

# 20 years On: The Dexter Model of Hypertext and its Impact on Web Accessibility.

Robert Dodd

Accessibility Research Centre, University of Teesside, UK

r.dodd@tees.ac.uk

In 2008 the Dexter Model of Hypertext will be 20 years old. Even after 20 years, the structure of Web documents, and the syntax we use to describe them and their relations to one another is still noticeably bound up with that 1980's view of content navigation. With the W3C having launched a consultation exercise on HTML 5, and a browser war imminent over the future of ECMA/JavaScript, it is time to reconsider what the discipline of accessibility requires of a hypertext model. This article considers both the original Dexter Model, and, briefly, the related Amsterdam Model of Hypermedia, in terms of the needs of assistive technology, and proposes an alternative, extended view of Hypertext of which the original Dexter Model can be considered a subset.

## Introduction

Web page accessibility is usually considered in terms of the WAI guidelines for constructing accessible websites [1] when using existing web mark-up languages, particularly HTML [2]. HTML itself is one of a family of hypertext mark-up languages that express the characteristics of the Dexter Reference Model of Hypertext [3]. The Dexter Model is a view of navigation between, and within, electronic documents, and is intended as a reference set for the character and capabilities of hypertext, rather than as a notation in itself.

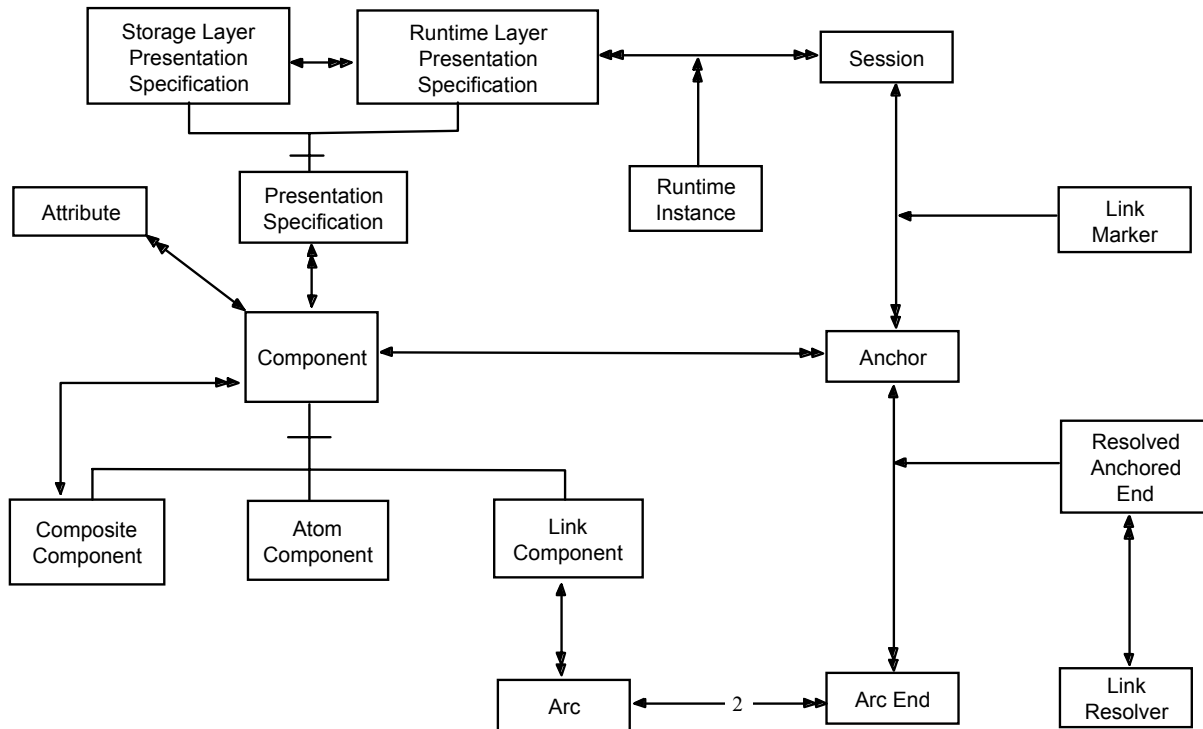
HTML is one of a number of competing hypertext systems that originated in the late 1980's. At that time, the capabilities of these emerging hypertext systems varied widely. For example in some systems, links could be navigated bi-directionally (HTML has only unidirectional links). Some systems also allowed for 'dangling' links that were resolved at run-time. In 1988 an attempt was made to define the meaning and character of hypertext systems as expressed by the most prominent hypertext systems of the time, with the Dexter Hypertext Reference Model as the result. Consequently, Dexter became the superset of what hypertext can be, with individual hypertext systems considered to be implementing subsets of these defined capabilities.

