

Fluid Annotations Through Open Hypermedia: Using and Extending Emerging Web Standards

Niels Olof Bouvin *, Polle T. Zellweger **, Kaj Grønbaek *, Jock D. Mackinlay **

* Department of Computer Science
University of Aarhus
Åbogade 34
8200 Århus N, Denmark
+45 8942 5659, +45 8942 5636
{n.o.bouvin, kgronbak}@daimi.au.dk

** Xerox Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, CA 94303 USA
+1.650.812.4335, +1.650.812.4426
{jmackinlay, pzellweger}@acm.org

ABSTRACT

The Fluid Documents project has developed various research prototypes that show that powerful annotation techniques based on animated typographical changes can help readers utilize annotations more effectively. Our recently-developed Fluid Open Hypermedia prototype supports the authoring and browsing of fluid annotations on third-party Web pages. This prototype is an extension of the Arakne Environment, an open hypermedia application that can augment Web pages with externally stored hypermedia structures. This paper describes how various Web standards, including DOM, CSS, XLink, XPointer, and RDF, can be used and extended to support fluid annotations.

Categories and Subject Descriptors:

H.5.4 [Information Interfaces and Presentation]:
Hypertext/Hypermedia – *architectures, navigation, user issues.*

General Terms: Design, Human Factors, Standardization.

Keywords: Fluid Documents, Web augmentation with open hypermedia, annotations, Annotea, XLink, XPointer, RDF.

1. INTRODUCTION

Annotation has a long and important history as a way for readers to augment written texts, dating from at least the monks who created interlinear glosses on hand-copied Bibles. It is therefore appropriate that annotation has figured prominently in the defining visions for hypertext systems. For example, in Bush's proposed Memex, a reader could connect and comment on existing material to form trails for personal or shared use [9]. Since 1989 the field of open hypermedia has been working on allowing readers to add links and annotations to a wide range of existing documents by integrating a variety of third-party applications [24]. Recently the Annotea system has been developed to foster an infrastructure based on Web standards for sharing annotations on the Web [30].

Although standards to share annotations on the Web are clearly important, we also have the opportunity to improve annotations by taking full advantage of the capabilities of computer-based documents. For most of their history, annotations have been

limited by the static nature of the written page. For example, space limitations prevented Fermat from providing us with the proof of his famous Last Theorem. The simple hypertext link can overcome such space limitations by linking text to a separate annotation page. Unfortunately, this separation creates a poor annotation experience for the reader because it makes the comparison of original text and annotation more difficult.

During the last five years, the Fluid Documents project has been exploring the use of animated typographical changes to provide an effective and appealing user experience for viewing document annotations and other supporting material in context, rather than on separate pages. Fluid Documents arose from the user interface research domain as a way to broaden focus+context techniques [19] and to apply them to the problem of finding and viewing annotations. Previous prototypes have demonstrated the value of the Fluid Documents approach in a variety of application domains, including hypertext [45], electronic books [14], spreadsheets [28], avant-garde fiction [46], and reading instruction [48]. Furthermore, user studies have validated its basic tenets of animation and contextual views [47].

Given the success of our self-contained research prototypes, we are now focusing on using open hypermedia to bring fluid annotations to the Web. Our recently-developed Fluid Open Hypermedia prototype supports the authoring and browsing of fluid annotations on third-party Web pages. The Fluid Open Hypermedia prototype is an extension of the Arakne Environment [5], an open hypermedia application aimed at augmenting Web pages with externally stored hypermedia structures.

An earlier paper [44] presented initial work on the Fluid Open Hypermedia prototype and described the changes needed to bring Fluid Documents concepts and behavior to the Web. This paper provides more implementation details and examines how existing as well as emerging Web standards can support the demands of a fluid annotation system.

2. OPEN HYPERMEDIA AND THE WEB

Open hypermedia dates back to 1989, when it was realized that the reliance on special-purpose editors found in existing monolithic hypermedia systems was a liability, as it limited the appeal of hypermedia structuring. By integrating third-party applications with hypermedia, users could continue to work with their accustomed tools while enjoying the advantages of hypermedia structuring. The open hypermedia community has produced a number of systems, including MicroCosm [26],

HyperDisco [40], Devise Hypermedia (DHM) [21], Chimera [2] and HOSS [34]. The desire to support third-party applications led to certain characteristics, such as externally stored hypermedia structures. From the view of open hypermedia, the Web is an interesting subject of integration, and several systems have been extended to provide external links on Web pages. These systems include DLS [11], DHM/WWW [20], Webwise [23], and the Arakne Environment [5][6]. Using these systems, readers can create links and other hypermedia structures on top of arbitrary Web pages, and can share these links with others.

The Arakne Environment forms the infrastructure for the Fluid Open Hypermedia prototype. The Arakne Environment [5] is a collaborative open hypermedia system aimed at augmenting Web pages with externally stored links and other hypermedia structures. This goal is accomplished through the integration of the Microsoft Internet Explorer, allowing links and annotations to be added to Web pages as they are rendered in the Web browser. Hypermedia structures created in the Arakne Environment are stored on hypermedia servers, and can be exported in the Open Hypermedia Interchange Format (OHIF) [22].

3. REQUIREMENTS FOR WEB ANNOTATIONS

Informed by our experiences with Fluid Documents and other prior annotation systems, we have developed a variety of requirements and desires for Web-based annotations. This section presents our requirements and offers some underlying rationale for their inclusion.

1. *Augment existing Web pages directly, so that they will be found naturally when those pages are viewed subsequently.*

Long espoused by the open hypermedia community [24] and others, this requirement eliminates the need to copy Web pages in order to annotate them, provides a central and contextualized place to store annotations, and requires no changes to the work practice of readers.

2. *Support rich augmentation, including both links and annotations. Annotations should be able to contain links, images, audio, and other complex content.*

The ability to add links allows annotators¹ to make additional connections, either to their own documents or to other third-party Web pages. Annotations allow annotators to add additional material without having to create and manage new files to hold the annotations. Rich annotation content allows annotators powerful options for expressing their ideas. In addition to links and annotations, the Arakne open hypermedia system [5] also offers the ability to add other hypermedia structures, such as guided tours.

3. *Support **fine-grained** annotations within a page – that is, annotations should be able to refer to any characters or objects on the page, including existing link anchors.*

Fine-grained annotations reduce annotator effort by eliminating the need to describe the subject of the annotation within the annotation itself. They also support detailed comments on many

different locations on a page. The ability to annotate existing link anchors allows annotators to add their own link rationales or link previews, as advocated by the work on Fluid Links [45]. In contrast, Microsoft Web Discussions permit reader annotations only at author-selected points on a page [10].

4. *Allow further edits to the underlying page without loss of annotations. A robust mechanism for positioning annotations is desirable.*

This requirement supports shared work and minimizes the need for manual updates when the underlying document changes. The system should detect changes to locations within a document that contained annotations and allow access to any annotations that have been “orphaned” as a result. Readers may also want to be notified of more distant changes in the underlying page, in case the relevance or accuracy of annotations is thus damaged. User expectations regarding correct positioning in the face of edits can vary [8]. Algorithms for providing improved positioning robustness are under active development [35][43].

