[illegible]

SLAC ARCHIVES COLL 6
SERIES 1 SUBSERIES 17
BOX 3 FOLDER 1

Winters, Joan

/a/ebnextk/a/LocalApps/Obsolete/WorldWideWeb.app

[↓ AMD ~~automount~~
/nsf]

Hi Jean Marie – 11/9/00

Here's some interesting stuff I found in Paul Kunz's old NeXT WWW archive tree!

More later. Default.html was what he'd pointed me at.

Tom

```
/a/ebnextk/a/LocalApps/Obsolete/WorldWideWeb.app
```

AND automount

/nsf

[ls -l *]

```
-rwxr-xr-x   1 pfkeb    eb        106496 Nov 25 1992  WorldWideWeb*
-rw-r--r--   1 pfkeb    eb           144 Aug 12 1991  blank.html
-rw-r--r--   1 pfkeb    eb          1029 Feb 15 1991  compact.style
-rw-r--r--   1 root     reason      2609 Dec 14 1993  default.html
-rw-r--r--   1 pfkeb    eb          2610 Sep 16 1992  default.html~
-rw-r--r--   1 pfkeb    eb          1060 Jan 30 1993  default.style
-rw-r--r--   1 pfkeb    eb          3636 Nov 24 1992  docflash.tiff
-rw-r--r--   1 pfkeb    eb          4802 Nov 24 1992  docflash2.tiff
-rw-r--r--   1 pfkeb    eb           826 Nov 25 1992  help.html
-rw-r--r--   1 pfkeb    eb          1962 Nov 24 1992  icon1.tiff
-rw-r--r--   1 pfkeb    eb          1058 Apr 25 1991  paper.style
-rw-r--r--   1 pfkeb    eb          1057 Jun 12 1991  project.style
-rw-r--r--   1 pfkeb    eb          4802 Nov 24 1992  worldbook2.tiff

English.lproj:
total 0

WorldWideWeb.nib:
total 54
-rw-r--r--   1 pfkeb    eb          2592 Nov 25 1992  data.classes
-rw-r--r--   1 pfkeb    eb         22170 Nov 25 1992  data.nib
-rw-r--r--   1 pfkeb    eb          1962 Nov 25 1992  icon1.tiff
```

```
<title>(No title)</title>
<h1>Heading</h1>
Text<p>

<address>Author</address>
```

```
<TITLE>Welcome to the Universe of HyperText</TITLE>
<NEXTID 18>
<H2>Home</H2>Access to this information is provided as part of the <A NAME=0 HRE
F=http://info.cern.ch./hypertext/WWW/TheProject.html>WorldWideWeb</A>
project. The WWW project does not take responsability for the accuracy
of information provided by others.
<H2>How to proceed</H2>References to other information are represented like <A N
AME=1 HREF=#choose>this</A> . Double-click on it to jump to related information.
<H2>General Information sources</H2><A NAME=choose>Now choose an area in which y
ou would like to start browsing.</A> The
system currently has access to three sources of information.  With
the indexes, you should use the keyword search option on your browser.
<DL>
<DT><A NAME=1 HREF=http://slacvm.slac.stanford.edu:80/FIND/slac.html>SLAC Front
Page</A>
<DD>Presents the current SLAC test Front Page.
<DT><A NAME=9 HREF=http://info.cern.ch./hypertext/DataSources/News/Overview.html
>Internet News</A>
<DD> You can access the internet news scheme (See<A NAME=6 HREF=news:news.announ
ce.newusers> information
for new users</A> ). News articles are distributed typically CERN-wide
or worldwide, and have a finite lifetime.
</DL>
Newsgroups which may be of general interest at CERN include
<UL>
<LI><A NAME=7 HREF=news:cern.ecp.news>CERN/ECP news</A>
<LI><A NAME=8 HREF=news:cern.sting>STING (Software Technology Interest Group) ne
ws.</A>
</UL>
<H2>NeXT-related</H2>If you have a NeXT machine, see also the following topics:
<DL>
<DT><A NAME=13 HREF=http://info.cern.ch/hypertext/WWW/NeXT/WorldWideWeb.html>HEL
P</A>
<DD> on this WorldWideWeb application
<DT><A NAME=12 HREF=http://info.cern.ch/NeXT/Instalation.html>Installation at CE
RN</A>
<DD> A few notes on installing NeXTs and applications
at CERN.
<DT><A NAME=14 HREF=news:comp.sys.next.announce>comp.sys.next.announce</A>
<DD> The NeXT announcements newsgroup
<DT><A NAME=15 HREF=news:comp.sys.next.programmer>comp.sys.next.programmer</A>
<DD> The NeXT programmer's newsgroup
<DT><A NAME=16 HREF=news:comp.sys.next.misc>comp.sys.next.misc</A>
<DD> The NeXT miscellaneous newsgroup
<DT><A NAME=16 HREF=news:comp.sys.next.sysadmin>comp.sys.next.sysadmin</A>
<DD> The NeXT miscellaneous newsgroup
</DL>

<H2>More information</H2>There is more online information about <A NAME=7 HREF=h
ttp://info.cern.ch/hypertext/WWW/WhatIs.html>hypertext</A> in general and about
the <A NAME=8 HREF=http://info.cern.ch/hypertext/WWW//TheProject.html>WorldWideW
eb</A> project.
This page can and should be be customised.
If you have any comments, please contact
Tim Berners-Lee timbl@info.cern.ch.
```

```
<TITLE>Welcome to the Universe of HyperText</TITLE>
<NEXTID 18>
<H2>Home</H2>Access to this information is provided as part of the <A NAME=0 HRE
F=http://info.cern.ch./hypertext/WWW/TheProject.html>WorldWideWeb</A>
project. The WWW project does not take responsability for the accuracy
of information provided by others.
<H2>How to proceed</H2>References to other information are represented like <A N
AME=1 HREF=#choose>this</A> . Double-click on it to jump to related information.
<H2>General Information sources</H2><A NAME=choose>Now choose an area in which y
ou would like to start browsing.</A> The
system currently has access to three sources of information.  With
the indexes, you should use the keyword search option on your browser.
<DL>
<DT><A NAME=1 HREF=http://slacvm.slac.stanford.edu:5080/FIND/slac.html>SLAC Fron
t Page</A>
<DD>Presents the current SLAC test Front Page.
<DT><A NAME=9 HREF=http://info.cern.ch./hypertext/DataSources/News/Overview.html
>Internet News</A>
<DD> You can access the internet news scheme (See<A NAME=6 HREF=news:news.announ
ce.newusers> information
for new users</A> ). News articles are distributed typically CERN-wide
or worldwide, and have a finite lifetime.
</DL>
Newsgroups which may be of general interest at CERN include
<UL>
<LI><A NAME=7 HREF=news:cern.ecp.news>CERN/ECP news</A>
<LI><A NAME=8 HREF=news:cern.sting>STING (Software Technology Interest Group) ne
ws.</A>
</UL>
<H2>NeXT-related</H2>If you have a NeXT machine, see also the following topics:
<DL>
<DT><A NAME=13 HREF=http://info.cern.ch/hypertext/WWW/NeXT/WorldWideWeb.html>HEL
P</A>
<DD> on this WorldWideWeb application
<DT><A NAME=12 HREF=http://info.cern.ch/NeXT/Instalation.html>Installation at CE
RN</A>
<DD> A few notes on installing NeXTs and applications
at CERN.
<DT><A NAME=14 HREF=news:comp.sys.next.announce>comp.sys.next.announce</A>
<DD> The NeXT announcements newsgroup
<DT><A NAME=15 HREF=news:comp.sys.next.programmer>comp.sys.next.programmer</A>
<DD> The NeXT programmer's newsgroup
<DT><A NAME=16 HREF=news:comp.sys.next.misc>comp.sys.next.misc</A>
<DD> The NeXT miscellaneous newsgroup
<DT><A NAME=16 HREF=news:comp.sys.next.sysadmin>comp.sys.next.sysadmin</A>
<DD> The NeXT miscellaneous newsgroup
</DL>

<H2>More information</H2>There is more online information about <A NAME=7 HREF=h
ttp://info.cern.ch/hypertext/WWW/WhatIs.html>hypertext</A> in general and about
the <A NAME=8 HREF=http://info.cern.ch/hypertext/WWW//TheProject.html>WorldWideW
eb</A> project.
This page can and should be be customised.
If you have any comments, please contact
Tim Berners-Lee timbl@info.cern.ch.
```

[diff oldfind.html [illegible]]

```
11c11
< <DT><A NAME=1 HREF=http://slacvm.slac.stanford.edu:80/FIND/slac.html>SLAC Fron
t Page</A>
---
> <DT><A NAME=1 HREF=http://slacvm.slac.stanford.edu:5080/FIND/slac.html>SLAC Fr
ont Page</A>
44c44
< Tim Berners-Lee timbl@info.cern.ch.
---
> Tim Berners-Lee timbl@info.cern.ch.
```

```
<TITLE>Online Help for WorldWideWeb</TITLE>
<H1>Help</H1>
To recall the initial page, use "Home" from the "Navigate" menu.
For other information, double click on one of the following:
<ul>
<li><a name=manual href=http://info.cern.ch/hypertext/WWW/NextStep/WorldWideWeb.
html>
The user manual</a>
<li><a name=default href="default.html">The system default home page</a>
<li><a name=top href="http://info.cern.ch/hypertext/DataSources/Top.html">
An overview of information world wide</a>
</ul>
<h2>If these links don't go anywhere</h2>
Conatct the person who installed this
application on this machine.
This file (WorldWideWeb.app/help.html) probably needs editing to
change the name of the directory where the documentation has been put.
<p>
The file WorldWideWeb/default.html is the default home page. That can be edited,
 too.
```

(92)

8/1/97

Dan —

[illegible] also my [illegible] Bob's for report + talk @:

http:// [illegible] /grp[illegible]/

(Ethan V. [illegible])

(new) http:// [illegible] /grp[illegible]/[illegible] ([illegible]lab97/winter97/index.htm

← lab97winter.htm

Designing the SLAC Information Architecture

A Workplace for Users

InterLab '96

Joan M. Winters

Stanford Linear Accelerator Center

21 Aug 1996

Published in the *InterLab '96 Proceedings* by the National Renewable Energy Laboratory.

