Weblinks: Augmenting Web Browsers with Enhanced Link Services

Daniel Roßner*
Hof University
Institute of Information Systems
Hof, Germany
daniel.rossner@iisys.de

Claus Atzenbeck
Hof University
Institute of Information Systems
Hof, Germany
claus.atzenbeck@iisys.de

Daniel Urban
Hof University
Institute of Information Systems
Hof, Germany
daniel.urban@hof-university.de

ABSTRACT

Without any doubt, creating links and navigational trails is fundamental to hypertext. The Web is the widest spread representative among all systems in the history of hypertext, although its underlying core concept is kept simple. This is why the Web's widely used link functionality is naive, supporting only embedded, unary and unidirectional links. A strength of the Web is its extensibility, giving us the opportunity to augment the current functionality of links. We showcase a browser plugin, which enables users to create and share complex links over the existing Web. Furthermore, we discuss the CB-OHS *Mother*, its link model and how this relates to existing work. The implementation adopts latest standardization efforts and is an update to older attempts of enabling external link services for the Web.

CCS CONCEPTS

• **Information systems** → *Web services*; *Web applications*; **Social tagging systems**; • **General and reference** → *Experimentation*; • **Human-centered computing** → *Collaborative and social computing systems and tools.*

KEYWORDS

hypertext; navigational hypertext; link service; Web; browser;

1 INTRODUCTION

In his keynote at the 30th ACM Conference on Hypertext and Social Media 2019, Andries van Dam appealed to the participants: "Everybody needs to read it and re-read it about once a year [...]"[1]. He was talking about Bush's article *As We May Think* [8] published in 1945. While technically outdated, Memex' underlying paradigm is iconic to the hypertext community and still serves as an inspiration for current research and technology development. From a current perspective, the WWW has "won" the race of hypertext systems by their popularity, partly (or mainly) due to its simplicity and pragmatic approach [1, 5, 7, 13].

This said, "simple" may be a misleading term, as the Web has evolved to a rather complex building of (competing) standards and technology. But it still holds true, because the Web was never meant to implement all of the bold visions hypertext researchers worked on.

One of the missing features is rich support for links. The Web offers a naive approach to what is so-called "navigational hypertext": links are unary, unidirectional and embedded in the document. These limitations hinder the creation of links with multiple targets, which can be traversed in either direction. Document-centered embedding of links in HTML is foremost a limitation for people who want to author hypertext, because they need write access for the *document* in order to insert a link anchor element.

Bush's article provides an appropriate quote that summarizes the motivation of our work: "The process of tying two items together is the important thing" [8]. Do we need a new, "better" hypertext system to be used in our daily life? In this paper we propose an evolution, not a revolution of systems by augmenting what already exists: the WWW. This brings back richer hypertext functionality, as already proposed by previous research [7, 9, e.g.]. Even solutions for sophisticated navigational links within the Web were already discussed [6, 10] and implemented in the mid 1990s [11].

Since then the user experience across new platforms and technology evolved, however, the capabilities of native links are still the same. The goal of our work is to continue former approaches by improving their underlying concepts or technologies. We do not aim at establishing a new Web standard, but want to share our efforts to give users and researchers an idea about what the Web could look like with external link services and enriched navigational link semantics.

2 RELATED WORK

The proposed implementation is framed by previously discussed ideas, projects, or specifications. This also includes some of our recent work on spatial hypertext or component-based open hypermedia systems (CB-OHS). We also consider (evolving) standards or already existing solutions.

2.1 IWInxt

This project is driven by requirements we have in the project *Next Generation Intelligent Maintenance System for the Industry 4.0 (IWInxt)*, aiming at implementing an intelligent maintenance solution for Industry 4.0 applications. Maintenance often involves many distributed documents like reports, manuals, and tutorials. Often responsible employees do not have the ability of linking relevant sections among these documents. Furthermore, such created links

*Corresponding author

[1] See keynote video at: https://www.youtube.com/watch?v=g0yx-F1FGnc&t=1100

HUMAN'20, December 4, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-8058-4/20/12.
https://doi.org/10.1145/3406853.3432663

have to be accessible by co-workers to gather and grow specialized knowledge.

