

Fluid Annotations in an Open World

Polle T. Zellweger, Niels Olof Bouvin, Henning Jehøj, and Jock D. Mackinlay

Department of Computer Science, University of Aarhus, Åbogade 34, 8200 Århus N, Denmark

E-mail: {zellweger, n.o.bouvin, jehoej, mackinlay}@daimi.au.dk

ABSTRACT

Fluid Documents use animated typographical changes to provide a novel and appealing user experience for hypertext browsing and for viewing document annotations in context. This paper describes an effort to broaden the utility of Fluid Documents by using the open hypermedia Arakne Environment to layer fluid annotations and links on top of arbitrary HTML pages on the World Wide Web. Changes to both Fluid Documents and Arakne are required.

KEYWORDS: Arakne, Open Hypermedia, Fluid Documents, Web augmentation with open hypermedia.

INTRODUCTION

Fluid Documents use animated typographical changes to provide a novel and appealing user experience for hypertext browsing and for viewing document annotations in context. The Fluid Documents approach has been used to advantage in a variety of application domains, including hypertext [35], electronic books [9], spreadsheets [20], and avant-garde fiction [12]. Furthermore, user studies have confirmed that its basic tenets of animation and contextual views show considerable promise [37].

However, all of the previous trials have been monolithic research prototypes that could rely upon considerable control of the documents in question, both through simpler document contents and through full control of the document layout and presentation processes. In addition, the previous prototypes did not permit readers to add their own annotations. Readers used the annotation mechanisms to view optional supporting material, such as descriptions of link destinations, footnotes, references, derivations of formulas, definitions, figures, or simply more detail.

We report here on an effort to broaden the applicability of Fluid Documents to documents on the World Wide Web. Our approach utilizes the open hypermedia Arakne Environment [2]. As a result, it also provides readers the new ability to add their own fluid annotations. Finally, we have taken the opportunity to incorporate a few design changes motivated by the results of a user study of Fluid Documents.

We considered the following research questions:

- How well does open hypermedia (in particular systems aimed at Web augmentation such as the Arakne Environment) handle the new requirements of specifying dynamic and typographical behavior?
- How do fluid concepts, fluid graphical styles, and fluid algorithms translate to the more freewheeling and chaotic Web context?
- What changes must occur to fluid structures and the fluid user experience when readers can add their own annotations rather than simply viewing annotations made by others?

We begin our discussion by reviewing the capabilities of Fluid Documents and of the Arakne Environment. We then relate our experiences in recasting Fluid Documents as an open hypermedia application. Related work and a discussion of open issues conclude the paper.

FLUID DOCUMENTS

Fluid Documents alter their layout, typography, and other graphical characteristics in order to present supporting material in the context of the primary material that is annotated. The document surface acts as a canvas in which traditionally static elements move and change smoothly to make room for additional information.

Fluid Documents use the following four steps to give the reader access to supporting information:

Visual cue. A textual or graphical cue is placed near the annotated primary material (referred to as the *anchor*) to indicate the existence of supporting material (referred to as the *gloss*). For example, the previous Fluid Documents prototypes that implemented electronic book and hypertext applications underlined the anchor as a visual cue to indicate the presence of a gloss.

Interaction. The reader interacts with the cue or the anchor to request the presentation of the gloss. The previous Fluid Documents prototypes used mouse rollover as a lightweight interaction mechanism.

Accommodation. The system adjusts the appearance of the primary material and the gloss to allow the reader to focus on the gloss while maintaining the context of the primary material. The primary material may become less salient (decrease its size, move away from the center, fade to a lighter shade), while the gloss becomes more salient (in-

creases its size, moves to a prominent location, displays in a clearly distinguished manner).

Animated transition. The primary and supporting material smoothly animate to their new states so that the reader's visual system can easily track the changes to the document.

A variety of techniques for making space for annotations was implemented in the previous Fluid Documents prototypes, including interline compression, overlay, and margin callout [35][36]. In the interline compression technique, the interline spacing of the text gradually shrinks as the gloss grows to readable size between the anchor and the following line of text (see Figure 1). In the overlay technique, the lines of primary text below the anchor fade to a lighter color as the gloss grows overlaid on them. In the margin callout technique, an animated callout line guides the reader's eyes from the anchor to the nearest margin, where the gloss grows in existing white space. This variety served both to explore the breadth and utility of the fluid concepts and to test the generality of our implementations.

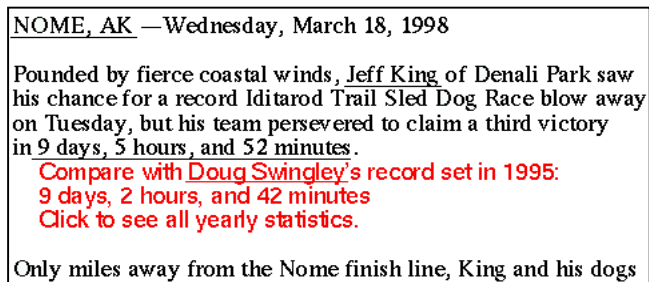


Figure 1 - Fluid links: interline compression. The expanded gloss (“Compare... statistics.”) contains a nested link. Glosses use a red sans-serif font to distinguish them from the black serif font in the primary text.

All of these techniques allow glosses to be viewed in a transitory way, or selected glosses can be “frozen” to allow them to stay in place while the reader continues to read elsewhere on the page.

The Value of Fluid Documents

The Fluid Documents features work together to permit readers to shift their attention smoothly and quickly from the primary material to glosses and back again as desired.

The typographical adjustments provided by Fluid Documents allow readers to view glosses with maximal context, minimizing occlusion or removal of the primary material. Readers can also freeze one or more glosses in view to compare their contents, to arrange the page contents to suit their current interests, or to mark glosses for later attention.

Animation helps readers easily process the changes to a page required to display a gloss. If glosses are placed nearby, animation can be used to move surrounding text out of the way in a visually clear way. On the other hand, if glosses are distant, animation can be used to guide the reader's attention to and from them, so that the gloss material is not inadvertently missed.

