Organising Information on the Next Generation Web - Design and Implementation of a New Bookmark Structure

Sheng-Uei Guan[a] and P. McMullen[b]

[a]*Department of Electrical & Computer Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore 119260 email: eleguans@nus.edu.sg*

[b]*School of Computer Science and Computer Engineering, La Trobe University, Vic. 3083, Australia*

Abstract

The next-generation Web will increase the need for a highly organized and ever evolving method to store references to Web objects. These requirements could be realized by the development of a new bookmark structure. This paper endeavors to identify the key requirements of such a bookmark, specifically in relation to Web documents, and sets out a suggested design through which these needs may be accomplished. A prototype developed offers such features as the sharing of bookmarks between users and groups of users. Bookmarks for Web documents in this prototype allow more specific information to be stored such as: URL, the document type, the document title, keywords, a summary, user annotations, date added, date last visited and date last modified. Individuals may access the service from anywhere on the Internet, as long as they have a Java-enabled Web browser.

Keywords: bookmark, World-wide-web, Web browser

[a] Corresponding author. E-mail address: eleguans@nus.edu.sg

1. Introduction

In 1988, the volume of scientific, corporate and technical information was doubling every 2.2 years and by 1992, it was doubling every 1.6 years. With the expansion of the Internet and other networks and the flourishing use of the World Wide Web (W3) as an information repository, this rate of increase will continue [3]. The success of the next-generation Web will depend on the ability to manage information in a manner that suits both individuals and workgroups and allows information to be accessed from any location on the W3. The implementation of a new bookmark structure, common to all the major Web browsers, may provide an excellent method to manage bookmarks in a user-friendly and organized manner.

1.1 Overview

The problem of remembering where previously seen Web pages are located is addressed in the idea of a ‘bookmark’ in Netscape, ‘hot-lists’ in Mosaic and ‘favourites’ in Microsoft explorer (all to be referred to as bookmarks). When a user encounters an interesting page, he can store the page's address (Universal Resource Locator, URL) and title in the bookmark structure. Later, the user can look at the bookmarks, select any title from it, and fetch that page. Bookmarks can be considered a form of long-term memory.

Bush first put forth the original idea of bookmarks in July 1945. He described a device called "memex" in which an "individual stores his books, records and communications, and which is mechanized so that it may be consulted with exceeding speed and flexibility. It is an enlarged intimate supplement to his memory" [2]. Bush described the essential feature of memex as its ability to tie two items together.

Present bookmark systems are deficient in a number of ways. These include: inability to access bookmark files from any Web browser on the W3, compatibility of

bookmark files between different Web browsers, lack of facilities for sharing bookmark files and insufficient information supplied to accurately identify each bookmark entry.

Current bookmark systems only allow the bookmark file to be used locally, i.e. the bookmark file can only be used on the machine the Web browser is running. If the user is to be able to use the same bookmark file between several locations, they must manually transfer a copy of the bookmark file and update this file at each location every time the bookmark file changes. This has the potential to confuse the user as to which is the current and up-to-date bookmark file.

The issue of a user having more than one place from which he/she utilises the same bookmark file raises another issue. Each of the major Web browsers (Netscape and Explorer) stores its bookmark file in a different format making them incompatible. This restricts a bookmark file from one browser to be viewed in another. This can be a problem if a user works in numerous different environments where each environment hosts a different Web browser. The incompatibility of the browsers would make it extremely difficult for the user to use the same bookmark file in each browser. The user would have to have two separate bookmark files on each browser and manually perform the same changes to each bookmark file.

There is an increasing desire to share W3 resources among groups of people with related interests [7]. This creates another problem as current bookmark schemes are only oriented toward a single user, and provide few facilities for sharing URLs within a workgroup. When a user wishes to share a particular URL with colleagues, they must manually email that URL to the workgroup, and then workgroup members must add it to their own bookmark list. To share an entire set of bookmarks among colleagues can be an even greater challenge. Some of the latest-version browsers offer

an “import bookmarks” feature. However, the bookmark file must still be manually emailed among users and the new bookmarks manually merged into the user’s existing folder structure. The overhead of this manual method is significant enough to discourage its use on a regular basis [7].

A further limit of current bookmarking systems is that often a bookmark’s title or URL lacks sufficient information to remind a user what a bookmark is about. Over time a user may forget why a particular Web document has been bookmarked. What makes a document interesting is highly dependent on the context in which a user was working at the time, and this context continually evolves [12]. Presently the only way for a user to refresh his memory to the document contents is to visit the site where the document is stored. This can be time consuming and resource wasting, particularly if numerous documents must be checked to find the required document.

1.2 Our Approach

To overcome these issues a new bookmark structure has been developed. This paper describes its development and implementation. The intent of building this prototype is to develop a bookmark structure that facilitates cooperative work and allows more information, than present systems, to be stored for each bookmark entry.

By allowing the new bookmark files to be stored at a centralized bookmark repository, it is believed that the problem of accessing bookmark files from any Web enabled terminal on the W3 will be solved. Users will be able to log in to a server and access their bookmark files from any location.

As a possible solution to the dilemma of browser incompatibility, bookmark files will be implemented as fully structured documents in HTML. This will allow the bookmark file to be viewed by any Web or HTML browser. In addition, this would assure compatibility between older and future versions of Web browsers.

Examination of many home pages has shown that they are often used for collecting pointers to places of personal or corporate interests. Most users view current bookmarks inadequate in presentation, non-practical through long selection menus. Furthermore, bookmarks are stored locally without any support for collaboration. Thus, it seems to be a good argument in favour of web-based bookmarks being implemented as fully structured documents in HTML.

This method of storing bookmark files will solve the problem of sharing bookmark files between users and workgroups. Users will be able to share the same bookmark file, see their own and others modifications to the file, and use the bookmarks in any working environment.

