Usability aspects of hypermedia

Arnoud Engelfriet
galactus@stack.nl

Begeleiders:

1. Dr. ir. H. Kragt
2. Prof. dr. ir. F.L. van Nes

November 1998
Abstract

The goal of this paper is to give an overview of the methods with which current World-Wide Web-browsers present hypermedia to their users, to review the usability aspects of these methods and to compare them with the methods to present and process hypermedia as offered in the literature. The paper consists of two parts.

The first part of the paper is a literature study. This study identifies two problems with hypertext navigation. The first problem, disorientation, can be solved by adding navigational aids to a Web browser. The most well-known aids are the stack-based history list, with which the user can navigate between previously visited nodes and bookmarks, to permanently remember the location of an interesting resource. The literature offers more solutions, but no Web browser implements these. Several products exist which offer more information, but these are not integrated in the browser, but rather in a specific site. This means that the user still has to re-learn the interface on every site he visits.

The ability to include data about a document (called metadata) is the second problem. Standardized formats for metadata is essential for search engines to identify documents on the World-Wide Web. There are several frameworks with which metadata can be included, but few systems make use of them at this time.

The second part of the paper puts the findings of the first part into practice. A review of the navigational capabilities of four browsers (Netscape, Internet Explorer, Opera and Lynx) showed that there is little difference between their capabilities. This is most likely due to the fact that Netscape, being the first popular browser, “set the standard” on World-Wide Web navigation, so other browser developers had to implement the same mechanisms in order to compete successfully.

A large-scale survey was presented to the visitors of a popular Web design site. The results show that the vast majority of the respondents consider a navigational toolbar important, and that use of the history list is not as popular as expected based on the first part of the paper. Site overviews and search features are also important.

Five users (three experienced and two inexperienced Web users) were asked to perform several tasks on four sites, using the four browsers reviewed earlier. The results of this experiment showed that there are little to no differences in current graphical browsers that makes one better or worse than the others with respect to navigation.

The conclusions of the paper show that the history list used in current browsers has serious deficiencies. The stack-based model is not intuitive, as sometimes nodes disappear from the list, for no apparent reason to the user. A “marking” facility, with which users can easily return to pages they marked as interesting earlier, would be a very useful addition, as shown in the second experiment. A “Webmap”, which gives an overview of all visited nodes, may also be a useful tool here. It is clear that search capabilities are an essential tool on Web sites.

It is a disappointment that there seems to be so little interest for the existing solutions to most of the problems encountered in this paper. Competition and a desire to control the browser market are probably the two main causes for this problem. Most of the current site developers have resorted to designing user interfaces in the documents, to make up for the limitations in the user interfaces in the browsers. This is a great setback after user interfaces such as the MacOS and Microsoft Windows have standardized the way applications look and feel.
# Contents

1 Introduction 7  
1.1 Terms and definitions 7  
1.2 Outline of this paper 7  

2 Problems with World-Wide Web navigation 9  
2.1 The “lost in hyperspace syndrome” (disorientation) 9  
2.2 Limited abilities to offer metadata 11  
2.3 Site design based on aesthetics rather than usability 11  

3 Solutions presented in the literature 13  
3.1 Navigational aids 13  
3.2 Standardization of metadata inclusion 15  
3.2.1 Cross-references in HTML 15  
3.2.2 Metadata in HTML 15  

4 An overview of navigation tools 23  
4.1 WebTOC 23  
4.2 Treemaps 23  
4.3 Navigational View Builder 23  
4.4 WebMap 24  
4.5 WebBook 24  
4.6 Other tools 25  
4.7 Discussion 25  

5 Field studies 27  
5.1 Jakob Nielsen’s 1994 Web usability study 27  
5.2 How People Revisit Web Pages 27  

6 An overview of the capabilities of several browsers 31  
6.1 Netscape Communicator 4.0 31  
6.2 Microsoft Internet Explorer 4.0 31  
6.3 Opera 3.21 32  
6.4 Lynx 2.7.2 32  
6.5 Evaluation 32  
6.6 Feature summary 33  

7 A survey of navigational aids 35  
7.1 Text of the survey 35  
7.2 Running the survey 37  
7.3 Conclusions of the survey 38  

8 A small-scale experiment 41  
8.1 Approach of the experiment 41  
8.2 Results of the experiment 42  
8.3 Observations during the experiment 43  
8.4 Conclusions of the experiment 44  

9 Conclusions 47
Bibliography 49
Introduction

At this moment the World-Wide Web consists of several million hypertext documents ("web pages"), organized in so-called "websites." For various reasons, many website developers have found it necessary to create a unique "look and feel" for their sites. This means that a reader has to re-learn how to find information and how to navigate around a site, every time he or she visits a new site.

An additional problem is that the currently popular browsers have only very limited capabilities to navigate. It is difficult to even simply revisit a web page that has been visited earlier. There are many programs and systems that allow readers to extend the browsers' capabilities, but their support is often limited to specific browsers or platforms, or not applicable in the general case, only for specific sites.

Literature on the topic of hypertext offers various methods to present large amounts of information in a consistent and easy to navigate way (for an overview of hypermedia tools before the World-Wide Web emerged, see [19]). Unfortunately, the solutions that were present in many hypertext applications are rarely found in World-Wide Web browsers.

The goal of this paper is to give an overview of the methods with which current World-Wide Web-browsers present hypermedia to their users, to review the usability aspects of these methods and to compare them with the methods to present and process hypermedia as offered in the literature.

1.1 Terms and definitions

Hypertext has been defined as "an approach to information management in which [textual] data is stored in a network of nodes connected by links. Nodes can contain text, graphics, audio, video as well as source code or other forms of data." [16]. Hypertext with other media (images, sounds, movies, and so on) is called hypermedia.

The connections between the nodes are often referred to as hyperlinks or simply links.

A hyperdocument is a part of the hypermedia network, containing nodes that are related in some way. This could for example be a book, with the related nodes being the chapters that make up the book, but these nodes would also include the figures in the book. A hyperdocument consists of several nodes, which are also referred to as pages if the content is mostly textual. Hyperlinks connect the various pages together.

On the World-Wide Web the term Website is used, which roughly corresponds to the above definition of "hyperdocument." However, the term can also be used to refer to all information offered by a single information provider. This could include more than one hyperdocument.

In this paper the term "hyperdocument" will be used to make it more clear what is being referred to; in most cases, its meaning will be the same as the term "Website" in the first sense mentioned above.

1.2 Outline of this paper

This paper consists of two main parts. The first part is a literature study on usability aspects of hypermedia. In this part, chapter 2 discusses the most important current problems that
play a role with using hypermedia on the World-Wide Web.

Next, chapter 3 discusses several possibilities presented in the literature, that can be used to make more effective use of hypermedia.

Chapter 4 gives an overview of existing tools and systems which are currently available to allow easier navigation on the World-Wide Web.

The second part of this paper builds on the results found in the first part. Chapter 5 reviews two existing field studies on hypermedia usability. Based on the list of navigation aids presented in chapter 3, chapter 6 discusses four popular browsers and their abilities to let the user navigate through hyperspace.

Two experiments were conducted for this paper. Firstly, a large-scale survey (a series of fill-out questions) is discussed in chapter 7. Secondly, a small group of users was asked to perform a few navigation tasks on the World-Wide Web. The outcome of this experiment can be found in chapter 8.

Finally, the conclusions are presented in chapter 9.
Problems with World-Wide Web navigation

Ever since the moment that hypermedia was first introduced, it became apparent that making effective use of hypermedia requires a complete switch in the way the reader approaches the information. The common metaphor used to explain the hypermedia concept is the book. Information is presented in a linear form, from the first page to the last, with hyperlinks as an easy way to browse through the book, or to read more in-depth information on specific topics. This raises the expectation for the reader that everything he sees and does in the “hyperbook” will behave exactly as he is used to in a normal book, which is not the case.

The most important problems that a reader will encounter, are that it is generally difficult to see “where” you are in the hyperdocument and where you can go from there. On the World-Wide Web, this problem is complicated by the fact that many Website developers seem more concerned with a visually pleasing appearance of their site than with its usability.

2.1 The “lost in hyperspace syndrome” (disorientation)

The first recognized hypermedia problem is commonly known as the “lost in hyperspace syndrome.” This refers to the fact that it is difficult to position the current node in relationship to the hyperdocument, since it is common for nodes to be displayed on their own, with little to no context. If there are insufficient methods to determine the context in the browser, the reader will lose track of his position in the document. Using the metaphor of “surfing through hyperspace” for locating information, this means that the reader is now lost.

A large amount of research on this subject has been published. Cockburn and Jones [6] extensively discuss this problem in their paper for RIAO ’97. They state three primary causes for the existence of this problem in World-Wide Web hypertext systems:

1. Browser user interfaces: Since the history mechanism used in browsers is a stack rather than a complete list, previously visited pages can suddenly disappear from the history list and become unreachable with the browser’s “back” button.

2. Limited overview capabilities: it is hard for a reader to determine exactly where he is, and what related other nodes (such as the next node, the table of contents and so on) in the system are.

3. Limited hyperlinking facilities: the language used to create documents on the World-Wide Web is HTML [3]. This language offers several types of hyperlinks, but current browsers only support the “directed” link, and cannot distinguish between the different types that can be used (see also section 3.2.1).