Studies of collaborative editing have shown that placing commentary near its referent improves the ability of reviewers to add comments and of later editors to process the comments [34]. Studies of Fluid Documents showed that users can process moving text even in a serious reading situation, and also indicated that providing glosses close to their anchors is beneficial [37]. Furthermore, the fluid studies showed that distant glosses presented without animation were frequently not seen.

Finally, Fluid Documents support use for fluid links. Glosses can be placed at link sources to support readers in choosing among links and understanding the structure of a hypertext. Mousing over a link source causes the presentation of the gloss. These glosses are used to blur the boundary between source and destination: computed glosses that automatically retrieve or construct gloss material from the destination supply a “bring from” approach to hypertext, while multi-way links and nested glosses allow readers to skip through intermediate nodes while still attending to their original source context.

Study Implications

The fluid studies indicated that opening glosses via light-weight mouse rollover was problematic, at least for novice users of Fluid Documents, despite requiring a short stationary dwell time for activation. The study logs included many gloss presentations that were too brief for subjects to read. In some cases, it appeared that gloss activation occurred as subjects were carefully positioning the mouse to follow links. In other cases, it appeared that subjects were moving the mouse as they read and accidentally placing the cursor on gloss anchors. Thus future interfaces for Fluid Documents should likely avoid mouse rollover activation by default. Rather, readers can be allowed to adjust the responsiveness of activation methods so as to match their reading style and fluid expertise.

The fluid studies also showed that some readers, while valuing the functionality of fluid links and annotations, found animated text (both movement and animation) quite disturbing. For such readers, we now wish to allow sufficient control to choose alternative presentations.

OPEN HYPERMEDIA AND THE WEB

The field of open hypermedia has since 1989 [24][27] worked on providing users with advanced hypermedia functionality by integrating third-party applications. By enabling commonly used tools with hypermedia, users gain the advantages of hypermedia structuring without having to abandon their tools of choice. Notable open hypermedia systems include Microcosm [18], HyperDisco [33], Devise Hypermedia (DHM) [14], Chimera [1], and HOSS [25]. These systems extend existing applications so that users can create links and other hypermedia structures between e.g. word-processing documents and spreadsheets. Given this extension of existing systems, it should come as no surprise that the open hypermedia community has also developed systems to extend the hypermedia functionality of the Web. Systems such as DLS [6][7], HyperWave [23],

DHM/WWW [13], Webwise [16], and the Arakne Environment [2] provide users with the ability to create hypermedia structures on top of existing HTML pages on the Web. This can be accomplished in various ways; e.g. by using a proxy (DLS and DHMProxy [16]), which modifies the Web pages en route to the Web browser, or by the integration of a Web browser, so that the Web page is modified inside the user's Web browser (Webwise and the Arakne Environment).

IMPLEMENTATION OF OPEN FLUID

This section describes the rationale behind the Arakne Environment and the Open Fluid prototype, as well as some of the architectural changes involved in providing fluid annotations on the World Wide Web.

The Arakne Environment

The Arakne Environment has been described in [2][3]. It is a collaborative open hypermedia system written in Java aimed at augmenting the Web with externally stored hypermedia structures, such as bi-directional multi-headed links, guided tours, or spatial hypermedia. The different hypermedia structures are handled by an open set of specialized hypermedia tools, known as 'views'. The Arakne Environment relies on the Construct servers, whose overall architecture is based on the work by the Open Hypermedia Systems Working Group (see <http://www.ohswg.org/>).

A general overview of the original Arakne architecture can be seen in Figure 2. In the context of the Open Fluid experiment, the integration with the Internet Explorer should be noted. The Internet Explorer is a COM (Component Object Model) component, and has a rich API. This API is accessed in the Arakne Environment by the Browser class, which forms a general interface to a Web browser. Earlier versions of the Arakne Environment used the Browser class to insert links and simple annotations into Web pages by modifying the Document Object Model [11].

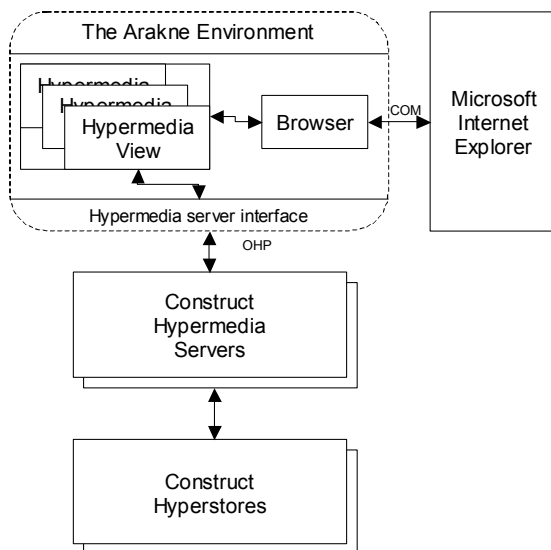


Figure 2- The original Arakne architecture.

While this architecture is quite sufficient for static modifications, it was found lacking for the demands of Open Fluid annotations, which rely on smooth animations of typographical changes. The complications involved with handling COM communication in Java (including but not limited to marshalling arguments from Java to C and back again) meant that we wished to keep communication between Arakne and the Internet Explorer as low as possible. On the other hand, the demands of fluid animations required many modifications of the appearance of anchors and glosses. To address both points, we resolved to move the code handling the Web pages into the Internet Explorer, as seen in Figure 3, where a new component, the Render Engine, has been introduced. The purpose of the Render Engine is to handle all modification of Web pages and animations of fluid annotations. It is a DLL written in C++, which communicates with the Arakne Environment through the Java Native Interface and COM. By having this timing-critical component close to the Web browser, performance was enhanced and the Browser class generalized, as the Internet Explorer specifics now reside in the Render Engine.