2.2 Mother

Our implementation of the linking mechanism is part of the CB-OHS *Mother*, which we described in further detail in [4]. The core concept is based on an optimized version of the Dexter Hypertext Reference Model [12]. We use *external link bases* that store links as first-class entities. Access to those is granted by an external API.

Mother's is built with a three-layered architecture, separating the user interface, data and (partly intelligent) structure-aware components from each other [3]. This enables Mother to host structure services of various types, including spatial, navigational, or hierarchical structures. Dexter originated in the time before the first CB-OHS were implemented and, thus, only considered a limited number of structure types, mainly links andes collections. In order to match both worlds we had to reassemble parts of Dexter, which led to a minimized, less complex model. For Mother's linking features we focused on the navigational only, including *n*-arity and endpoint directions, while removing the support of composites, presentation specifications and any content related fields [3]. Removed features features could be (and partly are) implemented in separate components. An example is Mother's metadata service which is used by the linking service to annotate endpoints/anchors with location specifications. A more detailed discussion will follow in Sect. 3.

2.3 Hypothesis

Due to chosen requirement to use current Web technologies, the obvious way for using our linking features was to extend the Web browser. Most modern browser implementations allow some sort of plugin mechanism, which can be used to intercept the rendering process and gives access to visited resources. Therefore, we chose to implement a browser plugin for *Google Chrome* and *Mozilla Firefox*. The plugin shares some ideas with and uses code from the *Hypothesis Project*, which is "a new effort to implement an old idea: A conversation layer over the entire web that works everywhere, without needing implementation by any underlying site."[2].

Hypothesis is a browser plugin, which allows users to annotate arbitrary web pages. These annotations can be shared in groups or with the public. The claim of supporting conversation is driven by the ability to reply to annotations. Hypothesis can be used without additional plugins, if the web page is visited via the offered Web proxy[3].

As we have similar demands and requirements (in particular Web compatibility for our services), it is no surprise that our solution for a *linking layer* borrows some fundamentals of Hypothesis. For example, the linking service shows a similar sidebar and uses the same method of selecting parts of a Web page.

2.4 Web Annotation Data Model

An required feature of annotating and linking is selecting specific parts of a document. In books we use pages, chapters, paragraphs, etc. Legal texts are typically referenced with various section or sentence numbers. Web pages are accessible using a URI, possibly including a *fragment identifier*; the part of an URI that is preceded by the number sign "#". The resolution of fragments depends on the MIME type of the requested resource and is handled by the Web browser. For example, for HTML the browser searches for elements with a matching id or name[4], whereas parts of X(HT)ML can be selected using *XPointer*. Both methods require pre-structured content, as it is not possible to select only parts of an element or element overarching text. Some browsers allow "text fragments"[5], which select the first appearance of a certain text, defined in the fragment.

Selector	Description
Fragment Selector	Fragment selection as defined by the corresponding MIME type
XPath Selector	XPath selection for resources that supports the DOM (HTML, XML)
Text Quote Selector	Range of text by quoting it; prefix and suffix text can be defined
Text Position Selector	Range of text by defining character numbers of the resource
Range Selector	Range of text by defining start and end with other selectors
Refinement of Selection	Composition of selections, which refine the selection

Table 1: Selectors defined by the Web Annotation Model and suitable for HTML resources

Today, Web content is often dynamic, with frequently changing content or layout. This makes anchoring specific parts challenging. Similar to Hypothesis we adapt selectors as they are defined in the *Web Annotation Data Model*[6]. The model does not define new selection methods, but proposes "a common model and syntax to express and possibly combine selections"[7]. Table 1 lists selectors, which work with HTML resources.