Nielsen and Lyngbæk [13] have demonstrated that the “lost in hyperspace” problem is also present when the hyperdocument contains a relatively small amount of nodes. This shows that this problem is not only related to the size of the hyperdocument being read.

The problem can be split into two subproblems:
Problems with World-Wide Web navigation

- When a reader finds a node that looks interesting, he must read it immediately, or risk being unable to find it again at a later time. This may slow down his browsing, as it requires reading every node to find out if it is interesting or not.

- While browsing, a reader can get confused about his current location in the hyperdocument.

Both problems have corresponding problems in the real world. When someone visits a city where he has never been before, and starts walking without a good map, he will soon lose his way. This means that it is no longer possible to return to a place he visited before, but also that his starting point can no longer be found again.

For a city, this problem is relatively easy to solve. A city street map has only two dimensions (possibly excluding bridges and tunnels, but these can usually be indicated without problems), so a good map on paper can be constructed without problems. The reason people still lose their way even with a map is because they do not pay attention at street corners and crossings, and because they do not keep track of the length of streets.

In a hyperdocument, all links are equally long, and there are no clear corners as there are in a street. Every node can have many “turns.” A reader cannot see what distance he has traveled, or that a particular link will take him to another node nearby the current node. Creating a map of a hyperdocument usually requires more than two dimensions. Chapter 4 discusses tools which create two-dimensional maps and overview diagrams of hyperdocuments.

Because the structure of a hyperdocument is much more complex than a city’s, and because there is no clear “direction,” it is much more difficult to prevent a reader from getting lost in a hyperdocument than in a city. Graphical overview tools can usually only make the immediate surroundings clear and show the rest in a more generic way, or they attempt to show the entire document with all nodes and links, but the limited screen size makes it then very difficult to read the names of individual nodes.

However, it is possible to return to previously visited nodes very easily. The reader can create “bookmarks” for specific nodes, or recall a list of all previously (or the most recently) visited nodes, then pick a node from any of those lists and be taken there immediately. In the analogy of the city, this would require the installation of teleportation stations on every street corner, which is unfortunately not feasible with the current state of technology.

In the magazine Webweek Andrew Marlatt discusses the idea that on the World-Wide Web, the question “Why am I here?” should be replaced with the question “Where, exactly, is ‘here?’” [10]. Because many readers do not visit a Website starting at the main entrance point, they often end up in situations where the context for the current node is missing. It is then very hard to figure out how to read more, or even which information provider is offering this information.

This agrees with the findings of Jakob Nielsen, distinguished engineer at Sun Microsystems. “Designers often assume visitors will come in through the front door, but the reality is, Web users may first show up in the basement, bedroom, or bathroom.”

It turns out that there are quite a few obstacles for information providers who want to offer navigational aids for their readers. Most browsers are not 100% compatible with each other, which makes it hard to implement effective navigational tools. Solutions that work in one browser cause problems in another; external applications (such as Java applets) are too slow or seen as security risks, and all too often there is absolutely no backward compatibility, so users with anything less than the latest version of the most popular browser are left out with no means at all to navigate.

Pitkow & Kehoe have conducted a large-scale survey on trends and user demographics for the Graphic, Visualization and Usability Center (GVU) at Gatech University [14]. Clare Davies used this survey to demonstrate that disorientation is not a significant problem on
the World-Wide Web.\textsuperscript{1} She argues that the World-Wide Web is not a true hypertext system, and is not used for the same purposes as “normal” hypertext systems, which are mainly used for teaching and learning, and which are designed and connected much more carefully than the average Web site. Because of this different approach, World-Wide Web users do not see disorientation as a problem, but rather as a normal part of Web browsing, according to Davies.

\section*{2.2 Limited abilities to offer metadata}

\textbf{Metadata} is data about data. In the context of hypermedia, this could be the title of a document, its author, keywords, or relationships to other documents. On the World-Wide Web this also includes information such as the size of a document, the date the information expires, whether or not it may be indexed by Web search engines, and so on.

Metadata allows easier searching and indexing of information, as well as the automated creation of links between relevant documents, since this information can be processed by computer without human intervention. This requires that there is a standard with which metadata in a document can be stored, read and recognized. Unfortunately, there is little in the way of standardization for metadata on the World-Wide Web.

The HyperText Markup Language (HTML) standard defines one element with which name/value-pairs of arbitrary length can be included in a document (the \texttt{META} element, section 5.2.5 of [3]). Authors and information providers can choose keywords to indicate metadata or to label cross-references. Some of these keywords were promoted by the large search engines, which has turned these keywords into ad-hoc standards.

Chapter 3 discusses several proposals to standardize the naming of metadata.

\section*{2.3 Site design based on aesthetics rather than usability}

In the last few years, having a World-Wide Web site has become more and more popular for companies and institutions. The widespread adoption of browsers such as Mosaic and later Netscape, as well as the increased availability of fast network connections and modems, have made it possible to use large amounts of graphical elements in a Web site. Graphics designers became important members of the Web site design team. However, most graphics designers were used to designing pages on paper, and used this experience when creating Web sites. This resulted in a large amount of Web sites being developed as if they were paper publications, which meant a large dependency on graphical elements and layout.

Unfortunately this often resulted in poor usability on the site. Many graphical elements require a large amount of time to be loaded and displayed, but navigation is impossible until these elements have been loaded completely. Also, many “artistic” sites want to experiment with the way information is presented, which makes it difficult to use them.

An interesting read on this topic is Jakob Nielsen’s 1995 study on World-Wide Web design.\textsuperscript{2}

\textsuperscript{1}See \url{http://zaphod.mk.dmu.ac.uk/~cdavies/webpaper.htm}.

\textsuperscript{2}See \url{http://www.sun.com/sun-on-net/uidesign/}. 
Solutions presented in the literature

This section discusses several possible solutions to the problems mentioned in the previous section. The problem of limited navigation abilities is discussed below, and the metadata inclusion problem on page 15.

3.1 Navigational aids

A large amount of tools to allow navigation in hypertext exist. The most well-known aids are:

**History list** The basic tool for keeping track of visited nodes in World-Wide Web browsers is the history list. It is based on the concept of stack-based browsing. In this concept, every new node that is visited is stored at the top of a history stack. When the reader goes “back” in the history list, the most recent node is removed (“popped”) from the stack. Sometimes this popping does not occur until the reader actually follows a link, so that moving backward and forward in the stack does not modify the stack.

**Backtracking** Backtracking allows a reader to reverse the path he has followed to arrive at the current node. Every time the reader uses his “back” button, the preceding node is recalled and displayed. In most World-Wide Web browsers, there is also a “forward” button which performs the exact opposite action.

One important limitation of the stack model becomes obvious here. When the user goes “back” several nodes, then follows another link, and then goes “back” once more, it is no longer possible to go “forward” to the original branch, since they have been popped off the stack.

Another problem occurs when textual or graphical links labeled “Back” are provided in a node. Following this link does not perform the same function as pressing the “back” button, since the browser treats this link as any other normal link, so no node is popped off the stack. Rather, the new node is pushed on the stack. This can cause great confusion when users mix the use of the “back” button and “back” links, since the resulting state of the stack is not at all what they expect (see [5] for a detailed discussion of this problem encountered during a study on navigational support in browsers).

**Sneak preview** In most cases, the anchor text that is used to mark up a link provides little information about the resource behind the link. Most World-Wide Web browsers show the Uniform Resource Locator (URL) of the destination in their status bar when the mouse is moved over the link. This at least gives some idea of what is behind the link, but URLs are often rather cryptic and hard to decipher.

HTML offers an author the ability to give a textual description of the destination, by adding a title to the link (with the `TITLE` attribute of the `A` element, see section 5.7.3 of [3]). Few browsers, however, currently display this title.
Link highlighting  To distinguish between new and already-visited links, different colours, underlining or background colours can be used. This way, a reader can at least determine that he has already read a particular node, even though he may not be able to figure out which node it was or what information he read in there.

On the World-Wide Web, the model made popular by Mosaic is to colour unvisited links light blue, and visited links purple. It is now possible to change the colour of links in a node, which often still confuses readers, as they can no longer distinguish between visited and unvisited links. Common practice is to use a darker version of the unvisited link colour for visited links.

Unique anchors  On the World-Wide Web it is not possible to distinguish between two different links with the same link text. The author must make this clear in the context, or by providing additional information about the linked node. Ideally, this would be done by the browser based on the relationship indicated in the link (see section 3.2.1), but this is very poorly supported.

An often-used solution is the use of the phrase “Click here for”, which has many disadvantages. For example, the phrase assumes that the reader uses a mouse and needs to “click” with it on a link to activate it. The phrase also disrupts the flow of the text because of its use of the imperative. Furthermore, the use of colour and underlining draws the reader’s attention to the hyperlinked phrase, which means that they will read the text “Click here” rather than the surrounding text, which contains the information about what is behind the link.1

Bookmarks  In a book, a reader can use a bookmark or a dog’s ear to indicate important pages. The ability to do this is somewhat limited: a book with dozens of bookmarks becomes hard to read.

