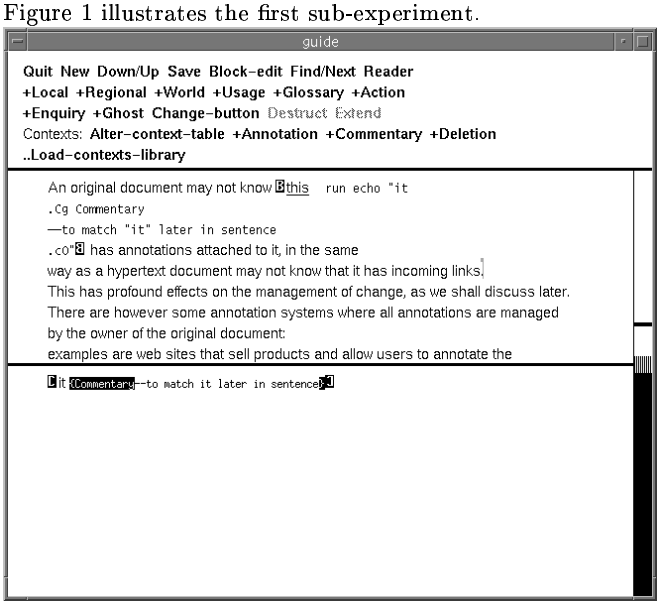

Figure 1: ***part of the first sub-experiment***

The first impression of the Figure is no doubt its old-fashioned appearance; Guide's user-interface was designed in the eighties. The Figure shows the extra menu-buttons, *+Deletion* etc. The Figure is displayed in Guide's 'author-mode', meaning that the underlying Guide mark-up is shown (white-on-black); this mark-up would be invisible if the user chose 'reader-mode'. We use author-mode for the Figures in this paper so that the interested reader can see the underlying authorship mechanisms used. Figure 1 shows one annotation expanded (the anchor is the word 'this' and the annotator suggests it be changed to 'it', giving as commentary the reason why); as we have said, when a glossary-button is expanded, Guide divides its screen into two sub-frames, and the association of the glossary button appears in the lower sub-frame. In reader-mode the relevant portion of the screen would be ' ... may not know this has annotations attached ... ', and when the user selects the this glossary-button, its association pops up in the lower sub-frame.

For the second sub-experiment we took the view that the annotation task was specific to some current needs of the user. We assumed the reader was annotating a paper for three purposes:

- to provide specific points that were relevant to a student's dissertation. The student is called Wei, so we created a context called *Wei-project* to mark these specific points.
- as above, but for a paper on Information Retrieval that the annotator was himself currently writing. These are marked by a context called *IR-paper*.
- to mark material suitable for use as quotations. The annotator is intending to build an anthology of 'pithy quotations in computer science', and, whenever he finds a candidate, marks it in the original paper, so that he can possibly use it later (perhaps employing an automatic tool to collect all the quotations — see below).

The second sub-experiment also used the *Deletion* context from the first sub-experiment. This is used to delete sections of the paper that are of no likely interest for the current tasks or any likely future ones. It also used the *Commentary* context; the need for this is generic across most annotation applications. We discuss the second sub-experiment in more detail later. We now move on to discuss the results of the experiment, and to highlight issues that arose.

3.6 Some general results

The most important conclusion of the experiment, which applied to each sub-experiment, is that, yes, it worked. Hypertext facilities, unchanged from their state more than ten years ago, have been successfully used to create the annotations used for real needs.

There are also several more specific conclusions. A number of these are typical of those that arise in any comparison of electronic working with paper working. Annotation of paper documents is exceptionally easy to do, and an electronic system will always be worse in some, or many, respects.

