THE MALLEABLE DOCUMENT

WILLIAM W. WADGE
Department of Computer Science, University of Victoria,
P.O. Box 1700, Victoria, B.C., Canada V8W 3P6
E-mail: wwadge@csc.uvic.ca

A malleable document is a document which can be incrementally transformed by an individual reader until it better suits that reader's needs. The transformation can alter content as well as form, and can result in a document which by normal standards would be considered entirely different from the original. We describe how malleable documents can be authored using an extension of HTML, Intensional HTML (IHTML), which supports Web pages and sites that vary over a multidimensional parameter space.

1 Introduction

Ever since the introduction of writing thousands of years ago documents have been considered by their very nature to be fixed and unchangeable — carved in stone, often literally do.

This lack of flexibility is often, for authors, a distinct advantage. Authors (then and now) want law codes and religious texts to be immutable, and contracts, accounts and property records to be incorruptible.

In other contexts, however, this same immutability causes serious problems. A case in point is the use of documents in classroom teaching (the English word *document*, which is descended from the same Greek root as *doctor*, originally meant a *lesson*.) Every teacher knows that elaborate support materials are often more trouble than they are worth, because they make it difficult to adjust the lesson (in real time) to the strengths and weaknesses of each individual class. Problems also arise with documents that are directed towards a large heterogeneous readership. How, for example, can a firm write an instruction manual for a product (such as a car) which is produced in many different versions and sold in many different countries? Ideally, each car sold would be delivered with an instruction manual tailored for that particular customer — say, a francophone who purchased the economy model with stereo and ABS but without air conditioning.

In practice, this is impossible, because there may be thousands of different possible versions of the manual. This is far too many to print and stockpile beforehand, especially considering that only a fraction of them will ever actually be needed.

The advent of the Web has made the problem even more critical. Now even relatively small organizations feel the competitive pressure to create Web sites accessible all over the world.

2 Versioning the Web

In theory, Web pages, being purely electronic, can be updated frequently and provided in customized versions on demand. In practice, however, this is rarely the case. The cost of delivering a new or customized version is near zero, but the maintenance is a real problem.

HTML itself provides very little support for the production and maintenance of multiversion sites. The problem is that different versions of the same site cannot share pages except through links; but then they have to share all the pages accessible from the shared page. Furthermore, pages with analogous layout cannot share the markup which describes the layout.

As a result, authors of multiversion sites are forced to create separate copies by cloning and editing — just like authors of hard-copy documents. And, just like hard-copy authors, they are faced with maintaining the consistency of many separate but parallel pages.

The author and a group of collaborators have recently produced an extension of HTML which makes practical the authoring of multiversion sites which vary according to a number of parameters. Called Intensional HTML [5], it allows the author to create source files for pages and parts of pages which, in general, are generic — valid for a whole family of versions of the component in question. When a request for a particular version of a page is received, the source for that particular version is assembled from the results of specializing the relevant source files for the components. The assembling work is done by a plug-in for the Apache server; the output generated is standard HTML, and thus no special browser software is required.

For example, the image on a page giving operating instructions for a copier might depend on the model parameter, but not on the language parameter; whereas the instructions for loading paper will depend on the language parameter but perhaps not on the model parameter (because they are all loaded the same way). In this situation an IHTML author most provide an image for each model, and a paragraph of loading instructions for each language, but *not* a separate entire page for each combination.

3 Intensional Versioning

IHTML is based on an intensional approach to versioning. The different versions of a document are specified by expressions in a simple algebraic version language. A request for an IHTML page consists of two parts, a conventional URL indicating the name of the page requested, and a version expression indicating which version of the page is requested.

The simplest version expressions consist of a dimension (parameter) identifier and a corresponding value, separated by a colon. For example, the expression `lg:fr` specifies that the value of the `lg` parameter (which might denote the document language) is `fr` (which might denote French). More generally, we can form sums of such expressions, specifying that each of the given dimensions has its respective value. For example,

```
lg:fr+ size:a4 + mod:ml4 + lv:expert
```

might stand for the French language A4 manual for the ML4 model, written at the expert level.

IHTML is, as a markup language, a very modest extension of HTML itself. The most important feature is the intensional link, which allows authors to specify, with a single source tag, whole families of links between different versions of a page.