In a hypertext system, it is possible to add nodes to a separate list, which is called the “bookmark list”. These nodes can often be organized with submenus or folders, to make it easier to search through them to find a specific node.

Even though most browsers have extensive abilities to include and manage bookmarks, many commercial and freeware products exist which provide additional features. Distributions of link collections in bookmark format are also popular.

Bird’s eye and fish eye views  It is often desirable to provide a graphical view of the structure of a hyperdocument. This allows the reader to figure out where he is and where he can go from there. There are two popular methods to create such a view:

1. A bird’s eye view of the entire document can be created. This is a graph displaying all nodes and links in the document. The reader can zoom in and out on the display, to get more detail or a better view of the “big picture” of the document.

2. This graph can also be adjusted so that the nearest nodes are shown bigger and with more detail, and faraway nodes are shown smaller and without detail. This gives a shortsighted overview, also called a fish eye view. Although the entire document is harder to visualize, the immediate vicinity is much easier to explore with this technique. The biggest problem in this system is to determine which details can be omitted without too much distortion. Faraway nodes often have to be grouped together due to lack of space, this may create the impression that those nodes belong together even if this is not the case.

1See http://www.htmlhelp.com/design/style/docstyle-no-online.html for a more extensive discussion.
3.2 Standardization of metadata inclusion

As discussed in section 2.2, there is currently no standard which defines how to “tag” metadata in World-Wide Web documents. Additionally, there is no way to define cross-references with other documents in a standardized way, so that for example table of contents lists, or trees of related documents can be created automatically.

3.2.1 Cross-references in HTML

The HyperText Markup Language (HTML) allows an author to “tag” every link with information on the relationship between the source and the target of the link. This is done with two attributes, REL and REV on the A element used to create anchors (see section 5.7.3 of [3]).

It is also possible to define document-wide relationships between documents. This is done with the LINK element (see section 5.2.4 of [3]).

With these elements, it is possible to indicate different types of links based on the relationship (for example, a bibliographic reference could be indicated with a dotted underline, and a link that would download a file could be followed by a small icon of a disk), and a navigation menu could be constructed. Such a menu would be created by the browser based on information in the current node, which has the advantage that the menu is always in the same location and uses the same icons and names for all nodes. The menu items always have the same label and icon, but the destinations change for every hyperdocument that is visited. This should make it easier to navigate through a large hyperdocument.

For both the A and the LINK elements, no standard exists for the keywords that can be used to indicate the type of link. The HTML 3.2 standard [15] gives a list of commonly used values and refers to [9], a now-expired draft proposal, for more information.

Due to this lack of standardization, few browsers make use of this information.

3.2.2 Metadata in HTML

To include metadata in an HTML document, the META element is available (see section 5.2.5 of [3]). With this element, arbitrary name/value pairs can be defined and included. There is a limit of 1,024 characters for both the name and the value. This is a general restriction on attribute values in HTML.

An author is free to choose any name for the metadata he wants to include. The HTML standards do not even suggest a list of commonly used names, as they do for the LINK element.

In practice a few ad-hoc standards have emerged. Search engines often provide short descriptions or summaries for the documents they find, and authors can suggest their own summary
by using a **META** element to provide a name/value pair with the name “description.” Similarly, additional keywords can be provided with the name “keywords.”

Other types of metadata, with names such as “author”, “generator” and “convertor” are frequently included by site editors, but the information is seldom used by browsers or other programs.

The **Dublin Core framework** is the result of a series of workshops organized by the World-Wide Web Consortium (W3C). The Dublin Core provides a framework to indicate metadata in a structured manner. Names are constructed in a hierarchical manner. The framework defines the top-level names, and authors can create sub-hierarchies for their own needs.

In the Dublin Core, all names start with “DC.” to indicate that they are part of the Dublin Core namespace. Components in the hierarchy are separated by periods. For the current version of the standard, refer to [18].

The following fifteen top-level names exist (based on an overview by Arthur D. Chapman of Environment Australia):²

1. **Title Label**: DC.title
   The name given to the resource by the creator or publisher.

2. **Author or Creator Label**: DC.creator
   The person(s) or organization(s) primarily responsible for the intellectual content of the resource. For example, authors in the case of written documents, artists, photographers, or illustrators in the case of visual resources.
   Sub-elements could be, for example, DC.creator.corporateName or DC.creator.personalName.

3. **Subject and Keywords Label**: DC.subject
   The topic of the resource, or keywords or phrases that describe the subject or content of the resource. The intent of the specification of this element is to promote the use of controlled vocabularies and keywords. This element might well include scheme-qualified classification data (for example, Library of Congress Classification Numbers or Dewey Decimal numbers) or scheme-qualified controlled vocabularies (such as Medical Subject Headings or Art and Architecture Thesaurus descriptors) as well.

4. **Description Label**: DC.description
   A textual description of the content of the resource, including abstracts in the case of document-like objects or content descriptions in the case of visual resources. Future metadata collections might well include computational content description (spectral analysis of a visual resource, for example) that may not be embeddable in current network systems. In such a case this field might contain a link to such a description rather than the description itself.

5. **Publisher Label**: DC.publisher
   The entity responsible for making the resource available in its present form, such as a publisher, a university department, or a corporate entity. The intent of specifying this field is to identify the entity that provides access to the resource.

6. **Other Contributors Label**: DC.contributor
   Person(s) or organization(s) in addition to those specified in the creator element who have made significant intellectual contributions to the resource but whose contribution is secondary to the individuals or entities specified in the creator element (for example, editors, transcribers, illustrators, and conveners).
   A subdivision similar to DC.creator can be used here as well. To distinguish between multiple contributors, the names could be assigned serial numbers.

7. **Date Label: DC.date**
   The date the resource was made available in its present form. It is recommended that an 8 digit number in the form YYYY-MM-DD as defined by ISO 8601 be used. In this scheme, the date element for the day would be 1996-12-03, or December 3, 1996.

8. **Resource Type Label: DC.type**
   The category of the resource, such as home page, novel, poem, working paper, preprint, technical report, essay, dictionary. It is expected that the resource type will be chosen from an enumerated list of types. A preliminary set of such types can be found at the following URL: http://www.roads.lut.ac.uk/Metadata/DC-ObjectTypes.html
   This is at present undergoing a radical revision to cater for a number of categories not previously catered for. A committee is developing what will probably be a hierarchical system — separating categories of documents into DOCUMENT.xxxx, IMAGE.xxxx, SOUND.xxxx, SOFTWARE.xxxx, DATA.xxxx and ENVIRONMENT.xxxx. This will be updated once a consensus is arrived at. (See current proposal at http://sunsite.Berkeley.EDU/Metadata/types.html).

9. **Format Label: DC.format**
   The data representation of the resource, such as HTML text, ASCII, Postscript file, executable application, or JPEG image. The intent of specifying this element is to provide information necessary to allow people or machines to make decisions about the usability of the encoded data (what hardware and software might be required to display or execute it, for example). As with resource type, format will be assigned from enumerated lists such as registered Internet Media Types (MIME types). In principal, formats can include physical media such as books, serials, or other non-electronic media.

10. **Resource Identifier Label: DC.identifier**
    String or number used to identify uniquely the resource. Examples for networked resources include Uniform Resource Locators (URLs) and Uniform Resource Names (when implemented). Other globally-unique identifiers, such as International Standard Book Numbers (ISBN) or other formal names would also be candidates for this element.

11. **Source Label: DC.source**
    The work, either printed or electronic, from which this resource is derived, if applicable. For example, an HTML encoding of a Shakespearean sonnet might identify the paper version of the sonnet from which the electronic version was transcribed.
    A number of subelements may be used, for example:

    - **DC.source.author.1**
      First author.
    - **DC.source.author.2**
      Second and subsequent authors.
    - **DC.source.date**
      Publication date.
    - **DC.source.title**
      Title of the published work.
    - **DC.source.series**
      Journal or Series title along with volume number.
    - **DC.source.publisher**
      Publisher.
    - **DC.source.ISBN**
      ISBN number.
Rachel Heery compared the Dublin Core scheme with several other metadata standardization systems.³

The Warwick Framework is the result of the Warwick Workshop, which was convened to build on the Dublin core program and provide a more concrete and operationally usable formulation of the Dublin Core, in order to promote greater interoperability among content providers, content catalogers and indexers, and automated resource discovery and description systems. The second workshop also was an opportunity to assess the results of a year of experimentation with the Dublin Core [8].

The goal of the Warwick Workshop was to provide a higher-level context for the Dublin Core. This context should define how the Core can be combined with other sets of metadata in a manner that addresses the individual integrity, distinct audiences, and separate realms of responsibility and management that characterize these distinct metadata sets.

The result of the Warwick Workshop is a proposal for a container architecture, known as the Warwick Framework. It has two fundamental components. A container is the unit for aggregating the typed metadata sets, which are known as packages.

A container may be either transient or persistent. In its transient form, it exists as a transport object between and among repositories, clients, and agents. In its persistent form, it exists as a first-class object in the information infrastructure. That is, it is stored on one or more servers and is accessible from these servers using a globally accessible identifier (the Uniform Resource Identifier (URI)), a generalized version of the common URL, may be suitable for this task.