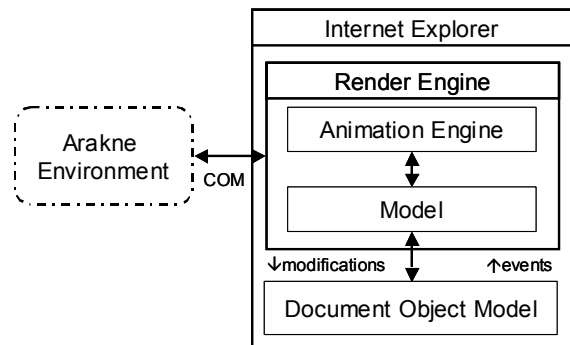


Figure 3 - The new Arakne architecture for Open Fluid.

Annotations in the Arakne Environment

The Arakne Environment supported annotations prior to the work described herein. Annotations could be pre- or post-fixed to the anchor, replace the anchor text with the annotation, or provide a pop-up text in the form of a Tooltip. This approach, while simple, had certain disadvantages. The insertion of the annotation directly into the Web page disrupted the layout of the page, especially as the annotation was shown in a single different color and style, chosen by the implementers to be likely to be distinct from the original text. The user could either activate all annotations in a context or choose not to display any annotations; there was no middle ground. The Tooltip annotations were not as intrusive as the other annotation types, however they could only be displayed one at a time.

The new fluid annotations eliminate these problems by being discreet while not in use, and by having state (such as 'open'), so that a reader can leave several open at the same time.

Despite these differences, the data model behind the annotations has not changed much. The Arakne Environment

models annotations as a special case of an anchor. An anchor contains (among other things) a node specification (the document that the anchor appears in), a location specification (the area in the document associated with the anchor), and a presentation specification (how the annotation should be presented). The annotation text itself is an attribute along with other key/value pairs (such as creator, time stamp etc.).

While the OHSWG data model allows anchors to refer to any hypermedia object, the Arakne Environment has in fact not previously supported other than anchors (and hence ultimately links) into proper documents with URLs. In order to handle glosses within glosses, this limitation has been lifted, as anchors now are permitted to refer inside anchors.

The Render Engine

The purpose of the Render Engine (shown in Figure 3) is to build a model of the displayed Web pages, modify them as needed, handle fluid animations, maintain annotation state, and inform the Arakne Environment about the state of the Internet Explorer. These tasks are described below.

Building a model: The Render Engine needs to maintain a model of the Web pages being displayed by the Web browser. This model is used to resolve LocSpecs [10][17][28], to maintain awareness of the configuration of framesets, to handle events, and to handle the modifications needed by links and glosses. This model will also at a future point be used to automatically find available white space on a Web page for glosses.

Modification of Web pages: An intrinsic part of Web augmentation with open hypermedia is the modification of Web pages, e.g. the insertion of externally stored links, annotations etc. The Arakne Environment does this at the client, in this case through the Render Engine. When a Web page is loaded, the Render Engine builds a model of the document. Given an anchor, which contains both a location specification and a presentation specification (described below), the Render Engine finds the location on the Web page and modifies it according to the presentation specification. If the anchor contains an annotation, the annotation text is also inserted into the page, usually initially hidden (though this can be specified by the annotator). The inserted elements are given unique identifiers, so that the Render Engine can later refer to them.

Fluid animations: If an annotation is actuated, the animation associated with it takes place. The animation is accomplished by successively and rapidly modifying the style of the annotation text. Currently the Render Engine supports a “revealed” animation style for the appearance of glosses: the gloss appears one line at a time, first line first.

The Render Engine must also make room for the gloss to appear. This is currently accomplished via a “push down” strategy, in which the text below the gloss's location is moved downward. We are currently working on two additional strategies: interline compression, which gradually

reduces the space between lines in the neighboring paragraphs, and fading, which gradually fades the color of the text below an overlaid gloss.

To ensure optimal performance these updates are handled directly by the Render Engine. An issue for future work would be to allow an expert annotator to specify an animation routine in JavaScript, which would then be executed on demand by the Render Engine.

Annotation state: Annotations can be opened or closed by the reader. A Web page may thus at a given time contain a number of open annotations. When the user returns to a page (by using the 'back' button or by other means), the state of the annotations is restored. Currently this state is only held per browsing session, but it could easily be stored between sessions.

Capturing events: The Render Engine, among its other roles, forms the connection between the Internet Explorer and the Arakne Environment. This includes alerting Arakne to the changes in the state of the Internet Explorer (e.g. which Web page is currently displayed), and the actions taken by the user (e.g. the creation of a gloss or a link). The Render Engine also listens to the user-generated mouse events. These events include mouse over, mouse down, mouse up, mouse drag, right and left clicking, all in combination with modifier keys (shift, alt and control). Furthermore, as the Render Engine is also privy to the x,y coordinates of the cursor, it could potentially recognize mouse gestures, though there are currently no plans to support this.

Specifying the Presentation Specification

The Dexter model [19] pioneered the presentation specification (PSpec), an opaque value (in the case of Dexter) specifying how a component should be presented. In the OHSWG data model, all hypermedia objects have presentation specifiers. PSpecs are objects in their own right (i.e. they are not just merely an attribute of a hypermedia object) and have (globally) unique IDs. In the Dexter model, presentations formed the “glue” between the run-time layer and the within-component layer. In general, a PSpec consists of a set of properties. In the context of fluid annotations, the PSpec consists of:

- Presentation of the anchor text
- Presentation of the gloss
- Placement (where should the gloss appear?)
- Actuation (when should the gloss appear?)
- Animation (how should the gloss appear?)

Presentation: The user must be made aware of the existence of annotations. A modern Web browser provides a rich set of typographical possibilities through Cascading Style Sheets (CSS) [8]. The Open Fluid prototype allows the user to specify the layout of the anchor and annotation text, using the primitives provided by CSS. The purpose of these different styles is to allow the reader to differentiate between original material (including original links) and external links and glosses. By the nature of CSS, anchors

and annotations inherit the style of the surrounding text, unless otherwise specified. This makes it easy to create annotations that blend well with the rest of the document. On the other hand, care must be taken to highlight the existence of annotations.