3 LINK SERVICE

The browser plugin is backed by an external link service component. As described earlier, its link model is derived by simplification of the Dexter Hypertext Reference Model. Using a component-based approach, many specifications and properties can be omitted and modeled with other, fitting structure services. Figure 1 depicts the link model. A detailed description can be found in [3].

A *component* is any entity that can be identified by a URI, thus, this is not limited to Web resources. Books, for example, can be referenced with their ISBN or many research papers by their DOI; both would be valid URIs for the link service. It is important to note, that links are first-class objects and consequently themselves components with URIs. This URI is managed by the link service

[2] https://web.hypothes.is/about/, visited 2020-09-23

[3] An example of such access via proxy is https://via.hypothes.is/https://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881/

[4] https://www.w3.org/TR/html52/browsers.html#navigating-to-a-fragment-identifier

[5] https://wicg.github.io/scroll-to-text-fragment/

[6] https://www.w3.org/TR/annotation-model/

[7] https://www.w3.org/TR/selectors-states/

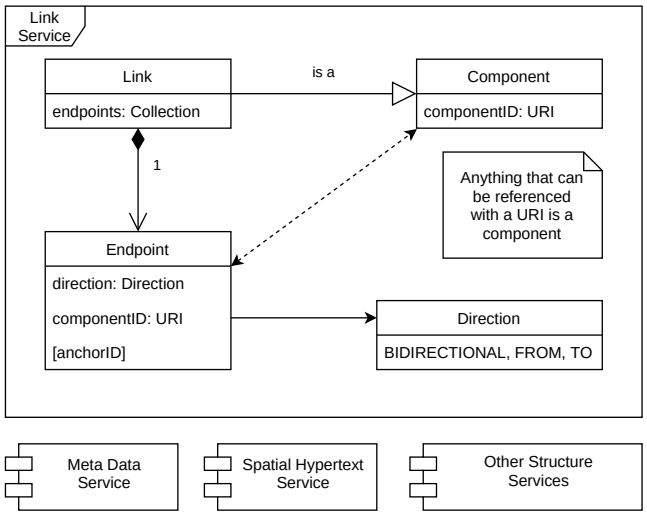

Figure 1: Link Model in context of Mother

upon creation and serves as HTTP resource, where clients can *GET*, *DELETE*, or *PUT* the underlying link object.

In Sect. 2.4 we emphasized the importance to specify a selector for *endpoints*. The Dexter model, for example, defines an "anchor value" to identify "some location, region, item, or substructure within a component"[12]. Others implement a "Selectors" collection into their link metamodel [14]. On the other hand, our simplified version does not cover such features, instead, selectors are modeled as annotations within the metadata service. That service uses a triplestore which manages triples of subjects, predicates, and objects. For a selector annotation, the object is the JSON serialized form of a selector as listed in Tab. 1. We follow the vocabulary of the *Web Annotation Data Model*, however, as components can be anything that has an URI, selectors for various types may be needed in the future. The predicate distinguishes selector annotations from other annotations; it is named *hasSelector*. Subjects are endpoints, uniquely identified by the URI of the corresponding link and the *anchorID* (see Fig. 2). An anchorID has to be unique across a link object and is optional, as long as no selectors are associated. Endpoints without selectors target the whole document. This is also the case if a client can not resolve a selector.

The metadata service augments the minimized link model, yet it is not bundled with the link service. This has the advantage that it can be used for other purposes as well. Therefore it is possible to introduce more predicates, for example, to enable comments for links or endpoints or support "presentation specifications", as suggested by Dexter.

The flexibility that comes with the use of a separate metadata service raises the question of which properties should be part of the link model and which should be added by the metadata service. For example, it is a design decision whether endpoint directions should be defined in the link model or be modeled as metadata of endpoints. Thus, the litmus test for any property is the following: Is property x needed to model navigation, that is, the trail from a point to one or many other points? For directions on endpoints, in our case, the answer is yes, while selectors are only necessary to implement a working piece of software.