Because annotation of paper documents is so convenient and easy, most of us, when we want to annotate an electronic document, will print it out on paper, and annotate the paper copy. Even if it was possible to perform electronic annotation, and the annotation system had the best user interface in the world, many of us would still prefer the paper approach. With the experiments we performed there was a further negative factor: we were placing annotation on the document source — both our sub-experiments used a TEX source form of a document — rather than on its WYSIWYG form. This was a further inconvenience in terms of user interface. (It is, however, a benefit in terms of precision and robustness over change [1]. Some WYSIWYG annotation systems just annotate a position on a rendering of a document; this method fails if a document changes, or has many possible renderings; it can also introduce fuzziness as to what document construct the user is annotating.) Although our experiments did not focus on usability, the comparatively tedious process of creating an annotation, compared with scribbling a comment on paper, made us realise why electronic annotation systems are not currently popular. Nevertheless there are still times when paper has enough

disadvantages for people to prefer the electronic alternative. Four examples are (a) when several people are annotating the same document, and there is a need to see the union of all the annotations; (b) when annotator and author are physically remote from each other, and the annotator would otherwise need to return his comments through snail mail or Fax; (c) when the annotations are long, and thus the task of copying them later into an electronic document would be long and error-prone; (d) when the document being annotated is inherently electronic, as a hyperdocument is.

A positive feature of electronic annotations is that automatic tools can be written to process them and perform some action that the user wants. Thus if the user has annotated several papers by marking a desired quotation as an annotation anchor and designating such annotations to be of type Quote (as is done in the second sub-experiment), then an automatic tool can collect all such quotations into a new document, and perhaps mark each quotation with the author and title of the paper it came from. Another type of automatic tool might process edit-annotations. Assume an author has received a set of suggested edit-annotations; she looks them through, deleting a few and changing the content of some others. When she is satisfied she calls an automatic tool to effect all the remaining edit-annotations. Thus if the annotation is a deletion, the deletion is applied, and if it is an insertion or replacement this happens too. (This has similarities to the Revisions facility in Microsoft Word.) The automatic tool can strip out commentary on the annotations (i.e. material enclosed in our *Commentary* context), or perhaps leave it as commentary in the revised document.

The issue discussed above, the potential inconvenience of creating electronic annotation against paper annotation, is in every sense the first issue. There are, however, four further issues that arose from our experiments — all but the last of them relating to paper/electronic differences. The second issue, familiar from the earliest days of hypertext research, is the lack of context in an electronic form. (Here we use the word 'context' with its general meaning, not its specialised Guide meaning.) In a paper document it is trivial to find the context of an annotation, e.g. what section of the paper it is in, whereas in an electronic document — even if special aids are added — it can be hard. The field-of-view is almost always smaller on screen than on paper, and using a scroll-bar is a poor substitute to flicking through pages of a paper.

A third issue relates to finding annotations. Usually the author wants to proceed sequentially through all the comments made by an annotator (irrespective of whether the annotations were created by someone else, or by the author herself at an earlier date). During this sequential search she might amend the annotations she disagrees with. A sequential search is simple to do on paper, especially if annotations are marked clearly in the margin. It is easy to overlook annotations on an electronic system. Fortunately in our experiments there is an aid to proceeding sequentially through annotations: Guide has a built-in command 'Find next instance of X', where X can be any object, e.g. the glossary-button object we used to enclose the anchor of each annotation.

A fourth issue is carelessness. We have emphasised that an advantage of electronic annotations is that an automatic tool can process them. The automatic tool can only work well if the annotations have been performed carefully. It is much less successful if annotations have been carried out carelessly. Careless errors often happen when the user drags the mouse over a piece of text to make it an anchor: often the user will cover one character too few or one too many. There is a special issue with spaces in suggested edit-annotations. If the annotator suggests inserting the word 'usually', what should really be inserted is the word 'usually' followed by a space. If the space is omitted — arguably this is a piece of carelessness, but it is almost universal practice — the automatic tool to effect the annotation may come up with 'it is usuallytrue that'. In some examples, however, such as the insertion of the prefix 're-', no extra space is required. Thus if the annotator has been careless it is hard to come up with automatic rules to do what was really intended. Overall, as Marshall [10] has noted with experiments on paper documents, annotators are typically careless. Our experiments, not surprisingly, found similar carelessness with electronic documents. Carelessness does not usually matter on paper documents: the human reading the annotation will almost subconsciously realise what was really meant. With electronic annotations, carelessness can matter more.