This page was last updated 28 Sep 1996.

Winters

Send copy 6

[illegible] 10/21/96

T[illegible] [illegible] keys

[illegible] data ? [illegible]

how things [illegible] now [illegible] up

[illegible] [illegible]ally [illegible] [illegible] failure, [illegible]?

[illegible] you want to publish

Ke[illegible] [illegible]

[illegible] [illegible] [illegible] [illegible]

[illegible] [illegible] files

[illegible] "[illegible] time out"

Lab[illegible]?

[illegible] Japan[illegible]

HIPZ [illegible]

[illegible] Research [illegible]

Terry [illegible] [illegible]

→ identificate

[illegible]?

Paul [illegible]?

summary paper [illegible]

identify key players

have key [illegible] identify key files

publish full-text key files on [illegible]

```
***************************************************************************************************************************************
**  VM/ESA    R1.1 SERVICE LEVEL 0000 ************** 10/08/96 ** 09:31:11 ** PRINTER 042E **** CPU SERIAL  74657  CPU MODEL  9021 **

BIN_'96   SLACARC   FILE NAME/TYPE=      MAIL$TMP  FILE         ORIGINID= SLACARC              VV          VV MM          MM    SLAC
BIN_ 96   SLACARC                                                                              VV          VV MMM        MMM  Computing
BIN_ 96   SLACARC   CREATION DATE/TIME= 10/08/96   08:31:18     SYSID=    SLACVM               VV          VV MM M      M MM   Services
3IN_:96   SLACARC                                                                              VV          VV MM  M   M   MM
BIN_196   SLACARC   COPY=    001   RECS= 0047      00000    333333333        4                  VV        VV  MM   M M    MM
BIN_196   SLACARC                                 0000000   333333333       44                   VV      VV   MM    M     MM
BIN_196   SLACARC   CLASS=   A     SPID= 0045    00     00  33     33      444                    VV   VV     MM          MM
BIN_196   SLACARC                                00     00       33       4444        EEEEEEEEEEEEEEV VSSSSSSSSSSS        MAAAAA
BIN_196   SLACARC   FORM=    STANDARD            00     00      3333     44 44        EE            VVSS      MM  SS     AA    AA
BIN_196   SLACARC                                00     00     333333   44  44        EE             VSS      MM        AAM     AA
BIN_196   SLACARC   DEST=    OFF                 00     00         33  444444444      EE              SS               AA        AA
BIN_196   SLACARC                                00     00  33     33  444444444      EEEEEEEEEEE      SSSSSSSSSSS    AAAAAAAAAAAAAA
BIN_196   SLACARC   CHAR=    TN5T                 0000000   333333333       44        EE                          SS  AA          AA
                                                   00000     3333333        44        EE                          SS  AA          AA
                                                                                      EE              SS          SS  AA          AA
                                                                                      EEEEEEEEEEEEE    SSSSSSSSSSS    AA          AA


TAG DATA:


  SSSSSSS   LL            AAA      CCCCCCC      AAA     RRRRRRRR    CCCCCCC
 SSSSSSSSS  LL           AAAAA    CCCCCCCCC    AAAAA    RRRRRRRRR  CCCCCCCCC
 SS     SS  LL          AA   AA   CC     CC   AA   AA   RR     RR  CC     CC
 SS         LL         AA     AA  CC         AA     AA  RR     RR  CC
 SSSSSSSS   LL         AA     AA  CC         AA     AA  RRRRRRRRR  CC
  SSSSSSSS  LL         AAAAAAAAA  CC         AAAAAAAAA  RRRRRRRR   CC
        SS  LL         AAAAAAAAA  CC         AAAAAAAAA  RR  RR     CC
 SS     SS  LL         AA     AA  CC     CC  AA     AA  RR   RR    CC     CC
 SSSSSSSSS  LLLLLLLLL  AA     AA  CCCCCCCCC  AA     AA  RR    RR   CCCCCCCCC
  SSSSSSS   LLLLLLLLL  AA     AA   CCCCCCC   AA     AA  RR     RR   CCCCCCC

 M       M     AAA      IIIIII    LL          $$$$$$$   TTTTTTTT   M       M  PPPPPPPP
 MM     MM    AAAAA     IIIIII    LL         $$$$$$$$$  TTTTTTTT   MM     MM  PPPPPPPPP
 MMM   MMM   AA   AA      II      LL         $$  $$ $$     TT      MMM   MMM  PP     PP
 MMMM MMMM  AA     AA     II      LL         $$  $$        TT      MMMM MMMM  PP     PP
 MM MMM MM  AA     AA     II      LL         $$$$$$$$      TT      MM MMM MM  PPPPPPPPP
 MM  M  MM  AAAAAAAAA     II      LL          $$$$$$$$     TT      MM  M  MM  PPPPPPPP
 MM     MM  AAAAAAAAA     II      LL             $$ $$     TT      MM     MM  PP
 MM     MM  AA     AA     II      LL         $$  $$ $$     TT      MM     MM  PP
 MM     MM  AA     AA   IIIIII    LLLLLLLLL  $$$$$$$$$     TT      MM     MM  PP
 MM     MM  AA     AA   IIIIII    LLLLLLLLL   $$$$$$$      TT      MM     MM  PP

_BBBBBBBB    IIIIII    N      NN                  1      9999999    6666666
 BBBBBBBBB   IIIIII    NN     NN                 11     999999999  666666666
 BB     BB     II      NNN    NN                111     99     99  66     66
 BB     BB     II      NNNN   NN                 11     99     99  66
 BBBBBBBBB     II      NN NN  NN                 11     999999999  66666666
 BBBBBBBB      II      NN  NN NN                 11      99999999  666666666
 BB     BB     II      NN   NNNN                 11            99  66     66
 BB     BB     II      NN    NNN                 11     99     99  66     66
_BBBBBBBBB   IIIIII    NN     NN               111111   999999999  666666666
-BBBBBBBB    IIIIII    NN      N               111111    9999999    6666666
```

SLAC on the Web

SLAC on the Web

- First WWW server in US in Fall, 1991
- SPIRES HEP Preprints first production application
    - Soon running thousands of accesses per month
- First production SLAC Home Page set installed November, 1993
    - Started an information architecture that has proven robust and extensible

Previous slide Next slide Back to the first slide View Graphic Version

[illegible]

```
==:T:== 10/16/96 11:53    WINTERS  @SLAC.St Old WWW File for the Paper/Docs Book?
=======================================================================
Received: from SLACVM by SLACVM.SLAC.STANFORD.EDU (Mailer R2.08 R208004) with
 BSMTP id 6008; Wed, 16 Oct 96 11:53:10 PST
Received: from SERV03.SLAC.STANFORD.EDU by SLACVM.SLAC.STANFORD.EDU
   (IBM VM SMTP V2R1) with TCP; Wed, 16 Oct 96 11:52:45 PST
Received: from DIRECTORY-DAEMON by SERV03.SLAC.STANFORD.EDU
 (PMDF V5.0-6 #10979) id <01IAPK7698C0000BEY@SERV03.SLAC.STANFORD.EDU> for
 jmdeken@SLACVM.SLAC.Stanford.EDU; Wed, 16 Oct 1996 11:51:39 -0800 (PST)
Received: from SLACVM.SLAC.STANFORD.EDU (MAILER@SLACVM)
 by SERV03.SLAC.STANFORD.EDU (PMDF V5.0-6 #10979)
 id <01IAPK4L229C00091J@SERV03.SLAC.STANFORD.EDU> for
 jmdeken@SLAC.Stanford.EDU; Wed, 16 Oct 1996 11:49:38 -0800 (PST)
Received: by SLACVM (Mailer R2.08 R208004) id 5975; Wed,
 16 Oct 1996 11:46:56 -0800 (PST)
Date: Wed, 16 Oct 1996 11:21 -0800 (PST)
From: "Joan M. Winters" <WINTERS@SLAC.Stanford.EDU>
Subject: Old WWW File for the Paper/Docs Book?
To: jmdeken@SLAC.Stanford.EDU
Message-id: <01IAPK4MQM1U00091J@SERV03.SLAC.STANFORD.EDU>
X-Envelope-to: jmdeken@SLACVM.SLAC.Stanford.EDU
Content-transfer-encoding: 7BIT
```

Jean - Look what I found when cleaning up an old VM disk. It's the CERN WWW Home Page and inwards especially following the WWW Project thread. See:

```
FILENAME FILETYPE FM FORMAT LRECL    RECORDS     BLOCKS   DATE     TIME    LABEL
WWW      CONSOLE  M0 V         82       1196         26 02/12/92 16:06:09 JCW400
```

I captured it using the VM spooled console.

SLAC is right there on the CERN home page, along with DESY; but not at FermiLab in sight! Note that this is about ~three months after we figured out the SLAC WWW server first came up.