5. *Support personal and shared annotations.*

Early work on the ComMentor system for Web annotation [37], as well as the more recent Microsoft Web Discussions[10], have demonstrated the utility of shared annotations. Marshall’s analysis of personal annotations [32] has shown the value and variety of personal commentary on existing documents.

6. *Provide facilities for organizing, filtering, and searching annotations.*

Pioneered by the ComMentor system, the organizing, filtering and searching capabilities of the present-day iMarkup plugin [29] provide a particularly pleasant example of good functionality and ease-of-use for annotators.

7. *To support a wide variety of existing pages and annotation goals, annotators should have considerable control over the **salience** of their annotations – that is, how much they stand out from the underlying page – ranging from little or no impact on layout and appearance to significant impact. This control should be easy for annotators to use.*

Avoiding impact on layout can permit the annotation of carefully formatted text, such as aligned columns. Reducing initial page impact also honors the original page as the primary material and permits it to be viewed with minimal distractions [31]. However, annotators may also wish to make an annotation initially the most visible thing on the page in order to call attention to it for their own purposes [32][38].

An effective way to achieve minimal impact is to separate an annotation into two parts: its *anchor*, which establishes its location, and its *contents*, which we also term its *gloss*. A gloss can remain hidden until the reader asks to view it. Issues of salience thus apply separately to both anchor and gloss: while a reader is viewing a gloss, annotators may wish to clearly distinguish the entire gloss from the underlying page, or they may wish it to blend in.

8. *Allow readers to view glosses in context – that is, combined with the original page. Ideally, a gloss should be displayed near its anchor, and items on the original page should not be occluded.*

Combining a gloss directly with the original page reduces cognitive effort for readers, allowing them to relate and compare

¹ We use the term “annotator” when we wish to emphasize the tasks required to augment pages; the term “reader” considers the tasks required to view augmentations. Users may assume either or both roles.

the gloss with its surrounding context. 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 [41]. Studies of Fluid Documents have indicated that providing glosses close to their anchors speeds reader performance [46]. However, these benefits of nearby placement are reduced and may be eliminated when the gloss occludes material on the original page, as for example in the popup ToolTip windows commonly used to display HTML link titles in current browsers [33].

9. *Provide readers with easy interactive control over viewing glosses. Ideally, the reader should be able to view multiple glosses simultaneously for comparison with the original material and each other.*

Annotations are typically smaller than full hypertext pages, and therefore an easy, lightweight mechanism for viewing them is desirable to match the effort required with their relative information content.

10. *Allow readers to interact fully with glosses: following embedded links, selecting text and inserting further open hypermedia links and annotations, etc.*

When glosses can include links, annotators can create effective multi-way links that improve their browsing experience [45]. Allowing further annotations within glosses supports conversations and other shared work [37].

4. FLUID OPEN HYPERMEDIA

This section describes the capabilities of the Fluid Open Hypermedia system. We begin with a simple example to motivate and demonstrate the use of fluid annotations.

The central characters in our scenario are a group of Web page designers who refer to the W3C standard documents regularly in their daily work. By augmenting the existing W3C documents with fluid annotations, the designers can both record information for their own use and pool their knowledge in a way that fits seamlessly into their normal work practice. The contents of their annotations serve a variety of roles, including: showing examples, warning of reduced compliance by some browsers, and providing additional comments.

Figure 1 shows several shared fluid annotations on the W3C Cascading Style Sheets 1 Web page. In this example, various colored highlights indicate the presence of shared annotations of different types. Not only do their distinct anchor appearances mark their roles within the document, but they also support filtering on the different types.

Fluid annotations have many desirable features:

- Any document element can act as an anchor, allowing a fluid annotation and/or an open hypermedia link to

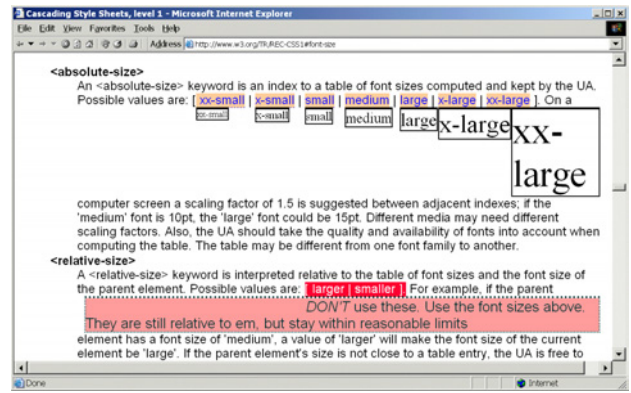


Figure 1. Shared annotations on the W3C Cascading Style Sheet 1 Web page. Using three different Presentation Specifications, readers have added fluid annotations with distinct anchor highlights to indicate *examples* (e.g., the seven separate annotations near the top, all currently open for comparison) and *warnings* (e.g. “[larger | smaller]” near the bottom of the page). Each gloss can be opened or closed interactively as desired. To avoid occluding the original material, following lines are dynamically pushed down on the page when a gloss is opened.

be added to it.

- The contents of fluid annotations (glosses) are typically hidden (or *closed*) until the reader interactively activates (or *opens*) them.
- When opened, a gloss expands to become a temporary first-class element of the original document near its anchor. Readers can thus interact fully with the contents of a gloss, following links, copying text, etc.
- Readers can interactively open and close each gloss as desired, allowing one or more to be available simultaneously for use and comparison with the original text and/or other glosses.
- Space to display a gloss is created by dynamically altering the layout, typography, and other graphical characteristics of the original material. Occlusion of the original material is minimized. In our current “push down” technique, the gloss is gradually revealed just below the anchor, while the following lines are pushed down the page to make room for it. See Figure 2.
- Animated opening and closing transitions clarify the changes as additional material arrives and departs

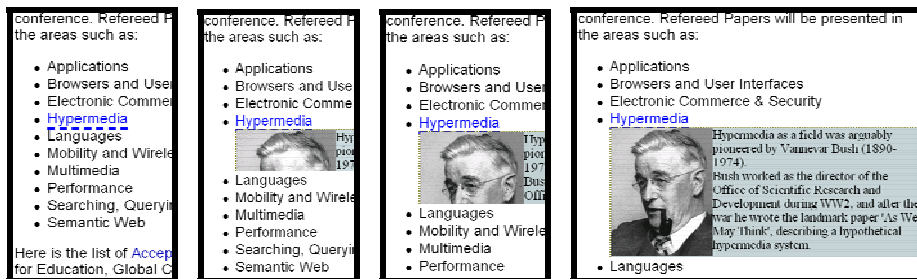


Figure 2. Smooth gloss animation. These snapshots show intermediate stages in the opening of an annotation that a reader has anchored on the word “Hypermedia” on the WWW 2002 preliminary Technical Program page.