Even today, almost 20 years after those initial discussions at the Dexter Inn, New Hampshire (hence the name), a casual search of the ACM digital library brings up over 100 references to the model, including some dated 2007. In those 20 years, amendments have been suggested to the Dexter Model, particularly with regard to the demands of multimedia. One notable multimedia variant of the Dexter Model is the Amsterdam Model of Hypermedia [4], and reference will be made to the Amsterdam Model throughout this article.

## The Dexter Hypertext Reference Model

A simplified conceptual diagram of the Dexter Hypertext Reference Model, created for this paper, is shown in Figure below. The key element is *Component*. A component is a document that a user may navigate from or to. That document may be an *Atom* or a *Composite* component. An Atom Component would be a simple web page. A Composite

Component would be a web page that included a <frameset>, an <iframe>, or a link to another <object> such as a video clip.



**Figure 1 – The Dexter Reference Model of Hypertext**

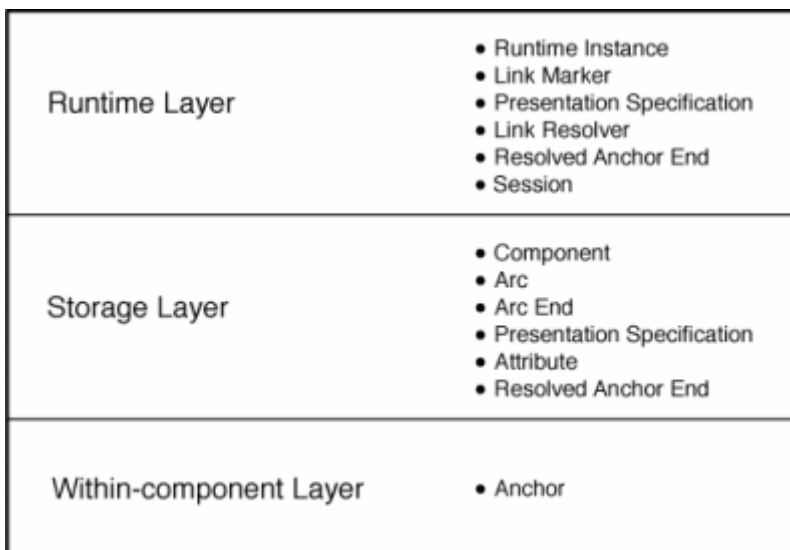
One further type of component is shown: the *Link Component*. The Link Component is where the simple analogy with HTML breaks down. In the Dexter Model, a hyperlink is itself a component that describes the possible navigation between other components. Links are not limited to describing a hyperlink between two documents, it may describe a link between many documents. A web example would be clicking on a link in one <frame> that caused multiple frames to reload with new content; in HTML this can only be achieved through the use of additional scripting, but is supported directly by the Dexter Model. This of particular interest to assistive technology, where it may be necessary to augment the target of a hyperlink to trigger, say, additional ear-cons for users with low vision or dyslexia.

Links between components are formally between *Anchors*. An *Anchor* expresses the fact that there is location within a component that may be the endpoint of a link. The Dexter Model itself, does not consider the internal structure of the component, but only identifies that there are locations within the component that are navigable. All documents have at least one anchor: their start. Figure describes the relationship between Link Components and Anchors in terms of *Arcs*. The term *Arc* does not appear in the Dexter Model, but is used here to describe potential navigation between any two components referenced by a Link Component. An *Arc* is considered to have two ends: a source and a target.

The Dexter Model allows for hyperlinks to be resolved at runtime. That is to say, a Link Component may define the linked components by use of a rule or formula. This would be equivalent to a hyperlink with a target *Arc* that is "the results of a Google search on the term accessibility". In this case, each result of the search would be a Link Component, with the source being the original target *Arc*, and the target being the link returned by Google. This

capability within the Dexter Model comes from allowing hyperlinks to be components, so that a Link Component may reference another Link Component. Such resolution, at runtime, is the responsibility of a *Link Resolver* that resolves *Arc Ends* to *Anchors*. This capability is of interest to assistive technology, in that an insight into the *Link Resolver* provides an insight into the semantics of the rendered content; in this case that it is an arbitrary length ordered list of navigable links, rather than, as with HTML, a table of links without an understanding that the table may continue onto further web pages.

Figure expresses the Dexter Model as a single relational diagram. In fact, the Dexter Model is formally expressed as three distinct layers as shown in Figure below; the relevant items from Figure are listed for each layer.



**Figure 2 – The three layer Dexter Model**

Each Component within the Dexter Model is associated with a *Presentation Specification*. The *Presentation Specification* describes how an individual component should be rendered to the user. A web example would be the styles assigned to the content of a web page (the actual content and its semantic meaning are contained within the Within-component Layer for that component).