Cheers,   Joan

```
++ ++ ++ ++ ++ ++ ++ ++ ++   Forwarded Text   ++ ++ ++ ++ ++ ++ ++ ++ ++

Ready;
www
                                                            CERN Information

                    CERN INFORMATION - SELECT BY NUMBER

       Help <1>            On this program, or the World-Wide Web project
                           <2>.

       Phone book <3>      People, phone numbers, accounts and email
                           addresses. See also the analytical Yellow Pages
                           <4>, or the same index in French : Pages Jaunes
                           <5>.

       "XFIND" index <6>   Index of computer centre documentation,
                           newsletters, news, help files, etc...

       News <7>            A complete list of all public CERN news groups,
                           such as news from the CERN User's Office <8>, CERN
```

```
                                 computer center news <9>, student news <10>. See
                                 also private groups <11> and Internet news <12>.
FROM OTHER SITES
   See online data by subject <13>, pointers to other forms of online data
   <14>, and the following specific databases:

       SLAC SPIRES <15>          The High Energy Physics preprint index at Stanford
                                 Linear Accelerator, California. (This is the same
                                 information avialable via the QSPIRES facility on
                                 BITNET. Include the word "FIND" as the first
                                 keyword, eg: K FIND AUTHOR FRED.).

       DESY documents <16>
                                 Documents and help files from the DESY lab in
                                 Hamburg.

       VMS Help <17>             VMS help data now available via a WWW gateway.

       Hacker Jargon <18>        An index to a cross-referenced set of hacker
                                 terms. A demonstration of the WWW gateway to the
                                 Graz Technical University Hyper-G database.
<ref.number>, Quit, or Help:
2

                                                    The World Wide Web project

                                    WORLD WIDE WEB

   The WorldWideWeb (W3) is a wide-area hypermedia <1> information retrieval
   initiative aiming to give universal access to a large universe of
   documents.
GENERAL PROJECT INFORMATION
   See also: an executive summary <2> of the project, Mailing lists <3> you
   can join, Policy <4> , latest W3  news <5> .

       Project Status <6>        A list of project components and their current
                                 state. (e.g. Line Mode <7> , NeXTStep <8> , Daemon
                                 <9> )

       People <10>               A list of people involved in the project.

       History <11>              A summary of the history of the project.

TECHNICAL DETAILS

       How to provide data <12>
                                 How can I make my own data available on the web?

       Protocols <13>            A description of the network protocols used.

       HTML format <14>          A description of the mark-up language used for
                                 some documents and for search hit-lists.

       Addressing <15>           The syntax of W3 document addresses.

       Design Issues <16>        A list of decisions to be made when designing or
```

```
                          selecting a hypertext/IR system. See also related
                          products <17> .

     Design notes <18>    Notes of meetings, etc, mostly historical.


<ref.number>, Back, <RETURN> for more, Quit, or Help:


     Data sources <19>    Sources of data on the web itself

     News <20>            Some internet/usenet newsgroups of possible
                          interest to the WorldWideWeb project.

     Coding standards <21>
                          A basic style guide for W3 code contributors.

DISTRIBUTION
  See the README <22> file, and copyright <23> notice. Tar files are
  available by anonymous ftp from info.cern.ch  in directory /pub .

   <End>


<ref.number>, Back, Quit, or Help:
2
```

Summary -- /WWW

WORLDWIDEWEB - SUMMARY

The WWW <1> project merges the techniques of information retrieval and hypertext to make an easy but powerful global information system.

The project is based on the philosophy that much academic information should be freely available to anyone. It aims to allow information sharing within internationally dispersed teams, and the dissemination of information by support groups. Originally aimed at the High Energy Physics community, it has spread to other areas and attracted much interest in user support, resource discovery and collaborative work areas.

Reader view

The WWW world consists of documents, and links. Indexes are special documents which, rather than being read, may be searched. The result of such a search is another ("virtual") document containing links to the documents found. A simple protocol (" HTTP <2> ") is used to allow a browser program to request a keyword search by a remote information server.

The web contains documents in many formats. Those documents which are hypertext, (real or virtual) contain links to other documents, or places within documents. All documents, whether real, virtual or indexes, look similar to the reader and are contained within the same addressing scheme.

To follow a link, a reader clicks with a mouse (or types in a number if he or she has no mouse). To search and index, a reader gives keywords (or other search criteria). These are the only operations necessary to access

```
    the entire world of data.
   Information provider view
    The WWW browsers can access many existing data systems via existing
    protocols (FTP, NNTP) or via HTTP and a gateway. In this way, the critical
    mass of data is quickly exceeded, and the increasing use of the system by
    readers and information suppliers encourage each other.

    Making a web is as simple as writing a few SGML <3> files which point to
    your existing data. Making it public involves running the FTP or HTTP
    daemon <4> , and making at least one link into your web from another. In
 <ref.number>, Back, <RETURN> for more, Quit, or Help:

    daemon <4> , and making at least one link into your web from another. In
    fact,  any file available by anonymous FTP can be immediately linked into
    a web. The very small start-up effort is designed to allow small
    contributions.  At the other end of the scale, large information providers
    may provide an HTTP server with full text or keyword indexing. This may
    allow access to a large existing database without changing the way that
    database is managed. Such gateways have already been made into Digital's
    VMS/Help, Technical Univerity of Graz's "Hyper-G", and Thinking Machine's
    "W.A.I.S." systems.

    The WWW model gets over the frustrating incompatibilities of data format
    between suppliers and reader by allowing negotiation of format between a
    smart browser and a smart server. This should provide a basis for
    extension into multimedia, and allow those who share application standards
    to make full use of them across the web.

    This summary does not describe the many exciting possibilities opened up
    by the WWW project, such as efficient document caching. the reduction of
    redundant out-of-date copies, and the use of knowledge daemons. There is
    more information in the online project documentation, including some
    background on hypertext and many technical notes.
   Try it
    You can try the simple line mode browser <5>  by telnetting to
    info.cern.ch with user name "www" (no password).  You can also find out
    more about WWW in this way.

    It is much more efficient to install the browser on your own machine. The
    line mode browser is currently available in source form by anonymous FTP
    from node  info.cern.ch [currently 128.141.201.74] as
                        /pub/WWWLineMode_v.vv.tar.Z.

    (v.vv is the version number - take the latest.)

    Also available is a hypertext editor <6> for the NeXT using the NeXTStep
    graphical user interface in file
                        /pub/WWWNeXTStepEditor_v.vv.tar.Z


 <ref.number>, Back, <RETURN> for more, Quit, or Help:
 list

      HYPERTEXT REFERENCES :=
     [1]        TheProject.html

     [2]        Protocols/HTTP/AsImplemented.html

     [3]        MarkUp/MarkUp.html

     [4]        Daemon/Overview.html

     [5]        LineMode/Browser.html

     [6]        NeXT/WorldWideWeb.html


<ref.number>, Back, <RETURN> for more, Quit, or Help:
2

                                   The HTTP Protocol As Implemented In W3

                         HTTP AS IMPLEMENTED IN WWW

   This document defines the Hypertext Transfer protocol  (HTTP) as currently
   implemented by the WorldWideWeb <1> initaitive software. This is a subset
   of the proposed <2> full HTTP protocol.  No client profile information is
   transferred with the query. Future HTTP protocols will be back-compatible
   with this protocol.

   The protocol  uses the normal internet-style telnet protocol style on a
   TCP-IP link. The following describes how a client acquires a (hypertext)
   document from an HTTP server, given an HTTP document address <3> .
CONNECTION
   The client makes a TCP-IP connection to the host using the domain name <4>
   or IP number <5> , and the port number <6>  given in the address.

   During development, the default HTTP TCP port number is 2784 -- this will
   change when an official port number is allocated.

   The server accepts the connection.

   Note: HTTP currently runs over TCP, but could run over any
   connection-oriented service.   The interpretation of the protocol below in
   the case of a sequenced packet service (such as DECnet(TM) or ISO TP4) is
   that that the request should be one TPDU, but the repose may be many.
REQUEST
   The client sends a document request consisting of a line of ASCII
   characters terminated by a CR LF (carriage return, line feed) pair. A
   well-behaved server will not require the carriage return character.

   This request consists of the word "GET", a space, the document address <7>
   , omitting the "http:, host and port parts when they are the coordinates
   just used to make the connection. (If a gateway is being used, then a full
   document address may be given specifying a different naming scheme).

   The search functionality of the protocol lies in the ability of the
   addressing syntax to describe a search on a named index <8> .

<ref.number>, Back, <RETURN> for more, Quit, or Help:
```

A search should only be requested by a client when the index document itself has been descibed as an index using the ISINDEX tag <9> .

RESPONSE

The response to a simple GET request is a message in hypertext mark-up language (HTML <10>). This is a byte stream of ASCII characters.

Lines shall be delimited by an optional carriage return followed by a mandatory line feed chararcter. The client should not assume that the carriage return will be present. Lines may be of any length. Well-behaved servers should retrict line length to 80 characters excluding the CR LF pair.

The format of the message is HTML - that is, a trimmed SGML document. Note that this format allows for menus and hit lists to be returned as hypertext. It also allows for plain ASCII text to be returned following the PLAINTEXT tag <11> .

The message is terminated by the closing of the connection by the server.

Well-behaved clients will read the entire document as fast as possible. The client shall not wait for user action (output paging for example) before reading the whole of the document. The server may impose a timeout of the order of 15 seconds on inactivity.

Error responses are supplied in human readable text in HTML syntax. There is no way to distinguish an error response from a satisfactory response except for the content of the text.

DISCONNECTION

The TCP-IP connection is broken by the server when the whole document has been transferred.

The client may abort the transfer by breaking the connection before this, in which case the server will not record any error condidtion.

Requests are idempotent <12> . The server need not store any information about the request after disconnection.

<ref.number>, Back, Quit, or Help:
12

(No title)

HYPERTEXT TRANSFER PROTOCOL

See also: Why a new protocol? <1> , Other protocols used <2>

This is a list of the choices made and features needed in a hypertext transfer protocol. See also the HTTP protocol as currently implemented <3>.

UNDERLYING PROTOCOL

There are various distinct possible bases for the protocol - we can choose

Something based on, and looking like, an Internet protocol. This has the advantage of being well understood, of existing implementations being all over the place. It also leaves open the possibility of a universal FTP/HTTP or NNTP/HTTP server. This is the case for the current HTTP.

Something based on an RPC standard. This has the advantage of making it easy to generate the code, that the parsing of the messages is done automatically, and that the transfer of binary data is efficient. It has the disadvantage that one needs the RPC code to be available on all platforms. One would have to chose one (or more) styles of RPC. Another disadvantage may be that existing RPC systems are not efficient at transferring large quantities of text over a stream protocol unless (like DD-OC-RPC) one has a let-out and can access the socket directly.