At the container level each package is an opaque bit stream. One implication of these properties is that any encoding (transfer syntax) for a container must allow the recipient of the container to skip over unknown packages within the container (in other words, the size of each package must be self-describing at the container level). This property also permits the contents of individual packages to be encrypted, permitting the transport of metadata
across systems that need not have access to specific sets, or that may need to acquire (i.e., purchase) such access. Certain implementations, such as the HTML one proposed later in this paper, may lack the power to fully enforce these abstract properties of containers.

Each package is a typed object; its type may be determined after access by a client or agent. Packages are of three types:

1. Metadata set — These are packages that contain actual metadata. Some examples of this are packages that are USMARC⁴ records, Dublin Core records, and encoded terms and conditions. A potential problem is the ability of clients and agents to recognize and process the semantics of the many metadata sets. In addition, clients and agents will need to adapt to new metadata types as they are introduced, at least to the extent of ignoring them gracefully, or perhaps copying them for downstream applications that may know how to process them. Initial implementations of the Warwick Framework will probably include a set of well-known metadata sets, in the same manner that most Web browsers have native handlers for a set of well-known MIME types. Extending the Framework implementations to handle an extensible metadata set will rely on a type registry scheme. We describe this in some greater detail in the implementation section of this document.

2. Indirect — This is a package that is an indirect reference to another object in the information infrastructure. While the indirection could be done using URLs we emphasize that the existence of a reliable URN implementation is a necessity to avoid the problems of dangling references that plague the Web. We note three possibly obvious, but important, points about this indirection. Firstly, the target of the indirect package is a first-class object, and thus may have its own metadata and, significantly, its own terms and conditions for access. Secondly, the target of the indirect package may also be indirectly referenced by other containers (i.e., sharing of metadata objects). Finally, the target of the indirection may be in a different repository or server than the container that references it.

3. Container — This is a package that is itself a container. There is no defined limit for this recursion.

The Resource Description Framework (RDF) is another scheme to encode structured metadata. It allows authors to define metadata elements as needed, using XML (eXtensible Markup Language, a subset of the standard SGML) as its encoding scheme [11].

The broad goal of RDF is to define a mechanism for describing resources that makes no assumptions about a particular application domain, nor defines (a priori) the semantics of any application domain. The definition of the mechanism should be domain neutral, yet the mechanism should be suitable for describing information about any domain.

RDF is the result of a number of metadata communities bringing together their needs to provide a robust and flexible architecture for supporting metadata on the web. It is an initiative of the World-Wide Web Consortium (W3C), influenced by, amongst other things, the PICS initiative to label content for easy filtering. Other metadata efforts, such as the Dublin Core and the Warwick Framework have also influenced the design of the RDF.

The foundation of RDF is a model for representing named properties and property values. The RDF model draws on well-established principles from various data representation communities. RDF properties may be thought of as attributes of resources and in this sense correspond to traditional attribute-value pairs. RDF properties also represent relationships between resources and an RDF model can therefore resemble an entity-relationship diagram. In object-oriented design terminology, resources correspond to objects and properties correspond to instance variables.

⁴The USMARC formats are standards for the representation and communication of bibliographic and related information in machine-readable form. See http://www.loc.gov/marc/,
The RDF data model is a syntax-neutral way of representing RDF expressions. The data model representation is used to evaluate equivalence in meaning. Two RDF expressions are equivalent if and only if their data model representations are the same. This definition of equivalence permits some syntactic variation in expression without altering the meaning.

The basic data model consists of three object types:

1. **Resources.** All Web objects being described by RDF expressions are termed resources. A resource may be an entire web page; an HTML document, for example. A resource may be a part of a web page; e.g. a specific HTML or XML element within the document source. A resource may also be a whole collection of pages; e.g. an entire web site. Resources are always named by URI. Anything can have a URI; the extensibility of URIs allows the introduction of identifiers for any entity imaginable.

2. **Property Types.** The names of the properties (or attributes, or relations) in an RDF expression are termed property types. The property type defines the specific meaning of the property, the permitted values for the property, the types of objects that can be described with the property, and the relationship(s) between properties of one type and properties of other types. Property types are also resources. This document does not address how characteristics of property types are expressed; for such information, refer to the RDF Schema document.

3. **Statements.** A specific resource together with a named property plus the value of that property for that resource is an RDF statement. Property values can be other resources or they can be atomic; that is, simple strings or other primitive data types defined by XML.

The specification of RDF is still under development. For the latest working draft maintained by the W3C, refer to [http://www.w3.org/TR/WD-rdf-syntax](http://www.w3.org/TR/WD-rdf-syntax).

**The ISO Z39.50** Client/Server Information Retrieval Standard began as a means for libraries to share cataloging data (USMARC records) and eventually expanded to include many types of shared databases and formats. With a standard for client/server, it would not matter if libraries were running different machines with different operating systems — all “clients,” or interfaces conforming to Z39.50, would be able to access data on Z39.50-compliant servers.

In essence, Z39.50 provides a common computer-to-computer search protocol between diverse information resources and diverse information access mechanisms. A range of software to implement Z39.50 in this way is available, from freeware to various commercial offerings worldwide.

The Z39.50 protocol is a so-called “stateful” protocol. Every Z39.50 search session involves an login phase, an initial query, zero or more refinements and additional queries and finally a logout phase. This makes it possible to work with earlier results (for example, to limit search results based on additional criteria) and to require authentication at the beginning of a session, if necessary.

There are also hundreds of Z39.50 WAIS databases available, and thousands more WAIS databases are maintained behind HTTP servers. Unfortunately, most Web browsers do not have built-in support for the WAIS Z39.50 search protocol.

The Z39.50 standard does provide a very useful method to allow easy and powerful searching, but it has the problem that it is stateful, whereas HTTP is stateless. In HTTP, every request for a document, image or other resource from a Web server stands on its own. There is no connection between the current and the previous request. This means that if a Web server were to implement support for Z39.50, every session would be initialization, one search and...
immediate termination. This prevents the full potential of the search system from being used.\(^5\)

There is no real solution for this problem. Adding support for Z39.50 search to Web browsers would still mean only the simplest queries can be performed.

\(^5\)The same problem has been encountered when Gopher and File Transfer Protocol (FTP) support was added to Web browsers. Both these protocols require logging in and support multiple download actions in one session, but Web browsers could not do more than log in, download one resource and log out.
An overview of navigation tools

In this chapter, several tools that were developed to make navigation of Websites easier are discussed.

4.1 WebTOC

WebTOC (Web Table of Contents) is a tool which automatically creates a hierarchical structure of the contents of a Web site and presents it in the format of a table of contents. Having the TOC allows the user to visually scan for and then directly move to areas of the Web site where the target information is likely to be found. WebTOC also shows the size and data types (e.g. text, images, sound, or other) of contents on the site.

The tool is implemented as a Java applet, making it possible to embed it in a Web site so that anyone with a Java-capable browser can make use of it. It was developed by David Nation, Catherine Plaisant, Gary Marchionini, and Anita Komlodi. An introduction to WebTOC, as well as several usability experiments conducted with WebTOC, can be found at the URL: http://www.uswest.com/web-conference/proceedings/nation.html.

4.2 Treemaps

Treemaps, a visualization method for large hierarchical data spaces [1], are used to augment the capabilities of the Analytic Hierarchy Process (AHP) for decision-making. Two direct manipulation tools, presented metaphorically as a “pump” and a “hook,” were developed and applied to the treemap to support AHP sensitivity analysis. Users can change the importance of criteria dynamically on the two-dimensional treemap and immediately see the impact on the outcome of the decision.

4.3 Navigational View Builder

The Navigational View Builder is a tool for letting the designer develop effective overview diagrams of hypermedia systems. The tool uses various strategies to reduce the problems concerned with developing overview diagrams. It is based on the assumption of a database-oriented hypermedia system where the nodes are described with attributes [12].

The creators describe how they construct this system from a World-Wide Web site as follows: “Since the World-Wide Web is unstructured and does not have this model, the model has to be constructed from the node and link structure of the WWW extracted by parsing the html documents. Some of the attributes of the nodes like the author (the owner of the file) and the file-size could be extracted automatically from the files. However, a major drawback of the World-Wide Web is the absence of many useful semantic attributes for the pages. Therefore, to fully show the power of our tool, attributes like the topic of the page (whether it is a research page or a personal page, etc.) were inserted manually. (Efforts are underway to incorporate metadata into WWW and hopefully in the near future we can extract all useful information from WWW automatically).”
The program generates overview diagrams from all nodes in a hyperdocument. It then attaches icons and other information to the diagrams, so that the user can more easily see what kind of node a particular node is, and what information can be found there. The diagrams can then be "clustered" and the view can be changed, with any node as the root node. This makes navigation from a particular node more easy, since the current node can be taken as root node and then all paths are immediately available in the overview.

4.4 WebMap

WebMap is a tool for NCSA Mosaic which can create and update a two-dimensional map of a user's visits to Websites during a session. It was developed by Peter Dömel at the University of Frankfurt [7]. WebMap communicates with the WWW browser Mosaic, so user interactions with Mosaic are immediately reflected in WebMaps graphics window.

