1.0 Introduction

    The WorldWideWeb (W3) is the universe of network-accessible
    information, an embodiment of human knowledge. It is an
    initiative started at CERN, now with many participants. It
    has a body of software, and a set of protocols and
    conventions. W3 uses hypertext and multimedia techniques to
    make the web easy for anyone to roam, browse, and contribute
    to. [1]

This paper overviews the World-Wide Web (frequently abbreviated
as "W3," "WWW," or the "Web") and related systems and standards.
[2] First, it introduces Web concepts and tools and describes
how they fit together to form a coherent whole, including the
client/server model of computing, the Uniform Resource Locator
(URL), selected Web client and server programs, the HyperText
Transfer Protocol (HTTP), the HyperText Markup Language (HTML),
selected HTML converters and editors, and Common Gateway
Interface (CGI) scripts. Second, it discusses strategies for
organizing Web information. Finally, it advocates the direct
involvement of librarians in the development of Web information
resources.

2.0 Background

In 1989, Tim Berners-Lee of CERN (a particle physics laboratory
in Geneva, Switzerland) began work on the World-Wide Web. The
Web was initially intended as a way to share information between
members of the high-energy physics community. [3] By 1991, the
Web had become operational.
    The Web is a hypertext system. The hypertext concept was
originally described by Vannevar Bush, [4] and the term
"hypertext" was coined by Theodor H. Nelson. [5] In a hypertext
system, a document is presented to a reader that has "links" to
other documents that relate to the original document and provide
further information about it.

    Scholarly journal articles represent an excellent
application of this technology. For example, scholarly articles
usually include multiple footnotes. With an article in hypertext
form, the reader could select a footnote number in the body of
the article and be "transported" to the appropriate citation in
the notes section. The citation, in turn, could be linked to the
cited article, and the process could go on indefinitely. The
reader could also backtrack and follow links back to where he or
she started.
    The HyperText Transfer Protocol (HTTP) that allows Web

servers and clients to communicate is older than the Gopher
protocol. The original CERN Web server ran under the NeXTStep
operating system, and, since few people owned NeXT computers,
HTTP did not become very popular. Similarly, the client side of
the HTTP equation included a terminal-based system few people
thought was aesthetically appealing. [6] All this was happening
just as the Gopher protocol was becoming more popular. Since
Gopher server and client software was available for many
different computing platforms, the Gopher protocol's popularity
grew while HTTP's languished.

     It wasn't until early 1993 that the Web really started to
become popular. At that time, Bob McCool and Marc Andreessen,
who worked for the National Center for Supercomputing
Applications (NCSA), wrote both Web client and server
applications. Since the server application (httpd) was available
for many flavors of UNIX, not just NeXTStep, the server could be
easily used by many sites. Since the client application (NCSA
Mosaic for the X Window System) supported graphics, WAIS, Gopher,
and FTP access, it was head and shoulders above the original CERN
client in terms of aesthetic appeal as well as functionality.
Later, a more functional terminal-based client (Lynx) was
developed by Lou Montulli, who was then at the University of
Kansas. Lynx made the Web accessible to the lowest common
denominator devices, VT100-based terminals. When NCSA later
released Macintosh and Microsoft Windows versions of Mosaic, the
Web became even more popular. Since then, other Web client and
server applications have been developed, but the real momentum
was created by the developers at NCSA. [7]

3.0 The Client/Server Model

To truly understand how much of the Internet operates, including
the Web, it is important to understand the concept of
client/server computing. The client/server model is a form of
distributed computing where one program (the client) communicates
with another program (the server) for the purpose of exchanging
information. [8]

+ Page 7 +

     The client's responsibility is usually to:

     o   Handle the user interface.

     o   Translate the user's request into the desired protocol.

     o   Send the request to the server.

     o   Wait for the server's response.

     o   Translate the response into "human-readable" results.

     o   Present the results to the user.

The server's functions include:

     o   Listen for a client's query.

     o   Process that query.

     o   Return the results back to the client.

A typical client/server interaction goes like this:

     1.   The user runs client software to create a query.

     2.   The client connects to the server.

     3.   The client sends the query to the server.

     4.   The server analyzes the query.

     5.   The server computes the results of the query.

     6.   The server sends the results to the client.

     7.   The client presents the results to the user.

     8.   Repeat as necessary.

This client/server interaction is a lot like going to a French
restaurant. At the restaurant, you (the user) are presented with
a menu of choices by the waiter (the client). After making your
selections, the waiter takes note of your choices, translates
them into French, and presents them to the French chef (the
server) in the kitchen. After the chef prepares your meal, the
waiter returns with your diner (the results). Hopefully, the
waiter returns with the items you selected, but not always;
sometimes things get "lost in the translation."