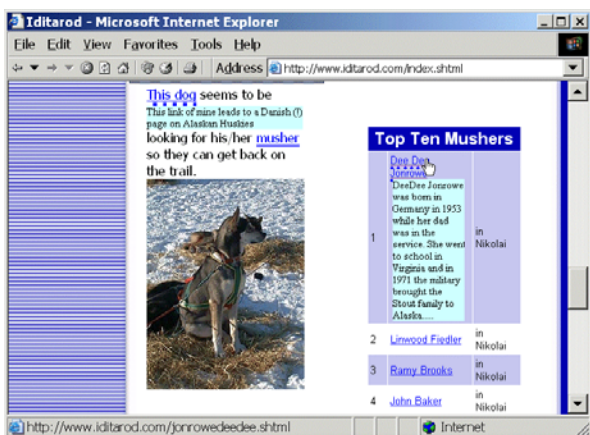


Figure 3. Using fluid annotations to explain or preview open hypermedia links and HTML links. To the left, the annotator created two open hypermedia links (“This dog” and “musher”) and then annotated the former with explanatory text. On the right, the top musher “Dee Dee Jonrowe” has a HTML link (notice the URL visible in the bottom of the window). This link has been annotated with the first sentences from the destination Web page to form a link preview. Note that the visual appearance of anchors composes: a double solid underline indicates an open hypermedia link anchor, a broken single underline indicates an annotation anchor, and the conventional single solid underline indicates a HTML anchor.

from the page. Animation smooths the experience of viewing glosses, allowing it to become a perceptual activity rather than a cognitive one.

- Annotators can use Presentation Specifications (PSpecs) to specify the visual appearances of anchors and glosses to suit a wide variety of annotation goals and settings. Anchors and glosses can be designed to blend in with the original Web page or to be visually distinct from it. To ease the creation of annotations, PSpecs can also be reused and shared with other annotators.
- To permit links (either existing links within the page or open hypermedia links) and fluid annotations to share the same anchor, PSpecs also allow the annotator to specify a different activation event (e.g., shift + mouseEnter or shift + leftMouseButton) for the fluid annotation than co-located links. See Figure 3.
- Glosses are expressed in HTML and can contain embedded links, rich formatting, and images. Open hypermedia links and/or fluid annotations can also be added to glosses.

It is illustrative at this point to compare fluid annotations with two previous efforts to display supporting material on Web pages: the well-known ToolTip-style popup windows for HTML link titles [33] and the iMarkup annotation system for augmenting existing Web pages [29]. Although popup link titles provide a way to present supporting information to the reader, they cannot be added by the reader. iMarkup annotations can be added by reader. Both ToolTip popups and iMarkup annotations appear on an overlaid virtual layer rather than being combined with the original page. This simplifies

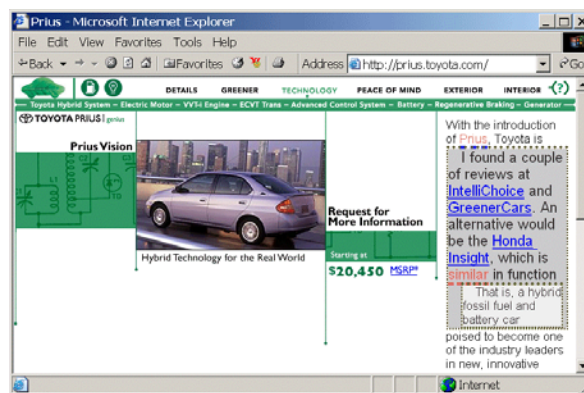


Figure 4. Nested open hypermedia links and annotations inside glosses. Contemplating a car purchase, the reader has created a gloss and then added three open hypermedia links to it: to two reviews of the car and to a similar car. Another reader has added nested annotation to clarify what makes the car special.

implementation, but occludes material on the original page. ToolTip popups have the additional flaw that they are very transitory: only one can be open at a time and users cannot interact with their contents (e.g., the user cannot copy/paste from a ToolTip). iMarkup annotations do not share these flaws.

To avoid these problems of occlusion and limitations on opening and interacting with annotations, we have chosen to combine the annotations with the original page. Animation is a strong asset in managing the temporary addition and subtraction of annotations on the page.

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 Fluid Documents [46] showed that users can process moving text even in a serious reading situation. They also showed that distant glosses presented without animation were frequently not seen.

In an effort to minimize clutter and distraction on the page when annotations are closed, as well as to avoid altering page layout (if desired), we have so far chosen not to insert special annotation markers. This has two implications: finding annotations is not necessarily obvious, and there is no special annotation locus for behavior. We address the former issue in two ways: anchor appearance can be adjusted to be visually distinct from the underlying page as well as from ordinary links, and a separate editor allows readers to list and filter existing annotations and other augmented structures. We have solved the latter problem with overloading behavior on the anchor as part of the PSpec.

Similarly, although earlier versions of Arakne annotations permitted changes to the anchor, such as insertion or replacement of text, current fluid annotations do not permit such modifications. This restriction is intended to maintain the integrity of the existing page. Given appropriate safeguards against rewriting that is invisible to readers, this policy could be changed in the future if users find it worthwhile.

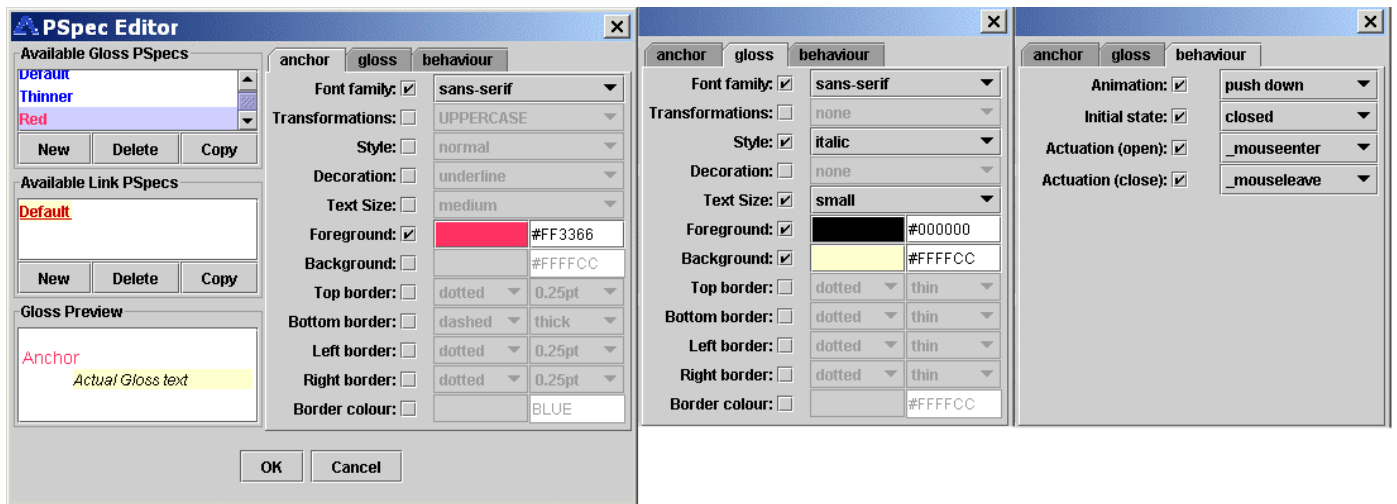


Figure 5. The Presentation Specification (PSpec) Editor. This editor is used to specify the appearance and behavior of glosses and link endpoints. Instances of all three tabs are shown (the left panels do not change).