During the user's hyperspace travel WebMap creates and updates a topology representation of the navigation history. This internal representation reflects the structure of the hypertext, i.e., the relation between all documents the user has visited so far. In this context the web resulting from the complete history is viewed as a graph whose nodes correspond to WWW documents and edges to hyperlinks resp. direct URL jumps, i.e., manual URL input by the user.

WebMap maintains the topology information by calculating a so-called spanning tree. The construction of a spanning tree divides the set of edges roughly into two categories:

1. Tree-edges: All edges of the spanning tree.
2. Non-tree-edges: All remaining edges of the graph.

This construction is done essentially in the following manner:

- If a document is accessed for the first time, a new topology node is created and connected with its predecessor through a tree-edge.
- Otherwise, if the document has been visited before with a different access path, a non-tree-edge is created to connect the node with its predecessor.

4.5 WebBook

At CHI '96, Card, Robertson, and York presented a paper with two tools that can present navigational and locational information more clearly [4]. The WebBook is a tool that can present Web sites in a more easy-to-use fashion. The related Web Forager is used to retrieve sites and extract information from the pages.

The WebBook proposal creates a Web "entity" at a higher level of abstraction than the level used in current Web browsers, which is one page at a time, with no difference between a page "in the collection" and one outside the collection. A natural candidate structure to represent this abstraction is the book metaphor.

Given a collection of web pages, WebBook preloads those pages and displays them as a collection using an augmented simulation of a physical book. 3D graphics and interactive animation are used to give the user a clear indication of the relationship between the pages of the book. Each page of the WebBook is a page from the web. Links are colour coded so the user can easily tell the difference between a reference to another page in the book (red links) and a reference outside the book (blue links). Picking a red link will animate the flipping of pages to the desired page. Picking a blue link will close the current WebBook and look for the page elsewhere. If the page is in another WebBook stored on a bookshelf, that WebBook is opened to the desired page. If the page is in none of the WebBooks, then the Web Forager is used to display the individual page in the user's information workspace.
4.6 Other tools

There are a number of features in the WebBook that make it intuitive to use. The user has several ways to flip through the pages of the book, all animated so the user can continue to see the text and images on pages while they turn. The simplest method is to click on a page (away from any link on that page); this will flip to the next or previous page depending on whether user clicked on the right or left page. The user can also click on the right or left edge of the book. The relative distance along that edge indicates how far to flip. The user can also scan the book with forward and backward scan controls (two of the buttons on the bottom of the book). The scan rate and pause time at each page is a user preference. When the user clicks on a page during a scan, the scan stops. Finally, the user can ruffle through the pages by clicking and holding the mouse button down. The ability to riffle rapidly through a set of pages has previously been a method of rapid scanning for information that could only be done with physical books.

4.6 Other tools

Too many tools exist to provide an exhaustive list here. Cockburn and Jones have reviewed twelve visualization tools (which create 2- or 3-dimensional maps of hyperspace) in [6].

4.7 Discussion

Although many of these tools do allow for easier navigation, there is one problem inherent in all of them. The tools are to be used embedded in the Website for which they provide the information. This means that there still is no consistency between sites, even if every site were to use the same tool. Placement, appearance and layout of the tool can still be customized, which means that a reader at least has to figure out where the navigation tool is on this site.

Compared to the present situation, using one of these tools on one's Website may make a positive difference with respect to the users' abilities to navigate the site.
5.1 Jakob Nielsen’s 1994 Web usability study

In December 1994, Jakob Nielsen investigated the usability of several Web sites. Although the study only included three participants and four Web sites, some observations are still very interesting:

- During the assignments (which involved finding certain information on the sites under investigation) users sometimes identified the need for search after they had traversed several links and were deep in the hierarchy. They expressed a desire to be able to start a search from their current location and not have to navigate back to the home page first, which was the only place where the sites under investigation offered links to search facilities.

- The interface of the browser being used (NCSA Mosaic 2.4 for the X Windowing System) was not clear enough about the type of resource available from a link. In a few cases, users retrieved information that caused a PostScript viewer to be launched without warning. The users complained that the WWW page had not warned them that the link would download a PostScript document rather than jumping to another hypertext screen and they also did not want to read the PostScript document (as mentioned above, the users disliked long texts).

Users had low tolerance for anything that did not work, was too complicated, or that they did not like. They often made comments like “if this was not a test I would be out of here” or stated that they would not want to visit a site again after a quite small number of problems. With non-WWW user interfaces, the technically oriented users in this study would normally persist for some time in trying to figure out how to use the system, but with the WWW, there are so many sites out there that users have zero patience. Thus, the demands for good usability are probably higher for WWW user interfaces than for normal user interfaces, even though the designers’ options are fewer.

The study found that users wanted search and that global search mechanisms should be globally available. Even so, users were poor at specifying search strings and they often overlooked relevant hits. Thus, a site cannot rely on search as the main navigation feature. “Navigational structure and overviews are necessary to avoid user confusion and should be provided both in the large (server structure and location) and in the small (structure for the individual pages with iconic markers for the various types of information).” (Nielsen, 1994)

5.2 How People Revisit Web Pages

In 1997 Linda Tauscher and Saul Greenberg investigated the revisitational behaviour of users on the World-Wide Web [17]. The study found that only 42% of the requested pages were visited for the first time by a user. A comparison of various techniques to keep track of the history of visited URLs revealed that the popular history stack was inferior to the simpler approach of showing the last few recently visited URLs with duplicates removed. Other predictive approaches fare even better.

---

Field studies

Their conclusions are presented here.

1. Users revisit a considerable number of Web pages. Our analysis of recurrence rate shows that there is a 58% probability that the next page visited was previously seen. This qualifies Web browsing as a recurrent system.

2. While many pages are revisited, users continually incorporate new pages into their repertoire at a regular rate. There are also local variations in the vocabulary growth rate and use of navigation activities across users, and across their browsing timeline. These variations indicate the presence of different browsing activities.

3. Users visit very few Web pages frequently. Consequently, many Web pages are only visited once (60%) or twice (19%). The few frequently accessed pages tend to fall into certain definable categories.

4. Users exhibit considerable recency of revisits. The major contributions to the recurrence distribution are provided by the last few pages visited, which also explains why 'Back' is frequently used (30% of all navigation actions).

5. Users revisit pages that have not been accessed recently. For example, 15% of recurrences are not covered by a list of the last 10 URLs visited. Still, doubling or tripling the size of the list does not increase its coverage much.

6. Users do not have strongly repeatable linear patterns when browsing selected clusters of pages. Both locality sets and longest repeated sequences are small, rarely repeated, and also exhibit recency.

7. Methods to present a history list of previously visited pages are available that are more predictive and usable than the current stack-based approach. Presenting the last 10 or so recent URLs, with duplicates saved only in the latest position, surpasses current stack-based approaches and are likely much more usable. Other methods fare even better, although their usability must be determined.

8. The Back button is very effective and simple to use. Perhaps it could be improved further by basing it on a recency rather than a stack model.

9. The poor use of history facilities is likely due to several interface problems. Firstly, hotlists in Mosaic and many other contemporary browsers require considerable effort to manage. Consequently, users may not bother to add a URL to the list, may forget that it is there, or only record URLs that are convenient starting points. Secondly, the “Window History” in Mosaic 2.6 is not visible and requires several actions to access. (While Netscape 3.0 does allow a history window to remain in view, it must be raised through an explicit menu action for every session). Thirdly, history lists in Mosaic, Netscape, and other browsers are based on the stack model, which means that the desired URL may have been popped off the list even though it was entered a short time ago. We expect these problems will be mitigated in future systems by both interface redesign (perhaps by making the last few items always visible on the display) and by taking advantage of new browser features (e.g., using frames to include hotlists as a fundamental part of the display). However, browser redesign will require careful consideration and evaluation, which is beyond the scope of this paper.

Based on their findings, Tauscher and Greenberg present eight recommendations, which are given below in slightly abbreviated form.

1. Maintain records of URLs visited, and allow users to recall previous URLs from those records. Our study shows that though users incorporate new URLs into their repertoire at a regular rate, 58% of Web pages are revisited. Web navigation is thus classified as a
5.2 How People Revisit Web Pages

recurrent system. Hence, a history mechanism has value, and as a first requirement, it must record the URLs that users visit. To obtain the maximum benefit from this data, users must be able to access the URLs during their current as well as later sessions. However, Web browsers fail to maintain adequate records of URLs visited, as most only supply users with session-based history. While they do maintain a persistent history list for internal purposes, browsers do not give users access to the history data; the information is only used to indicate which hyperlinks have been recently accessed. Even within sessions, the history list may discard some of the URLs visited because of its stack-based model.

2. It should be cheaper, in terms of physical and cognitive activity, for users to recall URLs from a history mechanism than to navigate to them via other methods. The prime motivation for providing a history system is to reduce the physical and/or cognitive effort of returning to a particular Web page. The heavy use of the Back button (30%) indicates it is an effective way for the user to reach the last few items, as long as page redisplay is quick. In contrast, users select URLs from the history dialog less than 1% of the time, likely due to various physical and cognitive overheads involved.