There are two types of *Presentation Specification*: those within the Storage Layer, and those contained within the Runtime Layer. Those within the Storage Layer provide hints to the Runtime Layer. Those in the Runtime Layer define precisely how the component is to be rendered. When there is more than one *Presentation Specification* for a single component, it is for the Runtime Layer to resolve any conflicts, either through a selection, or a merge, process. *Presentation Specifications* are of direct interest to assistive technology, in that they help separate content from presentation. Further, the concept of multiple *Presentation Specifications* provides the capacity to offer multiple alternative representations of content within the one model, selected at runtime to match user capability and preference. An example within HTML is the ability to provide alternative Cascading Style Sheets.

Any Component may have a *Presentation Specification* including Link Components. *Presentation Specifications* can describe the rendering of, say, an individual web page, some composite web pages such as <frameset> based menu systems, or describe the presentation

of a hyperlink. Note that this is the entire presentation of the link, and not just the anchors within the documents.

Components are represented at runtime by *Runtime Instances*, and Anchors by *Link Markers*. There may be multiple Runtime Instances for a single Component. A web example of this would be when text is presented visually and concurrently as text-to-speech assistive technology. Similarly there may be multiple Link Markers for a single Anchor.

The Dexter Model also provides the concept of a *Session*. A Session represents a single user interacting with the content, and includes a history of user interaction during the session. Selection decisions between competing Presentation Specifications may use this history to help guide the selection process. That is to say, the Dexter Model allows for adaptive behaviour in content presentation and user interaction.

Adaptation and adaptivity are further supported in that not all components or links need be presented at runtime. The Dexter Model allows for a comprehensive collection of Components, with selection of a subset made at runtime, based on Presentation Specifications, Session history, and component *Attributes*. Attributes are name/value pairs that help describe a component. The Dexter Model does not specify any Attributes, it simply allows for them to help inform and guide the selection of Presentation Specification and Runtime Instance. Note that Anchors cannot take attributes directly; attributes relating to Anchors belong to the Link Components that reference them. This capacity to tag a component with attribute values is of particular use in assistive technology in that it allows for equivalent meaning between individual Components to be identified, for example to say that an item of text is equivalent to an image; the 'alt' tag in HTML is an example of this.

## Expressing Hypermedia

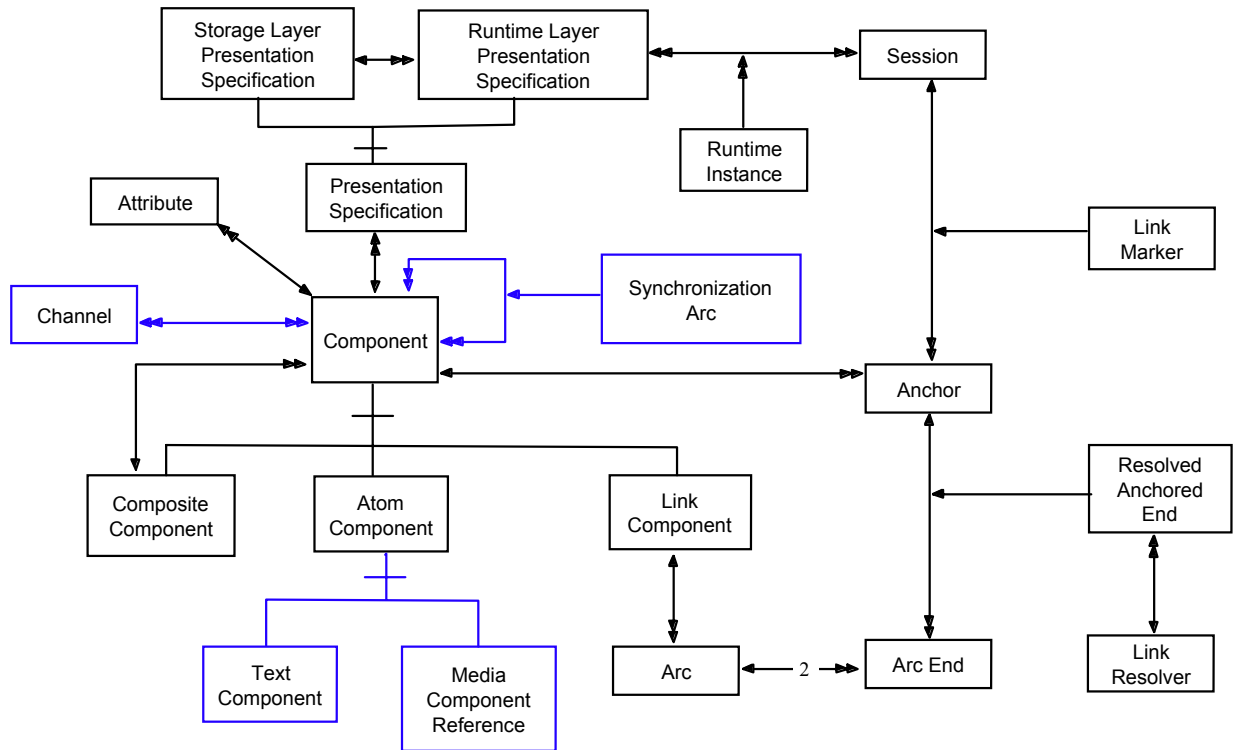
The sheer number of papers published that set out to modify or extend the Dexter Reference Model points to a number of underlying problems with the model. Perhaps the most common complaint is that the Dexter Model does not have the concepts of time and synchronization; this is particularly a problem for multimedia applications. One of the proposed solutions to this is the Amsterdam Model of Hypermedia [4]. The Amsterdam Model extends the Dexter Model by adding *Synchronization Arcs* to the model. Figure shows a simplified Amsterdam Model, with modifications to the Dexter Model highlighted in blue.

