# ConTexts: Adaptable Hypermedia

m.c. schraefel

Department of Computer Science and The Knowledge Media Design Institute,
University of Toronto, Toronto M5S 3G4 Canada
mc@cs.toronto.edu

**Abstract.** ConTexts is an implementation of and proposed design model for an adaptable hypermedia system. ConTexts provides an easy yet powerful way to author the interactive, adaptable effects of adaptive hypermedia without Natural Language Generation modeling. We present an overview of the architecture and design methodology for ConTexts, its relationship to other technologies, and future research directions for this adaptable hypermedia.

## 1 Introduction

Adaptive Hypermedia requires an explicit implementation of a user model in order to render versions of a hypermedia document appropriate to the various profiles represented in the model [4]. The user model, while the most critical element of adaptive hypermedia, may also be the most intrigued of the development process. The consequence is that the benefits of adaptable hypermedia are largely out of reach for many hypermedia projects.

In this paper we present an approach to adaptive hypermedia that does not require an explicit user model implementation, but which supports the other significant components of adaptability identified by Brusilovsky [2] such as selective link hiding, page component sorting, link annotation, and stretching and collapsing page components. The result is an *adaptable hypermedia* approach called ConTexts. This approach allows rapid deployment of sites/virtual documents which are either sufficiently adaptive for the type of document they enhance not to warrant an explicit user model implementation, or which can act in concert with user–model–based adaptive hypermedia as the user model becomes available.

### 1.1 ConTexts' Context: Intensional HTML and IML

The ConTexts project began in the Intensional Programming community independent of work in Adaptive Hypermedia. The points of confluence between these approaches, however, suggest that the models complement each other. ConTexts began in 1997 as part of a project called Intensional HTML [16], [12]. The term "Intensional" refers to intensional semantics in Montague Logic which in part envisions a way to describe a range of possible, multi-dimensional worlds in which certain laws hold true for certain instances of those worlds. An intension refers to the range of possible worlds whereas an extension refers to an instance from that range. Our goal was to create versionable web sites as intensions that could be rendered on demand by users as specific extensions. Our solution was Intensional HTML (IHTML): a set of tags that extended HTML to support multidimensional pages and an engine for processing those dimensional definitions and rendering them as web pages. Within the evolution of IHTML, I developed the ConTexts model as a "rhetoric" or design methodology for intensional web sites, described in section 3, below. Over the past year, IHTML has been developed into a complete language, Intensional System Extensions, or ISE (pronounced "izzy") [14]. The result for intensional web page authorship has been Intensional Markup Language (IML), an evolution in IHTML which makes these versionable web pages more richly functional while improving ease of authorship.

### 1.2 Related Work

Besides delivering adaptable hypermedia functionality, ConTexts brings to the web some of the not easily available (or authorable) functionality of early hypertext systems like Shneiderman's Hyperties (in Gestures, described below) [10]; Xerox's Notecards [5] (in Tangents, below) and Nelson's Strechtext (in Focus, below) [8]. IML encapsulates these effects in simple tags (see section 2). The simplicity of IML's tags for document rendering separates it from the interesting modeling for dynamic content of the Virtual Document Interpreter [9]. Similarly, ConTexts and IML are distinct from adaptive systems like PEBA-II [6] or ILEX [3] which also render on dimensions like expertise (novice/expert) because IML is *not* a Natural Language Generation, AI-based, system. Indeed, its non-NLG implementation separates ConTexts from most adaptive hypermedia implementations. Further, while ConTexts tracks users' anonymous interactive renderings of the content throughout a session with its Log Roll, it does not seek to use this information, as per Joanna Moore's work on Dialog History [7], to refine the current session interaction with the user. With a ConText, the user, not the system, is always the agent of change. Hence ConTexts is a dynamically adaptable, rather than adaptive system.

## 2 Architecture and Implementation

The ConText architecture is relatively simple: a source document and any documents referenced by the source's "include" tag are additionally marked up in HTML with IML dot notation version tags. The source is preprocessed by the Intensional Markup Language plug-in for the Apache Web Server. This process converts the version information in the source into standard HTML. This revised HTML source is then sent to the web browser for display. The display page also includes the embedded links which allow users to re-render another version of the page via any available dimension. As well, the version algebra [1], [15] in IML's processor returns at least a best match for a version request. For instance, in a version request where the degree of detail is set to "high" IML will render all components matching "high." If a "high" level component is not available for some part of the site requested, the best available match for the request, “medium” for instance, will be returned. Instrumented with a version request logger, the Log Roll, the log process tracks version requests to indicate which requests have been met exactly and which have not. This allows systems designers to focus on which parts of the site need better version support and which receive fewer requests for particular versions. The log can also act as an aid in cases where developers wish to supplement parts of the site with a user model–based adaptive system. The log makes apparent whether or not users are getting to components of the site which are necessary for some specific task. This can guide designers in determining how to better hide or annotate links and paths in the user modeled components. Users as well can render the log as a record of their session with the site.