In its simplest form, an intensional link is syntactically identical to a conventional (extensional) link; for example, if the IHTML source for `home.html` contains the tag

```
<a href="intro.html">Introduction</a>
```

the tag will be used as a template for links between corresponding versions of the home and intro pages. In this way, if the `lg:fr` version is requested, it will contain the link

```
<a href="intro.html<lg:fr>">Introduction</a>
```

which leads to the `lg:fr` version of the intro page.

The IHTML <a> tag has an optional attribute which is used to specify links to a version of the target page that is different from the version of the source page. For example, the intensional link

```
<a href="instr.html" vmod="lg:en">English instructions</a>
```

specifies a family of links between versions of the `home` and `instr` pages. But in each case, the links lead to a version in which the language is English (although all the other parameters remain unchanged).

IHTML also has intensional image and server side include tags which function in an exactly analogous manner.

Finally, IHTML has a kind of intensional case statement which when evaluated selects the text to include based on the particular version to be produced. Here, for example, is the IHTML code that provides links between the French and English versions of a bilingual page:

```
<iselect>
  <case version="lg:fr">
```

```
    <a href=home.html vmod="lg:fr">Version francaise</a>
  </icase>
  <case version="lg:en">
    <a href=home.html vmod="lg:fr">English version</a>
  </icase>
</iselect>
```

This same identical piece of code works for both the French and English versions.

4 Conversational Texts

Originally, Intensional HTML was designed (as we indicated above) to facilitate authoring and maintenance of Web sites that vary in a number of obvious parameters. The first such site was published by Taner Yildirim [4], who also produced the first implementation of IHTML. His site is bilingual (Turkish/English), can be viewed with or without graphics, and with a variety of background patterns and font colors. (His site is currently still available at `http://csr.uvic.ca/~taner/cgi-bin/scan.cgi`.)

As is so often the case in engineering, once a tool is available people realize that it can be put to uses other than those for which it was originally designed. As part of her doctoral dissertation [3], m.c.schraefel described how intensional versioning could be usefully applied to nontechnical documents which at first sight seem monolithic — for which no parameters immediately suggest themselves.

She described how a traditionally structured single essay on the literary theory of *Wuthering Heights* could be presented as a multidimensional document. The reader could then select a version corresponding to their particular interests and point of view.

The possible parameters include the particular character the reader wishes to know about; the particular social question of interest; the depth of treatment; and the amount of documentation. For example, version

```
char:heathcliff+sq:marriage+depth:summary+doc:none
```

would be a very short work focussing on Heathcliff and the question of marriage, with documentation omitted.

Schraefel generalized the approach and proposed the notion of a Conversational Text (or ConText), a document capable of reconfiguring itself to suit a particular reader on the basis of some form of dialogue. By a *malleable document* we mean a simple form of ConText which can at least listen to the reader and recast itself according to user-specified values of its parameters.

Yildirim's first version of IHTML (the only one available to schraefel at the time) was too primitive to support a practical implementation of ConTexts or even ambitious malleable documents. However these ideas contributed directly to the design and implementation (by Gord Brown, Yildirim, schraefel and the author) of the second incarnation of IHTML [1]. This IHTML (the language described above) made it practical to implement nontrivial and relatively sophisticated malleable documents.

5 StretchText in IHTML

Knowledgeable readers will be aware that malleable documents are not a totally new concept. Theodore Nelson anticipated many of our ideas more than thirty years ago, in [2].

Nelson is rightly given credit for hypertext but even this does not really do him justice. He had a much broader view of hypertext than simply pieces of conventional text linked together. This, the conventional notion of hypertext, he called *basic* or *chunk-style* hypertext.

Nelson described one form of hypertext which goes beyond the bare minimum. This is variable-length text, which Nelson called *stretchtext*, where (as above) the length of the document can be varied according to a global parameter. When the parameter of a stretchtext document is increased, increase new text appears (not necessarily only at the beginning or end). Stretchtext, at least a form of it, can easily be implemented with IHTML.

To illustrate how, consider the simplest case of stretchtext, where we want a single extra paragraph to appear — say, the caption of an image. In ordinary hypertext, we can place, just below the picture, a link to a separate page consisting of the paragraph. The problem with this approach is that just before the caption appears, the picture disappears. To see both, we have to toggle back and forth between the two pages.

A better idea is to have the link take us to a different version of the new page, identical to the original except that below the picture we can now see the caption paragraph (instead of the anchored text).