5. BUILDING FLUID OPEN HYPERMEDIA USING WEB STANDARDS

This section describes how the Fluid Open Hypermedia system was implemented using DOM and CSS: what issues were encountered, and how they were resolved.

The Fluid Open Hypermedia prototype allows readers to add fluid annotations and multi-headed open hypermedia links to third-party Web pages. Both links and annotations are tied to anchors, which can be actuated by the user. The visual appearance of anchors and glosses are governed by Presentation Specifications. All these elements, known from open hypermedia and previous fluid document prototypes, have been mapped to the Web using the Document Object Model [16] and Cascading Style Sheets [12].

5.1.1 Open Hypermedia Anchors

An open hypermedia anchor is a selection chosen to form the source of a fluid annotation and/or the endpoint of an open hypermedia link. The nature of the anchor will vary according to the media type addressed. While the Arakne Environment has previously explored externally-defined anchors into temporal data [6], the current version is limited to text anchors.

Regardless of media type, anchors are defined by a Node Specification, which specifies the object wherein the anchor occurs. This node may currently be a Web page, a frame hierarchy, or a gloss. The ability to address glosses as nodes enables open hypermedia links and fluid annotations within glosses, as shown in Figure 4.

Furthermore, an anchor has a Location Specifier [25], which designates a selection in a media type. As described above, these currently only address text selections, but given the extensible nature of open hypermedia location specifications, this limitation may be lifted in the future. Location specifiers should be extended to support arbitrary HTML elements, so that e.g. an image can act as an anchor as well.

Additionally, an anchor contains arbitrary key/value-pairs, which may be used for e.g. semantic information.

5.1.2 Open Hypermedia Links

Using the Arakne Environment, annotators can create bi-directional multi-headed links on arbitrary Web pages. If more than one destination is available for a link, the destinations are presented in a popup menu as the reader activates the link.

5.1.3 Fluid Annotations

A fluid annotation contains a text (i.e., the gloss) that is presented when its anchor is activated. Because the gloss is expressed as HTML, it can contain rich formatting and images. Open hypermedia anchors can be added to a gloss, so fluid annotations can contain open hypermedia links or other fluid annotations. An example of a nested annotation can be seen in Figure 4.

5.1.4 Presentation Specifications (PSpecs)

The user can format the appearance or presentation of anchors and glosses. This formatting is accomplished through the Presentation Specification or PSpec, which is authored using the PSpec Editor in Arakne. One of the challenges faced by an annotator is to properly style the annotations, so that they are distinct without being jarring. Because the formatting parts of the PSpec are expressed in CSS, anchors and glosses inherit style from their context. In addition to the static formatting, the PSpec is also used to designate how an anchor should be actuated and how a gloss should be animated.

The editor used to create PSpecs can be seen in Figure 5.

5.2 Rendering Fluid Annotations via DOM

The aim of the Fluid Open Hypermedia prototype is to provide smoothly animated fluid annotations on arbitrary Web pages, in combination with open hypermedia links and other hypermedia structures. This section explains how this goal is accomplished through the Render Engine component.

An overview of the relationship between the Arakne Environment and the Render Engine can be seen in Figure 6. The Render Engine is a DLL that serves to connect Arakne and the Microsoft Internet Explorer. The Render Engine provides the Arakne Environment with an API to define anchors, open hypermedia links, fluid annotations, and PSpecs, and to receive

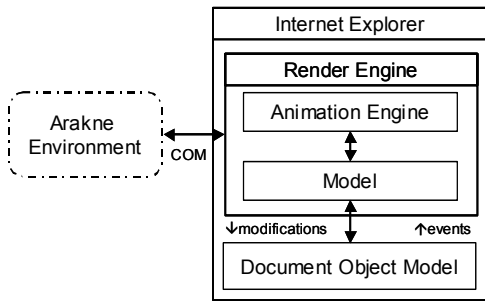


Figure 6. The structure of the Fluid Open Hypermedia system.

events generated by the Web browser (including, but not limited to the events designated to actuate links or glosses). The Render Engine maintains the initial state of the gloss (opened or closed), models the space available for glosses on the Web page, and handles the animation of opening and closing the glosses.

The first task of the Render Engine is to determine the current location of the anchors. This is done through open hypermedia Location Specifiers (LocSpecs) [24], similar to XPointer. An anchor may be associated with both open hypermedia links and fluid annotations, and are therefore declared separately from these. Once an anchor location has been found, a `` tag is created around it, and it is marked up according to its Presentation Specification, which is pure CSS. At this point, the Render Engine subscribes to all events generated by this `` tag. Of special interest is the event specified to be the actuation command for this anchor (e.g. `shift + mouseEnter`). If such an event is generated, the gloss associated with the anchor should be displayed. The actuation event, as well any other generated event, is also sent to Arakne. This gives Arakne the opportunity to react to any event, as well as to define the gloss on demand, rather than when the page is initially displayed.

When a gloss is to be displayed, its size and location must be determined, and room on the page must be made, if necessary. The resulting structure is illustrated in Figure 7. First, a `<div>` tag is created between the end of the line containing the anchor and the beginning of the following line. This creates the Context box, which provides a static origin for its containing box. Inside the Context box, another `<div>` tag is created, which will contain the white space upon which the gloss will be drawn. This creates the White Space box. Finally, the gloss is inserted into the White Space box in a `<div>` tag. By this time, the Gloss box has been generated and decorated using its Presentation Specification, so that its size is known. In the case of a “push-down” animation, the text below the anchor is gradually pushed down, revealing the gloss. To accomplish this effect, the Internet Explorer’s timer is started, and at each tick, the Gloss box is made bigger. The White Space box will resize to the point of accommodating the largest of its contained Gloss boxes. Likewise, the Context box will accommodate the White Space box. As this animation runs, the growth of the boxes will push down the following lines. The animation terminates when the Gloss box has reached its desired size.

The White Space box is necessary to handle the situation where there is more than one annotation on a line, as shown in Figures 1 and 7. The initial implementation of the Render Engine made room available for glosses by enlarging the Context box. This

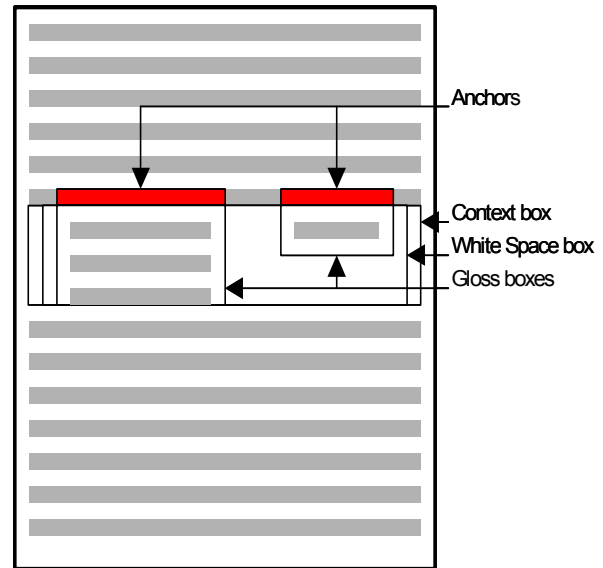


Figure 7. The `<div>` boxes generated by the Render Engine. The Context box forms a static origin for the White Space box, wherein the Gloss boxes are found. This example has two anchors and two open glosses.

worked fairly well, but not in the case illustrated by Figure 1, where there are multiple anchors on the same line. Opening multiple glosses would result in the following text being pushed further and further down the page, as each gloss would make room for itself again. By introducing the White Space box, this problem was solved, as the White Space box will grow or shrink no larger than its largest contained Gloss box. In Figure 1, notice how the text is pushed down to accommodate the largest gloss, and notice how the largest gloss has been pushed slightly out of the way to make room for the rest of the glosses. The boxes around the glosses show the space needed for each.