## 2.1 Implementation

Much of the implementation detail for ConTexts has been covered elsewhere [13], [15]. In the following sections, we present only the main aspects to facilitate our discussion of the ConTexts design methodology.

**Version Tags.** The key attribute of a ConTexts source page is the dimensional information which is embedded in the HTML source with what is referred to as a “version code”. For instance, if an author of page `fred` wishes to make a French component available in that web page, the author wraps the French content of that page with named dimension definition markers, such as: `.bdef dimension_name: + lang:French` *`French Content goes here`* `.edef`. To make this version of the content available to a user, a version link (intensional pointer) is written in the source as `.ipt dimension_name + lang:French "Select French version of component"`. This appears as a regular link in the rendered web page. If the user clicks “Select French version of component” from the resulting page, the current component will be re-rendered in French. There are several advantages to this tag approach to version management. The version information for the page is stored in the URL, eg `http://www/fred.ise<lang:French>`. The version information, therefore, is *neither stored on the server nor stored as cookies on the client.* Also, these dimensionally encoded URLs can be bookmarked and shared with others, anonymously. Version codes also easily maintain state from one page to the next.

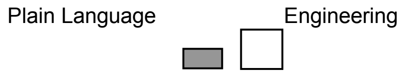
**Transversions**. Dimensional information can be triggered globally or locally within a site. For example, in a ConTexts prototype available online [13], we present a topic, Sampling, as part of an electronic text on Digital Signal Processing (DSP). The prototype includes the dimension `expertise`, which has two attributes: `plain_language` and `engineering`. By clicking a global expertise link, the user can re-render the entire site as the engineering version. Similarly, the user may select a local, component–level expertise link to re-render *just* that component in `plain_language`. These version controls are referred to as *transversional links.* The user’s selection of transversional links re-renders both the content and the navigational links appropriate to that dimension, thus enabling adaptive effects like link annotation, hiding, removing or disabling.

## 3 ConTexts Rhetoric

Adaptable hypermedia like ConTexts creates specific needs to describe clearly the function and consequence of link actions in order to reduce the *noise* of a link. To that end, we postulate the following link lexicon of *quiet* link signifiers. For the most part, we draw on link techniques that are in practice, but which so far have not been collected to represent a link rhetoric: a consistent way of communicating link intent.

**Focus.** In ConTexts, focus is a compound (two part) link signifier that indicates the site where information will be expanded into the current point. A **bold faced** term followed by a "[+]" in a document indicates that in clicking the plus, the user will see more information about the emphasized term *at that location in the text*. Once the text is expanded, the "[+]" becomes a "[-]". Clicking on the minus collapses the text at that point. This effect is similar to Nelson's Stretch Text. What we emphasize is the consistent and unambiguous way to indicate what these "+/-" signifiers will actually expand or collapse in the document. Focus links can also be established at the top and bottom of a page to indicate degree of detail can be expanded or collapsed globally (page or site wide) rather than only locally (at that location in the text).

**User Level.** User level is a link signifier that indicates current user level selected. In the DSP example, there are two user levels: Engineering and Plain Language. A user level link indicator, separate from the text, makes this option apparent and selectable at the header of any component that can be displayed in this way:



The taller block is the currently active dimension; the shorter block is the non-active dimension. Clicking the non-active Plain Language block makes it active (tall and light) and the Engineering block non-active (short and dimmed). This dimensional shift can be rendered both globally (site wide) and locally (per component).

**Gestures.** A gesture is another compound signifier for activating a link. A capitalized, colored term in the document indicates a gesture-enabled link. In the first part of the gesture, the user "rolls over" (also known as "brushes") a term, causing either a definition of the term to appear or specific information about the target to appear. The user may gain enough information from the rollover not to need to click on the term for further information. The second part of the gesture, therefore, is a click to get to the link target. The click on this link opens a new window ( a *tangent window*) on the topic specified by the that term and and at the same user level as the originating click.
**Tangents.** A tangent window described above is a target indicated by the click-through of a gesture link. The tangent window opens off side (at a tangent to) the current window allowing the originating context to remain visible while the user focuses on the tangent. Window placement, in this case, helps demonstrate relations of one point to another, and maintains the context of the given subject.