Something based on the OSI stack, as is Z39.50. This would have to be run over TCP in the internet world.

Current HTTP uses the first alternative, to make it simple to program, so that it will catch on: conversion to run over an OSI stack will be simple as the structure of the messages is well defined.

IDEMPOTENT ?

Another choice is whether to make the protocol idempotent or not. That is, does the server need to keep any state informat about the client? (For example, the NFS protocol is idempotent, but the FTP and NNTP protocols are not.) In the case of FTP the state information consists of

```
<ref.number>, Back, <RETURN> for more, Quit, or Help:
RDR FILE 1133 SENT FROM MAILER   PUN WAS 6161 RECS 0036 CPY  001 M NOHOLD NOKEEP
15:47:08  * MSG FROM MAILER   : * New mail from
<lee_collins1@gateway.qm.apple.com>
```

are not.) In the case of FTP the state information consists of authorisation, which is not trvial to establish every time but could be, and current directory and transfer mode which are basically trivial. The propsed protocol IS idempotent.

This causes, in principle, a problem when trying to map a non-dempotent system (such as library search systems which stored "result sets" on behalf of the client) into the web. The problem is that to use them in an idempotent way requires the re-evaluation of the intermediate result sets at each query. This can be solved by the gateway intelligently caching result sets for a reasonable time.

REQUEST: INFORMATION TRANSFERRED FROM CLIENT

Parameters below, however represented on the network, are given in upper case, with parameter names in lower case. This set assumes a model of format negociation in which in which the client says what he can take, and the server decides what to give him. One imagines that each function would return a status, as well as information specified below.

When running over a byte stream protocol, SGML would be an encoding possibility (as well as ASN/1 etc).

GET document name   Please transfer a named document back. Transfer the results back in a standard format or one which I have said I can accept. The reply includes the format. In practice, one may want to transfer the document over the same link (a la NNTP) or a different one (a la FTP). There are advantages in

```
                         each technique. The use of the same link is
                         standard, with moving to a different link by
                         negociation (see PORT <4> ).

     SEARCH  keywords    Please search the given index document for all
                         items with the given word combination, and
                         transfer the results back as marked up hypertext.
                         This could elaborate to an SQL query. There are
                         many advantages in making the search criterion
                         just a subset of the document name space.

<ref.number>, Back, <RETURN> for more, Quit, or Help:
list

     HYPERTEXT REFERENCES :=

    [1]      WhyHTTP.html

    [2]      RelevantProtocols.html

    [3]      HTTP/AsImplemented.html

    [4]      HTTP.html#5


<ref.number>, Back, <RETURN> for more, Quit, or Help:
3
                                   The HTTP Protocol As Implemented In W3

                        HTTP AS IMPLEMENTED IN WWW

  This document defines the Hypertext Transfer protocol  (HTTP) as currently
  implemented by the WorldWideWeb <1> initaitive software. This is a subset
  of the proposed <2> full HTTP protocol.  No client profile information is
  transferred with the query. Future HTTP protocols will be back-compatible
  with this protocol.

  The protocol  uses the normal internet-style telnet protocol style on a
  TCP-IP link. The following describes how a client acquires a (hypertext)
  document from an HTTP server, given an HTTP document address <3> .
CONNECTION
  The client makes a TCP-IP connection to the host using the domain name <4>
  or IP number <5> , and the port number <6>  given in the address.

  During development, the default HTTP TCP port number is 2784 -- this will
  change when an official port number is allocated.

  The server accepts the connection.

  Note: HTTP currently runs over TCP, but could run over any
  connection-oriented service.   The interpretation of the protocol below in
  the case of a sequenced packet service (such as DECnet(TM) or ISO TP4) is
  that that the request should be one TPDU, but the repose may be many.
```

```
REQUEST
   The client sends a document request consisting of a line of ASCII
   characters terminated by a CR LF (carriage return, line feed) pair. A
   well-behaved server will not require the carriage return character.

   This request consists of the word "GET", a space, the document address <7>
   , omitting the "http:, host and port parts when they are the coordinates
   just used to make the connection. (If a gateway is being used, then a full
   document address may be given specifying a different naming scheme).

   The search functionality of the protocol lies in the ability of the
   addressing syntax to describe a search on a named index <8> .

<ref.number>, Back, <RETURN> for more, Quit, or Help:


   A search should only be requested by a client when the index document
   itself has been descibed as an index using the  ISINDEX tag <9> .
RESPONSE
   The response to a simple GET request is a message in hypertext mark-up
   language ( HTML <10> ). This is a byte stream of ASCII characters.

   Lines shall be delimited by an optional carriage return followed by a
   mandatory line feed chararcter. The client should not assume that the
   carriage return will be present.  Lines may be of any length. Well-behaved
   servers should retrict line length to 80 characters excluding the CR LF
   pair.

   The format of the message is HTML - that is, a trimmed SGML document. Note
   that this format allows for menus and hit lists to be returned as
   hypertext. It also allows for plain ASCII text to be returned following
   the  PLAINTEXT tag <11> .

   The message is terminated by  the closing of the connection by the server.

   Well-behaved clients will read the entire document as fast as possible.
   The client shall not wait for user action (output paging for example)
   before reading the whole of the document.  The server may impose a timeout
   of the order of 15 seconds on inactivity.

   Error responses are supplied in human readable text in HTML syntax. There
   is no way to distinguish an error response from a satisfactory response
   except for the content of the text.
DISCONNECTION
   The TCP-IP connection is broken by the server when the whole document has
   been transferred.

   The client may abort the transfer by breaking the connection before this,
   in which case the server will not record any error condidtion.

   Requests are idempotent <12> .  The server need not store any information
   about the request after disconnection.


<ref.number>, Back, Quit, or Help:
4
```

BNF -- /Addressing

W3 ADDRESS SYNTAX: BNF

This is a BNF-like description of the W3 addressing syntax <1> . We use a vertical line "|" to indicate alternatives, and [brackets] to indicate optional parts. Spaces are representational only: no spaces are actually allowed within a W3 address. Single letters stand for single letters. All words of more than one letter below are entites described elsewhere in the syntax description. (Entity names are here linked to their definitions, probably making this difficult to read with the line mode browser.)

An absolute address specified in a link is an anchoraddress <2> . The address which is passed to a server is a docaddress <3> .

```
    anchoraddress       docaddress <4> [ # anchor <5> ]

    docaddress          httpaddress <6> | fileaddress <7> | newsaddress
                        <8> | telnetaddress <9> | gopheraddress <10> |
                        waisaddress <11>

    httpaddress         h t t p :   / / hostport <12>  [  / path <13> ] [
                        ? search <14> ]

    fileaddress         f i l e : / / host <15> / path <16>

    newsaddress         n e w s : groupart <17>

    waisaddress         waisindex | waisdoc

    waisindex           w a i s : / / hostport <18> / database <19> [ ?
                        search <20> ]

    waisdoc             w a i s : / / hostport <21> / database <22> /
                        wtype <23> / digits <24> / path <25>

    groupart            * | group <26> | article <27>
```

```
<ref.number>, Back, <RETURN> for more, Quit, or Help:
list

     HYPERTEXT REFERENCES :=

    [1]       Addressing.html

    [2]       #47

    [3]       #22

    [4]       #22

    [5]       #20

    [6]       #1
```

```
     [7]        #51

     [8]        #92

     [9]        #55

     [10]        #72

     [11]        #105

     [12]        #3

     [13]        #9

     [14]        #11

     [15]        #43

     [16]        #9

     [17]        #90

     [18]        #3

     [19]        #95

     [20]        #11

     [21]        #3

     [22]        #95

     [23]        #102

     [24]        #24

     [25]        #9

     [26]        #87

     [27]        #85

     [28]        #81

     [29]        #87


<ref.number>, Back, <RETURN> for more, Quit, or Help:
help


WWW Line Mode Browser:     COMMANDS AVALIABLE Version 0.14

    You are reading document:
    'http://info.cern.ch/hypertext/WWW/Addressing/BNF.html'
    <RETURN>       Produces the next page of the remaining text.
    TOP            Returns to the first page of the present document.
    LIST           Produces a list of hypertext references
                   which have been accumulated from the text
                   already shown on the screen.
    <number>       Select a referenced document by number
                   (from 1 to 29)
    RECALL         Gives a list of the previous nodes
                   visited.
    RECALL <number>
                   Returns to a previously visited document
                   numbered in the recall list.
    HOME           Returns to the starting node
    BACK           Moves back to the previous node
    Q              Quits the program.

<ref.number>, Back, <RETURN> for more, Quit, or Help:
recall


          HISTORY OF PREVIOUS NODES :-


     1)        CERN Information
     2)        The World Wide Web project
     3)        Summary -- /WWW
     4)        The HTTP Protocol As Implemented In W3
     5)        (No title)
     6)        The HTTP Protocol As Implemented In W3
     7)        BNF -- /Addressing


<ref.number>, Back, <RETURN> for more, Quit, or Help:
recall 5
                                                                    (No title)

                           HYPERTEXT TRANSFER PROTOCOL

   See also: Why a new protocol? <1> ,  Other protocols used <2>

   This is a list of the choices made and features needed in a hypertext
   transfer protocol.   See also the HTTP protocol as currently implemented
   <3>.
UNDERLYING PROTOCOL
   There are various distinct possible bases for the protocol - we can choose

       Something based on, and looking like, an Internet protocol. This has
       the advantage of being well understood, of existing implementations
       being all over the place. It also leaves open the possibility of a
       universal FTP/HTTP or NNTP/HTTP server. This is the case for the
       current HTTP.

       Something based on an RPC standard. This has the advantage of making
       it easy to generate the code, that the parsing of the messages is done
       automatically, and that the transfer of binary data is efficient. It
       has the disadvantage that one needs the RPC code to be available on
```

all platforms. One would have to chose one (or more) styles of RPC. Another disadvantage may be that existing RPC systems are not efficient at transferring large quantities of text over a stream protocol unless (like DD-OC-RPC) one has a let-out and can access the socket directly.