4 PLUGIN IMPLEMENTATION

4.1 Technologies

As described above, today, the best way to extend the functionality of the Web is extending the software we use to explore it: the Web browser. Such browser plugins can be developed independently and are loaded as external modules. Plugins have access to the internal APIs of the browser and contents of visited websites. Especially the latter is important to augment the functionality of the Web. This access is guaranteed by an interface called the *WebExtensions API* which is provided by many browsers[8].

Browser plugins can be implemented as pages that are shown in separate tabs, popup windows, or as sidebars. Since the plugin will be used to create links on Web resources, a page in a separate tab is not the preferable solution. A popup window is also not feasible, because it would close automatically when losing the user interaction focus. Thus, a sidebar is the best solution for our use case. It gets injected into the website as an *iframe*, which opens a higher level of flexibility with respect to functionality and design without restrictions by the API. The same approach is used by Hypothesis, as discussed in Sect. 2.3.

Like other web extensions, our project consists of a content script, running in the context of the website, and a background script that runs in the context of the browser. Both scripts are developed using common web technologies: HTML, CSS, and JavaScript. To manage the user interactions in the views, we use *Knockout.js*, a lightweight library for data binding between data model and view model. Structure and dependencies of each script are managed with *Node.js* in combination with *Browserify* to make Node.js modules employable on the client-side.

4.2 Anchors View

The sidebar contains several views, each responsible for a specific task. The anchors view (as shown in Fig. 3) shows a list of all link endpoints associated with the currently shown website. Furthermore, new endpoints and links can be created. If the user decides to create a new endpoint with a click on the "new anchor" button, he/she can either assign it to an already existing link or create a new link with this endpoint. Initially, new links are stored in the local browser storage only. The reason is that the link service does not allow the creation of invalid links. A link is complete or valid if it contains at least two endpoints with at least one FROM and one TO or BIDIRECTIONAL endpoint. Valid links can be submitted to the link service.

4.3 Links View

The links view presents links managed by the link service. For inspecting a specific link, the user either clicks on an endpoint in the anchors view or use the search bar searching the link manually. The corresponding link will be presented as shown in Fig. 4. Using this view, new links can be created or existing links can be edited

[8]The WebExtensions API is a cross-browser platform by Mozilla. It is used by Firefox and compatible with the extension API of Chromium-based browsers.