+ Page 8 +

     Flexible user interface development is the most obvious
advantage of client/server computing. It is possible to create
an interface that is independent of the server hosting the data.
Therefore, the user interface of a client/server application can
be written on a Macintosh and the server can be written on a
mainframe. Clients could be also written for DOS- or UNIX-based
computers. This allows information to be stored in a central
server and disseminated to different types of remote computers.
     Since the user interface is the responsibility of the
client, the server has more computing resources to spend on
analyzing queries and disseminating information. This is another
major advantage of client/server computing; it tends to use the
strengths of divergent computing platforms to create more
powerful applications. Although its computing and storage
capabilities are dwarfed by those of the mainframe, there is no
reason why a Macintosh could not be used as a server for less
demanding applications.
     In short, client/server computing provides a mechanism for
disparate computers to cooperate on a single computing task.

4.0 Uniform Resource Locator

The Uniform Resource Locator (URL) is a fundamental part of the
Web. It is utilized to concisely describe and identify both the
protocol used by and the location of Internet resources. [9]
     In general, a URL has the following form:
protocol://host/path/file. "Protocol" denotes the type of
Internet resource. The most common are: "gopher," "wais," "ftp,"
"telnet," "http," "file," and "mailto" (electronic mail). "Host"
denotes the name or IP (Internet Protocol) address of the remote

computer (e.g., 152.1.39.42 or www.lib.ncsu.edu). "Path" is a
directory or subdirectory on a remote computer. "File" is the
name of the file you want to access.
     Using variations of this general form, you can use URLs and
Web browsers to access just about any Internet resource. Here is
an example of a URL for an FTP session:

     ftp://ftp.lib.ncsu.edu/pub/stacks/alawon/alawon-v1n04

This URL results in the following actions: 1. FTP to
ftp.lib.ncsu.edu, 2. log on as anonymous, 3. change the directory
to /pub/stacks/alawon/, and 4. get the file alawon-v1n04.
     Since Web browsers understand and implement the File
Transfer Protocol (FTP), you do not have to remember all the
commands necessary to do FTP. All you have to remember is how to
create a URL for an FTP session.

+ Page 9 +

     Here is an example of a URL for an HTML document:

     http://www.lib.ncsu.edu/stacks/alawon-index.html

This URL opens up a HTTP connection to www.lib.ncsu.edu, changes
the directory to stacks, and retrieves the file
alawon-index.html. URLs are more complicated than the general
form illustrated above; URLs can also provide the means to
present the logon name for Telnet connections, a communications
port, an index/search query, and/or an HTML anchor. Here is an
example of a URL for a Telnet session:

     telnet://library@library.ncsu.edu:23/

In this example, "library" denotes the logon name and "23"
denotes the communications port. (Port 23 is the standard Telnet
communications port.) Thus, a Web browser can initiate a Telnet
session. This example opens up a Telnet connection to
"library.ncsu.edu," and, depending on the user's browser, the
user may be reminded to log on as "library." This URL does not
use the "path" or "file" parameters because they are meaningless
for Telnet sessions.
     On the other hand, to manually query the Geographic Name
Server, the URL would be:

     telnet://martini.eecs.umich.edu:3000/

Since the Geographic Name Server requires no password, no
password is specified; however, since the Geographic Name Server
"listens" on port 3000, a nonstandard port number must be
specified.
     WAIS searches can be specified using URLs. Unfortunately,
at the present time, only NCSA Mosaic for the X Window System
directly implements the WAIS protocol. WAIS URLs have the
following form:

     wais://host:port/database?query

"Port" is assumed to be 210 (the standard WAIS/Z39.50 port),
"database" is the source file to search, "?" delimits the
database from the query, and "query" is the your search strategy.
Here is an example of a URL for a WAIS search:

```
        wais://vega.lib.ncsu.edu/alawon.src?nren
```

Gopher servers and files can be specified with URLs as well.
Since Gopher resource specifications require "Type" identifiers
and paths to Gopher resources often include spaces, Gopher URLs
usually deviate from the norm. Here is an example of a URL for a
Gopher subdirectory:

```
        gopher://gopher.lib.ncsu.edu/11/library/
```

Notice the pair of 1's after the Internet name of the computer.
These 1's specify the resource as a directory. On the other
hand, the following URL specifies a specific text file within
that directory:

```
        gopher://gopher.lib.ncsu.edu/00/library/about
```