3. Other strategies for presenting the history list, particularly pruning duplicates and hierarchical structuring, increase the probability of it containing the next URL. A significant number of URLs to be revisited are not covered by the last 10 pages visited (26% of the recurring total). It is these missed URLs that could help the user most since they occurred long ago and are thus more difficult to recall and/or locate. Pruning duplicates from a recency ordered list is a simple improvement, as it makes room for additional URLs in a list of a fixed size (in this case 47% of the URLs in the list would be useful, vs. 43% in the original situation with a ten item list).

Also, hierarchical structuring is a way of bringing distant nodes closer by offering branching points. We have already presented two menu-based methods to do so: context-sensitive sublists and Web subspaces (51% and 53% of the URLs respectively vs. 43% for strict recency for a ten item list).

4. History based on recency is not effective for all possible recalls because it lists only a few previous events. Alternative strategies must be supported. Recency was a strong reuse pattern but we found that other patterns exist. For example, a few key pages are accessed with a high frequency. One of these, the user’s home page, is easily accessible by the option of it being both the start-up document, and reachable through the Home button on the browser toolbar. Other frequently accessed pages could be made available on a toolbar for easy access. A drawback of frequency ordering is that it has a certain degree of non-intuitiveness. That is, during post-study interviews, subjects were sometimes surprised to see certain URLs on their 15 most frequent URLs list. Two alternative strategies are worth mentioning. Firstly, identifying and presenting paths to the user may be useful, though additional research is required to improve path detection within the WWW domain. Secondly, for infrequently accessed URLs that have not been visited recently, the ability to perform a parameterized text search on one’s history could be beneficial.

5. Consider current browsers within this context. Most of the alternative strategies for recalling distant pages are provided (if at all) through helper applications or add-on software that are not well-integrated with the browser. While Netscape Navigator 2.0 incorporates a search feature, it only searches the user’s bookmarks list and not the internal global history list.

6. History items should have a meaningful representation. When pages are presented as items in a menu, they must be identified by a name or symbol. Yet finding appropriate symbols for pages is an issue. URLs are usually poor identifiers as their names are rarely meaningful.
7. Support grouping of URLs into high-level Web tasks, and switching between tasks. Locality did not prove to be prevalent enough to identify common tasks. Still, we believe that it would be useful to allow items on the history list to be grouped in a way that reflects a user's task.

8. Allow end-user customization of history data. The seam between history and hotlists should be eliminated. The idea is that the history list could offer sets of candidates, and that users can decide at any time which of those deserve greater emphasis and saving for posterity.
An overview of the capabilities of several browsers

In this chapter of this paper four popular browsers (Netscape Communicator 4.0, Microsoft Internet Explorer 4.0, Lynx 2.7.2 and Opera 3.21) are compared on the following points: the ability to present hypermedia information, to navigate certain sites, to create bookmarks, to give visual hints about links and their consistency in presenting information.

While the choices for Netscape and Internet Explorer in this overview are probably obvious (each is claimed to hold about 40% of the total browser market), Opera and Lynx may need some explanation. Lynx is a browser for the UNIX command line, its main advantage is that it is very fast, easy to use and is very extensible. In browser comparisons, Lynx is often used as an example of a browser on a totally different platform, since most browsers only run in graphical environments. This makes Lynx one of the primary candidates for use on platforms such as PDAs, where usually only small text screens are available.¹

Opera is a browser for the MS Windows platform, with many options to customize it to one’s liking. It has been included here for this reason, and to show that it is possible to develop a browser that is not based on NCSA Mosaic (as both Netscape and Internet Explorer are).

A detailed description of each browser is given below.

6.1 Netscape Communicator 4.0

Netscape started out as the commercial version of the famous Mosaic browser (developed at NCSA) and quickly became the standard browser on the World-Wide Web. It is available on 18 different platforms. The Communicator version also integrates a mail client, a newsreader and several other programs.

Because it (also in its previous incarnation as NCSA Mosaic) was the only widely-used graphical browser for several years, its appearance and behaviour has set the standard for Web browsers.

Netscape Communicator can be downloaded from http://home.netscape.com/download/

6.2 Microsoft Internet Explorer 4.0

On the Microsoft Windows platform, Internet Explorer is regarded as a serious competitor to Netscape’s browser. The very tight integration with the Windows 95 desktop (and complete integration in the Windows 98 interface) make it an easy choice for many Microsoft Windows users.

Internet Explorer can be downloaded from http://www.microsoft.com/ie/ie40/

¹The author of this article has repeatedly used Lynx on a Psion with GSM phone, much to the amazement of his friends and family.
6.3 Opera 3.21

Opera is a recent browser for the MS Windows platform. It promotes itself as a “lean and mean” browser. At only 10% of the size of Netscape and Internet Explorer, it performs many times faster, is much more customizable and uses a multi-window approach to allow easy concurrent Web browsing.

In contrast with Netscape and Internet Explorer, Opera is not freeware. A demonstration version which stops working after 30 days is available, the full version costs US$ 35.

Opera can be downloaded and ordered from http://www.operasoftware.com/

6.4 Lynx 2.7.2

Lynx is a freeware text-mode browser, primarily intended for UNIX platforms, although versions for MS-DOS, VMS, Amiga and OS/2 also exist. It can only display textual pages, but images and other resources can be viewed with an external helper application.

The browser is fully HTML 3.2 compliant and has a lot of customizable options, thanks to the large number of people who helped in its development. The program is freeware and comes with full source (under the GNU General Public License).

Lynx can be downloaded from http://lynx.browser.org/

6.5 Evaluation

The following features have been evaluated in this comparison (for a detailed explanation of each feature, please see section 3.1):

**Bookmarks** are, as previously explained, the ability to remember URLs of visited documents, so they can be revisited easier. All browsers support this feature, although the presentation and organization of bookmarked URLs is done slightly different. Netscape and Opera use a menu structure, with submenus for different categories. Internet Explorer also allows the user to navigate through bookmarks (called “Favorites”) in a menu, but they are organized as folders on the desktop, allowing navigation with the standard Windows tools. Lynx creates an HTML document with a list of all the documents bookmarked. The user can edit this file manually to re-organize the bookmarks in categories. There is no bookmark maintenance feature in Lynx.

**The history stack** allows the user to go back to previously visited documents in order of recency. All browsers have a “back” and a “forward” button, which allows the user to navigate back and forth, but Opera does not have an easy way to display the current stack.

**A Webmap** is an extended version of a history stack, which allows the user to navigate between all documents he visited during the current session. None of the tested browsers supported this feature.²

**Sneak Preview** capabilities give the user information on documents behind a link before he follows it. Netscape and Internet Explorer only display the URL of the target document. The document author can use scripts to display more information, which then replaces the URL. Lynx and Opera can also display information supplied by the document author, in addition to the document URL.

**Customization of the document's appearance** refers to the ability to control how a document is displayed. Netscape and Internet Explorer only allow the user to change the

---

²The only browser with this ability is IBM’s WebExplorer for Windows and OS/2, but this product has been taken off the market in 1996.
main font used to display text. More detailed customization can be done with a “style sheet”, usually supplied by the document author. Internet Explorer also has the ability to use a reader-supplied style sheet. Opera offers a menu to customize the appearance of almost every element. Lynx’s rendering of documents is not customizable (without changing the program itself).

**Document resizing on the fly** allows a user to enlarge or shrink the entire document, for easier reading or a better overview. Netscape nor Internet Explorer support this feature. The user can increase or decrease the font size, but images are not resized accordingly. Opera offers a menu to zoom in or out on the entire document, including images. Lynx does not have this capability.

**A navigation toolbar** can be generated based on link relation information embedded in the document. This gives the user a toolbar with the same functionality in the same location in every document, which provides a very consistent interface. Section 3.2.1 explains this in more detail. Only Lynx currently supports this feature; a list of links can be called up with one keystroke, which allows for easy navigation.

### 6.6 Feature summary

<table>
<thead>
<tr>
<th>Feature</th>
<th>Netscape 4.0</th>
<th>Internet Explorer 4.0</th>
<th>Lynx 2.7.2</th>
<th>Opera 3.21</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bookmarks</td>
<td>Yes (menu)</td>
<td>Yes (folder)</td>
<td>Yes (list)</td>
<td>Yes (folder)</td>
</tr>
<tr>
<td>History stack</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Web map</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Sneak preview</td>
<td>URL only</td>
<td>URL only</td>
<td>URL and title</td>
<td>URL and title</td>
</tr>
<tr>
<td>Customization of document appearance</td>
<td>Yes (style sheets)</td>
<td>Yes (style sheets)</td>
<td>No</td>
<td>Yes (menu)</td>
</tr>
<tr>
<td>Document resizing on the fly</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Navigation toolbar</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
A survey of navigational aids

A survey was presented on the WDG’s Web site (http://www.htmlhelp.com/) for a period of two weeks, with a prominent link to it from the main page. Visitors were asked to answer a series of questions in multiple-choice format.

The Web Design Group is a non-profit organization which promotes the creation of Web sites that are accessible to all users, regardless of browser, platform or setup. One of the means it uses to achieve this, is by maintaining a Web site with information on developing sites, annotated versions of the relevant specifications and discussion fora. The site is visited by thousands of people every day, most of which are Website developers, authors and designers.