This is easy enough to do in ordinary HTML but is almost never attempted. The main reason is that it requires creating and maintaining a separate copy of the original file. Even with only one caption this is already a bad idea, but if there are dozens of images with captions it becomes completely impractical.

Javascript is not much help either, because uncovering the caption may require reflowing the text below the caption. This is possible with DHTML, but not easy, and fails for many browsers and platforms.

The IHTML solution, by contrast, is straightforward. We use a `c` dimension which records whether or not to show the caption. Below the picture, we have an `iselect` which evaluates to the caption paragraph if the value of `c` is on, and to the link otherwise:

```
<iselect>
   <icase version="c:on">
      <i>This ellipse is a
         circle flattened along the x axis</i>
   </icase>
   <icase>
      <a href=conics.html vmod="c:on">more...</a>
   </icase>
</iselect>
```

We can use the same code for every caption, so that clicking the "more" link makes all the captions appear. Alternatively, we could have a single link at the top that controls all the captions. Naturally, there is no reason to restrict ourselves to captions alone, or to only one level of expansion. If we remove these restrictions, we have implemented something very close to the Nelson's original stretchtext idea.

It should be emphasized that the IHTML solution involves no cloning; there is only one source file to maintain, with no duplicated copy.

6 PlyText

Suppose now we want to control the captions. We can use the same code as above, except that each caption has its own dimension. The captions are initially invisible, as if the document were creased so that they are folded out of sight. Using separate dimensions we can unfold individual captions (hence the name *plytext*). Plytext gives us much finer control over the document than does stretchtext, with its single global dimension.

Plytext takes a more familiar form when we use section headings instead of images, and section bodies instead of captions. The document initially consists of section headings alone, and the reader chooses which sections to unfold and read. This corresponds roughly to the outline view provided by many word processors (the folding sections can be nested).

We can make plytext look even more familiar by using arrowhead images as the anchors for the links to the folded and unfolded versions. We can place them to the left of the section headings, pointing sideways when the body is folded, down when it is unfolded. For example, a reader might initially see

▶ **Introduction**

▶ **Versioning the Web**

▶ **Intensional Versioning**

▶ **StretchText**

After clicking on the third arrow, we see (simplified)

▶ **Introduction**

▶ **Versioning the Web**

▼ **Intensional Versioning**
IHTML is based on an intensional approach to versioning. The different versions of a document specified by expressions in a simple algebraic version language.

▶ **StretchText**

Each sideways pointing arrow is a link to a version of the page in which the corresponding section is unfolded, and the arrow is pointing down; and in this unfolded version, the downward pointing arrow is linked backed to the folded version:

▼ **Ellipses**
An ellipse is like a circle only flattened along one axis.

▶ **Hyperbolas**

▼ **Parabolas**
A parabola is formed when the plane intersecting the cone is at a strictly steeper angle than the side of the cone.

▼ **Ellipses**
An ellipse is like a circle only flattened along one axis.

▼ **Hyperbolas**
The next kinds of conic section we will examine is the hyperbola. A hyperbola is formed when

▼ **Parabolas**
A parabola is formed when the plane inter-

Of course this picture is familiar to anyone who has used a recent version of Microsoft's Windows or Apple's MacIntosh operating system, in the list view of directories. But it is rarely seen on the Web.

7 Documents sans Frontières

It is not hard to imagine many useful variations on the constructs like plytext. We could allow the user to specify a brief summary text to be displayed when the folded text is not visible. We could arrange that at most one section is unfolded, so that unfolding another section folds the one which is currently open (this avoids making the document too long). A more refined arrangement would allow only one open section at each level, so that the user can drill down to subsections and sub-subsections but still read them in context. Also, we can combine stretchtext and plytext: we could have a global control on the length of the texts being folded and unfolded. These possibilities are relatively easy to program in IHTML.

One of the more interesting ideas is allowing links which make text disappear completely. For example, we could have an **X** icon next to the arrow icon for each section. Clicking the **X** (i.e. following the link anchored to it) removes all trace of the entire section. In other words, the **X** links us to a version of the page in which the folding text, the summary text, the arrow icon and the **X** itself have all vanished.

This would be useful, for example, in preparing a lecture. If the lecture notes were already on-line as malleable documents, we could click away those sections (say, on hyperbolas) which we want to skip in the lecture. In fact, if we use video projection, we can erase them in real time — just as we can remove notes for topics we have finished covering. With malleable documents, we can begin to blur the distinction between browsing and editing.