The "00" denotes a text file.
     Constructing URLs is more difficult when the path and/or
file names of the Internet resources contain special characters
like spaces or colons. In these cases, escape codes must be used
to denote the special characters. For example:

```
        gopher://gopher.lib.ncsu.edu/0ftp%3amrcnext.cso.uiuc.edu%40/
        pub/etext/etext91/aesop11.txt
```

This long URL first asks a Gopher server (gopher.lib.ncsu.edu) to
FTP a file (aesop11.txt) from an anonymous FTP server
(mrcnext.cso.uiuc.edu). Notice the "%3a" and "%40" in the URL.
They are used to denote a colon (":") and at sign ("@"),
respectfully. Furthermore, notice the zero proceeding the "ftp."
This is used to identify the remote file as a text file.
     As you can see, Gopher URLs are particularly difficult to
decipher. The easiest way to construct a URL for a Gopher item
is to access the Gopher server via a Web client, traverse the
Gopher menus until you locate the resource, and then copy the
displayed URL from the appropriate part of your client's screen.
     In summary, URLs unambiguously describe the location of
Internet resources. Using URLs as a standard, Internet client
programs like Web browsers can interpret URLs and retrieve the
desired information. URLs describe the protocols and locations
of Internet resources without regard to the particular Internet
client software the user is employing to access them.

5.0 Example Web Client Software

Four examples of Web client software are described here: MacWeb,
NCSA Mosaic for Microsoft Windows, Lynx, and NCSA Mosaic for the
X Window System. These particular pieces of software are
described because I think they presently represent the best
clients for the most common computing environments (i.e.,
Macintosh, Microsoft Windows, character-terminal-based VMS or
UNIX, and X Window System).
     The real power of these Web clients (usually referred to as
"browsers") is their ability to understand multiple Internet
protocols. Each of the browsers described understands how to FTP

files, act as Gopher clients, and read and interpret the output
of Web servers. Additionally, each of these pieces of software
understand "forms," an HTML extension allowing the user to
complete electronic forms similar to Gopher+ ASK blocks. While
none of these clients can directly understand the Telnet
protocol, each can be configured to load and run Telnet software.

5.1 MacWeb

As the name implies, MacWeb is a Web browser for the Macintosh.
Written at the Microelectronics and Computer Technology
Corporation (MCC), MacWeb is distributed via the Enterprise
Integration Network (EINet). [10] MacWeb requires System 7 and
at least MacTCP version 2.0.2. MacTCP is an operating system
extension available from Apple Computer that allows Macintosh
computers to understand the Transmission Control
Protocol/Internet Protocol (TCP/IP) necessary for Internet
communications. A very important piece of software called
"StuffIt Expander," is strongly recommended when using MacWeb or
NCSA Mosaic for the Macintosh (MacMosaic). [11] StuffIt Expander
is a utility program used to translate and uncompress files;
compressed files are usually retrieved via FTP archives.
     The advantages of MacWeb are that it is fast, has an elegant
and easily customizable interface, supports the automatic
creation of HTML documents from its hotlists, and indirectly
supports the WAIS protocol by launching MCC's WAIS client,
MacWAIS.
     Its disadvantages are that you cannot select and copy text
directly from the screen and, when the displayed text is saved as
a text file, the displayed text looses all of its formatting.

+ Page 12 +

5.2 NCSA Mosaic for Microsoft Windows

NCSA Mosaic for Microsoft Windows is bound to be one of the more
popular Web browsers since most people have or will have
Microsoft Windows-based computers. [12] NCSA Mosaic for
Microsoft Windows requires a WINSOCK.DLL. Like MacTCP, the
WINSOCK.DLL software allows your computer to understand TCP/IP.
Common WinSock packages include LAN WorkPlace for DOS and Trumpet
WinSock. Additionally, NCSA Mosaic for Microsoft Windows
requires the 32-bit Windows extensions (Win32s). Win32s runs on
80386, 80486, or Pentium computers. The Win32s software is
available via anonymous FTP from NCSA.
     One of the nicest features of NCSA Mosaic for Microsoft
Windows is the ability to customize its menu bar. By editing the
MOSAIC.INI file, you can delete or add menu items to the menu
bar. Consequently, you can configure the client and have it
display commonly used Internet resources.
     At the present time, you cannot select nor copy text from
the screen. Therefore, if you want to save displayed text, you
must use the application's "Load to Disk" option.

5.3 Lynx

Lynx is a basic Web browser that is intended to be used on DOS
computers or "dumb" terminals running under the UNIX or VMS
operating systems. [13]
     Lynx clients are wonderful when your only Internet
connection is located on a remote computer (i.e., most dial-in

access) or when you need to provide a lowest common denominator
interface (e.g., VT100 terminals).