A fifth and final issue is that it is often hard to separate content from commentary. Commentary is used extensively. In our first sub-experiment 31% of the annotations were deletions, and approximately one fifth of these had a commentary to explain the deletion. Of the 59% of annotations that were insertions/changes, 36% of these involved commentary; in all these cases there was also, of course, content (the text to be inserted), so in all these cases commentary was mixed with content. A specific example of the the problems of inter-mixing commentary and content is where an annotation was attached to the anchor 'two-way', and the annotation itself says: replace this either with 'bi-directional' or with 'multi-directional'. This annotation has two pieces of content (the two suggested new terms) interwoven with commentary that makes it clear the two are alternatives. Is any author going to bother to separate these out? (On looking back, we failed to get it quite right.) Can acceptable conventions, e.g. with the use of quotes, help achieve this?

3.7 Some issues from the second sub-experiment

The second sub-experiment, an extract from which appears in Figure 2, raised a number of particular issues.

Firstly the 'data types' marked by contexts were not mutually exclusive. For example in one instance, the second annotation in Figure 2, the same anchor is marked both as *Wei-project* and as *IR-paper* annotations. This necessitates a refinement of our view of attaching data types to annotations. One approach to covering an annotation that has two separate data types is to define new data types that are unions of two or more of the existing data types. However a better approach, we believe, is to recognise that annotations are attached to aggregates. These aggregates are built from items of the atomic data type of the application (e.g. words, characters), together with other aggregates. Thus in our example:

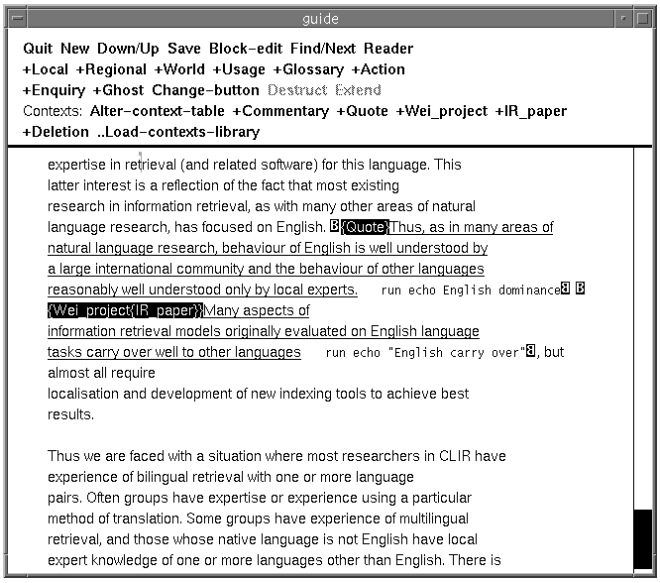

Figure 2: ***part of the second sub-experiment***

1. the aggregate of words, in this case 'Many aspects of information retrieval models originally designed for English language tasks carry over well to other languages', is given the data type *Wei-project*.

2. the data type *IR-paper* is attached to an aggregate that just consists of the aggregate in (1) above. Thus the instance of *Wei-project* is nested within the instance of *IR-paper* — logically a redundant piece of nesting, but it solves the problem of duplicate data types.

The above approach solves many problems, and in particular would cover an instance where, say, a *Wei-project* sentence was a subset of an *IR-paper* paragraph, an instance of data types being nested. Much more difficult is the case where two annotation anchors are not properly nested within one another, e.g. one applies to the first two words of a paragraph, and another applies to the second and third words of the same paragraph. One can see practical needs for such examples, but fortunately they did not arise in our experiment.

Secondly, and coming as a pleasant surprise, there was one extra advantage in annotating the TEX form. The original author had inserted some TEX comments in their paper, covering material that had been omitted due to lack of space. One of these comments was highly relevant to Wei's project, and was therefore marked. The comments were, of course, invisible in a rendered form of the paper.

Thirdly — and unsurprisingly — it was useful to place hypertext links in the association of some annotations, e.g. an association might contain a reference, effected by a hypertext link, to another paper that backs up the same conclusion. Hypertext links and annotation in tandem are more powerful than either in isolation.