This new bookmark structure will be capable of storing more specific information about each referenced Web object. In relation to Web documents this information should include such things as URL, document type, document title, keywords, a summary, user annotations, date added, date last visited and date last modified. With the implementation of these new features the problem of bookmark identification will be solved. These new features will add greater identification ability and a comprehensive history to each bookmarks' documentation.

1.3 Research Outcome

Firstly, we intend to develop a new bookmark structure to allow bookmarking of Web objects, specifically HTML documents. It is also believed that a standard for the description of bookmarked documents could possibly be developed whereby bookmarks are rigorously structured with such fields as URL, document type, document title, keywords, summary, annotations, date added, date of last access, and date of last visit. We will also implement a prototype of such a system to demonstrate

the feasibility of the ideas put forward in this paper in solving the problems of present bookmark systems outlined above.

1.4 Outline of the Paper

The rest of this paper is organized as follows: Section 2 discusses related research. Section 3 describes the architecture of the proposed prototype. Section 4 reviews design and implementation decisions faced in developing the prototype. Section 5 covers a discussion and evaluation of the prototype and section 6 concludes.

2. Related Work

There is much research and development happening in areas related to the searching, organising and sharing of Web-based information resources. Some work concentrates on bookmark files. However, there is little work being done on the evaluation of whether current bookmarking facilities, provided by Web browsers, are able to store enough relevant information with each bookmark entry.

WebTagger [7] is a prototype bookmarking service enabling authorized users to store, organize, access and evaluate URLs – either individually or within a group structure. Each user is assigned a repository to store and organise URLs. Users can also be optionally assigned access to one or more group repositories. In WebTagger, bookmarks are stored within a sparse lattice structure. The problem with this approach is that that the information relating to each bookmark entry is stored in a proprietary format unrecognizable to Web browsers. Access to these bookmarks can now only be provided to users registered with a WebTagger repository.WebTagger also does away with the hierarchical folder view, providing the user with a retrieval process that performs a search of the lattice structure. Bookmarks represent a user's personal interests and he may not want to relinquish control over the organization of his

repository to an automated tool [10]. Imagine a stranger that organizes your papers every morning on your desk; even if the organization is better than what you would have done, it is usually not acceptable. A system incorporating a mixture of manually and automatic classification would be an ideal compromise.

The uniformity of the folder hierarchy with a file tree structure provides a familiar view of information to users. Therefore, this folder view of bookmarks has been retained in the new BookMark structure. The usefulness of this folder view can be improved by the addition of searching and sorting facilities on individual folders or the entire bookmark file. These sorting and searching facilities could use the proposed additional information stored with each bookmark in the new BookMark structure. The 'keywords' tag along with 'date added' tag would aid in the use of automatic document classification and keyword searches.

The sharing of information between users is tackled by the development of a centralized information repository that operates in a client-server fashion. This method has been used in WebTagger, Grassroots [6], Jasper [3] and BRIO [11]. This implementation of file sharing is also used in the new BookMark structure. For a user to have access to shared bookmark files, as well as his personal bookmark file, he must first have an account set up in advance and must initially log in to the system. Each of the above-mentioned systems stores a user profile that has such typical information as the user's name, password, groups he has access to and his rights within individual groups.

WebTagger is implemented as a proxy-based system. The current Grassroots prototype is based on an http-proxy implementation, and can be used with any Web browser. The new BookMark system prototype has been implemented as a stand-alone server, not as a proxy-based application because the "use of a proxy can slow

down Web page delivery" [7]. Considering most servers have a proxy cache nowadays, the addition of another proxy may be considered too much overhead.

Bookmark Organizer [5] is a useful client-side-only bookmark organization. It achieves automatic classification by using clustering analysis to organize documents based on their conceptual similarity and at the same time allows the user to select when and where to apply it. Their work combines manual and automatic organization of bookmarks, confirming with the idea we mentioned earlier.

PowerBookmarks [14] provides personalized organization and management of bookmarks by combining the database with Web technologies. It can achieve advanced query, classification and navigation functions and classifies the bookmarks of all users. While PowerBookmarks allows sharing of bookmarks, it is not clear whether it facilitates group sharing and access.

SeiJi Yamada and Norikatsu Nagino proposed a database named Personal Web Map (PWM) [12] and developed the Anytime-Control algorithm to let users control their own Web map construction. PWM can help users to gather relevant information in the WWW to a small database for convenient retrieval. It will be interesting to see how their work can be extended for group work.

Twiki [15] is a Web forum for collaboration based on Zope [16] - one of the leading open source Web application server. Zope is a full-fledged software suite whose minimal execution is still overkill for some users who only require sharing of bookmarks and other Web object references while comfortable with the use of email groups for collaboration communications. Email actually simplifies private communications and the ease of including external parties into ad-hoc email "conversations". Using a Web forum based collaboration platform often requires a shift in the privacy concerns of participants.

3. Architecture

Many of the issues discussed in Section 1 can be solved by the development of a new bookmark structure. The new design attempts to reconcile with the best ideas embodied within existing systems, streamlining them for the bookmarking of Web objects. At its highest level, the architecture is broken up into two sections, the server and the client. It is then possible to break the design up further into a number of sub-sections. These sections relate to the bookmark structure, the server, the client, and the user interface.

3.1 Bookmark Design

The most common Web browsers (Netscape and Microsoft) store their bookmark files in different formats. Mosaic stores its bookmark file as a text file and uses its own form of keywords to distinguish between folders (known as menus) and bookmarks (known as items). Explorer stores bookmark files as a 'short-cut' in the windows file hierarchy structure as a text file. This text file simply holds the URL of the Web document, the folder name being the documents title. Netscape stores a bookmark file as a fully structured HTML document using the HTML tags shown in Table 1.