Synchronization Arcs allow components to be described in relation to each other e.g. "starts 3.5 seconds after" or "finishes at the same time". This is particularly useful for assistive technology where captioning is required, or where text and image animation is augmented with ear-cons.

Related to synchronization is the issue of competition for resources; the resource may be screen acreage, or audio channels, or haptic feedback. The approach taken in the Amsterdam Model is to provide *Channels* through which Component communicate with the user. Channels are described within the Storage Layer, as is the allocation of Components to Channels. It is the responsibility of the Runtime system to regulate competition for channels with regard to the defined Synchronization Arcs.

The Amsterdam Model extends the concept of the Atom Component, providing two sub-types: *Text Component* and *Media Component Reference*. This distinction exists in modern versions of HTML such as HTML 4, which uses the <object> tag to identify external media. This

extension is necessary as the Dexter Model maintains all content within the model which does not sit well with streaming media where the content may play out from a remote source.



**Figure 3 – The Amsterdam Model of Hypermedia**

A recurring issue with the Dexter Model is effective selection of alternate content. This is possible through judicious use of Component Attributes to identify equivalent content. The Amsterdam Model formalizes this further, by limiting equivalent content to the contents of a *Composite Component*. In this scenario, a video clip available in 320x240 and 640x480 formats are both Atom Components contained within a Composite Component. The Runtime Layer chooses the appropriate format based on device capabilities; this is an important feature for assistive technology where alternative content well beyond simple alt text may be required.

### Adaptability Issues

Even allowing for Hypermedia extensions to the Dexter Model, significant outstanding issues remain when adapting content.

The first, and most significant issue revolves around the stability of Components. The Dexter Model describes navigation between and within components, and it is possible to augment the model with additional Link Components, with user-appropriate sets of Link Components chosen at runtime. However, problems occur with Composite and Atom Components.

Consider as an example a bus timetable website. How the timetable should be presented to the user will depend upon a number of factors, including for example screen size and resolution, required font size for the user, and the level of detail to be presented. In the case of screen resolution, the smaller the screen resolution, the less information that can physically fit on the screen at any one time. Depending upon the chosen solution, the number of required web pages to present the timetable will vary, yet the underlying content remains the same. If the

point at which further adaptation for the user occurs, is after the division into web pages, there is no practical way to "knit" the timetable together in alternative forms. At best, the adaptation process can restructure the individual web page. Whilst website design is not normally expected to adjust to radically different screen sizes and shapes, assistive technology often is.

A second issue revolves around the synchronization of contemporaneously presented content. The Amsterdam Model goes some way towards addressing the issue, but is constrained by a lack of semantic knowledge. Taking adaptation using text-to-speech as an example, if some components are already generating audio, it is difficult to know when and how to present audio content, or know the relationship between presented text and existing audio. Consider an on-line immersive game, where characters speak out loud within a soundscape of ear-cons (say providing city-like, or jungle-like noises). Augmenting such an environment requires an understanding what audio may be safely removed or faded, what audio may be delayed, and what audio is already text-to-speech. Whilst it is possible through judicious use of component attributes to provide this information, there is no formal mechanism to dynamically route content between Amsterdam-like Channels. A second example is a mobile phone: how, in the Dexter Model, can the user interface be augmented with text-to-speech without an understanding that a "new message arrived" ear-con may be delayed until the end of speaking an email, or that text-to-speech of an email is less important than an "incoming call" ear-con? In each case a clear semantic model is needed.