Lynx clients don't support image or audio data, but they do
support the "mailto" URL. Mailto URLs are used for the Simple
Mail Transfer Protocol (SMTP), the Internet mail standard. When
a Lynx client user selects a mailto URL, the user will be
presented with a "form" to complete and the resulting text from
the form will be delivered via Internet mail to the person or
computer specified in the URL.

5.4 NCSA Mosaic for the X Window System

NCSA Mosaic for the X Window System, coupled with NCSA's Web
server (httpd), really gave the Web the momentum and visibility
it has today. [14] This full-featured browser supports copy and
paste from the display. Direct WAIS support is also provided,
and URLs such as wais://wais.lib.ncsu.edu/alawon?nren are valid.
At the present time, just about the only thing it doesn't support
is the mailto URL.

The disadvantage of NCSA Mosaic for the X Window System is
that it requires a relatively powerful computer. While a
Macintosh equipped with MacX or a Microsoft Windows computer with
HummingBird Communications' eXceed/W can run X Window terminal
sessions, NCSA Mosaic for the X Window System really requires
direct access to a UNIX or VMS machine running the X Window
System software.

6.0 Example Web Server Software

If you want to become a Web information provider, you need to
utilize Web server software. This section describes the most
popular Web server software for the most common computing
platforms (i.e., Macintosh, UNIX, VMS, and Microsoft Windows).

6.1 MacHTTP

MacHTTP is an Web server for Macintosh computers. [15] Written
by Chuck Shotton, MacHTTP is one of the easiest servers to set up
and configure. In fact, it is so easy it works "straight out of
the box." MacHTTP requires System 7 to support advanced features
like AppleScript. MacHTTP runs on Macintosh II-type computers
(e.g., Macintosh IIci, SE/30, LC, Centris, and Quadra computers).
It does not run on low-end Macintoshes based on the Motorola
68000 microprocessor (e.g., Macintosh Plus, SE, and PowerBook 100
computers). MacHTTP requires MacTCP.

Because of its simple installation, I recommend the use of
MacHTTP to learn the basics of Web servers. Since it is so
small, just about anyone can create a server on their desktop
computer and effectively experiment with serving HTML documents.
A Macintosh is not recommended as an institution's primary
server, since the potential user population may be very large.
On the other hand, a group of Macintosh servers that were linked
together via the HTTP protocol to form a single virtual server
could easily distribute the load, with each server supporting a
subset of an institution's HTML documents.

6.2 NCSA httpd

Based on the number of postings to comp.infosystems.www
newsgroups, NCSA's httpd seems to be the most popular Web server.
Running under the UNIX operating system, httpd is distributed
both as source code and in binary form for the many "flavors" of
UNIX. [16] This server is robust and only slightly difficult to
configure.
     If you have a UNIX computer at your disposal and your
server's intended audience is large, then I recommend the use of
NCSA httpd. I recommend this for several reasons. First, this
server is widely supported by the Internet community; you can
always find an expert, and it is easier to get help for this
server than for the CERN server. Second, since it runs under
UNIX, it is intended to coincide with other applications running
on the same computer, like Gopher, WAIS, or a list server.
Finally, many Common Gateway Interface (CGI) scripts are written
in Perl, a programming language most at home on a UNIX computer.
(CGI scripts are described in more detail later.)

6.3 CERN httpd

If you have a VMS computer, you cannot use the NCSA http server;
however, there is an appropriate Web server available. It is a
port of the CERN httpd server by Foteos Macrides of the Worcester
Foundation for Experimental Biology. Like the servers described
previously, the CERN httpd server for VMS comes in binary form as
well as in source code form. [17] Configuration is not as easy
as MacHTTP or NCSA httpd for Windows, but it is not any more
difficult than NCSA's httpd server for UNIX. Presently, the
server does not support the POST method, the preferred method of
transmitting information from forms to CGI scripts, but it works
just the same. One advantage of VMS is its strong scripting
language, DCL. DCL is works well for CGI scripts.

+ Page 15 +

     If you plan to maintain a server, your intended audience is
large, and you have a VMS computer at your disposal, then I
recommend using this server software. If you have a UNIX
computer, use the NCSA http server instead.

6.4 NCSA httpd for Windows

Robert B. Denny has ported the NCSA httpd server to Microsoft
Windows. [18] Like MacHTTP, it worked for me "right out of the
box," and it supports all the standard features, such as forms,
CGI scripts, graphics, and access control.
     Its disadvantages are that it is considered slow and it
requires a lot of system resources (memory and CPU power) as well
as a WinSock-compatible TCP/IP driver (just like NCSA Mosaic for
Microsoft Windows).
     This server would make a good platform for PC users to learn
the basics of HTTP and server maintenance. Like MacHTTP, I would
not recommend this application as the main server of an
institution, such as an academic library.