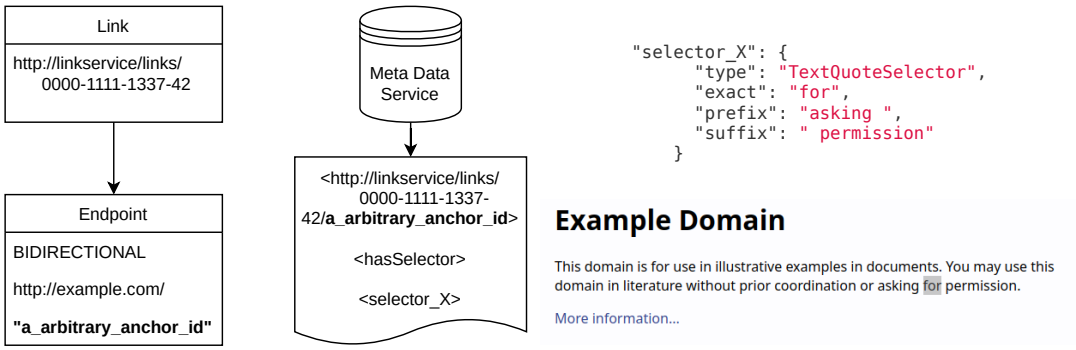

Figure 2: Endpoint on http://example.com with TextQuoteSelector as metadata

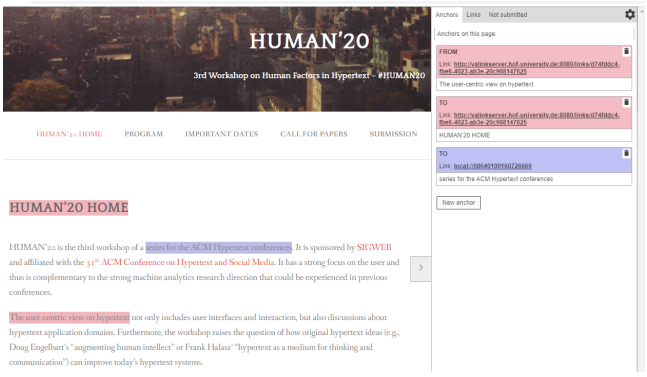

Figure 3: Anchors View – Annotated link endpoints on the HUMAN'20 website

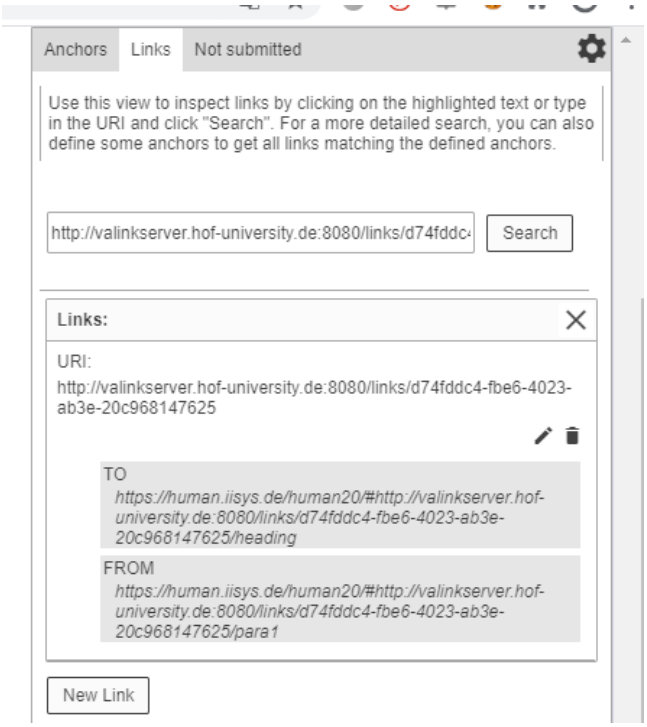

Figure 4: Links View – The links view with a selected link

or deleted. Any editing process must lead to a valid link at the end; otherwise the update gets rejected by the link service.

4.4 Not Submitted View

The third and final view of the plugin, the "not submitted view", is depicted in Fig. 5. It contains a list of all locally stored links that are not yet submitted. As such, they may be invalid. This view offers users to delete locally stored links or add valid links to the link service.

Users can create new links by either creating a new endpoint via the anchors view or by creating a link via the links view directly. Assume that a user wants to create a new endpoint first, he/she would click on the "new anchor" button in the anchors view. This leads to the view for creating new endpoints. Now, the user can enter the URI of the website that should be referenced. If only parts of a website should be referenced, the user would start a selection via the annotation button and mark the corresponding text. A button would appear next to this selection (as shown in Fig. 6) to confirm the annotation. This new endpoint would then be used for the new link to be created (or alternatively be added to an existing link). After having created a new endpoint, the corresponding text appears highlighted and the new link is created locally with one initial endpoint. Figure 3 shows the differently highlighted areas on the HUMAN'20 website with the corresponding endpoints in the sidebar. Blue highlighting refer to endpoints of local links, red ones to links stored by the link service.