Table 1. HTML Tags used to Implement the New Bookmark Structure Files

Begin Tag	HTML tag description	Information between tags	End Tag
<TITLE>	A string that identifies the contents of the document	The title of the bookmark file	**</TITLE>**
<H1>	Heading level 1	The heading for the bookmark file	**</H1>**
<DL>	A definition list is a list of terms and corresponding definitions	A new folder within the bookmark file	**</DL>**
<H3>	Heading level 3	The folder name	**</H3>**
<A HREF=	The Anchor element	"url of bookmark" >Title of document	**</A>**
<DD>	Definition list definition	The description of a folder	

To support compatibility with all Web browsers the new bookmark file has been implemented in HTML as a fully structured document, in a manner similar to the present Netscape bookmark implementation. New features are added to the proposed bookmarking systems through a process termed "information hiding".

Information hiding refers to adding new features into the HTML code, specifically the <A HREF=...>…</A> tag. This tag refers to the URL of the bookmarked Web document. These new features are simply ignored by Web or HTML browsers, but once parsed and displayed in the new bookmark system will provide the ability to add richer information content to the bookmarks stored.

The HTML tags in Table 1 were chosen to implement the bookmark file as an HTML document because they each provide a meaningful name and purpose suited to the bookmarking of URLs. In a sense we have taken the implementation of bookmark files in HTML the next logical step by adding more attributes to the HTML tags, as shown in Table 2. This allows more relevant information to be stored with each bookmark.

Modifications to the HTML tags used to implement the new bookmark structure are shown in Table 2. HTML tag <H3> shows the modifications FOLDED and ADD_DATE. These are both adapted from the Netscape HTML implementation but only the ADD_DATE field is intended to be used. The FOLDED field is left only to allow compatibility with Netscape bookmark files. The field ADD_DATE will be used to indicate when the folder was added to the bookmark file.

The HTML tag <A HREF => is where the new features of the proposed bookmark system are added. This tag, like the Netscape implementation, contains the fields ADD_DATE, LAST_VISIT, LAST_MODIFIED. These tags will correspond to when the bookmark was added to the file, when it was last visited, and when the Web page was last modified respectively.

The field SUMMARY will refer to a short summary of the document. The field KEYWORD refers to a comma-delimited list of key words that relate to the Web page. The field ANNOTATIONS refers to a text file located on the bookmark server

that contains any user annotations to the Web page. The field TYPE refers to the type of bookmark that this URL is pointing to. These fields are intended for future development. Refer to section 3.5 for the intended uses of these new fields.

Table 2. Modification to the HTML Tags used to Implement the New Bookmark Structure Files

HTML Tag	Modification to tag for new bookmark structure	End Tag
<H3>	<H3 FOLDED ADD_DATE=”add date”>The Folder name	</H3>
<A HREF=”url of bookmark” >Title of document >	<A HREF=”url of bookmark” ADD_DATE=”date” LAST_VISIT=”date” LAST_MODIFIED=”date” SUMMARY=”filename” KEYWORDS=”keywords” ANNOTATIONS=”file name” TYPE=”type”>Title of document	</A>

Developing a next-generation bookmark structure through information hiding in HTML code allows the system to develop over time. This enables addition of other useful fields within the HTML tags such as host, user and newsgroup. The use of HTML also ensures backward compatibility with Web browsers.

3.2 Server Design

Sharing of bookmark files between users and groups of users is achieved by the development of a centralized bookmark repository. A bookmark collaboration working environment is typically characterized by short duration of read access by users to request references and similarly short duration of write access by users to update information. For maximum scalability in such an application, we developed the centralized server for the BookMark System to be stateless and each http connection is kept open for the user’s login session (see Figure 1). Users must go through an authentication process before they can access their bookmark files.

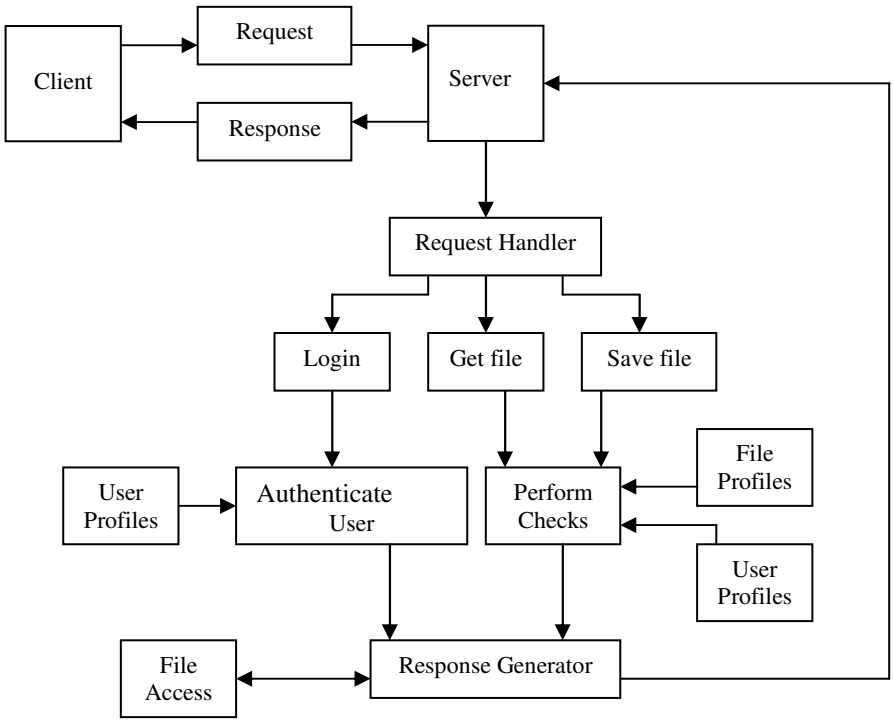


Figure 1. Server Architecture and Information Flow

Once the server is running, it will sit in an indefinite loop listening to a specified port for a connection request from clients or for requests from connected clients. A communication channel remains open between the server and client for the duration of the client's connection. A persistent connection is used for the sake of efficiency – a client with several requests will not need to reopen a connection each time it has a new request. The server, at present, allows requests to login, logout, send a file to the client and save a file from the client. Once a user connects, a separate thread is created to take care of the connection.