The current version of the Render Engine only models the white space that it has itself created on the page. A challenge for the future is to extend this functionality, so that previously existing white space on an unmodified Web page can be utilized when it is sufficiently near the anchor.

5.3 Writing and Reading Fluid Annotations

The Arakne Environment is a collaborative hypermedia system. Users may create annotations for their own use or for a wider audience. Annotations can either be shared via the hypermedia servers, or exchanged in the OHIF format [22]. This section briefly outlines the typical interaction with the system.

Most authoring and browsing of fluid annotations and open hypermedia links can be accomplished directly in the Internet Explorer. The annotator can specify how links should be followed and glosses should be opened. The default is “click” for links, and “shift + MouseEnter” for glosses, but can be any combination of mouse events and modifier keys that the annotator wishes.

Open hypermedia links and glosses are typically authored via context-dependent popup menus. Right-clicking on a text selection and selecting “Create Gloss” in the context menu

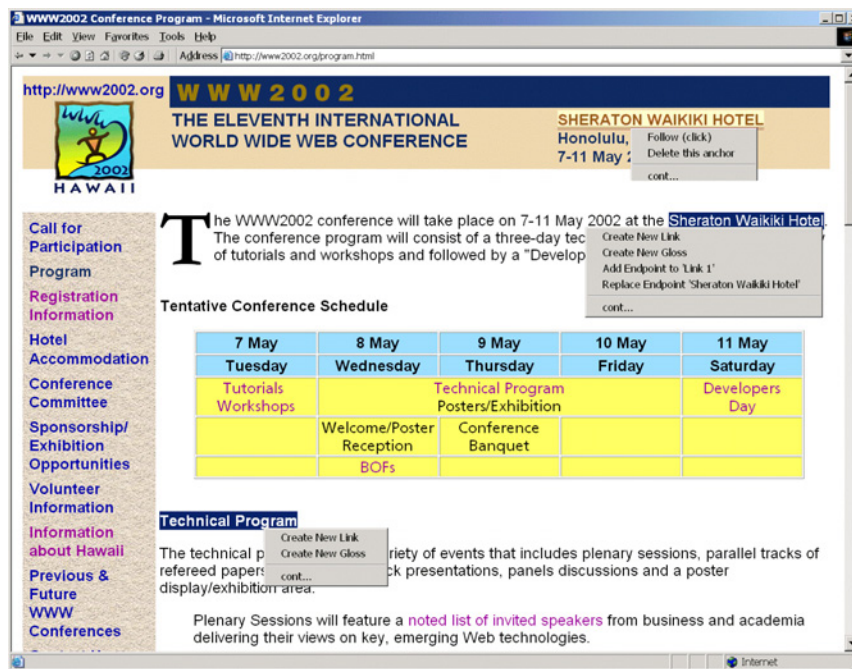


Figure 8. Composite illustration of context-dependent right-click menus. Leftmost, the reader has selected “Technical Program” and can add a new open hypermedia link or fluid annotation. Topmost, the reader has right-clicked on a link endpoint, and can choose between following the link (which leads to the Sheraton Web site) or removing the anchor underlying the endpoint. Notice how the link anchor PSpec has upcased “SHERATON WAIKIKI HOTEL” to make it more visible. At the right edge, the reader has selected “Sheraton Waikiki Hotel” and can create a new link or gloss, add the selection as an additional endpoint for the link, or replace the existing endpoint (above) of the link with this selection. In all instances, selecting “cont...” in the menu produces the standard Internet Explorer menu.

creates an open hypermedia anchor for the selection and opens an auxiliary window in which the annotator can type the gloss (using HTML, if desired) and select the styling of the annotation. Once the window is closed, the Web page is refreshed and the new annotation appears. Inserting open hypermedia links and adding endpoints to existing open hypermedia links is handled similarly. Endpoints and glosses can also be removed through the right-click menu. The commands available in the right-click menu depend on the state of the system (is the user currently authoring a link?) and what element the user has right-clicked on (a new selection vs. an existing endpoint?). This is illustrated in Figure 8, which is a composite figure demonstrating the various context menus.

The appearance of fluid annotations and open hypermedia links is specified by PSpecs, which are created using the PSpec Editor shown in Figure 5. This tool allows for a varied set of appearances (based on CSS) for open hypermedia link anchors, annotation anchors, gloss text and gloss behavior. To reduce annotator effort and to support meaningful shared visual semantics, PSpecs are first-class objects that can be reused and shared. Thus annotators will most often use an existing PSpec.

6. USING AND EXTENDING EMERGING WEB STANDARDS

Based on our experiences with Fluid Open Hypermedia and the ongoing standards work by the W3C, this section discusses areas where the standards are moving in the right direction and where they might yet be improved.

6.1 CSS3

The Fluid Open Hypermedia prototype relies heavily on DOM for modeling and manipulating the structure of Web pages, and on CSS for affecting the appearance as well as animation of fluid annotations. The existing CSS standards (CSS1 and CSS2) [12] are aimed at static page rendering: once a Web page has been parsed, and its associated style sheets have been applied to its DOM, the rendered result does not change. In contrast, the animations performed by the Render Engine are handled by gradually modifying the appearance (through DOM and CSS) of the Web page. Dynamic Web pages are now quite common, with behaviors such as rollover effects and unfolding menus as prime examples. These modifications are usually handled through JavaScript programs, which manipulate the DOM and style sheet of the Web page. These scripts are however on an ad hoc basis, and often vary depending on the Web browser used.

The work on CSS3 is currently in progress, and many new technologies have been suggested for inclusion in this new standard. These extensions include some that are of special relevance to Fluid Open Hypermedia. BECSS (Behavioral Extensions to CSS) [4] proposes a model to extend CSS with behavior, so that the above-described dynamic HTML can be handled directly through CSS.

Another interesting proposed extension is the support for ruby [13]. A ruby is a small pronunciation guide close to a primary text. It is commonly used in Japan where it helps young or foreign readers to grasp the meaning of the primary text. While very useful in this context, this technology also has use for

general annotations. Using this technique, authors could place small annotations on e.g. links.

The consequences of these suggested extensions of CSS are interesting. Firstly, the flexibility and modularization provided by CSS1 and CSS2 with regards to layout would be available to behavior. Secondly, it would make it possible to provide functionality similar to fluid annotations for ordinary Web browsers without special software. While not necessarily addressing externally stored annotations like Fluid Open Hypermedia, it would allow Web page authors more tools to ease the navigation of their readers.