Something based on the OSI stack, as is Z39.50. This would have to be run over TCP in the internet world.

Current HTTP uses the first alternative, to make it simple to program, so that it will catch on: conversion to run over an OSI stack will be simple as the structure of the messages is well defined.

IDEMPOTENT ?

Another choice is whether to make the protocol idempotent or not. That is, does the server need to keep any state informat about the client? (For example, the NFS protocol is idempotent, but the FTP and NNTP protocols are not.) In the case of FTP the state information consists of
<ref.number>, Back, <RETURN> for more, Quit, or Help:

are not.) In the case of FTP the state information consists of authorisation, which is not trvial to establish every time but could be, and current directory and transfer mode which are basically trivial. The propsed protocol IS idempotent.

This causes, in principle, a problem when trying to map a non-dempotent system (such as library search systems which stored "result sets" on behalf of the client) into the web. The problem is that to use them in an idempotent way requires the re-evaluation of the intermediate result sets at each query. This can be solved by the gateway intelligently caching result sets for a reasonable time.

REQUEST: INFORMATION TRANSFERRED FROM CLIENT

Parameters below, however represented on the network, are given in upper case, with parameter names in lower case. This set assumes a model of format negociation in which the client says what he can take, and the server decides what to give him. One imagines that each function would return a status, as well as information specified below.

When running over a byte stream protocol, SGML would be an encoding possibility (as well as ASN/1 etc).

GET document name — Please transfer a named document back. Transfer the results back in a standard format or one which I have said I can accept. The reply includes the format. In practice, one may want to transfer the document over the same link (a la NNTP) or a different one (a la FTP). There are advantages in each technique. The use of the same link is standard, with moving to a different link by negociation (see PORT <4>).

SEARCH keywords — Please search the given index document for all items with the given word combination, and transfer the results back as marked up hypertext. This could elaborate to an SQL query. There are many advantages in making the search criterion just a subset of the document name space.

<ref.number>, Back, <RETURN> for more, Quit, or Help:
4

(No title)

HYPERTEXT TRANSFER PROTOCOL

See also: Why a new protocol? <1> , Other protocols used <2>

This is a list of the choices made and features needed in a hypertext transfer protocol. See also the HTTP protocol as currently implemented <3>.

UNDERLYING PROTOCOL

There are various distinct possible bases for the protocol - we can choose

Something based on, and looking like, an Internet protocol. This has the advantage of being well understood, of existing implementations being all over the place. It also leaves open the possibility of a universal FTP/HTTP or NNTP/HTTP server. This is the case for the current HTTP.

Something based on an RPC standard. This has the advantage of making it easy to generate the code, that the parsing of the messages is done automatically, and that the transfer of binary data is efficient. It has the disadvantage that one needs the RPC code to be available on all platforms. One would have to chose one (or more) styles of RPC. Another disadvantage may be that existing RPC systems are not efficient at transferring large quantities of text over a stream protocol unless (like DD-OC-RPC) one has a let-out and can access the socket directly.

Something based on the OSI stack, as is Z39.50. This would have to be run over TCP in the internet world.

Current HTTP uses the first alternative, to make it simple to program, so that it will catch on: conversion to run over an OSI stack will be simple as the structure of the messages is well defined.

IDEMPOTENT ?

Another choice is whether to make the protocol idempotent or not. That is, does the server need to keep any state informat about the client? (For example, the NFS protocol is idempotent, but the FTP and NNTP protocols are not.) In the case of FTP the state information consists of
<ref.number>, Back, <RETURN> for more, Quit, or Help:

are not.) In the case of FTP the state information consists of authorisation, which is not trvial to establish every time but could be, and current directory and transfer mode which are basically trivial. The propsed protocol IS idempotent.

This causes, in principle, a problem when trying to map a non-dempotent system (such as library search systems which stored "result sets" on behalf of the client) into the web. The problem is that to use them in an idempotent way requires the re-evaluation of the intermediate result sets at each query. This can be solved by the gateway intelligently caching result sets for a reasonable time.

REQUEST: INFORMATION TRANSFERRED FROM CLIENT

Parameters below, however represented on the network, are given in upper case, with parameter names in lower case. This set assumes a model of format negociation in which in which the client says what he can take, and the server decides what to give him. One imagines that each function would return a status, as well as information specified below.

When running over a byte stream protocol, SGML would be an encoding possibility (as well as ASN/1 etc).

GET document name   Please transfer a named document back. Transfer the results back in a standard format or one which I have said I can accept. The reply includes the format. In practice, one may want to transfer the document over the same link (a la NNTP) or a different one (a la FTP). There are advantages in each technique. The use of the same link is standard, with moving to a different link by negociation (see PORT <4>).

SEARCH keywords   Please search the given index document for all items with the given word combination, and transfer the results back as marked up hypertext. This could elaborate to an SQL query. There are many advantages in making the search criterion just a subset of the document name space.

<ref.number>, Back, <RETURN> for more, Quit, or Help:
4

(No title)

HYPERTEXT TRANSFER PROTOCOL

See also: Why a new protocol? <1> , Other protocols used <2>

This is a list of the choices made and features needed in a hypertext transfer protocol. See also the HTTP protocol as currently implemented <3>.

UNDERLYING PROTOCOL

There are various distinct possible bases for the protocol - we can choose

Something based on, and looking like, an Internet protocol. This has the advantage of being well understood, of existing implementations being all over the place. It also leaves open the possibility of a universal FTP/HTTP or NNTP/HTTP server. This is the case for the current HTTP.

Something based on an RPC standard. This has the advantage of making it easy to generate the code, that the parsing of the messages is done automatically, and that the transfer of binary data is efficient. It has the disadvantage that one needs the RPC code to be available on all platforms. One would have to chose one (or more) styles of RPC. Another disadvantage may be that existing RPC systems are not efficient at transferring large quantities of text over a stream protocol unless (like DD-OC-RPC) one has a let-out and can access the socket directly.

Something based on the OSI stack, as is Z39.50. This would have to be

```
     run over TCP in the internet world.

  Current HTTP uses  the first alternative, to make it simple to program, so
  that it will catch on: conversion to run over an OSI stack will be simple
  as the structure of the messages is well defined.
IDEMPOTENT ?
  Another choice is whether to make the protocol idempotent or not. That is,
  does the server need to keep any state informat about the client? (For
  example, the NFS protocol is idempotent, but the FTP and NNTP protocols
  are not.) In the case of FTP the state information consists of
<ref.number>, Back, <RETURN> for more, Quit, or Help:
2

                          Protocol Specifications relevant to HyperText

                       RELEVANT PROTOCOLS

  The WorldWideWeb <1> system can pick up information from many information
  sources, using existing protocols. Among these are file and news transfer
  protocols.
FILE TRANSFER
  The file transfer protocol currently most used for accessing fairly stable
  public information over a wide area is "Anonymous FTP". This means the use
  of the internet File Transfer Protocol without authentication. As the WWW
  project currently operates for the sake of public information, anonymous
  FTP is quite appropriate, and WWW can pick up any information provided by
  anonymous FTP. FTP is defined in RFC 959 <2> which includes material from
  many previous RFCs. (See also:  file address syntax <3> ). See also the
  prospero project and the shift <4> project, for more powerful file access
  systems.
NETWORK NEWS
  The "Network News Transfer Protocol" (NNTP) is defined in RFC 977 <5> by
  Kantor and Lampsley. This allows transient news information in the USENET
  news format to be exchanged over the internet. The format of news articles
  is defined in RFC 850, Standard for Interchange of USENET Messages <6> by
  Mark Horton. This in turn refers to the standard RFC 822 <7> which defines
  the format of internet mail messages. (See news address syntax <8> )
SEARCH AND RETRIEVE
  The WWW project defines its own protocol for information transfer, which
  allows for negotiation on representation. This we call HTTP, for HyperText
  Transfer Protocol <9> .See also  HyperText Transfer Format <10> , and the
  HTTP address syntax ) <11>
  Whilst the HTTP <12> protocol provides an index search function, another
  common protocol for index search is Z39.50, and the version of it used by
  the WAIS <13> project.  (See also the WAIS-WWW gateway <14> ).

    <End>


<ref.number>, Back, Quit, or Help:
9


                          The HTTP Protocol As Implemented In W3
```

HTTP AS IMPLEMENTED IN WWW

This document defines the Hypertext Transfer protocol (HTTP) as currently implemented by the WorldWideWeb <1> initaitive software. This is a subset of the proposed <2> full HTTP protocol. No client profile information is transferred with the query. Future HTTP protocols will be back-compatible with this protocol.

The protocol uses the normal internet-style telnet protocol style on a TCP-IP link. The following describes how a client acquires a (hypertext) document from an HTTP server, given an HTTP document address <3> .

CONNECTION

The client makes a TCP-IP connection to the host using the domain name <4> or IP number <5> , and the port number <6> given in the address.

During development, the default HTTP TCP port number is 2784 -- this will change when an official port number is allocated.

The server accepts the connection.

Note: HTTP currently runs over TCP, but could run over any connection-oriented service. The interpretation of the protocol below in the case of a sequenced packet service (such as DECnet(TM) or ISO TP4) is that that the request should be one TPDU, but the repose may be many.

REQUEST

The client sends a document request consisting of a line of ASCII characters terminated by a CR LF (carriage return, line feed) pair. A well-behaved server will not require the carriage return character.

This request consists of the word "GET", a space, the document address <7> , omitting the "http:, host and port parts when they are the coordinates just used to make the connection. (If a gateway is being used, then a full document address may be given specifying a different naming scheme).

The search functionality of the protocol lies in the ability of the addressing syntax to describe a search on a named index <8> .