The new BookMark structure file system has files, directories, users and groups. The file system also supports access control lists and multi-versioned files. Each file or directory has a list of users and groups associated with it, each with read or read and write permission. A file can be either single-version (like Unix) or multi-version (like VMS).

Each user of the system has a profile that stores authentication information as well as the groups that the user has access to for the sharing of a group bookmark file. For each bookmark file stored in the system there is an associated profile that stores information such as the file's name, type, whether the file is multi-versioned, version of the file, file creator, users allowed to access the file and each user's permissions (i.e. simply read or read and write access). With these profiles, it is possible to control access to the bookmark server and the bookmark files stored on the server.

The BookMark server listens for TCP connections from clients on a default port. Once a connection is made, the connection is kept active for the lifetime of the client, and all data transfer is done over that connection (unlike HTTP).

For example, to load a file a client might send:

Request: Get

File: ~/bm/home/mcmullen/bookmark.html

To which the server would reply:

Reply: Sucess

Data: bookmark file contents

The first request a client must send after connecting to a server is the Login request, which contains username and password. This authorises the client, so the server knows which files and directories to which it has access. In the case of a group bookmark file, the user's name must be contained within the file's profile.

At present the file type indicates that the file stored is of type 'html', but it has the potential to indicate other file types. This has been implemented so the file type can easily be identified and has room to allow the handling of different file types.

The multi-versioned field, along with the file version field allows the system to keep track of file versioning information. In the case of a group bookmark file, users may have varying degrees of access to the file. Thus there is a need to allow file

locking and versioning while a user with write access is using the bookmark file. A group bookmark file while being locked will not be accessible to the other users. Before a group bookmark file is locked for 'write' access, the old version will be archived automatically while a working version is created for actual writing. Once the user has finished writing, the working version will then be saved as a new version. Concurrent read access to a group bookmark file is allowed if the file is not locked.

3.3 Client Design

The client architecture presents the features of the system to the user (see Figure 2). Given the system requirements, these features were to include the ability to login and logout of the server, download and upload bookmark files and present the bookmark file to the user in a meaningful way. This has been encapsulated in a viewing window of its own that interacts with the client's Web browser.

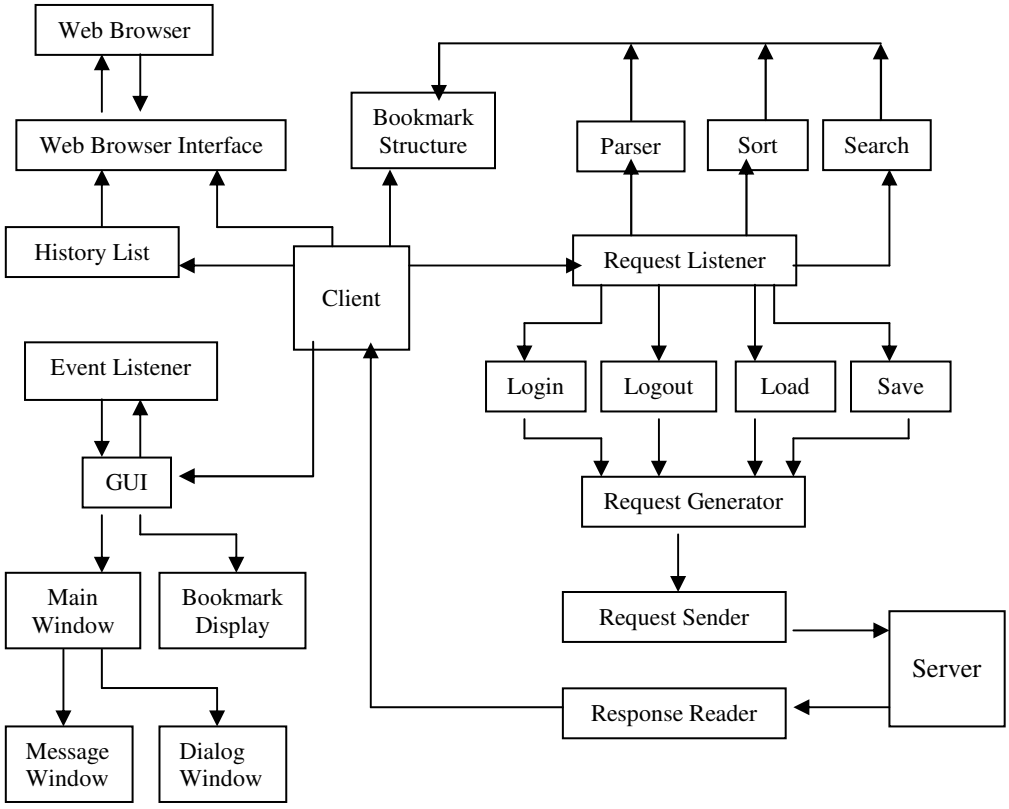


Figure 2. Client Architecture and Information Flow

Current bookmark schemes allow URLs to be stored in either flat lists or hierarchical folder structures for subsequent reference. Having just one flat list is inherently not scalable. As the bookmark structure grows, it quickly becomes unwieldy and users may forget why a certain page is referenced in their bookmarks.

Most Web browsers now provide a hierarchical folder organization of bookmarks, clearly superior to unstructured lists, as they allow users to collect entries into meaningful taxonomic groups. The new bookmark structure adopts such an implementation because of the uniformity of the folder hierarchy with a file tree structure, thus providing a familiar view of information to users.