**Light, Medium, and Heavy Weight Clicks.** The above link descriptions can also be classified relative to the degree that they change the current document state when activated. Focus and user level clicks are *light* since they indicate the link click only increments the current state of the current page. The click that is the second part of a gesture link is *medium* weight since the user knows from the rollover what *kind* of target will appear, and how it will appear, but the context does shift to a new, related topic/window. ConTexts employ light and medium weight clicks. A *heavy* click refers to a link which does not provide clear information about the kind of target to be invoked. Heavy link signifiers also have a high risk of being noisy. For example, a heavy click is implied by a list of links that give only chapter number information. The user may guess that the link leads to the sixth chapter, but has to click to find out what the chapter topic is. Heavy clicks violate the *quiet signifier* paradigm underlying the ConText document model.

## 4 Future Work and Conclusions

**Relationship to Other Technology.** There are many possible ways to implement adaptable/intensional hypermedia. XML may be one; HyperWave may be another. The specific implementation has been incidental to the ConTexts model. IHTML/IML is, however, a robust easy-to-author enabling technology that works well with Java, Javascript and XML, without depending on those technologies or their browsers to create intensional, adaptable hypermedia.

**Future Research**. Aggregation of ConTexts components is supported by IML but has not been explored fully in terms of possibilities for sorting and presenting loosely structured web components. We are currently investigating this. As well, ConTexts versioning has mainly been applied to hypertext. We are also applying this user-directed adaptable technique to multimedia for video-enabled, versioned, software support.

**Conclusions.** This paper presents an overview of the architecture, implementation and design of ConTexts, an intensionally modeled, adaptable web-based hypermedia. ConTexts is a low cost system to implement, author and administer. It is a feasible approach to bring adaptable hypermedia even to small scale sites. Its link rhetoric assists communicating the effects of these state change requests. While similar in functionality to adaptive hypermedia systems, as well as to some pre-web hypertext systems, the differences in this user-defined, intensional, adaptable model may complement and expand opportunities for adaptive hypermedia application and research.

## Acknowledgements

Graeme Hirst, David Modjeska, Melanie Baljko, U of Toronto; J.H Snyder, AT&T Research Labs. Research in ConTexts is supported by the National Science and Engineering Research Council of Canada, Bell University Labs, Canada and Communication and Information Technology Ontario.

## References

1. Brown, G. Intensional HTML, a Practical Approach. Masters Thesis, University of Victoria. (1998)
2. Brusilovsky P. Methods and techniques of adaptive hypermedia. User Modeling and User-Adapted Interaction 6, 2-3, (1996): 87-129
3. Dale, R., J. Oberlander, M. Milosavljevic and A. Knott. Integrating Natural Language Generation and Hype text to Produce Dynamic Documents. Interacting with Computers, (1998) 11(2), 109-135
4. DeBra, Paul. Design Issues in Adaptive Hypermedia Application Development. 2nd Workshop on Adaptive Systems and User Modeling on the WWW. Toronto (1999) 29-39.
5. Halasz, Frank G, Thomas P. Moran and Randall H. Trigg. Notecards in a Nutshell CHI/GI 1987 conference proceedings on Human factors in computing systems and graphics interface (1987) 45 – 52
6. Milosavljevic, M., Tulloch, A., and Dale, R. Text Generation in a Dynamic Hypertext Environment. In Proceedings of the Nineteenth Australasian Computer Science Conference, Melbourne, Australia. (1996) 417-426
7. Moore, Johanna D. Participating in Explanatory Dialogues: Interpreting and Responding to Questions in Context. MIT Press, Cambridge, Mass. (1995)
8. Nelson, T. Dream Machines / Computer Lib. Microsoft Press, Redmond, Washington (1987)
9. Paradis, Franois and Anne-Marie Vercoustre and Brendan Hills, A Virtual Document Interpreter for Reuse of Information. Proceedings of Electronic Publishing '98, Saint-Malo, France. Lecture Notes in Computer Science 1375, 1-3 April (1998) 487-498
10. Schneiderman, Ben. User interface design for the Hyperties Electronic Encyclopedia. *Proceeding of the ACM conference on Hypertext*, (1987) 189 - 194
11. schraefel, m.c. A thousand papers for ISLIP '97, The Tenth International Symposium for Intensional Programming Languages, Victoria, B.C., 1997, 41-45
12. schraefel, m.c. Talking with Antigone. Ph.D. Dissertation, U of Victoria, Canada. (1997)
13. schraefel, m.c., P.F. Driessen. Sampling. A first course in Digital Signal Processing; A ConText Prototype. (1999) http://lucy.uvic.ca/mc/proto/sampling
14. Swaboda, Paul. Intensional Systems Exstensions. Masters Thesis, U of Victoria, 1999
15. Wadge W., G. Brown, m. c. schraefel, T. Yildirim, Intensional HTML, Proc 4th Int. Workshop PODDP '98, Springer Verlag LNCS 1481 (1998) 128-139
16. Yildirim, Taner. *Intensional HTML* Masters Thesis, U of Victoria, Canada. (1997)