7.0 Web Servers Versus Gopher Servers

There are several reasons why Web servers should be used instead
of Gopher servers.

First, in terms of computing resources, Web servers are more efficient since most of the information processing is distributed to the client software. A Gopher client can effectively have access to FTP and WAIS services, but the Gopher server is doing all the work. On the other hand, Web clients (for the most part) understand these protocols and take the load off the server.

Second, because Web clients understand HTML, Web servers are not limited to making their information available via menus. Thus, more descriptive texts and abstracts can be added to hypertext links making it easier for the user to evaluate possible choices.

Third, Web servers are significantly easier to maintain. For example, every "study carrel" of the North Carolina State University Libraries' Web server consists of a single HTML file created either with a public domain editor or via a report from a database program. This is so much easier to maintain and manage than all the link files and directories of the study carrels in the Libraries' Gopher server.

+ Page 16 +

8.0 HyperText Markup Language

The HyperText Markup Language (HTML) is used to format documents delivered by Web servers. The formal HTML standard can be read from the CERN server, [19] and a few style guides are available from the WWW Developer's JumpStation. [20] A subset of the Standard Generalized Markup Language (SGML), HTML's strengths and weaknesses are well documented by Price-Wilkin [21] and Barry. [22] Therefore, only a brief overview of HTML will be provided here.

HTML files are simple ASCII files containing rudimentary "tags" describing the format of a document. Creating an HTML document is a lot like using the old word processing program WordStar. (Remember WordStar?) For example, to print a word in boldface type using WordStar, the user would first select text from the screen. Then the user would enter a code like "^b." This code would be inserted before and after the selected text. When the document was printed, WordStar would interpret the "^b" and print boldface letters until another "^b" was encountered. HTML works in a similar fashion. The author goes through his or her document surrounding text with special codes denoting format. Since the Web employs the client/server model, there is little control over the fonts and styles of formatted text at the client end. Therefore, HTML provides logical rather than stylistic formatting capabilities.

The basic structure of an HTML document looks like this:

```
<HLML>
<HEAD>
<TITLE>My First HTML Document</TITLE>
</HEAD>
<BODY>
Hello, World!
</BODY>
</HTML>
```

The <HTML> and </HTML> tags define the document as an HTML document; the <HEAD> and </HEAD> tags denote the leading matter of a document; the <TITLE> and </TITLE> tags specify the document's title; and the <BODY> and </BODY> tags specify the

location of the formatted text. Notice how the second tag of
each tag pair is identical to the first tag except the second tag
includes a backward slash ("/"); the backward slash denotes the
completion of a logical formatting option.

     Within the body of an HTML document there can be many other
formatting constructs. Examples include the <P> tag for
paragraph marks and the <BR> tag for simple line breaks. There
are also the ordered list (<OL>) and unordered list (<UL>) tags
that allow the user to create lists of numbered items and
unnumbered items, respectively. An ordered list results in
formatting something like this:

     1.   apples
     2.   pears
     3.   bananas

An unordered list results in something like this:

     *   red
     *   white
     *   blue

The real utility of HTML is not its ability to format text.
Rather, its real strength lies in its ability to transport a user
from one section of text to another (or to a completely new
document) by clicking on (or selecting) highlighted words. This
hypertext capability is HTML's greatest asset.
     The hypertext features of HTML are implemented with tags
called "links." Links are tags containing either an anchor, URL,
or both. Section headings are usually used as anchors in HTML
documents. Thus, anchors are used to navigate to another section
of the presently viewed document or, when used in conjunction
with a URL, to navigate to a section of a different document.

9.0 HTML Converters and Editors

Creating HTML documents by hand can be a laborious process; it is
easy to forget all the various tags and formatting rules.
Consequently, there are a growing number of software tools
available to make the HTML document creation process easier.

9.1 Simple HTML Editor (S H E)

Simple HTML Editor (S H E) is an HTML editor in the form of a
HyperCard stack. [23] It requires a Macintosh and HyperCard
version 2.1 (or HyperCard Player). Optional editor features
require MacWeb and the AppleScript extensions.

     The creation of a document is a four-step process. First
you create a new document. Second, you enter text into the
document. Third, to enhance your document, you select text from
the screen and choose a markup option from the menu. Finally,
you save the document. Specific knowledge of HTML is not
necessary, but it helps.
     Unique features of S H E include Balloon Help, forms
creation, and one-step preview if you have MacWeb. Like all HTML

editors (with the possible exception of HoTMetaL), S H E is not a WYSIWYG editor. In other words, the user is presented with raw HTML when editing. Another limitation of S H E is its inability to create documents longer than 30,000 characters.