7.1 Text of the survey

Below the full text of the survey is presented. For the first five questions, the reader could choose from five different answers, ranging from “Not at all important” to “Very important”, except for question two, which ranges from “Never” to “(Almost) every time”.

Question six was in a slightly different format. The goal of this question was to find out when, if at all, people open a new browser window if asked. The answer “Sometimes” could be specified further with a list of possible reasons given below the question. Due to limitations in HTML forms, reasons in this list could also be checked if an answer other than “Sometimes” was given.
Question 2

How often do you use the history list in your browser when you spend some time on the World-Wide Web?

(_) Never
(_ Rarely
(*) Sometimes
(_) Often
(_) (Almost) every time

Question 3

How important is it that sites offer detailed site overviews?

(_) Not at all important
(_ Moderately unimportant
(*) Neither important nor unimportant
(_) Moderately important
(_) Very important

Question 4

How important is a search feature for the site?

(_) Not at all important
(_ Moderately unimportant
(*) Neither important nor unimportant
(_) Moderately important
(_) Very important

Question 5

Do you think that a navigational toolbar in the same spot on all sites is important?

(_) Not at all important
(_ Moderately unimportant
(*) Neither important nor unimportant
(_) Moderately important
(_) Very important
7.2 Running the survey

The survey was made available on the WDG's Web site from Tuesday, September 8th until Tuesday, September 22nd. During this two-week period, 351 responses were received. There was one three-hour outage of the Web server during the night between September 12 and 13, which made it impossible to submit survey results sent in during that period.

All survey responses were entirely anonymous. To prevent duplicate voting, the IP address\(^1\) of the participant and the date/time when the response was sent in were stored together with the vote. This only helps in part, since a determined person can always switch to another host, or fill in the survey every day at a different time. However, it does eliminate ballot-stuffing through automated scripts.

There were a total of nine IP addresses found from which more than one response was sent. In eight of these cases, this meant two responses per IP address, with a large time between the two. This is most likely simply two people who were assigned the same IP address by their provider. The ninth case consisted of four responses sent within ten minutes. All responses were the same, so it is possible that the duplicates were sent accidentally. The three duplicates have been filtered, so that the response was counted as only one. This gives a total of 348 unique responses.

The number of responses per day is given in the below table:

<table>
<thead>
<tr>
<th>Day</th>
<th>Number</th>
<th>Day</th>
<th>Number</th>
<th>Day</th>
<th>Number</th>
<th>Day</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>23</td>
<td>12</td>
<td>19</td>
<td>16</td>
<td>20</td>
<td>20</td>
<td>30</td>
</tr>
<tr>
<td>9</td>
<td>33</td>
<td>13</td>
<td>15</td>
<td>17</td>
<td>21</td>
<td>21</td>
<td>26</td>
</tr>
<tr>
<td>10</td>
<td>33</td>
<td>14</td>
<td>28</td>
<td>18</td>
<td>17</td>
<td>22</td>
<td>23</td>
</tr>
<tr>
<td>11</td>
<td>25</td>
<td>15</td>
<td>23</td>
<td>19</td>
<td>16</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The average number of responses per day is 23.4.

The following table shows the different possible responses and the percentage of respondents who chose them. Question six is discussed separately due to its different structure.

---

\(^1\)Every computer on the Internet has a unique address, called the “Internet Protocol” or IP address, which can be used to identify the sender or recipient of a request.
The results of question six are presented in the next table.

<table>
<thead>
<tr>
<th>Answer</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>No, never</td>
<td>224</td>
</tr>
<tr>
<td>Yes, always</td>
<td>5</td>
</tr>
<tr>
<td>Don't know</td>
<td>16</td>
</tr>
<tr>
<td>Sometimes...</td>
<td>104</td>
</tr>
</tbody>
</table>

As shown, 104 people chose the answer “Sometimes” in question six. They further specified as follows (numbers between brackets are responses from people who did not check “Sometimes”):

<table>
<thead>
<tr>
<th>Answer</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>When I feel like it.</td>
<td>27 (6)</td>
</tr>
<tr>
<td>When the site is not usable otherwise.</td>
<td>53 (4)</td>
</tr>
<tr>
<td>When I have the listed browser handy.</td>
<td>35 (1)</td>
</tr>
<tr>
<td>When I'm not in a hurry.</td>
<td>21 (3)</td>
</tr>
<tr>
<td>Other.</td>
<td>5 (5)</td>
</tr>
</tbody>
</table>

7.3 Conclusions of the survey

In response to question one, 75% of the responses indicated that a navigational toolbar is considered important on a Web site (42% thinks it is important, 33% even thinks this is very important). However, question five shows that only 30% thinks the consistency of the toolbar’s placement is important, and an even smaller minority (16%) thinks this is very important.

An interesting result is found when the answers to question one are compared to those of question five. Of the 261 people who said a toolbar is important or very important, 50% also said its placement is (very) important. 30% Does not think the placement is important.

The results for question two indicate that only 9% of the surveyed people never use the history list in their browser, 50% use it but not very often, and 41% use it often or very often. This may be explained by the unintuitive nature of the stack model, which makes it a less useful tool than it could be (see also section 3.1 and [5]).

Site overviews are considered important by a slight majority (54%), but only 14% find a site overview to be very important. 25% of the responses express no preference one way or the other, and the rest (21%) thinks a site overview is not an important feature of a Web site.

The preference for search features is clear from the results of this question: 37% of the responses states a search feature is important, and 30% even thinks it is very important for a site to have a search feature. This means that two-thirds of the respondents consider a search feature important for a site. This should be no surprise, as it is entirely consistent with other research in this area (see for example the discussion of Nielsen's survey in section 5.1).

The responses to question six make clear that few people will take the time to start a new browser if asked. 64% of the surveyed says they never do this. 30% do this only in certain situations, mostly when a site is not usable with the browser they are currently using (38%).

---

Table 1: Importance of navigational aids

<table>
<thead>
<tr>
<th>Question</th>
<th>1</th>
<th>2a</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not at all important</td>
<td>6</td>
<td>30</td>
<td>18</td>
<td>20</td>
<td>69</td>
</tr>
<tr>
<td>Moderately unimportant</td>
<td>33</td>
<td>88</td>
<td>54</td>
<td>25</td>
<td>46</td>
</tr>
<tr>
<td>Neither important nor unimportant</td>
<td>49</td>
<td>88</td>
<td>88</td>
<td>69</td>
<td>74</td>
</tr>
<tr>
<td>Moderately important</td>
<td>146</td>
<td>108</td>
<td>140</td>
<td>129</td>
<td>104</td>
</tr>
<tr>
<td>Very important</td>
<td>115</td>
<td>35</td>
<td>49</td>
<td>106</td>
<td>56</td>
</tr>
</tbody>
</table>

*aAnswers for this question actually range from “Never” to “(Almost) every time.”*
or when it is not a problem (that is, when the requested browser is available, in 26% of the responses).
A small-scale experiment

In this experiment, a small group of users ($n = 5$) was asked to perform a few simple tasks on several Web sites. The aim of this experiment was to determine whether the different methods of navigation offered on the different sites affected the speed with which the tasks could be completed.

Three of the five subjects were members of MCGV STACK, the Eindhoven University's computer club. They were very familiar with the World-Wide Web and the browsers used in the experiment. The other two sometimes "surfed the Web" but only used one browser (one used Netscape, the other used Internet Explorer) to do so, and had never tried another browser.

8.1 Approach of the experiment

The following Web sites were used in the experiment:

1. CNN Interactive - a large information provider, with lots of links, no internal search feature but a very consistent interface.
   http://cnn.com

2. Microsoft's Web site - a very popular site. Has an internal search engine, but no index and is often reorganized and redesigned.
   http://www.microsoft.com

3. The Pretty Good Privacy list of frequently asked questions with answers - a textual list of questions and answers on a popular file encryption program. Presentation is very straightforward, consistent and simple, there is extensive cross-referencing and information to generate a toolbar.
   http://www.pgp.net/pgpnet/pgp-faq/

4. The Dilbert store - an online shop where people can buy items related to the popular Dilbert comic strip. It is very graphics intensive.
   http://www.dilbert.com/store/

In order to minimize delays and the risk of the site being unavailable or getting an upgrade during the experiment, all sites were downloaded prior to the beginning of the experiment. A local Web server was set up to serve all documents. Since each subject used a computer with a different IP address, the logfiles of this server could be used to reconstruct the path that each user took to reach the documents he was looking for. It could also be used to compute the time necessary to do this.

Unfortunately, this approach prevented the use of a site's search engine during the experiment. This turned out to be rather frustrating to the subjects, who all indicated they preferred to use a site's search feature if they couldn't find the information they want after trying a few paths likely to be successful.

The Dilbert site also has an online ordering mechanism, which could not be copied either. This limited the possibilities on this site to browsing through the catalog.
Each subject was asked to locate some specific information on each site, with each of the four browsers discussed in chapter 6. The tasks were chosen in such a way that the contents of these documents were comparable. The time necessary to do this was recorded in each instance, along with the path that each subject needed to locate the page with the information he was looking for.

With this approach, the subject would have to perform the same tasks with his browser on each site. This should make it possible to compare the actions necessary to perform the task between the different subjects.