Placement: The sophisticated Fluid Documents research prototypes featured a cooperative negotiation between the primary material and the gloss in order to determine a suitable layout [9]. This is not yet supported in the Open Fluid prototype.

Actuation: In order to read the gloss, the reader must activate it in some fashion. As previously mentioned, the Render Engine is capable of capturing many different mouse events from the Internet Explorer. This allows the annotator to specify a different kind of interaction with glosses than with links. The annotator can also specify that the gloss should be opened as soon as the Web page is loaded.

Animation: Finally, the annotator must decide on what animation to use. The set of animation styles is currently fixed, though it may, as previously mentioned, be opened in the future.

An example of a PSpec can be seen in Figure 4.

```
<PSPEC id="daimi.5.979147242" type="PSPEC">
<PROPERTIES>
  <PROPERTY name="name">
    <VALUESET> <VALUE>warning</VALUE> </VALUESET>
  </PROPERTY>
  <PROPERTY name="anchor style" type="css">
    <VALUESET>
      <VALUE>color: #FFFFFF; background: #CC0000;</VALUE>
    </VALUESET>
  </PROPERTY>
  <PROPERTY name="gloss style" type="css">
    <VALUESET>
      <VALUE>font-size: 0.8em; border: 2px dotted;</VALUE>
    </VALUESET>
  </PROPERTY>
  <PROPERTY name="animation">
    <VALUESET>
      <VALUE name="primary">pushdown</VALUE>
      <VALUE name="gloss">revealed</VALUE>
    </VALUESET>
  </PROPERTY>
  <PROPERTY name="placement">
    <VALUESET> <VALUE>inline</VALUE> </VALUESET>
  </PROPERTY>
  <PROPERTY name="actuate">
    <VALUESET>
      <VALUE name="open">Shift&MouseEnter</VALUE>
      <VALUE name="close">Shift&MouseLeave</VALUE>
    </VALUESET>
  </PROPERTY>
  <PROPERTY name="initial state">
    <VALUESET> <VALUE>closed</VALUE> </VALUESET>
  </PROPERTY>
</PROPERTIES>
</PSPEC>
```

Figure 4 - An example of a PSpec in OHIF format. This is the PSpec used for the warning gloss in Figure 6.

Authoring Fluid Annotations

The Arakne Environment is a collaborative hypermedia system. Users may write annotations for their own use, or for a wider audience. Annotations can either be shared over the Construct servers, or exchanged in the Open Hypermedia Interchange Format (OHIF) [15].

The authoring of fluid annotations is a three-step process consisting of selecting, writing, and styling. First, the annotator must designate the anchor for the gloss, which is done by right clicking on an appropriate text selection in the Internet Explorer, and selecting "Create Gloss" in the context menu. It should be noted that text in an open gloss is a valid target for anchoring, so that annotators may add further glosses or links (either sources or destinations) to gloss contents. Second, the annotator writes the annotation text, in HTML if desired, in a separate Arakne window. Third, the style (or PSpec) of the annotation must be specified. As PSpecs are separate unique objects, an annotator will, more often than not, merely reuse an already existing PSpec, though he or she is of course free to create as many PSpecs as desired.

Reading Fluid Annotations

Once the fluid annotations have been created, other users of the Arakne Environment can access it. Hypermedia structures, such as links or glosses, are grouped in contexts, and opening a context applies the hypermedia structures within to the browsed Web pages. Contexts may be retrieved from a hypermedia server, or saved as OHIF files and emailed to other users or placed on a Web server.

A reader may well wish to differentiate, not only between original and added content and structure, but also between different authors and annotators. This can be accomplished in an ad hoc fashion by associating a particular style, e.g. a background color, with glosses by one particular annotator, and another with a different annotator.

The reader can open or close annotations as he or she sees fit. If the reader at a later point returns to a Web page where glosses had been left open, their state will be reinstated.

Using Fluid Annotations

We now present several practical uses of the current Open Fluid system.

Glosses as a personal information portal: Our first scenario, shown in Figure 5, focuses on a reader who visits the main U.S. CNN page regularly for news and information. She often uses the CNN site to view the weather for her hometown and several places where other family members live, to check for schedules and results for favorite sports, and to learn about new films. Each of these interests require her to navigate several levels more deeply into the CNN site. While she is checking the results from the major league teams, she would also like to check schedules and results from the local soccer club where her children play, and after she reads reviews of new films, she would like to find out when and where they are showing in her area.

This information is not available on the CNN site. Although she could create conventional browser-based bookmarks to each of these locations, both within the CNN site and outside it, it is convenient for her to place these frequently-used links in the context that she thinks of as her information portal. She thus places her links inside personal glosses on the existing weather, sports, and entertainment category links on the main CNN page.

The CNN pages change many times each day, and they use rich formatting and animated content. Nevertheless, the main page has a stylized structure that maintains the content of the category links at the left of the page, so these glosses can be used over many months or even years. In contrast, Figure 5 also shows a simpler gloss on the caption of the lead figure of that issue. This gloss, added to allow appearance comparison with the other glosses, will only work temporarily; it will be orphaned as soon as a new issue is posted with different figures and text. The current version of the Render Engine relies on the context around an anchor for location. Future versions should employ a more solid anchor location strategy and should gracefully handle lost anchors [4][10][28].