9.2 HTML Assistant

HTML Assistant is a Windows-based HTML editor. [24] It works like other editors in that you enter text on the screen and make changes to the text's characteristics by selecting the text and choosing a markup option. Like S H E, HTML Assistant is not a WYSIWYG editor, but it to has the ability to test your work with a Web browser at the click of a button.

Other features include:

  o   A user defined toolbox enables you to easily include
       new markup text as more features are added to HTML.
       You can also create your own markup tags for special
       editing tasks.

  o   Facilities for extracting, organizing, and combining
       URLs from different sources.

  o   A multiple-document interface (more than one file may
       be opened at one time) so you can easily cut and paste
       between documents.

  o   Context-sensitive help.

10.0 Converters

Another popular way to create HTML documents is to convert files from a wordprocessor file format (e.g., Microsoft Word, WordPerfect, and RTF) to HTML with the help of "converter" programs. A collection of these program can be seen at the WWW Developer's JumpStation. [25]

     On one hand, converter programs are very convenient. On the other hand, they keep you in the dark about HTML, and, unless you know something about HTML, you are stuck with the tags the converter gives you as output. Although converter programs are useful, you still have to manually enter some hypertext links in order to take full advantage of HTML's capabilities.

11.0 CGI scripts

The real potential of Web servers lies in their ability to run programs behind the scenes and return the results of these programs to the user. This is known as the Common Gateway Interface (CGI).
     Basic CGI scripts include the ability to display the current time or the number of users who have accessed a server. More advanced and useful CGI scripts include features like SFgate (a gateway to WAIS servers) and forms for interlibrary loan requests.
     CGI scripts are made available to a Web browser by either the ISINDEX HTML tag, a specialized URL containing a question mark (?), or forms. After the user completes an HTML document

pointing to a script, the script's query is passed to the Web
server, which passes the input to the designated script. CGI
scripts can be written in almost any language. Common languages
include C, Perl, AppleScript, Visual Basic, or DCL. The scripts
then process the user's input and pass the results (usually in
the form of an HTML document) back to the Web server, which
subsequently sends the results to the Web client.

11.1 Tim Kambitch's CGI Scripts

One of the best CGI scripts I have seen for libraries has been
written by Tim Kambitch of Butler University. Tim has written a
number of scripts allowing the user to search DRA online catalogs
(OPACs). These scripts allow the user to input a Boolean query,
including qualifiers like "au" for author, "ti" for title, and
"su" for subject. These queries are then applied to the OPAC and
the results are returned. Thus, it is not necessary to Telnet to
the OPAC to perform a search; a single program (a Web client) can
be used to access both Internet resources and OPACs. Since the
DRA searching program used by Kambitch's scripts is a Z39.50
client, it is possible to use the client to provide access to
Z39.50 servers. The North Carolina State University Libraries
have used these scripts to provide Web browser access to its OPAC
and its government documents database.

11.2 NCSU Libraries' Mr. Serials Project

Collecting serial literature is another application of Web
servers and CGI scripts. For the past two years, the North
Carolina State University Libraries (NCSU Libraries) have
systematically collected electronic serials with a process called
"Mr. Serials." The result of the Mr. Serials process is the
creation of HTML documents available on the NCSU Libraries' Web
server. While the collection is rather small and it is limited
to library and information science titles, it effectively
demonstrates how libraries can organize, archive, index, and
disseminate electronic serials. It is hoped librarians can use
something like Mr. Serials to convince the academic community of
the feasibility of electronic publishing.
     With the advent of the 856 field, the MARC record will be
able to effectively describe the locations and holdings of
electronic documents. It is anticipated that URLs will be
entered into a public note subfield of the 856 field. As an
experiment, the NCSU Libraries have added two records to our
OPAC. The first describes ALAWON and the other describes The
Public Access Computer Systems Review. We then added 856 fields
to the MARC records and added URLs describing the locations of
these electronic serials. Last, we made these URLs hypertext
links. Consequently, we can use a Web browser like Mosaic to
search the NCSU OPAC for "alawon" or "public access computer
systems review." Once a record is retrieved and displayed, a
hypertext link appears. The user can then choose the hypertext
link and go directly to the electronic serial. (We have done
something similar to an item in our catalog for the University's
recent self study.) This project demonstrates how traditional
cataloging mechanisms can be used to help organize the Internet.