5 CONCLUSION AND FUTURE WORK

In this paper we described and showcased a browser plugin, which enables a richer linking functionality for Web browsers. Based on our underlying infrastructure *Mother* we proposed an external link service and a client, implemented as a browser plugin. The link server functionality is based on a minimized variant of the Dexter Hypertext Reference Model, focusing on the navigational features only. Metadata, as used for link anchor selectors, is managed by the Mother's metadata service. Externalizing the link service allows links that are more flexible than HTML native, unary Web links. As

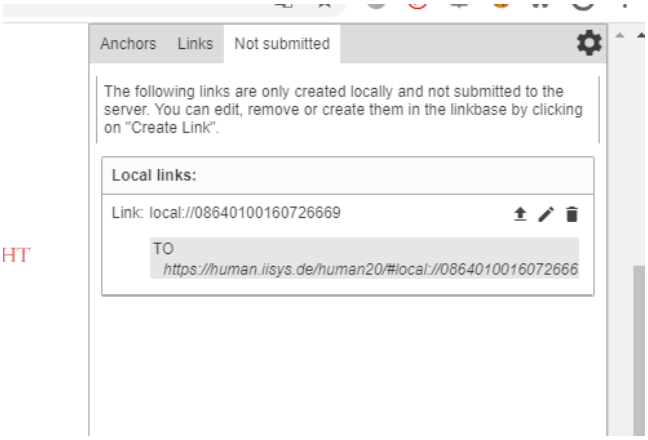

Figure 5: Not Submitted View – A local link with only one endpoint that is listed in the view for local links

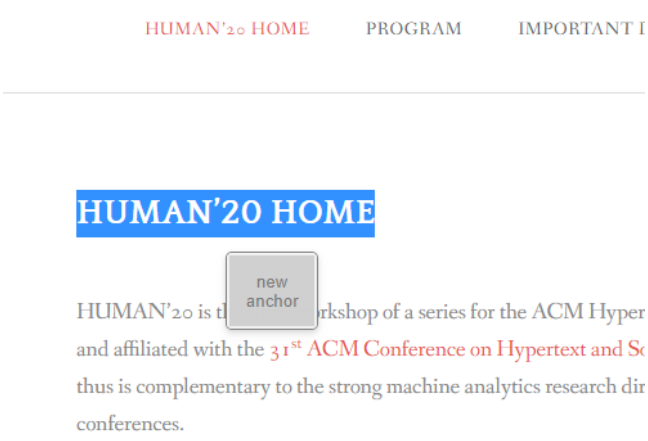

Figure 6: Next to the marked text passage, a button will appear to confirm the annotation

such it empowers users to create sophisticated links that go beyond today's wide-spread HTML-embedded URIs.

The browser plugin is a technical demonstration of Mother's link service. It is a first step toward using Mother for augmenting the Web with more sophisticated hypertext fundamentals.

We aim at improving the plugin with features that allow us to use it in productive work environments. This includes user management, integration with various systems, a new designed user interface, and optimized workflows for users to create, modify, traverse, or share links.

Furthermore, our goal is to use and test the plugin in various application domains, foremost for education. We plan to integrate the plugin with our university's learning management system (LMS) *Moodle* in order to ease user management such as to offer access rights for specific courses or study groups. It is planned that the plugin will be used in our recently started project *International Teaching and Research in Hypertext* (INTR/HT), which aims at providing joint hypertext classes for students worldwide. The link service will enable students to link various document sources and share those connections.

We believe that the link service will be further beneficial in any other application domain in which users may want to create associations. This includes, for example, linking from objects in a spatial hypertext environment to external documents. As such, the ongoing project IWInxt, further described in [2], provides is a possible scenario.

With its application in various domains and the commitment to get it working in productive environments, we hope that the link service and the plugin will be used by a number of people that is sufficiently large enough for meaningful user testing. In particular, we hope to answer questions regarding the overall use of the service and retrieve insights in the link structure built, the nature of the links (arity, directions, traversal, etc.), or the communication intended by users when sharing links. It is a step toward augmenting today's Web with "traditional" hypertext ideas.