A third issue concerns how alternate content is identified and selected. As noted earlier, judicious use of component Attributes can identify equivalent components, and this can be formalized further, as in the Amsterdam Model, by grouping alternate content within a Composite Component. The primary objection to this approach is that only like-for-like alternatives are supported; whilst a video clip may be replaced with text, or subtitles played with the video, you cannot restructure the content. As a simple example, the ordering of content in a list may need to vary between visual and audio design spaces: in text-to-speech environments, the most important menu options may need to be listed first, but in a visual environment the menu may be required to be grouped more logically. Resolving this case within the Dexter Model requires duplication of content to create two Components, one for each view, each with its own set of Link Components. Any further decomposition of content below the menu (however presented) requires further duplication.

All three issues stem from a common complaint: it is not possible to describe content navigation entirely independent of content semantics. How content is organized into components for presentation, and factors such as the relative importance of each component, impacts upon the identification of Atom and Composite Components in the Dexter Model.

## **HTML**

HTML is the World Wide Web Consortium's (W3C) primary mark-up language for the web. It is based on a book metaphor of pages of content that is further organized under sub-headings within a page (sub-headings do not cross page boundaries, which is where the metaphor begins to break down). As a notation it addresses a number of concerns: content structure, presentation, interaction, and navigation.

In terms of the Dexter Model, HTML content exists partly inside the "Within Component Layer", and partly as a hierarchy of *Composite Components* in the "Storage Layer" as websites and web pages. It is only the component hierarchy that is directly addressed by the Dexter Model. One complication with HTML is that it allows for dynamic modification of web pages using scripting languages such as ECMA/JavaScript; scripting is typically attached to user actions such as the "onMouseOver" and "onClick" attributes of many HTML elements. If the result is effective re-presentation of content, then there is navigation to a new component within the Dexter Model. So *Components* are not only web pages and websites, they may also be regions of an individual web page; so-called Ajax-based web pages can be considered to follow this model. Having the component hierarchy expressed within the scripting language is a significant concern for existing assistive technology that cannot access the underlying semantics of such nested components. A practical example of the problem is "Google Suggest" [6], a version of the Google search page which provides dynamic hints to the user as they enter search terms. Screen reading application Jaws for Windows [5] and VoiceOver [7], both fail to handle the dynamic page updates correctly on what is otherwise a trivial web page.

Presentation within HTML is guided by a combination of explicit elements such as <b> (meaning bold), and attributes describing presentation style for individual elements and classes of element. All such presentation information is formally only a hint to the web browser, and consequently are *Presentation Specifications* within the Storage Layer of the Dexter Model. As with the Within-Component Layer, the exact details of how presentation specifications are described is considered outside of the general Dexter Model, however, it is important to note that the use of scripting in HTML means that such presentation information may change, or be constructed, dynamically, affecting *Runtime Instances*. This means that *Presentation Specifications* can be hidden in the scripting, resulting in assistive technology such as screen readers working with the *Runtime Instances*, and not selecting from, say, a choice of *Presentation Specification*.

User interaction with HTML is provided through pre-defined elements that allow for list selection, text entry, button presses, image maps, and anchors. Only anchors and buttons can directly cause navigation; their existence is represented by *Anchors* within the Dexter Model. List selection and text entry may also represent *Anchors* indirectly through the use of scripting. As with hiding component hierarchy within scripting, hiding *Anchors* and the associated *Link Components* in this way, also effectively hides them from existing assistive technology.

The principal form of navigation within HTML is the hyperlink and causes (re) presentation of a web page potentially embedded within existing presented web pages. An HTML hyperlink is unidirectional *Link Component* in the Dexter Model, although some web browsers use *Session history* to provide a 'back' button and/or a breadcrumb trail. HTML follows the Dexter convention of navigation between uniquely identifiable *Anchors*. HTML itself has the concept of an 'anchor' represented by the <a> and <area> elements, however it is also possible to cause navigation to a web page programmatically using ECMA/Java scripting attached to interactive elements as described above; script-level changes are not guaranteed to appear in *Session history*, and are therefore not always visible to assistive technology.

## HTML 5

The HTML notation is now 16 years old, and reflects a subset of the Dexter Hypertext Reference Model, which itself is 20 years old. Over the 16 years, HTML has undergone a number of

revisions and spawned several complementary and replacement specifications. HTML version 5 [8] is now out for discussion and, "intends to replace HTML 4, XHTML 1.x, and DOM2 HTML specifications". As the dominant mark-up language for the Web, any changes to HTML will have a knock-on impact on the effective accessibility of the Web, both directly in terms of web pages as scripted by their designers, and indirectly in terms of adaptation by assistive technology on behalf of disabled users.

There are currently over 20 additional tags in the HTML 5 recommendation compared to HTML 4.01 [9]. Some of the new elements provide additional presentation hints (*Presentation Specifications* in Dexter Model terminology), others are concerned with content semantics.