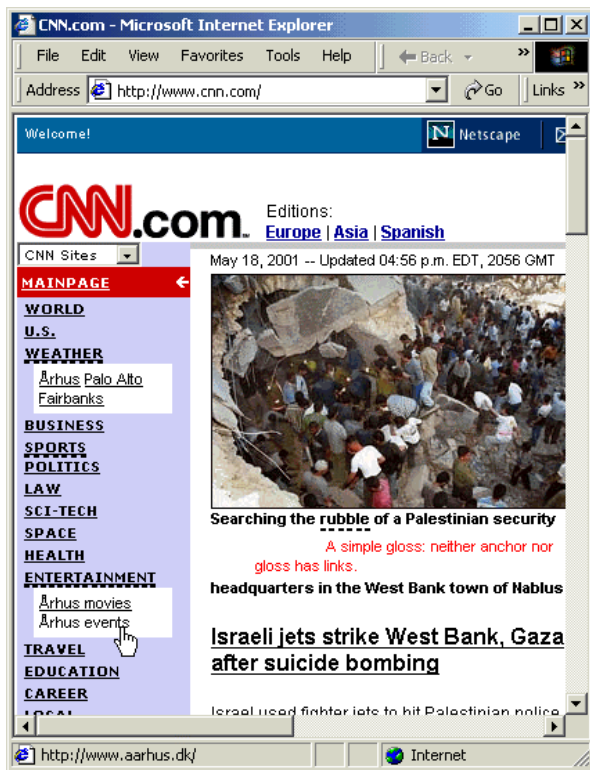


Figure 5 - Fluid annotations on the CNN Web page. The page shows four glosses, indicated by dashed underlines. The reader has added three glosses to existing links in the news category section on the left of the page (two of these are shown open). These glosses contain personal links to allow quick access to information of interest in that category.

A dashed underline was used for the gloss anchor appearance to suggest the availability of more information without looking exactly like a traditional link marker. This appearance also composes well with the conventional link underline. Because the bookmark glosses were placed on link anchors, they were specified with an actuation of “shift + mouse-enter”, to allow left click to continue to follow the underlying WEATHER etc. links. The Internet Explorer context menu, available via a right mouse click, provides a consistent method of operating on glosses and also shows what the chosen accelerators are for opening or closing them (e.g., mouse rollover, shift-left click, etc.).

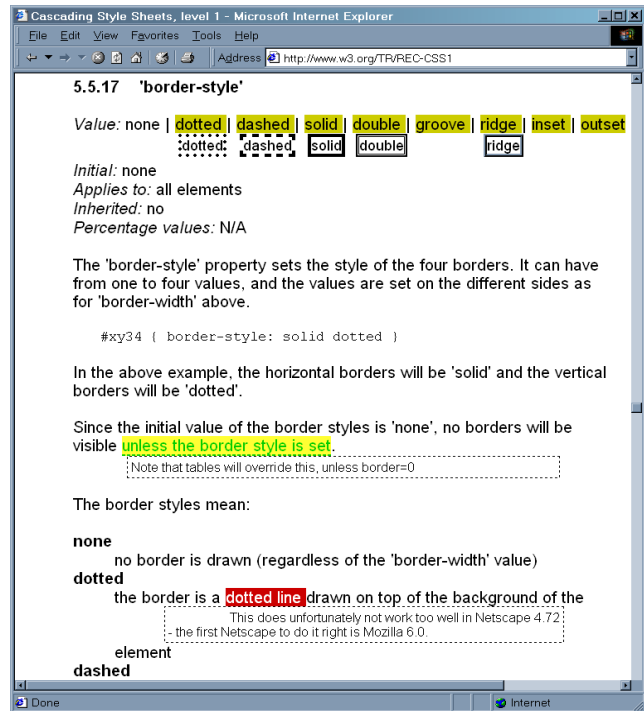


Figure 6 - Shared annotations on the W3C Cascading Style Sheet 1 web page. Using three different PSpecs, annotators have created glosses with distinct anchor highlights to indicate examples (e.g., 8 separate glosses near the top of the page that show formatting results; 5 are open for comparison), comments (e.g., "unless the border style is set" at mid-page), and warnings (e.g., "dotted line" near the bottom of the page).

Glosses as typed annotations: In this scenario, the annotators are members of a group of Web designers who rely on W3C standard documents on a daily basis. This example, shown in Figure 6, uses the Cascading Style Sheet 1 standard specification. The level of compliance with this standard specification varies tremendously across different Web browsers, so the annotators have annotated the standard specification with warnings and comments on the level of compliance – both for their own use, but also for colleagues. Furthermore, the annotators have sprinkled the standard specification with small examples of how formatting is affected by particular commands. The annotators have used a different shared PSpec for each type of annota-

tion: warning, comment, and example. Not only do their distinct anchor appearances mark their roles within the document, but they also support filtering on the different types.

Glosses as an adjunct to open hypermedia links: Our experience suggests that a user who inserts an open hypermedia link into a public document may often find it valuable to insert an explanatory gloss along with it. Since the existence of the link, the contents of its destination, and the rationale of their relationship were not envisioned or intended by the original author of the document, the document content around the anchor may be less helpful than usual to readers in making effective browsing decisions. Open Fluid glosses are better than pop-up annotations (such as Arakne previously offered) for this purpose because they can contain additional links and they can be viewed more permanently in the page if desired.

Glosses in the foreground: In order to promote engagement with the primary material first and foremost, earlier fluid glosses were placed in the page in a relatively subtle way (colored underline only, while the anchor text itself retained its original color). Readers could later open and freeze glosses to focus on the gloss contents. This usage is similar in concept to conventional footnotes: the footnote symbol is fairly unobtrusive, allowing the reader to ignore it if desired. However, in another common annotation style on hardcopy pages, readers add a handwritten annotation in big bold letters. Here, in a sense, the importance of the annotation makes it *primary* for the reader rather than secondary – the annotation is the *first* thing the reader wants to notice about this page in future.

Several features of Open Fluid allow glosses to play this second kind of role also (although handwritten glosses are not yet supported). First, standard PSpec style machinery can be used to make the gloss contents appear in a large font with brightly colored text and/or background. Second, the gloss can be specified to be initially open on arrival at the page; thereafter the reader can control it as usual. Finally, if desired, the gloss's anchor can be specified to have a smaller font or a lighter color, so as to make it less prominent.