11.3 Possible Expert System Uses

Another, as of yet unrealized, application of CGI scripts is an expert system for locating information on the Internet or in databases. Imagine a scenario where you are asked a number of questions via an HTML form. Based on the answers to these questions, other questions are asked. At the end of this question/answer process, the CGI script generates either a "game plan" for locating the information you seek or it generates queries that can then be applied to various databases across the Internet (e.g., OPACs, Web servers, and Veronica servers).

12.0 Organizing Web Information

The introduction of technologies like the Web can have a profound effect on libraries. Keeping in mind libraries are about information and not about books and other printed materials, how can libraries use Web clients and servers to provide better library service?
     The Web can be used to distribute information about libraries. This information includes such things as hours of operation, reference guides, policies, descriptions of services, lists of subject specialists, and building maps. Like our earliest online catalogs, this particular use of the Web transfers old services to a new technology without truly taking advantage of the new technology's strengths.
     The organization of Internet resources is another use of this new technology. We are all aware of the tremendous, ever growing amount of data and information available on the Internet. Organizing this information into a coherent whole is a daunting task being attempted by many, many people. Who can do this better than librarians who have special training and experience in organizing information? Once a Web server is in place, it is a simple matter of dividing it into sections where each section contains information on a common theme.
     There are no rules restricting the creation of thematic organizational schemes; however, based on my experience with the Gopher at the NCSU Libraries, I can suggest some guidelines.
     First and foremost, the organizational scheme must be comprehensible to your intended audience. Think about the people who will be using the Web server. What are their backgrounds? What do they want? What specialized terminology do they use? In general, how do they think? Incorporate the answers to these questions into the structure of your Web server. "Libraries are for use," and, in order for this to happen, your classification system must be understandable by most of your clientele.
     Second, create a structure striving to be both enumerative and synthetic.

     Enumerative classification attempts to assign designations
     for (to enumerate) all the single and composite subject
     concepts required in the system. . . . Synthetic
     classifications are more likely to confine their explicit
     lists of designations to single, unsubdivided concepts,
     giving the local classifier generalized rules with which to
     construct headings of composite subject. [26]

Third, organize materials based on format as a last resort.
People usually don't care what format the data is in just as long

as the answer to their query can be found.

Last, but not least, be consistent in the way things are classified. In short, practice good cataloging.

After deciding what you are going to collect and how you are going to organize the material, you need to decide how you are going to maintain your data. At first glance, the solution appears to be to use an HTML editor and begin the construction of subject-specific pages. An alternative approach is to take advantage of a database program, and use the database program's report generation capabilities to create HTML files automatically. With this method, each Internet resource corresponds to one record. The record is then divided into fields like title, author, date, URL, abstract, major subject(s), and minor subject(s). Records are added to the database and as many fields are completed as possible, especially the title, URL, and subject fields. Finally, a report is generated by creating a subset of records sharing a common theme (e.g., engineering resources) and then outputting the report in HTML form.

This database method has many advantages over creating HTML files by hand with an HTML editor. First, it reduces human error in the creation of HTML. Second, if a particular resource is to be classified with more than one subject heading, there is only one place where the information needs to be maintained. With the manual creation of HTML documents, there will be more than one file to edit. Third, a report can be generated containing one and only one occurrence of every item in your database. This report can then be indexed using a WAIS server, and it can provide your users with a way to effectively search your Web server. Finally, when the next "killer" Internet protocol becomes available, you will not have to reenter your collection of Internet resources. You will only have to modify your report's output.

13.0 Conclusion

Now is the time for your library to begin maintaining a Web server. Read the USENET newsgroups comp.infosystems.www. providers, comp.infosystems.www.users, and comp.infosystems.www.misc. Start with a 80386-based or Macintosh-based server to get acquainted with the principles of server maintenance and HTML. Identify your target audience and anticipate their needs. Gather information accordingly. If you anticipate a large demand, move your server to a more powerful UNIX- or VMS-based computer with at least one gigabyte of storage, more if you are collecting electronic texts. Keep reading the newsgroups.

The Web and the Internet as a whole are about accessing electronic information resources. Libraries are about collecting, organizing, archiving, disseminating, and sometimes evaluating information resources. Libraries are not just about books and journals; books and journals are only one manifestation of the information universe.

Doesn't it make sense that librarians should be involved in providing Internet resources? Users often complain about the disorganization of the Internet. Librarians have been organizing information resources for centuries. Scholars worry about the long-term preservation of electronic information. Archiving information is a major aspect of librarianship. Some say the Internet has a high "noise to signal" ratio. This is true for

the information universe in general, and librarians have special
skills when it comes to extracting information from data.
      In short, I advocate the creation and maintenance of Web
servers and other Internet resources by librarians. Although
this requires the development of new skills, librarians already
possess the more critical skills necessary to make these Internet
services truly useful, and, while there are some risks involved
in this effort, these risks are well worth taking.