Multimedia was not addressed by previous versions of HTML beyond the <object> element (resulting in SMIL [10], an XML notation to describe multimedia closely resembling the Amsterdam Model of Hypermedia). For HTML 5, two new multimedia elements have been included, <audio> and <video>, with <video> allowing control of embedded captions/subtitles. Both the <audio> and <video> elements allow for manipulation of *cue ranges* within the elements, and match the general thrust of the Amsterdam Model of Hypermedia as *Media Component References*, with embedded <source> elements locating the target content. On first reading of the current draft is unclear whether multiple sources may be included within a single audio or video element. It is possible to synchronize multimedia content, but synchronization is hidden within scripting with the equivalent of "onMouseUp" events on cue range completions. This means that, unlike the Amsterdam Model, there are no explicit *Synchronization Arcs* for assistive technology to follow; they do exist, but programmatically. Also missing is the Amsterdam Model's concept of *Channels*; the audio and video elements carry a large number of attributes to help with play-out and synchronization, but there is no explicit concept of queuing content on specific channels, again leaving all such work at resolving contention for resources to the browser and to explicit scripting, effectively hiding it from 3<sup>rd</sup> party assistive technology.

After the addition of multimedia to HTML, perhaps the most notable additions are elements that describe semantic meaning. Of particular interest is the <section> element which "provides thematic grouping of content" [8]; existing HTML groups content using the <div> element which in HTML 5 is limited to marking up a "group of consecutive elements" [8]. For adaptive assistive technology, thematic grouping of content may help in terms identifying groups of content that that should appear conceptually "close" during presentation, for example on the same page if content needs to be split over pages, or spatially close on a page that uses scrollbars. Associated with <section> is the <nav> component, which may exist only as a sub-element of <section>, and describes a section of a page that "links to other parts within the page" [8]. This more clearly exposes anchors and their thematic relationships to assistive technology, allowing for example, the restructuring of a web page by a screen reader. Whilst such adaptations are already possible, with anchors moveable to the beginning for a page say, the <nav> element more clearly associates with the parent section and its descriptive attributes, allowing for more informative restructuring. Also of interest are the <command> and <menu> elements that identify and group navigation within, and between pages. As with the <nav> element, this helps expose the underlying *Link Components* in a structured fashion.

## Underlying Assumptions

The underlying model of hypertext, as expressed in the Dexter Model, begins with three assumptions; (i) that navigation of content can be described independently of content structure, (ii) that content structure is static, and (iii) that navigation can be described as data elements and the relationships between.

The bus timetable example given earlier demonstrates some of the limitations of these assumptions. Whilst the bus timetable is certainly static in this context, how the timetable is to be presented affects the organization of content. Can the timetable fit on one screen? Does it need to be presented in both a detailed and a summary form? For the Dexter Model, the content (the *Components*) is only considered static once such decisions have been made, as content structure in terms of components is not at the abstract level of bus stops and times, but is at the level of groups of content (pages in HTML) that may be navigated. Any assistive technology that begins with the Dexter Model, and hence HTML, is already beginning with important organizational details pre-defined, and with the underlying abstract model (in this case busses, bus routes and times) unavailable; assistive technology can only work within the pre-defined groups/pages.

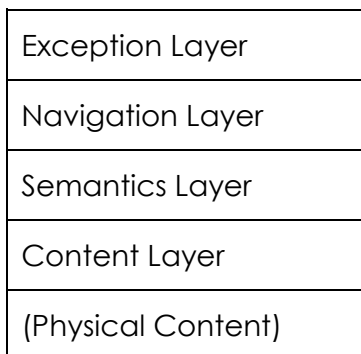
Further, the manner in which real-world bus timetable systems, such as Travelline [11] in the UK, are constructed also demonstrates limitations in the underlying assumptions. Timetable websites, and many corporate websites, are constructed using content from databases, with the specific web page presented to the user being generated on-the-fly by a web server; the layout of that web page is specified by a template that describes the page in terms of fixed content and "holes" to be filled with database content. None of this construction process is visible to the web browser, nor consequently to client-side assistive technology: all semantic meaning between pages is lost. The final constructed pages do follow the Dexter Model, but the overall *Component* hierarchy within the website is obfuscated. In practical terms, assistive technology is constrained to the current page.

HTML itself starts by constraining its use of the Dexter Model further by use of a book metaphor, with each page of the book constructed of headings, sub-headings and content (text, image, tabular). This already promotes the importance of page composition over the headings; it is the printer's view of a book, not an author's or a reader's. The hierarchy and division of *Components* becomes, largely, a collection of pages, not a collection of content. A user navigates from page to page, not chapter to chapter. Consequently, design of a website begins with choosing the number of pages independently of consideration of the user's device capabilities (e.g. screen size), or consideration of the needs of the user. HTML 5 contributes many more semantic and thematic constructs to describe content within a page, but it is still entirely concerned with that specific page.