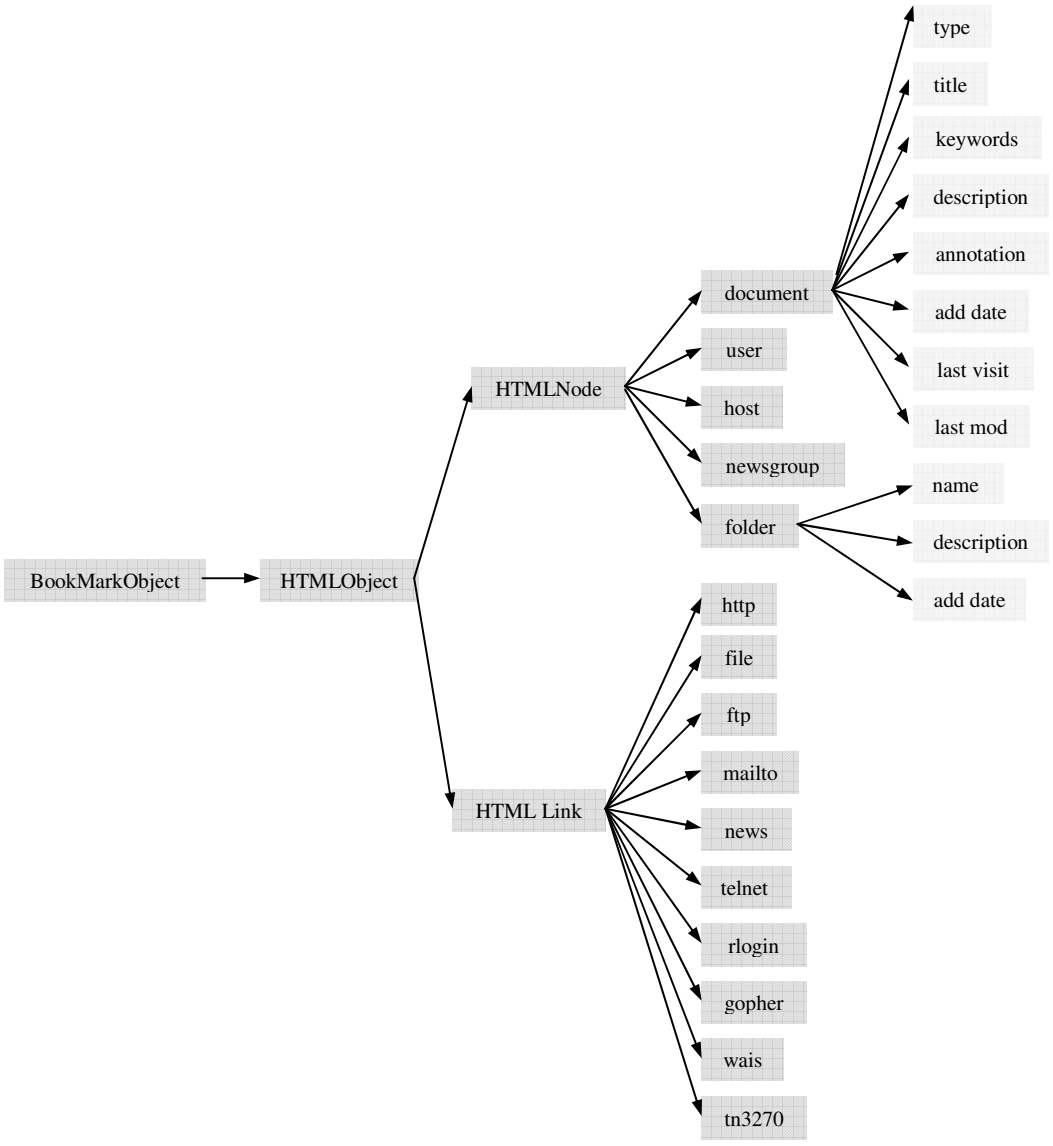
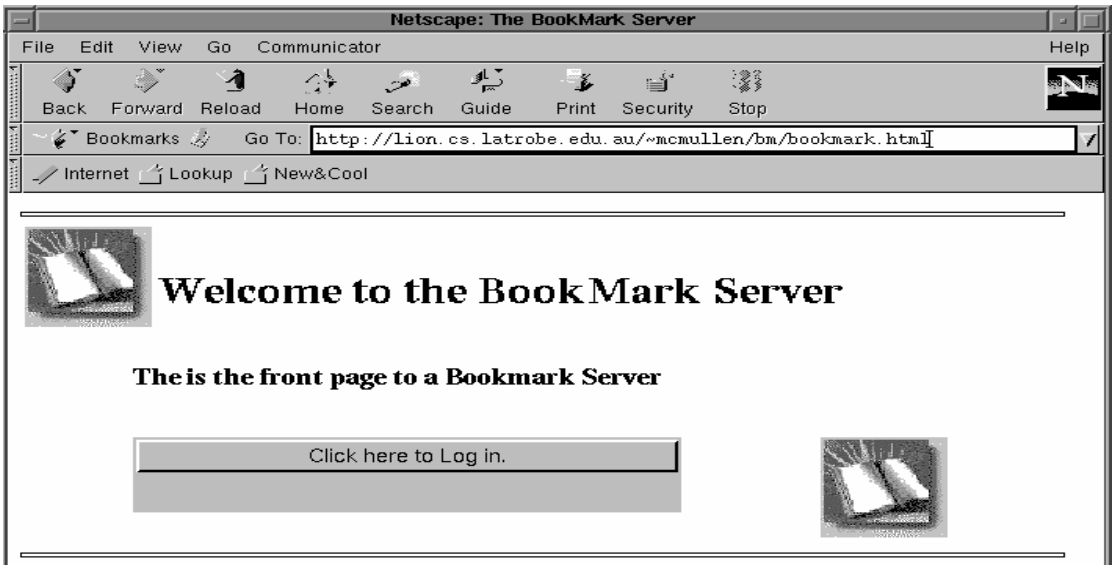


Figure 3. BookMark Objects

The new BookMark structure provides the textual view of the bookmark file in a separate BookMark window that has its own menu options. The Web browser is only used to access the front page of the new BookMark server. The interaction with the server takes place through this separate BookMark window leaving the browser to be used for its intended purpose of browsing Web pages. There is some interaction



between the Web browser and the BookMark window. A message is passed from the browser to the window indicating the current URL and document title in case the Web page is stored in the bookmark file.