One thing we have found lacking during the development of the Fluid Open Hypermedia prototype is the notion of time in CSS. It is not possible to designate that a certain markup should last for a specified time. One obvious use of such a feature would be to temporarily highlight a link destination. CSS offers pseudo classes for the `<a>` tag, so that users can easily distinguish between e.g. links that have been visited and others that have not. Yet, these pseudo classes do not extend to the `<a>` tag when it is used as a destination (through the name attribute) on Web pages, leaving the reader to wonder where the destination on the displayed part of the Web page is. If the destination could be temporarily highlighted (also to distinguish from other material on the page), there would be no confusion.

The Fluid Open Hypermedia prototype has demonstrated that interesting results may be achieved by creatively using existing Web standards. Reaching this point however required a fair amount of development work, trying to make the Internet Explorer do as we intended. This is acceptable for a research prototype, but if the technologies described in CSS3 became standardized, it would be much easier for others to follow suit. Behavior should be standardized rather than handled through ad hoc programming, and ruby offers discreet annotations on Web pages (if nothing else as an alternative to ToolTip popups).

6.2 Annotea and Fluid Annotations

This section briefly describes the W3C Annotea framework, compares it to fluid annotations, and finally proposes an extension to Annotea to implement fluid annotations.

6.2.1 Annotea

Annotea [30] is a Web-based shared annotation framework based on an open RDF infrastructure. Annotations are modeled as a class of metadata. Annotations are viewed as statements made by an author about a Web document. Annotations are external to the documents and can be stored in one or more annotation servers, e.g. implemented as a general RDF database. A number of clients implement the Annotea framework:

- Amaya [1] provides native support for Annotea, for publishing, querying, and discussion threads.
- Bookmarklets is a browser-independent JavaScript interface to Annotea, which provides document-level annotations and not fine-grained annotations that are possible with Amaya.
- Annozilla [3] uses Annotea within Mozilla.

Annotea annotations are represented in a combination of RDF [36], Dublin Core [18], XPointer [43], and XLink [42].

Annotations are (in Amaya) marked with a pencil icon on the location pointed out by the `a:context` attribute in the source document. The annotation text is stored either locally or on a

server as a separate HTML file (the `a:body` resource). At runtime the annotation is spawned in a new browser window with the Dublin Core attributes shown in a table before the annotation text.

6.2.2 Comparing Annotea and Fluid Annotations

In contrast to Amaya's new browser window, fluid annotations grow smoothly out of the source text between the lines. The open hypermedia LocSpec mechanism [24] is used to locate the annotation in the source material. PSpecs (a concept originating with the Dexter model [27]) are used to govern the appearance and behavior of fluid annotations. LocSpecs (introduced in [25]), PSpecs, and annotation text (or reference) is stored in the XML-based OHIF format [22]. In the following we propose to extend the Annotea framework with fluid annotations based on our experiences in designing Fluid Open Hypermedia.

6.2.3 Fluid Annotation Proposal for Annotea

In order to implement fluid annotations, the Annotea framework must be extended with PSpec information and a model for rendering the PSpec info.

Annotea could be extended with a new name space specifying the PSpec:

```
xmlns:fluid=http://www...com/fluid/fluid-ns#
```

as illustrated in Figure 9.

The inclusion of such a name space, together with the functionality provided by the Render Engine regarding presentation and animation, would provide Annotea with fluid annotations. Systems not supporting gloss animation should naturally ignore the fluid namespace.

6.3 XLink, XPointer, RDF and Fluid Annotations

The Fluid Open Hypermedia system uses the OHIF [22] XML format as the basis for storing annotations and other hypermedia structures. The OHIF format uses the general LocSpec mechanism proposed in [24], which is aimed at specifying locations in arbitrary document types such as text, graphics, audio, video, and CAD. With the emerging XPointer and XPath standards we can start using these as LocSpecs for fluid annotations in XML data. XPointer supports identification of regions, e.g. the anchor for a fluid annotation, in XML documents. XPointer allows for selection based on ids, hierarchical structure (from XPath), or an arbitrary user selection (e.g. selecting a string in the rendered XML document). XPointer can address arbitrary XML documents, and a given region may be identified using several locators - XptrParts, which improves reliability, as one locator might fail after a document has been edited. However, we still need to use other means of location specification if we wish to provide fluid annotations and links in non-XML documents.

For the purpose of providing fluid annotations we could replace the linking and annotation mechanism of OHIF with the XLink and RDF standard for representing links and annotation relations, similar to what is done in Annotea. RDF is well suited for representing the actual annotation and attributes as metadata, however the presentation format and behavioral aspects require extensions, as discussed in Section 6.1.

Another important issue is how to address nested links or nested annotations in a document. In the current Fluid Open

```

<?xml version="1.0" ?>
<r:RDF xmlns:r="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:a="http://www.w3.org/2000/10/annotation-ns#"
      xmlns:http="http://www.w3.org/1999/xx/http#"
      xmlns:d="http://purl.org/dc/elements/1.0/"
+     xmlns:fluid="http://www...com/fluid/fluid-ns#">
  <r:Description>
    <r:type resource="http://www.w3.org/2000/10/annotation-ns#Annotation" />
    <r:type resource="http://www.w3.org/2000/10/annotationType#Comment" />
    <a:annotates r:resource="file:///C:/Program%20Files/Amaya/amaya/AmayaPage.html" />
    <a:context>file:///C:/Program%20Files/Amaya/amaya/AmayaPage.html#xpointer(string-
range(/html[1]/body[1]/div[1]/p[1], "", 12, 10))</a:context>
+     <fluid:AnchorPSpec style="url(http://www...com/important.css)"/>
    <d:title>Annotation of Welcome to Amaya</d:title>
    <d:creator>Kaj Gronbak</d:creator>
    <a:created>2001-11-11T20:48:51</a:created>
    <d:date>2001-11-11T20:49:01</d:date>
    <a:body r:resource="file:///C:/WINNT/profiles/Kaj%20Gronbak/amaya/annotations/annots19c.2.html"
  />
+     <fluid:GlossPSpec id="important"
+       style="border: thin dotted black; font-size: small;"
+       initial-state="closed">
+     <fluid:animation type="push-down" duration="50"/>
+     <fluid:actuation-open event="onMouseEnter" modifier="shift"/>
+     <fluid:actuation-close event="onMouseLeave" modifier="shift"/>
+     </fluid:GlossPSpec>
+   </r:Description>
</r:RDF>

```

Figure 9. An Annotea annotation created with the Amaya browser and augmented with a fluid PSpec. A ‘+’ symbol in the left column distinguishes the lines that comprise the fluid extension.

Hypermedia implementation, links to material located within glosses are resolved by first retrieving the parent Web page of the gloss, opening the gloss and locating the destination. This does not fit well with the XLink philosophy of using URI and XPointer as targeting mechanism, as there is no provision for the opening of the containing gloss.

Finally, XLink is designed primarily for navigational hypermedia, which is the classic hypermedia application. If we consider general open hypermedia, other structuring mechanisms have been introduced which cannot easily (or at all) be described in terms of links. This goes e.g. for composites, guided tours, spatial, or taxonomic hypermedia [24]. Thus general open hypermedia structures combined with fluid annotations still calls for the mechanisms of the OHIF format or similar extensions to the XLink standard.