A developing trend over the last two years has been the promotion of so-called Ajax websites. In simple terms, an Ajax website is one that replaces the "book" metaphor with a "screen" metaphor and turns web pages into computer programs. The web page is still presented to the web browser as an HTML web page, but its construction is heavily based upon ECMA/JavaScript running within the web page. Ajax web pages are also more active than traditional web pages, in that they can communicate with a web server without the user selecting a hyperlink. As with the simple template approach above, the final presented page follows the Dexter Model, but the construction process, and any semantic relationships involved are obfuscated from both the browser and any assistive technology.

## An alternative view

An alternative view the Dexter Model is as one layer of a larger document model; Figure describes documents within a five-layer model.



**Figure 4 – The Dexter Model in context**

At the lowest level of the five-layer model is physical content such as text, images, video and audio clips which may reside anywhere within the related electronic network.

The Content Layer provides identifiers for all used content. This is equivalent to the *Media Component Reference* in the Amsterdam Model shown in Figure , and equivalent to the <source> element of HTML 5, and the "src" attribute of the <img> element in HTML, but also encompasses text fragments.

The Semantics Layer provides for document composition, and for description of relationships of elements in individual documents, and between documents; the Semantics Layer may reference the same content in the Content Layer multiple times.

The Navigation Layer represents navigation within the Dexter Model, but with each *Component* in the Navigation Layer referring to an element of the Semantics Layer. Many Components may reference the same element in the Semantic Layer. An example would be multiple views of a bus timetable.

The Exception Layer provides for the Dexter Model's *Link Resolver* and *Resolved Anchor Ends*. The layer exists to illuminate the case where links break, or adaptive systems identify problems with user navigation within a website.

Separate, and in parallel with this five-layer model is the Runtime System described by the Dexter Model. The *Session*, *Runtime Instances*, *Link Markers*, and *Runtime Layer Presentation Specifications* all belong to the Runtime System. *Presentation Specifications*, as hints, may exist in any of the four layers top layers.

The five-layer model is a useful reference model in that it encompasses the entire scope of notations such as HTML, rather than only the navigation aspects. The migration of HTML 4 to 5 into this view also becomes clearer, with the new semantic elements strengthening the Semantics Layer, the <source> element strengthening the Content Layer, and the <nav> element strengthening definition of *Composite Components* in the Navigation Layer.

The enduring strength of the Dexter Model is also apparent. With Dexter *Components* in the Navigation Layer considered as counterparts to elements within the Semantic Layer, the model of Components, Arcs and Anchors remains intact. The Exception Layer provides for

clarification of link resolution issues, and as an anchor for adaptive systems, but the Dexter Model's link resolution principals remain intact.

The five-layer model also highlights inconsistencies within HTML 5, notably the new <source> element exists only for audio and video clips, but not for images or text (text may be quoted or cited, but never referenced for inclusion as an image or video clip may be). Another inconsistency within HTML 5 is the drift towards scripting rather than expressing direct document structure which can be seen in the synchronization of media in the Navigation Layer, and link resolution in Exception Layer when there is more than one arc per link( for example when reloading multiple frames). With the increase in semantics-related elements such as <section> this leaves the Semantics Layer expanding the direct description of content, and the Navigation and Exception Layers retreating into the obfuscation of scripting.

## Summary

The Dexter Reference Model of Hypertext provides a view of how users may navigate electronic documents. HTML builds upon that view, using a book metaphor to describe the content of each document, and providing hints to the web browser to help in presenting content to the user. Both the Dexter Model and HTML both date from the late 1980's and embody the view of content and content navigation prevalent at the time. HTML has undergone a number of revisions since its inception, with discussion of HTML 5 ongoing. A number of variants, and extensions to, the Dexter Model have been proposed, a notable example being the Amsterdam Model of Hypermedia.

HTML itself is part of a family of mark-up languages for the web, each supporting a subset of the features of the Dexter Model. Key constraints include unidirectional hyperlinks and simplified link components. The use of scripting within an HTML page tends to obfuscate the underlying Dexter Model, for example anchors may be hidden in script attached to attributes such as "onMouseUp" and "onBlur". Practical use of HTML when pages must be constructed from database content and/or through templating, also tends to obfuscate the underlying Dexter Model; the constructed web pages follow the Dexter Model, but may obfuscate the semantic relationships that hold the page elements together.

The current version of HTML 5 available for review, adds multimedia capabilities to HTML, and in the process extending the Dexter Model. The current proposals approximate to a simplified version of the Amsterdam Model of Hypermedia, but with synchronization left to scripting, rather than using explicit synchronization arcs. This leaves HTML 5 as being more than Dexter, but less than Amsterdam in terms of capability. HTML 5 has also begun to address a deficiency in describing semantic relationships between content, although only for content within an individual web page.