Figure 4. Front page to the BookMark Server

As the textual view of the bookmark file will be a hierarchical folder view, it was decided that a tree data structure would be best suited for the storage of bookmarked objects. However, since the bookmark file is stored as HTML code at the server to allow compatibility with all Web browsers, a parser is needed to convert the bookmark file from HTML to the required tree data structure. Once the bookmark file is parsed into the tree data structure, the nodes of the tree, apart from the root node, will represent bookmark objects. The nodes (at present) will be either a folder or a document, as shown in Figure 3. A similar structure has been implemented in [12].

When the user points his Web browser to the front page of the new bookmark system, he is greeted by the Web (Figure 4). Pressing the button “Click here to Log in” causes another window to be displayed in the main client window. This window interacts with the server and Web browsers as well as displaying the bookmark file.

When a user logs in he must go through an authentication process. If successful, he is logged in and the connection remains open for the duration of his interaction with the server. Authentication takes place upon logging in. When logged in the user may request that his bookmark file be sent to him. Once the file is sent, the user will be presented with his bookmark file as in Figure 5.

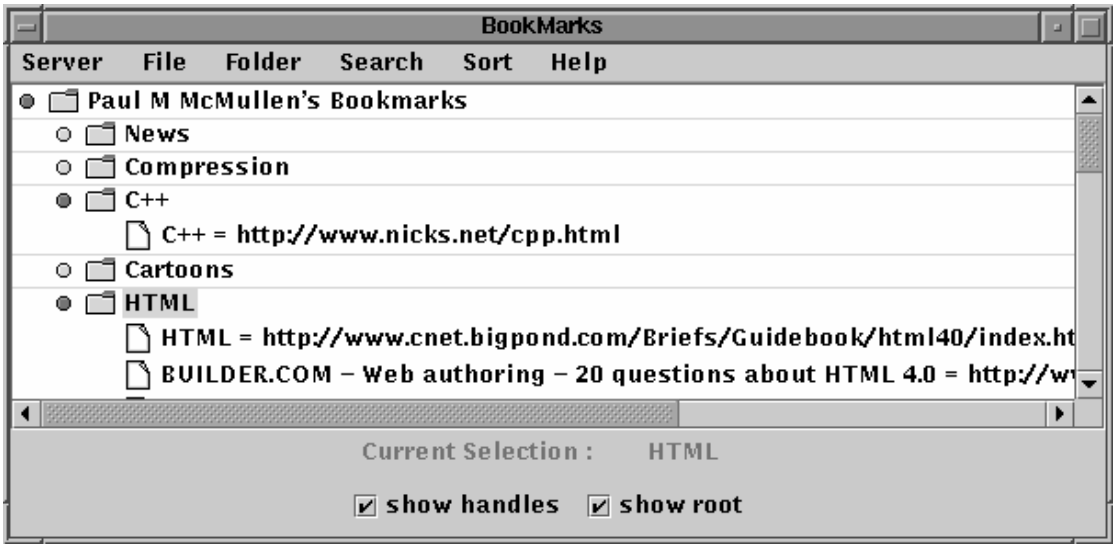


Figure 5. Textual View of a Bookmark File

4. Implementation

The prototype bookmarking system has been implemented in Java. Java was decided upon as it provides the use of socket communication to develop a client-server system, and enables development of a GUI that can interact with Web browsers. Numerous classes have been developed for the prototype implementation. These classes use many in-built classes, termed Java Foundation Classes (JFC) within Java.

The JFCs used range from String manipulation classes to the classes that help create a socket connection.

The implementation of the prototype BookMark system has been implemented in a number of stages that follow the design described in Section 3. These stages are also titled BookMark structure, the Server and the Client.

4.1 Bookmark Structure

A requirement of storing the bookmark file in HTML is to convert the bookmark file from HTML to the required data structure. Several classes are used to implement this, including: the data storage classes BookMark and Folder and the classes PageParser, ExtractDetails and SaveBookMarks. Table 3 describes these classes.

Table 3. BookMark Structure Class Interfaces

Class Name	Constructor	Methods of note
BookMark	BookMark()	
Folder	Folder()	
PageParser	PageParser(Reader in) PageParser(Reader in, boolean prefixes)	param(String name) next()
ExtractDetails	ExtractDetails(BufferedReader in)	getRoot()
SaveBookMarks	SaveBookMarks(Vector r) SaveBookMark(Vector r, boolean bl)	

4.1.1 Data Storage Classes

Two data storage classes are used to convert the bookmark file into the required data structure, the class BookMark and the class Folder. The class BookMark is a data storage class that holds information pertaining to each bookmark entry within a bookmark file. This information includes URL, document type, document title, keywords, summary, user annotations, date added, date last visited and date last modified. The class Folder holds information about a Folder within the bookmark data structure. This information includes the folder name, a description of the folder and the date the folder was added to the bookmark file.

4.1.2 PageParser Class

HTML documents consist of HTML tags with encapsulated parameters and parameter values. The PageParser Class is responsible for finding the HTML tags within the bookmark file and extracting these parameter values. These parameter values contain relevant information pertaining to each bookmarked entry.

4.1.3 ExtractDetails Class

The ExractDetail class is in charge of extracting the HTML parameter values, in conjunction with the class PageParser and storing them into the data structure. At present this data structure is a vector of vectors, where each vector represents a folder within the bookmark file. This data structure is later converted into the tree data structure (Figure 3) for displaying. The ExtractDetails class also allows the importing of bookmark files from the Netscape Web browser. For example, if the following HTML code fragment was in a bookmark file:

```
<DL><p>
<DT><H3 FOLDED ADD_DATE="884244386">Linux</H3>
<DD> Linux related pages
<DL><p>
<DT><A HREF="http://www.serv.net/~cameron/ezppp/index.html"
ADD_DATE="884244410" LAST_VISIT="884244579"
LAST_MODIFIED="884244160" SUMMARY = “null”
KEYWORDS="linux,ppp"  ANNOTATION="null">The EzPPP Project</A>
```

Then after parsing, the following parameter values would be extracted:

folder	linux	linux related pages	884244386

document	http	The EzPPP Project	www.serv.net/~cameron/ezppp/index.html

linux,ppp	null	null	884244410	884244579	884244160

4.1.4 SaveBookMarks Class

This class converts the bookmark data structure into its HTML form, enabling it to be sent back to the server and saved. This class accepts a data structure of type vector. The vector is then enumerated and looped through extracting folder and bookmark information. The output of this class is HTML code to an output stream.