RELATED WORK

ComMentor allowed readers to add and share annotations on sequences of characters in HTML Web pages [30][31]. An inserted, customizable, character-sized icon acts as both an annotation cue and a link to the annotation text. Annotation content can be shown either as a separate page, in which case links or further annotations can be included and activated, or in context as a temporary pop-up window while the middle mouse button is depressed. In contrast, Open Fluid's dynamic page manipulation allows multiple glosses to be open at once in context without occluding the primary material. Open Fluid's glosses, which can also contain links and further glosses, offer a faster and smoother user experience because they can be used directly from within the context of the original page.

DLS provided open hypermedia links on HTML Web

pages [7][18]. To distinguish between inserted and original links, DLS offered a form for the user to select how added links should be presented in a document. Open Fluid extends this notion, allowing readers to use and/or define a collection of CSS-based PSpecs to distinguish inserted glosses from existing links and ordinary content. These PSpecs can also be used to control gloss appearance for other reader purposes, such as categorizing gloss types or bringing glosses to the foreground.

As mentioned earlier, the Webvise system [16] and the related Arakne Environment [2] included the possibility for collaborative annotations on sequences of characters as well as new links directly in an HTML Web page. Open Fluid is a direct extension of Arakne that supports animated contextual glosses with rich appearance control.

Microsoft Office Web Discussions [5] allows readers to add shared comments to selected paragraphs (chosen by the author) or an entire HTML document on the Web. Readers use a small cue icon inserted at the end of an annotated paragraph to expand or collapse its annotations (shown with a stylized post-it appearance) in context within the page, pushing later paragraphs downward. Multiple annotations can be expanded at once for comparison. Although Web Discussions has been successfully used in a large community of users to comment collaboratively on design specifications, it was not intended as a full open hypermedia system. Unlike Open Fluid, readers cannot insert links in an annotation or in the body of the document. Since annotations refer to an entire paragraph, the system cannot be used to annotate existing links. Furthermore, there is no animation to help readers assimilate changes to pages as annotations are expanded or collapsed in context.

iMarkup is a commercial Internet Explorer plug-in that allows readers to add and share annotations with widely varying appearances to HTML Web pages [21]. Handwritten strokes and "post-it notes" of many styles can be overlaid on the page and can be shrunk (without animation) to minimize occlusion. Sequences of characters can also be highlighted and given hidden textual annotations to be shown later in a popup ToolTip window. Open Fluid allows a more limited form of foreground annotation via initially open glosses. However, Open Fluid glosses can contain links, do not occlude material on the page, and offer opening/closing animations.

WEBTOUR [32] allows readers to create active tours through HTML Web pages that include dynamic mouse gestures and handwriting, synchronized audio or video playback, and link traversals among pages. Although dynamic behavior flows over the page, the typography of the page itself does not participate in this dynamic behavior, as in Open Fluid.

Although not Web-based, the XLibris prototype digital reading appliance [29] provides rich kinds of stylus-based inking and highlighting that approach the qualities of paper-based annotation [26][22]. Open Fluid comes closer to its desirable qualities than earlier Fluid prototypes because readers can annotate existing pages, it supports a wide variety of anchor and gloss appearances, and glosses

can now appear in the foreground of the page if desired.

DISCUSSION

In this paper, we have described how we used the Arakne Environment to bring fluid annotations to existing HTML pages on the World Wide Web. Open Fluid allows readers to adjust Web pages to emphasize material that is appropriate for their tasks. Animated transitions make it easier for readers to perceive the changes to pages, which can be extensive when annotations are opened near their associated primary material.

We now return to the questions that motivated this research.

How well does open hypermedia (in particular systems aimed at Web augmentation such as the Arakne Environment) handle the new requirements of specifying dynamic and typographical behavior?

The development of Open Fluid showed that fluid annotations fit well within open hypermedia and the OHSWG data model.

Practically speaking, the implementation has been successful. Glosses can successfully be placed on HTML pages containing frames, tables and other complex HTML and appearance, such as animated GIFs or Java programs. Generally speaking, gloss opening and closing animations are responsive and smooth.

The Arakne Environment required modifications in several areas. Most notable is the introduction of the Render Engine. Whereas previous versions of Arakne relied on a relatively simple integration of a Web browser (currently the Internet Explorer), the fluid demand for sophisticated annotation markup, animation and state required a new component, the Render Engine. This component builds a model of a Web page and uses it to locate and markup anchors, perform animations, and handle events. By moving this animation and markup component into the Internet Explorer, the overall generality of the Arakne Environment has been improved, as all Internet Explorer-specific code now resides solely in the Render Engine. If another Web browser were to provide functionality similar to the Internet Explorer, another special purpose Render Engine can be written for it without affecting the central Arakne classes. Currently, the Render Engine maintains gloss state (open/closed) during a browsing session. This state is currently transient, but could easily be made persistent.

Could Open Fluid be adapted for other types of documents than HTML? In principle, certainly yes; however this depends on the availability of a suitable API for the manipulation of document presentation. Applications without this kind of support would be more challenging, though an approach focusing on techniques such as margin notes might be used.

Another fluid requirement was the enhancement of the presentation specification. The PSpec of a fluid annotation extends along several axes, specifying the markup of the anchor, the markup of the gloss text, animation, etc. While

sophisticated PSpecs are not new, an interesting feature of the PSpecs employed in the Arakne Environment is the notion of PSpecs recombination or cascade. (Recall that appearance control within PSpecs is accomplished via Cascading Style Sheets, which rely upon a related cascade notion.) PSpecs are created by the annotator and applied as 'styles' to annotations. The reader of the annotations can not only specify default or override PSpecs (e.g. ensuring that a specific font size is always used, or controlling animation style or speed), but can also conditionally enforce PSpecs, allowing a reader to e.g. easily distinguish between different annotators. Upon display, the various PSpecs are combined into the final result. The work on PSpecs is by no means over, as there remain many potential venues of inquiry, such as PSpecs dynamically adjusting to the surrounding page.