Once we have the ability to make text selectively appear, expand, shrink, and disappear, we have also begun to erase the boundaries between documents themselves. For example, a professor's home page might include a plytext list of her research students. The browser could expand one particular student's entry and kill off all the other information about the professor; then drill down to the student's thesis, extract one of the references then focus in on that author's home page.

What we have done, then, is gradually transformed the professor's page to the author's page. Of course we can already follow hypertext links, but that takes us from one document to another in a series of abrupt leaps. With malleable documents, we can morph one into another in small steps, in which each new piece of information appears in a context which explains its relevance.

It should also be clear that in so doing we have almost completely detached documents from specific files. At each stage in the transformation, new files contribute to the content being viewed, while others' content disappears.

8 Some Practical Considerations

As we already indicated, nothing we have described is absolutely *impossible* without IHTML. If we are willing to write cgi scripts, all of the effects above can be achieved using standard HTML and standard servers. The real question is, how practical is it do implement plytext and the rest using IHTML?

In fact, it is fairly straightforward. To begin with, the author does not have to do any programming at all, no cgi scripts and not even Javascript. There is no source cloning and no maintenance of multiple files. Beyond the HTML for the actual content of the document, the author supplies only IHTML markup, mainly `iselect` statements.

Admittedly the intensional markup for the more elaborate constructs is complex. Also, for an elaborately structured document, the author has to generate many extra lines of markup — a dozen or more per intensional construct. It is therefore not practical to expect to produce any but the simplest malleable documents by hand.

Fortunately, the author has found a simple way to allow even unsophisticated authors to write or convert documents. He has produced a macro package which generates the required markup. Currently it uses `troff`'s macro system, which is included with almost every UNIX system (to expand the macros, we run `troff` with formatting disabled and with ASCII output).

The macro definitions themselves are intricate (details can be found in [6]) but their use is simplicity itself. Suppose that an author has an existing standard HTML document and they want to convert it to plytext. Each plytext section needs three macro calls of three characters each. For example, if the original source includes

```
....
<h3>ELLIPSES</h3>
<p>
An ellipse is like a circle, only flattened
along one axis.
......
```

then the author adds three lines as follows:

```
....
<h3>
.SC
ELLIPSES</h3>
<p>
.SD
An ellipse is like a circle, only flattened
```

```
along one axis.
.SE
......
```

A document the size of this paper can be converted in a matter of minutes. In fact, if the original HTML uses the `<h1>`, `<h2>`, `<h3>`, etc. tags in a disciplined manner, the whole procedure can be easily automated.

In practice the effectiveness of plytext and related constructs is very dependent on the connection speed and the speed with which the browser renders each document. Current browsers were designed based on the implicit assumption that the page jumped to will normally look nothing like the current page. The current display is erased before the new one is rendered, and the window may be empty for some time.

When we really are jumping to a totally different page, the empty window helps the eye adjust to the new image. However, the whole point of malleable documents is that the new display is closely related to the old one. A delay of more than a fraction of a second makes it much more difficult to perceive the relationship between the old and the new pages.

Fortunately, with high-speed connections and the latest processors, the transition is usually fast enough to give the desired animation effect. Eventually, browsers could appear which take advantage of parallelism between the before and after displays.

9 Acknowledgement

This was made possible by generous support from the Canadian Natural Sciences and Engineering Research Council (NSERC).

10 References

[1] G. Brown, *Intensional HTML, a Practical Approach*, MSc Thesis (Computer Science), University of Victoria, Canada (1998).
[2] T. Nelson, *Dream Machines/Computer Lib*, Microsoft Press (1987).
[3] m.c.schraefel, *Talking with Antigone*, PhD Dissertation (interdisciplinary), University of Victoria, Canada (1997).
[4] T. Yildirim, *Intensional HTML*, MSc Thesis (Computer Science), University of Victoria, Canada (1997).
[5] W. Wadge, G. Brown, m. c. schraefel, T. Yildirim, "Intensional HTML", *PODDP '98*, Springer Verlag (LNCS 1481), pp. 128–139 (1998).
[6] W. Wadge, "Putting the Hyper back in Hypertext". In *Intensional Programming II*, World-Scientific, Singapore (1999).