4.2 Server

The Java File system (JFS), a network file system protocol for Java applets, has been developed as a client-server system. The JFS server is a Java program run on the same host as a Web server. JFS clients are Java applets that connect back to the server on the host from which they were loaded. The new BookMark structure server was modelled in part on this server design because it provides a simple solution to the problem of file sharing and message passing between client and server.

A number of classes are involved in the implementation of the prototype server used in the development of the new BookMark system. These classes relate to configuration information, the server and the threads it creates, file implementation, file access, user profiles, request generation, request evaluation and logging facilities.

4.3 Client

There are many classes involved with the implementation of the client, including request generation, the sending and receiving of requests and the construction of the Graphical User Interface (GUI). GUI has been developed with 'Swing' Java classes. Swing was chosen over the Abstract Window Toolkit (AWT) because it offers better data display features and capabilities. These features are able to store data in a tree-like data structure and then display this information in a hierarchical folder view.

5. Evaluation & Discussion

5.1 Evaluation

The new BookMark structure design and prototype addresses the main deficiencies cited in the Overview Section (Sec. 1.1). The system provides the ability to store bookmark files as fully structured HTML documents, allowing any Web browser to view the bookmark files. The new BookMark system also provides a method of storing more relevant information per bookmark entry; termed 'information hiding'. With the development of a centralized bookmark repository, users are able to share bookmark files stored on the server and access these files from any Java enabled-Web browser.

The new BookMark system ensures compatibility of bookmark files between all Web browsers since the new bookmarks are viewed as a Web document. Using a web browser, extra information hidden within the new tags is not displayed. Only information available in current bookmark systems such as folder names, URLs and the name of the document; is displayed. Information within these new tags is displayed within the client application window developed for the new BookMark system server.

Storing the bookmark files as HTML documents necessitated the use of information hiding to store relevant information per bookmark entry. Information hiding involved adding new features into the HTML code, specifically the <A HREF=...>…</A> tag. The new information stored includes document type, keywords, summary, annotations, date added, date of last access, and date of last visit.

The centralized server suited the prerequisite of file sharing. However, a distributed system involving the updating of shared bookmark files was considered. For a group bookmark file if a change is made by a user, then this change is sent to

the server, which in turn broadcasts the change to all the users currently using the bookmark file, as well as updating the servers copy.

We developed the centralized server for the BookMark System to be stateless and each http connection is kept open during each login session. Storing of login states (cookies etc.) is avoided in this design. The system's operating environment is characterized by short login user sessions - keeping connections open allow the fastest access without overheads in processing cookies. This allows maximum scalability for the developed system.

The approach of developing the client's view of the bookmark file as a separate application to the Web browser was implemented so that the user could always have a view of the file on screen in the background. Thus the Web browser can be used for its intended purpose of viewing Web documents, while the bookmark file can be manipulated in its own window. This approach also allows the bookmark system to function as a stand-alone application if needed.

Finding the most useful information in a folder with a large number of bookmarks often involves scanning a long list of URLs. Ordering URLs based on previous experience can aid in rapid location of the required information. Typically a small subset of bookmarks will be referenced frequently. These bookmarks should be prominent and visible within the organisation structure. With current Web browsers, the presentation order of bookmarks is determined by initial user placement. Users must manually rearrange the URL and folder ordering to keep useful bookmarks near the top of the list or folder.

Determining which bookmarks are most useful is made more difficult in a workgroup situation. Yet bookmark presentation order is even more important for structuring workgroup collections than for individual collections [7]. Workgroup

collections will tend to grow rapidly as a function of workgroup size, and users can become overwhelmed with information without some organisational methods such as the ability to order, search, and sort.

The requirements for organizing a bookmark repository are quite different than those required for "conventional" document collections, such as a digital library [10]. A bookmark collection will tend to be smaller in size, highly evolutionary (i.e. the bookmark collection changes both frequently and dynamically), require a tighter user-interaction mechanism to interface and control the organisation utility to suit the user preferences. In addition, the HTML documents pointed to by the bookmark are typically small in size, which allows the scanning and analyzing of documents more frequently with small performance penalties [10].

Given the requirements for organising bookmarks, most fully automatic document clustering tools are not suitable for bookmark organisation [10]. Thus a hybrid approach would seem to be needed where manual and automatic organisation is used equally.

At present the bookmark file is sent to the client as an HTML file and then parsed by the client into the tree structure. Sending the file, as a tree object would reduce the amount of computation needed to be performed by the client, thus making it more lightweight.

The prototype implemented was made available to a project team and a group of forum users to evaluate its usage patterns and performance. We found that the frequency of individual bookmark access versus group bookmark access is 2:1 for the project team while for the forum users it is 3:1. The difference is likely due to the tighter work relation and collaboration among the project team users compared to looser collaboration among the forum users. The average message size from the client

to the server is usually less than 80 bytes, while the average message size from the server to the client is more varied according to the stage the user group is involved: it tends to become larger when the collaboration (or discussion) enters the later stages. The average message size from the client to the server is usually short because such messages are typically requests. The delay (or response time) from using such a system is negligible due to the fact that the local area network bandwidth is sufficient.

5.2 Discussion

This section refers to ideas that may be inferred from the current design of the new bookmark structure.