A final extension of the Arakne Environment is the ability to address glosses and text within glosses. The OHSWG data model allows (similar to Dexter) anchoring within all hypermedia objects (making e.g. links to links possible). A gloss is in Arakne modelled as an anchor (a hypermedia object) with the gloss text contained within an attribute. Previously, the Arakne Environment handled only anchoring within Web pages, but it has now been extended to handle anchoring within glosses as well. This makes it possible for users to create links to or from glosses, or for glosses to contain other glosses.

How do fluid concepts, fluid graphical styles, and fluid algorithms translate to the more freewheeling and chaotic Web context?

Open Fluid supports a variety of cue appearances to allow salience on many kinds of pages, as well as visual distinctions between glosses, links, and combined links and glosses. These appearance specifications are based on CSS to allow composition with existing appearance choices by the original author of the page. As a result of this greater variety, Open Fluid glosses can be useful in more situations, such as the typed glosses shown in Figure 6. Open Fluid also supports a rich set of interaction techniques so that the interaction with glosses and links can coexist.

Earlier fluid prototypes were inspired by a vision of electronic books. They therefore emphasized page orientation – namely, finding room on the existing page for both the primary and open glosses. In the open hypermedia context, with all of HTML's potential complexity, the strategy of pushing down later content to make room is much simpler than finding existing whitespace or creating it within the current view (e.g., via interline compression). Moreover, since users are already accustomed to scrolling and window resizing in the typical desktop environment, the push down strategy feels reasonable in practice, given its advantages over pop-ups: lack of occlusion, interacting with the contents, and the ability to compare. Nevertheless, we would like to experiment with more sophisticated methods for finding room in the future.

Earlier fluid prototypes used a font zoom strategy for gloss appearance, which gradually increased the font size from invisible to its final size. The additional complexity of implementing font zoom given HTML's automatic line

wrapping motivated a simpler solution. The revealed strategy is not only easier to implement, but it also permits the gloss to be read *during* the animation. In contrast, the text motion inherent in font zoom precluded reading ahead. In retrospect, the revealed strategy may solve some of the objections to animation raised by earlier test subjects.

The push down strategy for making room in the page can in principle accommodate annotations of arbitrary size. Because glosses can contain links, the annotator can explicitly break large glosses up into a summary and the full content. Alternatively, the system could perform such a partitioning automatically if the making-room strategy requires it.

What changes must occur to fluid structures and the fluid user experience when readers can add their own annotations rather than simply viewing annotations made by others?

Fluid annotations are a valuable addition to open hypermedia links, allowing annotators to provide additional context and rationale for their inserted links.

Readers use annotations in a wide variety of ways for both personal and shared use. As a result, they wish to control the appearance of anchors and glosses. We chose to base PSpecs on Cascading Style Sheets to provide this functionality. It has proved to provide considerable power, allowing annotators to both adapt glosses to and distinguish them from the surrounding material. Wielding this control does require a separate forms-based user interface, which is more complicated than we would prefer. However, the ability to use and define a set of custom PSpecs mitigates this complexity.

Note that it can occasionally be challenging to specify CSS directives to achieve a precise desired appearance on a given HTML page, because the page may use sophisticated CSS of its own, which is invisible to the annotator except through trial and error. Moreover, complex HTML structuring can similarly obscure underlines, etc.

One drawback of such a wide range of typographical possibilities is that it can reduce the consistency of anchor appearances and may thus make it more difficult to find glosses. To countermand this, a reader of annotations can issue default or override values for PSpecs, thus ensuring consistency. On the other hand, annotators can use these different styles to good advantage to create annotations of different types.

One problem with the current version is the awkward creation of glosses, as the gloss text is written in a separate window away from the Internet Explorer. We are currently investigating the possibility of inline editing of glosses in order to minimize focus shifting during gloss creation.

The external Arakne tool is of course desirable for both readers and reader-annotators to filter, organize and navigate among glosses. It also provides a useful backup for finding glosses in the presence of inconsistent anchor appearances. Future integration of the tool as an Explorer Bar will smooth interactions for readers.

Finally, supporting readers directly also motivated us to add the ability to specify an initially open state for glosses, thus permitting a new foreground style for Open Fluid glosses.

CONCLUSION

Our goal has been to allow readers to annotate existing HTML pages on the World Wide Web, including those that they do not own, and have those annotations behave in a fluid way. We accomplished our goal by using the open hypermedia Arakne Environment. In particular, Arakne was augmented with a Render Engine to handle all modifications of Web pages in a browser, including the fluid animations. Annotation appearance was specified using open hypermedia presentation specifications (PSpecs) and Cascading Style Sheets.

Fluid annotations in an open world support one of the key benefits of hypermedia, which is the ability to share content. Fluid animation, in particular, allows annotations to be placed in the context of even a complex Web page with a user experience that clearly differentiates the original from the addition. Although our focus in this paper has been to give readers the ability to smoothly annotate and view HTML Web pages for themselves and others, we expect that the ability to animate typography using open hypermedia augmentation techniques will be useful for the original authors of Web pages as well.

ACKNOWLEDGEMENTS

This work has been supported by the Danish Research Council's Center for Multimedia (Project No. 9600869) and by the Center for Human-Machine Interaction of the Danish Research Foundation. We would like to thank Kaj Grønbæk for helpful discussions and support, and the anonymous reviewers for many thoughtful comments that improved the presentation of this paper.

REFERENCES

- [1] K. M. Anderson, R. N. Taylor, and E. J. Whitehead, Jr. Chimera: Hypermedia for heterogeneous software development environments. *ACM Transactions on Information Systems*, 18(3), July 2000.
- [2] N. O. Bouvin. Unifying strategies for Web augmentation. *Proceedings of ACM Hypertext '99*, p 91-100, 1999.
- [3] N. O. Bouvin. Designing user interfaces for collaborative Web-based open hypermedia. *Proceedings of ACM Hypertext 2000*, p 230-231, 2000.
- [4] A.J. Brush, D. Barger, A. Gupta, JJ Cadiz. Robust annotation positioning in digital documents. *Proceedings of CHI 2001*, p 285-292, 2001.
- [5] J. J. Cadiz, A. Gupta, and J. Grudin. Using Web annotations for asynchronous collaboration around documents. *Proceedings of CSCW'00*, p 309-318, 2000.
- [6] L. A. Carr, W. Hall, and S. Hitchcock. Link services or link agents? *Proceedings of ACM Hypertext '98*, p 113-122, 1998.