ACKNOWLEDGMENTS

IWInxt and this work are funded by the Bavarian State Ministry of Science and the Arts (grant ID Kap. 15 49 Tit. 547 78-2/2018).

REFERENCES

[1] Claus Atzenbeck and Mark Bernstein. 2018. Interview with Andy van Dam. *ACM SIGWEB Newsletter* (March 2018).

[2] Claus Atzenbeck and Peter Nürnberg. 2019. Hypertext as Method. In *Proceedings of the 30th ACM Conference on Hypertext and Social Media (HT '19)*. ACM, 29–38. https://doi.org/10.1145/3342220.3343669

[3] Claus Atzenbeck, Daniel Roßner, and Manolis Tzagarakis. 2018. Mother - An integrated approach to hypertext domains. In *HT 2018 - Proceedings of the 29th ACM Conference on Hypertext and Social Media*. ACM Press, New York, New York, USA, 145–149. https://doi.org/10.1145/3209542.3209570

[4] Claus Atzenbeck, Thomas Schedel, Manolis Tzagarakis, Daniel Roßner, and Lucas Mages. 2017. Revisiting hypertext infrastructure. In *HT 2017 - Proceedings of the 28th ACM Conference on Hypertext and Social Media (HT '17)*. ACM, New York, NY, USA, 35–44. https://doi.org/10.1145/3078714.3078718

[5] Belinda Barnet. 2019. Hypertext before the Web – or, What the Web Could Have Been. In *The SAGE Handbook of Web History*, Niels Brügger and Ian Milligan (Eds.). SAGE Publications Ltd, 1 Oliver's Yard, 55 City Road London EC1Y 1SP, Chapter 15, 215–226. https://doi.org/10.4135/9781526470546.n15

[6] Niels Olof Bouvin. 2002. Augmenting the web through open hypermedia. *New Review of Hypermedia and Multimedia* 8, 1 (jan 2002), 3–25. https://doi.org/10.1080/13614560208914733

[7] Niels Olof Bouvin and Clemens Nylandsted Klokmose. 2016. Classical hypermedia virtues on the web with webstrates. In *HT 2016 - Proceedings of the 27th ACM Conference on Hypertext and Social Media*. Association for Computing Machinery, Inc, New York, New York, USA, 207–212. https://doi.org/10.1145/2914586.2914622

[8] Vannevar Bush. 1945. As we may think. *The Atlantic Monthly* 176, 1 (7 1945), 101–108. http://www.theatlantic.com/doc/194507/bush

[9] Robert Cailliau and Helen Ashman. 1999. Hypertext in the Web - a History. *Comput. Surveys* 31, 4 (dec 1999), 35. https://doi.org/10.1145/345966.346036

[10] L. Carr, W. Hall, H. Davis, and R. Hollom. 1994. The microcosm link service and its application to the World Wide Web. *Computer Networks and ISDN Systems* 27, 2 (nov 1994), 307. https://doi.org/10.1016/s0169-7552(94)90146-5

[11] Les A. Carr, David C. DeRoure, Wendy Hall, and Gary J. Hill. 1995. The Distributed Link Service: A Tool for Publishers, Authors and Readers. In *Proceedings of the Fourth International World Wide Web Conference*. Boston, MA, 647–656. https://www.w3.org/Conferences/WWW4/Papers/178/

[12] Frank Halasz and Mayer Schwartz. 1994. The Dexter hypertext reference model. *Commun. ACM* 37, 2 (feb 1994), 30–39. https://doi.org/10.1145/175235.175237

[13] Frank G. Halasz. 2001. Reflections on "Seven Issues". *ACM Journal of Computer Documentation* 25, 3 (aug 2001), 109–114. https://doi.org/10.1145/507317.507328

[14] Beat Signer and Moira C. Norrie. 2007. As we may link: A general metamodel for hypermedia systems. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 4801 LNCS. Springer Verlag, 359–374. https://doi.org/10.1007/978-3-540-75563-0_25