The ability to distinguish the document type, such as whether it is plain text, HTML, or SGML, etc, is considered a form of completeness to the new bookmark structure. It could also be used as a search option, such as performing a search for all plain text documents. The ability to add keywords to a document, whether manually or generated by a parser would aid in the use of automatic document classification and the ability to perform keyword searches upon bookmarked URLs.

A summary of a bookmarked document, whether manually entered by the user or generated by a parser, would aid in the use of identifying what a particular bookmarked document contents were related to. This could remove the need of actually having to fetch the document from its source to refresh the user's memory as to what the document was about, saving time, and resources.

Annotation is another useful structure that could be incorporated into the bookmark system. An annotation to a Web document can be considered a remark added to a document by the reader. Uses of annotations include: responses to a document or another annotation in the sense of a conversation (e.g. a group can share their common area of interest), the ability to have newsgroup-like forum associated

with specific items on the W3 [11], additions to an ordered list of items, evaluations of the worth or appropriateness of a document for some purpose [9], seals of approval (SOAPs - i.e. a rating system), or filters in support of enabling people to make sense of whatever information is presented to them, as well as usage indicators [11].

There are a number of ways annotations could be added to Web pages. A URL could go through a CGI program that fetches the real document and returns it with the annotations appended. Furthermore, a URL could reference the real document containing a directive that causes the server to append the annotations or the annotations could be added to the real document via Server-side includes (SSIs) [8]. Annotations can also be represented by in-lined images, where annotation icons themselves could become "hotlinks" [11].

Another purpose of annotations is the ability to add additional user or group information to a bookmarked document. Annotations can aid the user or group on commenting about a Web document. The annotation field contains a reference to a text file located at the server. This file once created will only be able to be added to, so those users in a group can view all annotations to the bookmarked document.

The fields concerning date added, date last visited and date last modified are associated with the housekeeping of the new bookmark configuration. The type of housekeeping these fields can help with are removal of bookmarked entries after a specified expiration date. These fields can also be used for searching or sorting.

The connection between the server and each client remains open throughout the transactions in our design. This design decision has been made to reduce the overhead of connection setup. Should the connection be broken, the client will need to re-connect and relogin. This constraint may annoy mobile users if their mobile equipments do not have Mobile IP supporting. A solution is to have an option for each

client when it connects to the server in the first place – to specify whether it would like to have a persistent connection or one-time connection.

6. Conclusion & Future Work

An evaluation of a new bookmark system for the next generation Web has been carried out. From this evaluation a prototype has been developed. This prototype, through the use of information hiding has implemented a new bookmark structure that stores its information in HTML. This supports backward compatibility so that any Web browser will be able to display the bookmark file because it is stored as HTML code. However, they will not be able to see the extra information hidden within the HTML tags. Only the developed prototype system can view this extra information which adds more meaning to bookmarked objects, specifically Web documents.

Future work is intended to exploit the addition of the new tag parameters into the HTML code of the bookmark file. This future work includes the ability to bookmark other Web objects such as email, news items, and personal cards.

Acknowledgement

The authors are grateful to the valuable comments from the editor-in-chief and referees.

Bibliography

[1] Bederson, B. B., Hollan, J. D., Stewart, J., Rogers, D., Druin, A., and Vick, D. A zooming web browser. Proceedings of the SPIE - The International Society for Optical Engineering Vol. 2667 (1996) p. 260-71.

[2] Bush, V. As we may think. The Atlantic Monthly. (July 1945).

[3] Davies N. J., Weeks, R., and Revett, M. C. Information agents for the World Wide Web. BT Technology Journal Vol. 14, Iss. 4 (Oct 1996) p. 105-14.

[4] Doemel, P. WebMap - A graphical hypertext navigational tool. 2nd International Conference on the World-Wide Web, Chicago, IL. (1993) p. 785-789

[5] Maarek Y.S. , Ben Shaul, I.Z. Automatically organizing bookmarks per Content. Proceedings Fifth International World Wide Web Conference, Paris, (May 1996) http://www5conf.inria.fr/fich_html/papers/P37/Overview.html

[6] Kamiya, K., Roscheisen, M. and Winograd, T. Grassroots: a system providing a uniform framework for communicating, structuring, sharing information, and organizing people. Computer Networks and ISDN Systems Vol. 28 Iss. 7-11 (May 1996) p. 1157-74.

[7] Tilley S.R. , Lamia, W.M. Personalized information structures II: hyperstructure hotlists. Emerging from Chaos: Solutions for the growing complexity of our jobs: 13th Annual International Conference on Systems Documentation. (1995). p. 171-180.

[8] Kruse, M. Using Server Side Includes : A simple but powerful technique. Dr. Dobb's Journal (February 1996) p. 52-56.

[9] LaLiberte, D., Braverman, A. A protocol for scalable group and public annotations. Computer Networks and ISDN Systems 27 (1995) p. 911-918.

[10] Maarek, Y. S., Ben Shaul, I. Z. Automatically Organizing Bookmarks per Contents. Computer Networks and ISDN Systems 27 (1995) p.1321-1333. http://www.w3.org/Conferences/WWW5/fich_html/paper-sessions.html

[11] Röscheisen, M., Mogensen, C. and Winogard, T. Beyond browsing : shared comments, SOAPs, trails, and on-line communities. Computer Networks and ISDN Systems 27 (1995) p.739-749.

[12] Yamada, S., Nagino, N. Constructing a personal Web map with anytime-control of Web robots", CoopIS'99 Proceedings. IFCIS International Conference on Cooperative Information Systems (1999) p. 140-147.

[13] Utting, K. and Yankelovich, N. Context and Orientation in Hypermedia Networks. ACM Transactions on Information Systems (January 1989).

[14] Li, Wen-Syan, Vu, Quoc, Agrawal, D., Hara, Y., and Takano, H., "PowerBookmarks: A System for Personalizable Web Information Organization, Sharing and Management", Computer Networks, 31, May 1999.

[15] http://www.twiki.org

[16] http://zdp.zope.org/