- [7] L. A. Carr, D. DeRoure, W. Hall, and G. Hill. The distributed link service: A tool for publishers, authors and readers. *Proceedings of the 4th International World Wide Web Conference*, 1995.
- [8] Cascading Style Sheets. <http://www.w3.org/TR/REC-CSS1>
- [9] B. Chang, J. Mackinlay, P. Zellweger, T. Igarashi. A negotiation architecture for fluid documents. *Proceedings of UIST'98*, p 123-132.
- [10] H. Davis. Referential integrity of links in open hypermedia systems. *Proceedings of ACM Hypertext '98*, p 207-216, 1998.
- [11] Document Object Model. <http://www.w3c.org/TR/REC-DOM-Level-1/>
- [12] R. Gold, B. Chang, P. Zellweger, J. Mackinlay. Fluid Fiction: Harry the Ape. Exhibit at the San Jose Tech Museum of Innovation, March 1 - September 7, 2000. http://www.thetech.org/xfr/xfr-red/fluid_fiction.html
- [13] K. Grønbaek, N. O. Bouvin, and L. Sloth. Designing Dexter-based hypermedia services for the World Wide Web. *Proceedings of ACM Hypertext '97*, p 146-156, 1997.
- [14] K. Grønbaek, J. A. Hem, O. L. Madsen, and L. Sloth. Cooperative hypermedia systems: A Dexter-based architecture. *Communications of the ACM*, 37(2):64-74, Feb. 1994.
- [15] K. Grønbaek, L. Sloth, and N. O. Bouvin. Open hypermedia as user controlled meta data for the Web. *Computer Networks*, (33):553-566, 2000.
- [16] K. Grønbaek, L. Sloth, and P. Ørbæk. Webwise: browser and proxy support for open hypermedia structuring mechanisms of the World Wide Web. *Proceedings of the 8th International World Wide Web Conference*, p 253-267, 1999.
- [17] K. Grønbaek and R. H. Trigg. Toward a Dexter-based model for open hypermedia: Unifying embedded references and link objects. *Proceedings of ACM Hypertext '96*, p 149-160, 1996.
- [18] W. Hall, H. C. Davis, and G. Hutchings. *Rethinking Hypermedia: The MicroCosm Approach*. Kluwer Academic, Norwell, USA, 1996.
- [19] F. G. Halasz and M. Schwartz. The Dexter hypertext reference model. *Communications of the ACM*, 37(2):30-39, Feb. 1994.
- [20] T. Igarashi, J. Mackinlay, B. Chang, P. Zellweger. Fluid visualization of spreadsheet structures. *Proceedings of Visual Languages '98*, 1998.
- [21] iMarkup: Annotate, organize and collaborate on the Web. <http://www.imarkup.com/>
- [22] C. Marshall. Toward an ecology of hypertext annotation. *Proceedings of ACM Hypertext '98*, 1998.
- [23] H. Maurer. Hyperwave—The Next Generation Web Solution. Addison Wesley, 1996.
- [24] N. K. Meyrowitz. The missing link: Why we're all doing hypertext wrong. In E. Barrett, Ed., *The society of text: Hypertext, hypermedia and the social construction of information*, p 107-114. MIT Press, Cambridge, USA, 1989.
- [25] P. J. Nürnberg, J. J. Leggett, E. R. Schneider, and J. L. Schnase. Hypermedia operating systems: A new paradigm for computing. *Proceedings of ACM Hypertext '96*, p 194-202, 1996.
- [26] K. O'Hara and A. Sellen. A comparison of reading paper and on-line documents. *Proceedings of CHI '97*, 1997.
- [27] A. Pearl. Sun's Link Service: A protocol for open linking. *Proceedings of ACM Hypertext '89*, p 137-146, 1989.
- [28] T. A. Phelps and R. Wilensky. Robust intra-document locations. *Proceedings of the 9th World Wide Web Conference*, p 105-118, 2000.
- [29] M. Price, G. Golovchinsky, and B. Schilit. Linking by inking: Trailblazing in a paper-like hypertext. *Proceedings of ACM Hypertext '98*, p 30-39, 1998.
- [30] M. Roscheisen, C. Mogensen, and T. Winograd. Interaction design for shared World-Wide Web annotations. *CHI '95 Conference Companion*, p 328-329, 1995.
- [31] M. Roscheisen, C. Mogensen, and T. Winograd. Beyond browsing: Shared comments, SOAPs, trails, and on-line communities. *Proceedings of the 3rd International World Wide Web 95 Conference*, 1995.
- [32] C. Sastry, D. Lewis, and A. Pizano. WEBTOUR: A system to record and playback dynamic multimedia annotations on Web document content. *Proceedings of ACM Multimedia '99*, p 175-178, 1999.
- [33] U. K. Wiil and J. J. Leggett. The HyperDisco approach to open hypermedia systems. *Proceedings of ACM Hypertext '96*, p 140-148, 1996.
- [34] P. Wojahn, C. Neuwirth, and B. Bullock. Effects of interfaces for annotation on communication in a collaborative task. *Proceedings of CHI '98*, p 456-463, 1998.
- [35] P. Zellweger, B. Chang, J. Mackinlay. Fluid links for informed and incremental link transitions. *Proceedings of ACM Hypertext '98*, p50-57, 1998.
- [36] P. Zellweger, B. Chang, J. Mackinlay. Fluid links for informed and incremental hypertext browsing. *CHI '99 Video Program*, 6:38 minutes, 1999.
- [37] P. Zellweger, S. Regli, J. Mackinlay, B. Chang. The impact of fluid documents on reading and browsing: An observational study. *Proceedings of CHI 2000*, 2000.