<ref.number>, Back, <RETURN> for more, Quit, or Help:

A search should only be requested by a client when the index document itself has been descibed as an index using the ISINDEX tag <9> .

RESPONSE

The response to a simple GET request is a message in hypertext mark-up language (HTML <10>). This is a byte stream of ASCII characters.

Lines shall be delimited by an optional carriage return followed by a mandatory line feed chararcter. The client should not assume that the carriage return will be present. Lines may be of any length. Well-behaved servers should retrict line length to 80 characters excluding the CR LF pair.

The format of the message is HTML - that is, a trimmed SGML document. Note that this format allows for menus and hit lists to be returned as hypertext. It also allows for plain ASCII text to be returned following the PLAINTEXT tag <11> .

The message is terminated by the closing of the connection by the server.

Well-behaved clients will read the entire document as fast as possible. The client shall not wait for user action (output paging for example) before reading the whole of the document. The server may impose a timeout of the order of 15 seconds on inactivity.

Error responses are supplied in human readable text in HTML syntax. There is no way to distinguish an error response from a satisfactory response except for the content of the text.

DISCONNECTION

The TCP-IP connection is broken by the server when the whole document has been transferred.

The client may abort the transfer by breaking the connection before this, in which case the server will not record any error condidtion.

Requests are idempotent <12> . The server need not store any information about the request after disconnection.

```
<ref.number>, Back, Quit, or Help:
recall


            HISTORY OF PREVIOUS NODES :-


      1)       CERN Information
      2)       The World Wide Web project
      3)       Summary -- /WWW
      4)       The HTTP Protocol As Implemented In W3
      5)       (No title)
      6)       The HTTP Protocol As Implemented In W3
      7)       BNF -- /Addressing
      8)       (No title)
      9)       Protocol Specifications relevant to HyperText
     10)       The HTTP Protocol As Implemented In W3


<ref.number>, Back, Quit, or Help:
recall 9
```

Protocol Specifications relevant to HyperText

RELEVANT PROTOCOLS

The WorldWideWeb <1> system can pick up information from many information sources, using existing protocols. Among these are file and news transfer protocols.

FILE TRANSFER

The file transfer protocol currently most used for accessing fairly stable public information over a wide area is "Anonymous FTP". This means the use of the internet File Transfer Protocol without authentication. As the WWW project currently operates for the sake of public information, anonymous FTP is quite appropriate, and WWW can pick up any information provided by anonymous FTP. FTP is defined in RFC 959 <2> which includes material from

```
   many previous RFCs. (See also:  file address syntax <3> ). See also the
   prospero project and the shift <4> project, for more powerful file access
   systems.
NETWORK NEWS
   The "Network News Transfer Protocol" (NNTP) is defined in RFC 977 <5> by
   Kantor and Lampsley. This allows transient news information in the USENET
   news format to be exchanged over the internet. The format of news articles
   is defined in RFC 850, Standard for Interchange of USENET Messages <6> by
   Mark Horton. This in turn refers to the standard RFC 822 <7> which defines
   the format of internet mail messages. (See news address syntax <8> )
SEARCH AND RETRIEVE
   The WWW project defines its own protocol for information transfer, which
   allows for negotiation on representation. This we call HTTP, for HyperText
   Transfer Protocol <9> .See also  HyperText Transfer Format <10> , and the
   HTTP address syntax ) <11>
   Whilst the HTTP <12> protocol provides an index search function, another
   common protocol for index search is Z39.50, and the version of it used by
   the WAIS <13> project.  (See also the WAIS-WWW gateway <14> ).

     <End>



<ref.number>, Back, Quit, or Help:
10

                                                 HyperText Mark-up Language
                                   HTML

   The WWW <1> system uses marked-up text to represent a hypertext document
   for transmision over the network. The hypertext mark-up language is an
   SGML <2> format. This defines the basic syntax used. The particular
   language, the set of tags and the rules about their use, and their
   significance is not part of the SGML standard. There being no standard on
   this, we have adopted a set which seems sensible. We call them HTML --
   hypertext markup language. HTML is not an alternative to SGML, it is a
   particular format within the SGML rules (an SGML "DTD"). HTML parsers
   should ignore tags which they do not understand, and ignore attributes
   which they do not understand of tags which they do understand.

   See also:

       The tags <3>        A list of the tags used in HTML with their
                           significance.

       Example <4>         A file containing a variety of tags used for test
                           purposes, and its source text <5>.
DEFAULT TEXT
   Unless otherwise defined by tags, text is transmitted as a stream of
   lines. The division of the stream of characters into lines is arbitrary,
   and only made in order to allow the text to be passed through systems
   which can only handle text with a limited line length. The recommended
   line length for transmission is 80 characters. The division into lines has
   no significance (except in the case of  example sections <6> and PLAINTEXT
   <7> ) apart from indicating a word end. Line breaks between tags have no
   significance.

     <End>



<ref.number>, Back, Quit, or Help:
4

                                       Hypertext HTML formatting example

                              TEST DATASET

   This is an example bit of hypertext - compare  the formatted version with
   the original HTML source <1>. Let's try introducing an initial paragrpah
   between the H1 and the H2 headings.

   This file was typed into WWW (on the NeXT) and saved.
INTRODUCTION
   This file contains a test set of HTML mark-up, as a test of hypertext
   browsers and an example of the syntax of the tags. See also:

       An arbitrary news article <2>

       The newsgroup on hypertext <3>

       More details about the WWW project. <4>

   That is the end of the list.
SOME ANCHORS
   Anchors come in two forms: whole nodes or parts of nodes. The line mode
   browser can't currently (Nov 91)  jump to a part of a node: it always
   jumps to the top.
  Leading to whole nodes
   Here is an anchor which leads to the VM FIND command. <5> Note the nested
   highlighting (hp2) within the anchor. If you want to click on this, <6>
   you will go to the system default page.
  Leading to anchors within nodes
   Now THIS <7> leads to anchor #2 in this file, and if you want to click on
   THIS, <8> you should go to the system default page with anchor #2
   selected. Now the word "destination" is a named destination anchor,
   connected to the word "source" <9> . Clicking on the destination shouldn't
   do anything. Selecting the source should lead to the destination.

   Now let's go through the limited set of markup tags which we accept. The
   title, "Hypertext HTML formatting example" was between TITLE tags. "Test
   Dataset" at the top of this page was a Level One Heading (H1). The quick

list
recall

?????
```

```
quit
hx

hx
CMS

CMS
jkljkl
SCSABN: invalid FCB (  6) at X'0000CB', processing terminated
SCSABN: recursive abend during abend recovery
HCPGIR450W CP entered; disabled wait PSW 00060000 15E4E3E2
IPL CMS
SLAC XA CMS 5.6 11/01/91 15:32


TEST disk for testing files prior to installing on the U or Y disks.

CMS29E ( 59C R ) RO
CMS5A5 ( 1A7 P ) RO

660 USERS, 295 LOGGED, 365 DISC, 0 DIALED, 0 NET
AVGPROC-100% 03
XSTORE-000320/SEC MIGRATE-0006/SEC
XSTORE MDC READS-000169/SEC WRITES-000009/SEC HIT RATIO-099%
STORAGE-031% PAGING-0020/SEC STEAL-000%
Q0-00000(00000)                               DORMANT-00642
Q1-00003(00000)            E1-00000(00000)
Q2-00002(00000) EXPAN-002 E2-00000(00000)
Q3-00017(00000) EXPAN-002 E3-00000(00000)
PROC 0001-100%           PROC 0000-100%
PROC 0002-100%


Conference LEPNEWS has 45 new items.
Conference PUBEXECS has 73 new items.
Conference XA-ASM has 8 new items.
Conference XA-FORT has 15 new items.

FILES: 0058 RDR, 0001 PRT,   NO PUN
Ready;
vmstatus restore
Disk S(190) could not be accessed
Ready;
qdiskid
Label   CCU  Mode Stat Owner    Vaddr

TMP193  193  A    R/W  Tdisk    -
JCW198  198  B    R/W  WINTERS  0198
JCW192  192  D    R/W  WINTERS  0192
BEB191  195  E    R/O  BEBO     0191
JCW194  194  J    R/W  WINTERS  0194
JCW400  400  M    R/W  WINTERS  0400
JCW191  191  N    R/W  WINTERS  0191
MAIL    1A6  O    R/O  RENATA   0195
CMS5A5  1A7  P    R/O  XAMAINT  05A5
```

```
CMS29E   59C  R    R/O  MAINT     028E
MNT190   190  S    R/O  XAMAINT   0190
CMS59E   59E  T    R/O  $TDISK    049E
CMS59F   59F  U    R/O  $UDISK    049F
CMS19E   19E  Y/S  R/O  $YDISK    029E
JCW223   223  Z    R/W  WINTERS   0223
Ready;
l
CMS      PRESERVE A1
FAQ      MAIL     A0
Ready;
qdiskid
Label    CCU  Mode Stat Owner     Vaddr

TMP193   193  A    R/W  Tdisk     -
JCW198   198  B    R/W  WINTERS   0198
JCW192   192  D    R/W  WINTERS   0192
BEB191   195  E    R/O  BEBO      0191
JCW194   194  J    R/W  WINTERS   0194
JCW400   400  M    R/W  WINTERS   0400
JCW191   191  N    R/W  WINTERS   0191
MAIL     1A6  O    R/O  RENATA    0195
CMS5A5   1A7  P    R/O  XAMAINT   05A5
CMS29E   59C  R    R/O  MAINT     028E
MNT190   190  S    R/O  XAMAINT   0190
CMS59E   59E  T    R/O  $TDISK    049E
CMS59F   59F  U    R/O  $UDISK    049F
CMS19E   19E  Y/S  R/O  $YDISK    029E
JCW223   223  Z    R/W  WINTERS   0223
Ready;
q cons
CONS 0009 ON LDEV L041F   TERM START
     0009 CL T NOCONT NOHOLD COPY 001    READY FORM STANDARD
     0009 TO WINTERS  RDR DIST BIN_048   FLASHC 000 DEST OFF
     0009 FLASH       CHAR        MDFY       0 FCB
     0009 3215    NOEOF OPEN 1131 NOKEEP NOMSG NONAME
Ready;
sp cons close
==:B:==
```

```
Received: from SLACVM by SLACVM.SLAC.STANFORD.EDU (Mailer R2.08 R208004) with
 BSMTP id 6008; Wed, 16 Oct 96 11:53:10 PST
Received: from SERV03.SLAC.STANFORD.EDU by SLACVM.SLAC.STANFORD.EDU
   (IBM VM SMTP V2R1) with TCP; Wed, 16 Oct 96 11:52:45 PST
Received: from DIRECTORY-DAEMON by SERV03.SLAC.STANFORD.EDU
 (PMDF V5.0-6 #10979) id <01IAPK7698C0000BEY@SERV03.SLAC.STANFORD.EDU> for
 jmdeken@SLACVM.SLAC.Stanford.EDU; Wed, 16 Oct 1996 11:51:39 -0800 (PST)
Received: from SLACVM.SLAC.STANFORD.EDU (MAILER@SLACVM)
 by SERV03.SLAC.STANFORD.EDU (PMDF V5.0-6 #10979)
 id <01IAPK4L229C00091J@SERV03.SLAC.STANFORD.EDU> for
 jmdeken@SLAC.Stanford.EDU; Wed, 16 Oct 1996 11:49:38 -0800 (PST)
Received: by SLACVM (Mailer R2.08 R208004) id 5975; Wed,
 16 Oct 1996 11:46:56 -0800 (PST)
Date: Wed, 16 Oct 1996 11:21 -0800 (PST)
From: "Joan M. Winters" <WINTERS@SLAC.Stanford.EDU>
Subject: Old WWW File for the Paper/Docs Book?
To: jmdeken@SLAC.Stanford.EDU
Message-id: <01IAPK4MQM1U00091J@SERV03.SLAC.STANFORD.EDU>
X-Envelope-to: jmdeken@SLACVM.SLAC.Stanford.EDU
Content-transfer-encoding: 7BIT
```