7. RELATED WORK

Recent interest in digital libraries has spawned a broader look at various forms of annotation. In addition to Marshall’s previously-mentioned analysis [32], the Notable project has produced a useful conceptual framework to aid in the design of annotation systems [15].

Although the Arakne system arose from the open hypermedia community and its approaches, there have been other efforts with related augmentation goals within the Web community. [39] presents an abstract annotation architecture and discusses how well current open and standard Web infrastructures support varied implementations of it. Yawas [17] also uses the DOM to provide fine-grained anchoring of annotations with customizable annotation styling. However, these systems do not support the richness of Fluid Open Hypermedia’s animated

gloss presentations, nested links and annotations, and broadly customizable gloss and anchor appearance via CSS.

For a detailed comparison with ComMentor [37], DLS [11], the Webwise system [23], the Arakne Environment [5], Microsoft Office Web Discussions [10], and the XLibris prototype digital reading appliance[38], see [44]. These systems all provide some notion of annotating content, but none of them provides a fluid annotation interface.

In the previous section we compared fluid annotations to Annotea-based annotations on the Web [30]. Although neither of the current Annotea clients (Amaya and Annozilla) provides a fluid annotation interface, we have demonstrated how we could extend Annotea to do so.

In addition to systems already covered, we have discovered two new annotation facilities for the Web: a new version of iMarkup and a new system called BrowseUp, which we will briefly discuss here.

iMarkup [29] provides the richest user interface for Web annotation to date. It supports both textual annotation (underlines, highlights, etc.) and document-level annotation via Post-It™-style sticky notes. Annotators can also draw freeform ink on the page. Many different styles are available to support varied users and usage. Recent additions include voice annotations; transparent sticky notes that create a result similar to the earlier Fluid overlay technique[45]; the ability to add one or more links to a sticky note (these are global to the note – that is, there are no fine-grained text anchors within the note); and the ability to mark on PDF documents as well as on HTML pages. These features, together with smooth integration with Internet Explorer as an Explorer Bar or a menu extension and

good capabilities for organizing and sharing annotations, make iMarkup a very usable and versatile tool.

However, despite a feature for shrinking annotations that are not currently of interest (neither this minimization nor its dual expansion is animated), iMarkup has a tendency to clutter and obscure its page content. This may be appropriate for many situations, but is not universally positive. iMarkup does not allow as much appearance control or interaction control for text annotations as does Fluid Open Hypermedia, nor does it contribute to our understanding and development of standards by making its mechanisms and algorithms public.

BrowseUp [7] is a recently released system that provides what they call Virtual Links or oLinks (Object Links) on top of Web pages. The system is similar to open hypermedia systems such as WebCosm, Webvise, and Arakne. BrowseUp supports linking in a separate layer on top of Web pages. Augmented content, such as local files, is automatically converted into HTML and uploaded to the BrowseUp server used by the client program. BrowseUp uses its own proprietary data format stored in an Oracle 8 based server. The oLinks can be assigned keywords, thereby enabling searches. There is no separate notion of annotations in BrowseUp. Instead, annotations are made as links to a new page, which is uploaded to the BrowseUp server and is thus accessible for specified groups of users. The link-based annotations are presented as popups, similar to other links.

8. FUTURE WORK

In the previous sections we have described the successful Fluid Open Hypermedia prototype we have constructed to provide fluid annotations using Web standards and open hypermedia techniques. We see several promising directions of development that will enable us to provide a more seamless reading and annotation environment on the Web: developing an Annotea-compliant Render Engine, providing link previews and other Fluid Links-type behavior, better Web browser integration, and developing a layout negotiation model for the Web.

We plan to develop an Annotea-compliant Render Engine for fluid annotations, taking advantage of the proposed fluid annotation namespace and the CSS-based PSpecs. This can be done by implementing a small preprocessor to the current Render Engine to parse the Annotea annotations, filtering out the XPointer LocSpec information and the extended PSpec information.

Fluid Links are a specific way to use glosses to improve hypertext navigation [45]. Glosses can be placed on link anchors to support readers in choosing among links and understanding the structure of a hypertext. At each link anchor, readers can either follow the link in the usual way or they can first view the gloss “preview” in the context of the source page. The contents of such glosses can potentially be computed dynamically, removing the need for authors to construct them individually. For example, gloss material can be automatically retrieved or constructed from the destination page. Multi-way links and nested glosses allow readers to skip through intermediate nodes while still attending to their original source context.

We see promise in using the concept of fluid annotation to provide link previews. The links on a Web page (either conventional or open hypermedia links) could be augmented with fluid previews: a generated gloss containing metadata, or

extended with a preview grabbed from the first few sentences of the target location.

The use of context-dependent menus in the Web browser is a step towards tighter user interface integration with the Internet Explorer. Other venues of integration could be the use of Explorer Bars similar to iMarkup. Whether this is a viable course remains future work.

The original Fluid Documents work matured to a point where we could codify a simple yet powerful interaction language and process between the primary document and a supporting annotation that wished to display itself [14]. The primary document and the annotations would express their constraints and desires about their relative space and salience. Given a better white space model for a Web page, it could be possible to create similar RDF extensions to support such a negotiation in the Web environment.

9. CONCLUSIONS

The scientific method and many other effective human activities are based on a cycle of reading and writing of documents. Annotation supports this cycle in individual documents, where a reader instantly becomes a writer and the document becomes an artifact that moves the activity forward. During the last five years, we have been researching fluid annotations, animated typographical changes to computer-based documents that provide an improved annotation experience for readers. We recently developed the Fluid Open Hypermedia prototype, an extension to the Arakne Environment that supports the authoring and reading of fluid annotations on third-party Web pages. This paper focuses on how we used existing Web standards, including DOM and CSS, to develop our research prototype, and how various emerging Web standards, such as XLink, XPointer and RDF, can be used and extended to support fluid annotations. Ultimately, we hope that these emerging Web standards will take full advantage of the capabilities of computer-based documents to support a wide range of effective and appealing annotation activities, from individual use to global collaboration.

10. 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 wish to thank Henning Jehøj for his work on programming the Render Engine. We also thank the anonymous reviewers for thoughtful comments that improved the presentation and clarity of this paper.

11. REFERENCES

- [1] Amaya. <http://www.w3.org/Amaya/>
- [2] 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.
- [3] Annozilla. <http://annozilla.mozdev.org/>
- [4] Behavioral Extension to CSS. <http://www.w3.org/TR/becca>
- [5] N. O. Bouvin. Unifying strategies for Web augmentation. *Proceedings of ACM Hypertext '99*, p 91-100, 1999.