The stability of the underlying content below the Dexter Model's navigable components is a cause for concern. This instability stems from describing navigation based upon a single concrete realization of content decomposition. A simple example, showing the impact upon HTML, is the splitting of search results across web pages in Google. If the content requires restructuring for a particular visual or cognitive impairment say, restructuring becomes limited to the predefined individual page, not to the search results as a whole. With online searching, and the results, forming a significant part of modern web usage, the quality of assistive technology for the web is directly affected.

Whilst all rendered HTML web pages do conform to a subset of the Dexter Model, the visibility of that model may be obfuscated by the use of scripting within the page, for example when changing the content of multiple frames of a frameset, or implementing synchronization arcs in HTML 5. With the advent of Ajax-based web scripting, visibility problems for assistive technology have increased significantly.

The observations expressed about the Dexter Model, and about HTML 5, become clearer when the Dexter Model is viewed in a broader context. The proposed five-layer model is an attempt to do so, placing Dexter within a layered model capable of expressing both content and navigation. In doing so, the enduring quality of the Dexter Model becomes clear, as do some of its limitations.

## Conclusion

Even after twenty years, the Dexter Reference Model of Hypertext still forms the core model for web content navigation. Only with the advent explicit multimedia elements within HTML version 5, is the model challenged, and even then the underlying model requires only minor modification. Those changes move the model towards a simplified view of the Amsterdam Model of Hypermedia, with synchronization and sequencing issues left to the browser and web scripting.

The Dexter Model was always designed to be a universal set of hypertext features, of which individual systems and notations implement a subset. This is certainly the case with HTML, including HTML 5. Of the subset implemented by HTML, the use of scripting sometimes obfuscates the underlying model (both content and hyperlinks). Further, some of the non-implemented features, such as multiple alternate presentation specifications, and limited runtime link resolution, limit the capabilities of browsers to adapt to user needs.

Hypertext benefits from being placed into a broader model of context, rather than simple navigation of pre-defined content groupings as in the Dexter Model. The five-layer model presented here, showing the relationships between content, navigation, and exception is one example of such an approach, demonstrating the enduring strengths, and some of the weaknesses of Dexter, notably the instability of the model in an adaptive environment.

The five-layer model also helps to place notations such as HTML in context, allowing all the features of the notation to be discussed, rather than only navigation. Placing the proposed HTML 5 within this model helps identify inconsistencies and omissions. A more detailed model with each layer describe similarly to Figure , would go some way to providing a broader reference model for electronic documents and user interfaces.

## Continuing Work

The five-layer model presented in this article forms part of a study of adaptable hypermedia at the University of Teesside. Detailed models of each layer are used, in conjunction with semantic models of device and user capability, to allow automated construction of hypermedia interfaces adapted to user needs. Associated with this, is a prototype XML notation for describing such a system that allows for limited transcoding of HTML directly into the notation.

## References (all online references viewed 20 November 2007)

1. World Wide Web Consortium, "Web Accessibility Initiative", online "<http://www.w3.org/WAI/>".

2. World Wide Web Consortium, "HTML", online "<http://www.w3.org/html/>".
3. F. Halasz and M. Schwartz, "The Dexter hypertext reference model," *Communications of the ACM*, vol. 37, pp. 30-39, 1994.
4. [4] L. Hardman, D. C. A. Bulterman, and G. v. Rossum, "The Amsterdam hypermedia model: adding time and context to the Dexter model," *Communications of the ACM*, vol. 37, pp. 50-62, 1994.
5. Freedom Scientific Inc. product catalogue, online "<http://freedomscientific.com/>".
6. Google Inc, "Google Suggest", online "<http://www.google.com/webhp?complete=1&hl=en>".
7. Apple Inc., "Voice Over", online "<http://www.apple.com/accessibility/voiceover/>",
8. World Wide Web Consortium, "HTML 5", online "<http://www.w3.org/html/wg/html5/>".
9. World Wide Web Consortium, "HTML 4.01 Specification", online "<http://www.w3.org/TR/REC-html40/>".
10. World Wide Web Consortium, "SMIL Specification", online "<http://www.w3.org/TR/1998/REC-smil-19980615/>".
11. Traveline, "Plan a Journey", online "<http://jplanner.travelinenortheast.info/jpclient.exe>".

### About the author



Robert Dodd's PhD focuses upon models of user interaction that allow hand-held mobile devices to self-adapt to individual user capabilities, with the goal being to widen access to established content, and to establish a framework for future content design that will further widen access. The key output of the work is a user-centred model of adaptable hypermedia that is structured around the visual, audio, haptic, cognitive, and learning design spaces. Supporting, and driving this approach is a model of portable user profiling that provides the raw information upon which the adaptation is based