Jean - Look what I found when cleaning up an old VM disk. It's the CERN WWW Home Page and inwards especially following the WWW Project thread. See:

gone

```
FILENAME FILETYPE FM FORMAT LRECL    RECORDS     BLOCKS  DATE      TIME     LABEL
WWW      CONSOLE  M0 V          82      1196         26 02/12/92 16:06:09 JCW400
```

Winters 400

I captured it using the VM spooled console.

SLAC is right there on the CERN home page, along with DESY; but not at FermiLab in sight! Note that this is about ~three months after we figured out the SLAC WWW server first came up.

Cheers, Joan

++ ++ ++ ++ ++ ++ ++ ++ ++   Forwarded Text   ++ ++ ++ ++ ++ ++ ++ ++ ++

```
Ready;
www

                                                      CERN Information

                      CERN INFORMATION - SELECT BY NUMBER

     Help <1>                On this program, or the World-Wide Web project
                             <2>.

     Phone book <3>          People, phone numbers, accounts and email
                             addresses. See also the analytical Yellow Pages
                             <4>, or the same index in French : Pages Jaunes
                             <5>.

     "XFIND" index <6>       Index of computer centre documentation,
                             newsletters, news, help files, etc...

     News <7>                A complete list of all public CERN news groups,
                             such as news from the CERN User's Office <8>, CERN
                             computer center news <9>, student news <10>. See
                             also private groups <11> and Internet news <12>.

FROM OTHER SITES
  See online data by subject <13>, pointers to other forms of online data
```

```
    <14>, and the following specific databases:

        SLAC SPIRES <15>    The High Energy Physics preprint index at Stanford
                            Linear Accelerator, California. (This is the same
                            information avialable via the QSPIRES facility on
                            BITNET. Include the word "FIND" as the first
                            keyword, eg: K FIND AUTHOR FRED.).

        DESY documents <16>
                            Documents and help files from the DESY lab in
                            Hamburg.

        VMS Help <17>       VMS help data now available via a WWW gateway.

        Hacker Jargon <18>  An index to a cross-referenced set of hacker
                            terms. A demonstration of the WWW gateway to the
                            Graz Technical University Hyper-G database.
 <ref.number>, Quit, or Help:
 2

                                               The World Wide Web project

                               WORLD WIDE WEB

    The WorldWideWeb (W3) is a wide-area hypermedia <1> information retrieval
    initiative aiming to give universal access to a large universe of
    documents.
  GENERAL PROJECT INFORMATION
    See also: an executive summary <2> of the project, Mailing lists <3> you
    can join, Policy <4> , latest W3  news <5> .

        Project Status <6>  A list of project components and their current
                            state. (e.g. Line Mode <7> , NeXTStep <8> , Daemon
                            <9> )

        People <10>         A list of people involved in the project.

        History <11>        A summary of the history of the project.

  TECHNICAL DETAILS

        How to provide data <12>
                            How can I make my own data available on the web?

        Protocols <13>      A description of the network protocols used.

        HTML format <14>    A description of the mark-up language used for
                            some documents and for search hit-lists.

        Addressing <15>     The syntax of W3 document addresses.

        Design Issues <16>  A list of decisions to be made when designing or
                            selecting a hypertext/IR system. See also related
                            products <17> .

        Design notes <18>   Notes of meetings, etc, mostly historical.


<ref.number>, Back, <RETURN> for more, Quit, or Help:
       Data sources <19>   Sources of data on the web itself

       News <20>           Some internet/usenet newsgroups of possible
                           interest to the WorldWideWeb project.

       Coding standards <21>
                           A basic style guide for W3 code contributors.

DISTRIBUTION
   See the README <22> file, and copyright <23> notice. Tar files are
   available by anonymous ftp from info.cern.ch  in directory /pub .

     <End>


<ref.number>, Back, Quit, or Help:
2

                                                              Summary -- /WWW

                          WORLDWIDEWEB - SUMMARY

  The WWW <1> project merges the techniques of information retrieval and
  hypertext to make an easy but powerful global information system.

  The project is based on the philosophy that much academic information
  should be freely available to anyone. It aims to allow information sharing
  within internationally dispersed teams, and the dissemination of
  information by support groups.  Originally aimed at the High Energy
  Physics community, it has spread to other areas and attracted much
  interest in user support, resource discovery and collaborative work areas.
 Reader view
  The WWW world consists of documents, and links.  Indexes are special
  documents which, rather than being read, may be searched. The result of
  such a search is another ("virtual") document containing links to the
  documents found.  A simple protocol (" HTTP <2> ") is used to allow a
  browser program to request a keyword search by a remote information
  server.

  The web contains documents in many formats. Those documents which are
  hypertext,  (real or virtual) contain links to other documents, or places
  within documents. All documents, whether real, virtual or indexes, look
  similar to the reader and are contained within the same addressing scheme.

  To follow a link,  a reader clicks with a mouse (or types in a number if
  he or she has no mouse). To search and index, a reader gives keywords (or
  other search criteria). These are the only operations  necessary to access
  the entire world of data.
 Information provider view
  The WWW browsers can access many existing data systems via existing
  protocols (FTP, NNTP) or via HTTP and a gateway. In this way, the critical
  mass of data is quickly exceeded, and the increasing use of the system by
  readers and information suppliers encourage each other.

  Making a web is as simple as writing a few SGML <3> files which point to
  your existing data. Making it public involves running the FTP or HTTP
  daemon <4> , and making at least one link into your web from another. In
<ref.number>, Back, <RETURN> for more, Quit, or Help:
```

daemon <4> , and making at least one link into your web from another. In fact, any file available by anonymous FTP can be immediately linked into a web. The very small start-up effort is designed to allow small contributions. At the other end of the scale, large information providers may provide an HTTP server with full text or keyword indexing. This may allow access to a large existing database without changing the way that database is managed. Such gateways have already been made into Digital's VMS/Help, Technical Univerity of Graz's "Hyper-G", and Thinking Machine's "W.A.I.S." systems.

The WWW model gets over the frustrating incompatibilities of data format between suppliers and reader by allowing negotiation of format between a smart browser and a smart server. This should provide a basis for extension into multimedia, and allow those who share application standards to make full use of them across the web.

This summary does not describe the many exciting possibilities opened up by the WWW project, such as efficient document caching. the reduction of redundant out-of-date copies, and the use of knowledge daemons. There is more information in the online project documentation, including some background on hypertext and many technical notes.

Try it

You can try the simple line mode browser <5> by telnetting to info.cern.ch with user name "www" (no password). You can also find out more about WWW in this way.

It is much more efficient to install the browser on your own machine. The line mode browser is currently available in source form by anonymous FTP from node info.cern.ch [currently 128.141.201.74] as

/pub/WWWLineMode_v.vv.tar.Z.

(v.vv is the version number - take the latest.)

Also available is a hypertext editor <6> for the NeXT using the NeXTStep graphical user interface in file

/pub/WWWNeXTStepEditor_v.vv.tar.Z

```
<ref.number>, Back, <RETURN> for more, Quit, or Help:
list


    HYPERTEXT REFERENCES :=

   [1]       TheProject.html

   [2]       Protocols/HTTP/AsImplemented.html

   [3]       MarkUp/MarkUp.html

   [4]       Daemon/Overview.html

   [5]       LineMode/Browser.html

   [6]       NeXT/WorldWideWeb.html
```

<ref.number>, Back, <RETURN> for more, Quit, or Help:
2

The HTTP Protocol As Implemented In W3

HTTP AS IMPLEMENTED IN WWW

This document defines the Hypertext Transfer protocol (HTTP) as currently implemented by the WorldWideWeb <1> initaitive software. This is a subset of the proposed <2> full HTTP protocol. No client profile information is transferred with the query. Future HTTP protocols will be back-compatible with this protocol.