- [6] N. O. Bouvin and R. Schade. Integrating temporal media with open hypermedia on the World Wide Web. *Proceedings of the 8th World Wide Web Conference*, Toronto, Canada, p 375-387, 1999.
- [7] BrowseUp. <http://www.browseup.com/>
- [8] A.J. Brush, D. Bargerion, A. Gupta, JJ Cadiz. Robust annotation positioning in digital documents. *Proceedings of CHI 2001*, p 285-292, 2001.
- [9] V. Bush. As we may think. *The Atlantic Monthly*, Volume 176, No. 1, July 1945, 101-108.
- [10] J. J. Cadiz, A. Gupta, and J. Grudin. Using Web annotations for asynchronous collaboration around documents. *Proceedings of CSCW'00*, p 309-318, 2000.
- [11] 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.
- [12] Cascading Style Sheets. <http://www.w3.org/TR/REC-CSS1>
- [13] Cascading Style Sheet 3 module: Ruby. <http://www.w3.org/TR/2001/WD-css3-ruby-20010216/>
- [14] B. Chang, J. Mackinlay, P. Zellweger, T. Igarashi. A negotiation architecture for fluid documents. *Proceedings of UIST'98*, p 123-132.
- [15] S. Cousins, M. Baldonado, A. Paepcke. A Systems View of Annotations. *Xerox PARC Tech Report P9910022*, April 2000. <http://www.parc.xerox.com/istl/members/baldonado/tr00-notable.pdf>
- [16] Document Object Model. <http://www.w3c.org/TR/REC-DOM-Level-1/>
- [17] L. Denoue and L. Vignollet. An annotation tool for Web browsers and its applications to information retrieval. *Proceedings of RIAO2000*, Paris, April 2000.
- [18] Dublin Core Metadata Initiative. <http://dublincore.org/>
- [19] G.W. Furnas. Generalized fisheye views. *Human Factors in Computing Systems, Proc. CHI '86 Conference*, Boston, April 13-17, 1986, 16-23.
- [20] 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.
- [21] 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.
- [22] K. Grønbaek, L. Sloth, and N. O. Bouvin. Open hypermedia as user controlled meta data for the Web. *Computer Networks*, (33):553-566, 2000.
- [23] 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 World Wide Web Conference*, p 253-267, 1999.
- [24] K. Grønbaek and R. Trigg. *From Web to Workplace: Designing Open Hypermedia Systems*. MIT Press, Boston, USA, July 1999.
- [25] K. Grønbaek and R. Trigg. Toward a Dexter-based model for open hypermedia: Unifying embedded references and link objects. *Proceedings of ACM Hypertext '96*, p 149-160, 1996.
- [26] W. Hall, H. C. Davis, and G. Hutchings. *Rethinking Hypermedia: The MicroCosm Approach*. Kluwer Academic, Norwell, USA, 1996.
- [27] F. Halasz and M. Schwartz. The Dexter hypertext reference model. *Communications of the ACM*, 37(2):30-39, Feb. 1994.
- [28] T. Igarashi, J. Mackinlay, B. Chang, P. Zellweger. Fluid visualization of spreadsheet structures. *Proceedings of Visual Languages'98*, 1998.
- [29] iMarkup: Annotate, organize and collaborate on the Web. http://www.imarkup.com/products/annotate_page.asp
- [30] J. Kahan, M. Koivunen, E. Prud'Hommeaux, and R. Swick. Annotea: An open RDF infrastructure for shared Web annotations. *Proceedings of the WWW10 International Conference*. Hong Kong, 2001.
- [31] D. Levy. I read the news today, oh boy: Reading and attention in digital libraries. *Proceedings of ACM Digital Libraries'97*, 202-211, 1997.
- [32] C. Marshall. Toward an ecology of hypertext annotation. *Proceedings of ACM Hypertext '98*, 1998.
- [33] J. Nielsen. Jakob Nielsen's Alertbox for January 11, 1998. <http://www.useit.com/alertbox/980111.html>
- [34] 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.
- [35] T. A. Phelps and R. Wilensky. Robust intra-document locations. *Proceedings of the 9th World Wide Web Conference*, p 105-118, 2000.
- [36] Resource Description Framework (RDF). <http://www.w3.org/RDF/>
- [37] M. Roscheisen, C. Mogensen, and T. Winograd. Beyond browsing: Shared comments, SOAPs, trails, and on-line communities. *Proceedings of the 3rd World Wide Web Conference*, 1995.
- [38] B. Schilit, G. Golovchinsky, M. Price. Beyond paper: supporting active reading with free-form digital ink annotations. *Proceedings of ACM CHI '98*, 1998.
- [39] V. Vasudevan and M. Palmer. On Web annotations: Promises and pitfalls of current Web infrastructure. *Hawaiian Int'l Conf. on Systems and Software*, 1999.
- [40] U. K. Wiil and J. J. Leggett. The HyperDisco approach to open hypermedia systems. *Proceedings of ACM Hypertext '96*, p 140-148, 1996.
- [41] 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.

- [42] XML Linking Language (XLink).
<http://www.w3.org/TR/xlink/>
- [43] XML Pointer Language (XPointer).
<http://www.w3.org/TR/xptr/>
- [44] P. Zellweger, N. O. Bouvin, H. Jehøj, J. Mackinlay. Fluid annotations in an open world. *Proceedings of ACM Hypertext 2001*, pp. 9–18, 2001.
- [45] P. Zellweger, B. Chang, J. Mackinlay. Fluid links for informed and incremental link transitions. *Proceedings of ACM Hypertext '98*, p50-57, 1998.
- [46] P. Zellweger, A. Mangel, P. Newman. Authoring fluid narrative hypertexts using treetable visualizations. *Proceedings of ACM Hypertext 2002*, to appear.
- [47] 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.
- [48] P. Zellweger, J. Mackinlay. The Fluid Reading Primer: Animated decoding support for emergent readers. *Proceedings of ED-MEDIA 2001*, Tampere, Finland, 2001.

Vitae



Niels Olof Bouvin is a post doc at the Department of Computer Science, University of Aarhus, Denmark. His research interests include open hypermedia systems, Web augmentation, structural computing, and collaboration on the Web. Niels Olof Bouvin received his Ph.D. in 2001 from the Department of Computer Science, University of Aarhus, Denmark.



Polle Zellweger received her PhD in computer science from the University of California at Berkeley, where she focused on interactive source-level debugging of optimized programs. Motivated by a long-term goal to improve people's interactions with documents and each other, she has explored a wide variety of topics in user interfaces, hypertext, multimedia, electronic books, and collaborative work since joining Xerox PARC in 1984. She has served as a member of the editorial board of ACM Transactions on Information Systems. She was a visiting professor at the University of Aarhus, Denmark in the 2000-2001 academic year.



Kaj Grønbaek is professor at the Department of Computer Science, University of Aarhus, Denmark. He finished his master's degree in 1988 and his Ph.D. in 1991 from the Dept. of Computer Science, University of Aarhus, Denmark. His research interests are: Hypermedia; Multimedia, CSCW; Interactive Workspaces, Participatory Design; User interface design; object oriented tools and techniques for system development. He is serving as a member of the editorial board of the NRHM and CSCW journals..



Jock Mackinlay received his PhD in computer science from Stanford University, where he pioneered the automatic design of graphical presentations of relational information. He joined Xerox PARC in 1986, where he collaborated with the User Interface Research Group to develop many novel applications of computer graphics for information access, coining the term "Information Visualization". Much of the fruits of this research can be seen in his recently published book, *Readings in Information Visualization: Using Vision to Think* (Morgan Kaufman, 1999, co-authored with Stuart Card and Ben Shneiderman). He is a member of the editorial board of ACM Transactions on Computer-Human Interaction. He was a visiting professor at the University of Aarhus, Denmark in the 2000-2001 academic year.