4. CONCLUSIONS

Clearly the experiment described in this paper was a small and limited one, and used a hypertext system, Guide, which, given the way the market has developed, is now off-beat. In particular, Guide's facility for integrating authorship (writing) with reading, vital for our experiment, is highly off-beat. Nevertheless it was a happy surprise to us that the three types of hypertext link built into Guide, as instantiated in its three types of hypertext button, all appeared to be good candidates for implementing annotations. The button type we selected, the glossary-button, was used satisfactorily in two sub-experiments, one concerned with edit-annotations and one with enhancement-annotations. Apart from a clumsy user interface, the nitty-gritty detail of the experiments revealed no serious problems in using a hypertext facility to perform annotation. At the more detailed level:

- data types, in our case represented using Guide contexts, add considerable value.

- following on from this, perhaps one of the greatest potential advantages of electronic annotations is that automatic tools can process them, and create further information from them. However a barrier to this is that people are often careless in marking their annotations exactly.

Overall at the practical level we believe the experiment *indicated* that it is possible to offer a combined facility for hypertext linking and for annotation.

We believe that the same applies at the theoretical level. Currently hypertext linking and annotation are treated as largely separate topics, but we believe they could profitably be combined. Indeed they should be regarded as two separate instantiations of a single super-concept, though this requires further work. Moreover we would like to see this generalisation taken further. The purpose of the annotations we have described here is to allow writing while reading. In fact there is another type of annotation, created pro-actively by an agent, which, while the user is writing, suggests documents they should read [11]. The ultimate step in generalisation is to integrate fully the reading of documents with their writing [3].

5. ACKNOWLEDGEMENTS

We are grateful to the University of Exeter for support of this work.

6. REFERENCES

[1] Bernheim Brush, A. J., Bargeron, D., Gupta, A. and Cadiz, J. J., 'Robust annotation positioning in digital documents', *Proc. SIGCHI '01*, Seattle, Wa., pp. 285-292, 2001.

[2] Bouvin, N. O., 'Augmenting the web through open hypermedia', *New Review of Hypermedia and Multimedia*, **8**, pp. 3-25, 2002.

[3] Brown, P. J., 'From information retrieval to hypertext linking', *New Review of Hypermedia and Multimedia*, **8**, pp. 231-255, 2002.

[4] Brown, P. J., 'Turning ideas into products: the Guide system', *Proc. Hypertext 87*, Univ. of North Carolina, pp. 33-40, 1987.

[5] Bush, V., 'As we may think', *Atlantic Monthly*, **1976**, pp. 101-108, 1945.

[6] Conklin, J. and Begeman, M., 'gIBIS: a hypertext tool for exploratory policy discussion', *Proc. of the Conference on Computer Support for Cooperative Work*, Portland, Oregon, pp. 303-331, 1988.

[7] Davis, H. C., Hall, W., Heath, I., Hill, G. J. and Wilkins, R. J., 'Towards an integrated environment with open hypermedia systems', *Proc. ACM Conference on Hypertext: ECHT92*, ACM Press, pp. 181-190, 1992.

[8] DeRose, S. J., 'Expanding the notion of links', *Hypertext '89 Proceedings*, ACM Press, pp. 249-257, 1989.

[9] Kahan, J., Koivunen, M-R., Prud'Hommeaux, E. and Swick, R. R., 'Annotea: an open RDF infrastructure for shared web annotations', *Proc. WWW10 International Conference*, Hong Kong, pp. 623-632, May 2001.

[10] Marshall, C., 'Toward an ecology of hypertext annotation', *Proc. ACM Hypertext '98*, Pittsburgh, Pa., pp. 40-49, 1998.

[11] Rhodes, B. J. and Maes, P., 'Just-in-time information retrieval agents', *IBM Systems Journal*, **39**, 4, pp. 685-704, 2000.

[12] Röscheisen, M., Morgensen, C. and Winograd, T., 'Interactive design for shared World-Wide Web annotations', *Proc. CHI '95*, Denver, Co., volume 2, pp. 328-329, 1995.

[13] Wusteman, J. and Brown, H., 'Acrobat, Mosaic and Guide as vehicles for electronic journals', *The New Review of Information Networking*, **1**, pp. 33-60, 1995.

[14] Zellweger, P., Bouvin, N. O., Jehoej, H. and Mackinlay, J. D., 'Fluid annotations in an open world', *Proc. 12th ACM Hypertext Conference*, pp. 9-18, 2001.