Notes

1. Tim Berners-Lee, World Wide Web Initiative (Geneva: CERN,
1994). (URL: http://info.cern.ch/hypertext/WWW/TheProject.html.)

2. For readers with a Web client, the author has also made this
paper available as an HTML file at the following URL:
http://www.lib.ncsu.edu/staff/morgan/www-and-libraries.html.

3. Kris Herbst, "The Master Weaver," Internet World 5 (October
1994): 78.

+ Page 24 +

4. Vannevar Bush, "As We May Think," Atlantic Monthly 176 (July
1945): 101-108.

5. Theodor H. Nelson, "As We Will Think," in From Memex to
Hypertext: Vannevar Bush and the Mind's Machine, ed. James M.
Nyce and Paul Kahn (Boston: Academic Press, 1991), 245-260.

6. Richard W. Wiggins, "Examining Mosaic: A History and Review,"
Internet World 5 (October 1994): 48-51.

7. Ibid.

8. Eric Lease Morgan, WAIS and Gopher Servers: A Guide for
Internet End-Users (Westport, CT: Mecklermedia, 1994), 1-2.

9. See http://info.cern.ch/hypertext/WWW/Addressing/
Addressing.html.

10. See ftp://ftp.einet.net/einet/mac/macweb/
macweb.latest.sea.hqx or http://galaxy.einet.net/
EINet/MacWeb/MacWebHome.html.

11. MacMosaic is a Macintosh Web browser from NCSA. Read more
about MacMosaic at http://www.ncsa.uiuc.edu/SDG/Software/
MacMosaic/MacMosaicHome.html or ftp://ftp.ncsa.uiuc.edu/
Mosaic/Mac/.

12. See http://www.ncsa.uiuc.edu/SDG/Software/WinMosaic/
HomePage.html or ftp://ftp.ncsa.uiuc.edu/PC/.

13. The DOS version (DOSLynx) can be found at ftp://
ftp2.cc.ukans.edu/pub/WWW/DosLynx/. Similarly, the UNIX and VMS
versions can be found at ftp://ftp2.cc.ukans.edu/pub/WWW/lynx/.
When obtaining the UNIX or VMS version of Lynx, be sure to copy
the version matching your specific hardware and TCP/IP
configuration. If you don't know your hardware and TCP/IP
configuration, then ask for the specification from your systems
administrator.

14.   See http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/
Docs/help-about.html or ftp://ftp.ncsa.uiuc.edu/Mosaic/Unix/.

15.   See http://www.uth.tmc.edu/mac_info/machttp_info.html.

16.   See http://hoohoo.ncsa.uiuc.edu/docs/Overview.html.

+ Page 25 +

17.   See http://sci.wfeb.edu/dir/216vms.

18.   See ftp://ftp.netcom.com/pub/rdenny/ or
ftp://ftp.ncsa.uiuc.edu/Web/httpd/Unix/ncsa_httpd/contrib/
winhttpd/.

19.   See http://info.cern.ch/hypertext/WWW/MarkUp/MarkUp.html.

20. See http://oneworld.wa.com/htmldev/devpage/dev-page1.html.

21.   John Price-Wilkin, "Using the World-Wide Web to Deliver
Complex Electronic Documents: Implications for Libraries," The
Public-Access Computer Systems Review 5, no. 3 (1994): 5-21.   (To
retrieve this article, send the following e-mail message to
listserv@uhupvm1.uh.edu: GET PRICEWIL PRV5N3 F=MAIL.)

22.   Jeff Barry, "The HyperText Markup Language (HTML) and the
World-Wide Web: Raising ASCII Text to a New Level of Usability,"
The Public-Access Computer Systems Review 5, no. 5 (1994): 5-62.
(To retrieve this article, send the following e-mail message to
listserv@uhupvm1.uh.edu: GET BARRY PRV5N5 F=MAIL.)

23.   See http://www.lib.ncsu.edu/staff/morgan/simple.html.

24.   See http://cs.dal.ca/ftp/htmlasst/htmlafaq.html.

25.   See http://oneworld.wa.com/htmldev/devpage/dev-page2.html.

26.   Bohdan S. Wynar, Introduction to Cataloging and
Classification (Littleton, CO: Libraries Unlimited, 1980), 394.

About the Author

Eric Lease Morgan, Systems Librarian, NCSU Libraries, Box 7111,
Room 2316-B, Raleigh, NC 27695-7111.   Internet:
eric_morgan@ncsu.edu.

+ Page 26 +