The assigned tasks to perform were:

- **CNN**
  
  1. Find your own horoscope for today.
  2. Find the verdict of judge Jackson in the case of the US Department of Justice vs Microsoft, Inc. (concerning the possible abuse of Microsoft’s monopoly on the desktop to promote its Web browser).

- **Microsoft**
  
  1. Find the list of new features in the latest version of Internet Explorer.
  2. Find the upgrade for Microsoft Outlook which fixes the security breach that allowed remote users to crash the program with a specially constructed e-mail message.

- **PGP FAQ**
  
  1. Find the list of commands for a "key server."
  2. Find in which states a digital signature is legally binding.

- **The Dilbert Store**
  
  1. Find the page with information on the Dilbert tie.
  2. Find the ISBN number of the latest Dilbert comic book.

By having two tasks on each site, it is less likely that a subject would remember too much about his previous session, which would make the results in the next experiment unreliable.

### 8.2 Results of the experiment

The server’s log files made it possible to obtain the time necessary for each subject to find the desired information. Also, the path the subject took could be reconstructed with this information. With this information, we can determine how difficult it was to complete the assignments.

All subjects used all four browsers in the same order: first Netscape, then Internet Explorer, then Opera and finally Lynx. In order to minimize the chance that a subject would remember the path he took previously, all subjects were asked to perform all tasks with one browser, then with the next, and so on. By the time they got back to the first site, they could no longer remember exactly which path they had taken.

The average times (in minutes:seconds format) necessary to find the requested page are given in the table below. We compare this against the browser used, in order to determine if this is a significant factor.
8.3 Observations during the experiment

Of the five subjects, two were self-expressed “Netscape lovers”, while one strongly preferred Microsoft Internet Explorer. One subject used Opera as his standard browser. The others had heard of it, but rarely used it. Lynx was familiar to the three STACK members, who all had used it often enough to know how to use it. The other two had never used Lynx before, and needed a printed copy of the “Key-strokes command” overview from its help files in order to use it.

All five subjects indicated they preferred a graphical browser to navigate a Web site. Reasons given were that Lynx (the only text-mode browser used) made it rather tiresome to quickly browse a large page, that it was inconvenient to have to browse through an entire document to find the link you want (Lynx only allows the user to choose a link after he has navigated to it with the “Up” and “Down” arrows), and because most Web sites were primarily aimed at graphical browsers. Two subjects said they often used Lynx to download text-oriented information (reference sites, news, and the likes), or when they were in a hurry to locate information. Having to start a graphical browser and downloading an image-heavy document took longer than starting Lynx and downloading just the text, even if it meant some extra work to read through the “mess” that resulted from viewing a document with lots of graphics in a text-only environment.

Lynx’s toolbar feature (which allows the user to quickly navigate to the previous or next document, or to an index or another related document) was only used by one subject. Most of the sites did not provide the information necessary for Lynx to generate a toolbar, so there was not much chance to use it.

The subjects who had never used Opera before were surprised at its speed, customizability and ease of use. It allows the user to open more than one window at a time, at a much smaller performance penalty than with the equivalent feature in Netscape and Internet Explorer. This made it much easier to manipulate multiple documents at the same time, so comparisons could be made quicker.

The default configuration of Opera makes it appear slightly different from the other two graphical browsers. Since the overall appearance is almost identical, this caused some frustration when “the small things don’t work the same”. Reconfiguring Opera so that it worked and looked the same as Netscape solved most of these frustrations. One subject was very surprised that it was at all possible to change the behaviour of a browser so much.
While searching for the documents, most subjects used only the “forward” and “back” buttons to navigate between visited documents. Often they would also use “back” links in the document, which do not perform the same action in the stack-based history model used by all tested browsers (see the explanation of the history list in section 3.1). This led to some frustration, as the “back” button would not always return them to the node they were expecting.

To overcome this problem, four of the five subjects used the bookmarks feature in their browsers to permanently “remember” important pages. This allowed them to go back to these nodes if a particular path did not turn out to be successful, regardless of what the history stack looked like. The main disadvantage of this method was that a subject has “to clean up [his] bookmarks after every session”. An “anchor” feature, which allows a user to access pages he considers important with one keystroke was suggested by two subjects.

One subject found that Opera’s multiple browser windows made it possible to do this without using the bookmarks. He simply opened a new window to explore a path if the current page seemed important to him. If the path didn’t turn out to be useful, he returned to the “important” page through the “Window” menu and then used the same technique on the next promising link. This technique was also tried by one of the other subjects while using Netscape, but the overhead required by Netscape to maintain multiple browser windows makes this rather impractical.

It was not possible to duplicate a site on the local server with its search feature intact. This was very frustrating to all subjects, since they were used to finding information with a search engine, rather than browsing through several indices. When asked, subjects said that this type of browsing is very tiresome, since “you have to read every document in the hope that it contains a link you want” and “most sites don’t have their site structure organized in a logical way”. Searching was considered a faster way, since “you immediately get all pages related to what you are looking for”, although sometimes a search feature only allows primitive searches (only searching by keywords, no boolean constructs which allow more precise searches) or also finds wrong documents.

8.4 Conclusions of the experiment

There are little to no differences in current graphical browsers that makes one better or worse than the others with respect to navigation. The tested browsers all use the same navigational tools, with the same stack-based history list.

Users expect this history model in a browser to follow their intuition. Using “back” links on a Web page violates their model of expectations and can lead to great frustrations.

The navigational tools in the tested browsers all miss a technique to temporarily “mark” potentially interesting pages. Such a technique would be very useful to return to these pages more quickly and would allow more intuitive navigation. This technique could be implemented in a way similar to the way the “anchor” feature found in many word processors is implemented. The current approach of using the bookmark feature in this way is not entirely satisfactory, since bookmarks are persistent across sessions, which requires “cleaning up” after every session.

Users want a search feature, to locate the information they want quickly. This is consistent with Jakob Nielsen’s findings in his 1994 Web usability report [13]. This search feature should allow complex searches so that users can filter irrelevant links and more quickly locate the documents they want.\footnote{This conclusion may be biased, since most of the subjects in this experiment were very experienced with search tools.}
A toolbar feature, such as the one available in Lynx, is currently of little use. Web site developers would have to provide this information in all documents, and then browsers can make wider use of it. However, browser developers will most likely not add this feature until developers make the information available, and those in turn will not add this information until browsers can make use of it. It is not very likely that this “chicken-and-egg” problem will be solved any time soon.
Conclusions

The goal of this paper was to give an overview of the methods with which current World-Wide Web-browsers present hypermedia to their users, to review the usability aspects of these methods and to compare them with the methods to present and process hypermedia as offered in the literature.

We have seen that current World-Wide Web-browsers have only very limited abilities to allow navigation through hyperspace. The reviewed hypermedia literature offers many possibilities to make navigation of hyperspace easier. However, most of today's browsers implement only the simplest of these tools. Enhancing the browsers to add support for these tools seems like a straightforward suggestion, but unfortunately it requires a lot of cooperation between browser developers and site creators. Due to competition between all the involved parties, it is very difficult to reach a common standard to solve this problem.

The various tools that were developed by third parties to enhance browsing capabilities are of limited use. They can only be used on a specific site if the site creator has added it, and most of the times they are designed specifically for that site. This means that a user still has to learn how to navigate on every site.

The use of metadata (data about data) would make it easier to index documents and to allow automated building of tables of contents and navigation bars and the distinction between different types of hyperlinks. Although possible in theory, in practice very little use is made of the information that can be extracted from metadata. Several proposals have been made to enhance the abilities to include metadata in Websites.

The related problem of cross-referencing has also been discussed briefly. Information that can be used to determine the location of a node in a hyperdocument can also be considered metadata. Where there is interest in developing methods to process data about documents (especially where intellectual property rights and identifying information about the author and owner are concerned), virtually no one seems interested in using metadata for cross-referencing.

The experiment has shown that the stack-based history model used in all current browsers has some serious deficiencies. Users expect this history model in a browser to follow their intuition. Using "back" links on a Web page violates their model of expectations and can lead to great frustrations. Sometimes for no apparent reason to the user, nodes disappear from the list. To overcome this, users have started using other tools (such as the bookmarking facility) to be able to always return to specific points.

A "Webmap", which gives an overview of all visited nodes, may be a useful tool to solve this problem. Site overviews — which could be created with a Webmap, although other methods also exist — are considered important by a slight majority of the participants in the WDG survey. Almost a quarter of the responses stated that such an overview is not important.

The survey also shows that a large majority of people who regularly use the Web prefer to have a navigational toolbar on the site. However, only 46% of this majority thinks the consistency of the toolbar's placement is important.

The literature, the survey and the experiment all made it clear that there is a large preference for search capabilities on a site. The experiment showed this in a negative way, since users could not use the search facilities (because the approach used did not allow this) and protested about this quite vocally.
It is a disappointment that there seems to be so little interest for the existing solutions to most of the problems encountered in this paper. Competition and a desire to control the browser market are probably the two main causes for this problem. Most of the current site developers have resorted to designing user interfaces in the documents, to make up for the limitations in the user interfaces in the browsers. This is a great setback after user interfaces such as the MacOS and Microsoft Windows have standardized the way applications look and feel.
Bibliography