The protocol uses the normal internet-style telnet protocol style on a TCP-IP link. The following describes how a client acquires a (hypertext) document from an HTTP server, given an HTTP document address <3> .

CONNECTION

The client makes a TCP-IP connection to the host using the domain name <4> or IP number <5> , and the port number <6> given in the address.

During development, the default HTTP TCP port number is 2784 -- this will change when an official port number is allocated.

The server accepts the connection.

Note: HTTP currently runs over TCP, but could run over any connection-oriented service. The interpretation of the protocol below in the case of a sequenced packet service (such as DECnet(TM) or ISO TP4) is that that the request should be one TPDU, but the repose may be many.

REQUEST

The client sends a document request consisting of a line of ASCII characters terminated by a CR LF (carriage return, line feed) pair. A well-behaved server will not require the carriage return character.

This request consists of the word "GET", a space, the document address <7> , omitting the "http:, host and port parts when they are the coordinates just used to make the connection. (If a gateway is being used, then a full document address may be given specifying a different naming scheme).

The search functionality of the protocol lies in the ability of the addressing syntax to describe a search on a named index <8> .

<ref.number>, Back, <RETURN> for more, Quit, or Help:

A search should only be requested by a client when the index document itself has been descibed as an index using the ISINDEX tag <9> .

RESPONSE

The response to a simple GET request is a message in hypertext mark-up language (HTML <10>). This is a byte stream of ASCII characters.

Lines shall be delimited by an optional carriage return followed by a mandatory line feed chararcter. The client should not assume that the carriage return will be present. Lines may be of any length. Well-behaved servers should retrict line length to 80 characters excluding the CR LF pair.

The format of the message is HTML - that is, a trimmed SGML document. Note that this format allows for menus and hit lists to be returned as hypertext. It also allows for plain ASCII text to be returned following the PLAINTEXT tag <11> .

The message is terminated by the closing of the connection by the server.

Well-behaved clients will read the entire document as fast as possible. The client shall not wait for user action (output paging for example) before reading the whole of the document. The server may impose a timeout of the order of 15 seconds on inactivity.

Error responses are supplied in human readable text in HTML syntax. There is no way to distinguish an error response from a satisfactory response except for the content of the text.

DISCONNECTION

The TCP-IP connection is broken by the server when the whole document has been transferred.

The client may abort the transfer by breaking the connection before this, in which case the server will not record any error condidtion.

Requests are idempotent <12> . The server need not store any information about the request after disconnection.

<ref.number>, Back, Quit, or Help:
12

(No title)

HYPERTEXT TRANSFER PROTOCOL

See also: Why a new protocol? <1> , Other protocols used <2>

This is a list of the choices made and features needed in a hypertext transfer protocol. See also the HTTP protocol as currently implemented <3>.

UNDERLYING PROTOCOL

There are various distinct possible bases for the protocol - we can choose

Something based on, and looking like, an Internet protocol. This has the advantage of being well understood, of existing implementations being all over the place. It also leaves open the possibility of a universal FTP/HTTP or NNTP/HTTP server. This is the case for the current HTTP.

Something based on an RPC standard. This has the advantage of making it easy to generate the code, that the parsing of the messages is done automatically, and that the transfer of binary data is efficient. It has the disadvantage that one needs the RPC code to be available on all platforms. One would have to chose one (or more) styles of RPC. Another disadvantage may be that existing RPC systems are not efficient at transferring large quantities of text over a stream protocol unless (like DD-OC-RPC) one has a let-out and can access the socket directly.

Something based on the OSI stack, as is Z39.50. This would have to be run over TCP in the internet world.

Current HTTP uses the first alternative, to make it simple to program, so that it will catch on: conversion to run over an OSI stack will be simple as the structure of the messages is well defined.

IDEMPOTENT ?

Another choice is whether to make the protocol idempotent or not. That is, does the server need to keep any state informat about the client? (For example, the NFS protocol is idempotent, but the FTP and NNTP protocols are not.) In the case of FTP the state information consists of

```
<ref.number>, Back, <RETURN> for more, Quit, or Help:
RDR FILE 1133 SENT FROM MAILER   PUN WAS 6161 RECS 0036 CPY  001 M NOHOLD NOKEEP
15:47:08  * MSG FROM MAILER   : * New mail from
<lee_collins1@gateway.qm.apple.com>
```

are not.) In the case of FTP the state information consists of authorisation, which is not trvial to establish every time but could be, and current directory and transfer mode which are basically trivial. The propsed protocol IS idempotent.

This causes, in principle, a problem when trying to map a non-dempotent system (such as library search systems which stored "result sets" on behalf of the client) into the web. The problem is that to use them in an idempotent way requires the re-evaluation of the intermediate result sets at each query. This can be solved by the gateway intelligently caching result sets for a reasonable time.

REQUEST: INFORMATION TRANSFERRED FROM CLIENT

Parameters below, however represented on the network, are given in upper case, with parameter names in lower case. This set assumes a model of format negociation in which in which the client says what he can take, and the server decides what to give him. One imagines that each function would return a status, as well as information specified below.

When running over a byte stream protocol, SGML would be an encoding possibility (as well as ASN/1 etc).

GET document name — Please transfer a named document back. Transfer the results back in a standard format or one which I have said I can accept. The reply includes the format. In practice, one may want to transfer the document over the same link (a la NNTP) or a different one (a la FTP). There are advantages in each technique. The use of the same link is standard, with moving to a different link by negociation (see PORT <4>).

SEARCH keywords — Please search the given index document for all items with the given word combination, and transfer the results back as marked up hypertext. This could elaborate to an SQL query. There are many advantages in making the search criterion just a subset of the document name space.

```
<ref.number>, Back, <RETURN> for more, Quit, or Help:
list
```

HYPERTEXT REFERENCES :=

[1] WhyHTTP.html

[2] RelevantProtocols.html

[3] HTTP/AsImplemented.html

```
    [4]       HTTP.html#5


 <ref.number>, Back, <RETURN> for more, Quit, or Help:
 3
                                           The HTTP Protocol As Implemented In W3

                              HTTP AS IMPLEMENTED IN WWW

    This document defines the Hypertext Transfer protocol  (HTTP) as currently
    implemented by the WorldWideWeb <1> initaitive software. This is a subset
    of the proposed <2> full HTTP protocol.  No client profile information is
    transferred with the query. Future HTTP protocols will be back-compatible
    with this protocol.

    The protocol  uses the normal internet-style telnet protocol style on a
    TCP-IP link. The following describes how a client acquires a (hypertext)
    document from an HTTP server, given an HTTP document address <3> .
 CONNECTION
    The client makes a TCP-IP connection to the host using the domain name <4>
    or IP number <5> , and the port number <6>  given in the address.

    During development, the default HTTP TCP port number is 2784 -- this will
    change when an official port number is allocated.

    The server accepts the connection.

    Note: HTTP currently runs over TCP, but could run over any
    connection-oriented service.   The interpretation of the protocol below in
    the case of a sequenced packet service (such as DECnet(TM) or ISO TP4) is
    that that the request should be one TPDU, but the repose may be many.
 REQUEST
    The client sends a document request consisting of a line of ASCII
    characters terminated by a CR LF (carriage return, line feed) pair. A
    well-behaved server will not require the carriage return character.

    This request consists of the word "GET", a space, the document address <7>
    , omitting the "http:, host and port parts when they are the coordinates
    just used to make the connection. (If a gateway is being used, then a full
    document address may be given specifying a different naming scheme).

    The search functionality of the protocol lies in the ability of the
    addressing syntax to describe a search on a named index <8> .

 <ref.number>, Back, <RETURN> for more, Quit, or Help:


    A search should only be requested by a client when the index document
    itself has been descibed as an index using the  ISINDEX tag <9> .
 RESPONSE
    The response to a simple GET request is a message in hypertext mark-up
    language ( HTML <10> ). This is a byte stream of ASCII characters.

    Lines shall be delimited by an optional carriage return followed by a
    mandatory line feed chararcter. The client should not assume that the
    carriage return will be present.  Lines may be of any length. Well-behaved
    servers should retrict line length to 80 characters excluding the CR LF
```

pair.

The format of the message is HTML - that is, a trimmed SGML document. Note that this format allows for menus and hit lists to be returned as hypertext. It also allows for plain ASCII text to be returned following the PLAINTEXT tag <11> .

The message is terminated by the closing of the connection by the server.

Well-behaved clients will read the entire document as fast as possible. The client shall not wait for user action (output paging for example) before reading the whole of the document. The server may impose a timeout of the order of 15 seconds on inactivity.

Error responses are supplied in human readable text in HTML syntax. There is no way to distinguish an error response from a satisfactory response except for the content of the text.

DISCONNECTION

The TCP-IP connection is broken by the server when the whole document has been transferred.

The client may abort the transfer by breaking the connection before this, in which case the server will not record any error condidtion.

Requests are idempotent <12> . The server need not store any information about the request after disconnection.

<ref.number>, Back, Quit, or Help:
4

BNF -- /Addressing

W3 ADDRESS SYNTAX: BNF

This is a BNF-like description of the W3 addressing syntax <1> . We use a vertical line "|" to indicate alternatives, and [brackets] to indicate optional parts. Spaces are representational only: no spaces are actually allowed within a W3 address. Single letters stand for single letters. All words of more than one letter below are entites described elsewhere in the syntax description. (Entity names are here linked to their definitions, probably making this difficult to read with the line mode browser.)

An absolute address specified in a link is an anchoraddress <2> . The address which is passed to a server is a docaddress <3> .

anchoraddress	docaddress <4> [# anchor <5>]
docaddress	httpaddress <6> | fileaddress <7> | newsaddress <8> | telnetaddress <9> | gopheraddress <10> | waisaddress <11>
httpaddress	h t t p : / / hostport <12> [/ path <13>] [? search <14>]
fileaddress	f i l e : / / host <15> / path <16>
newsaddress	n e w s : groupart <17>
waisaddress	waisindex | waisdoc

```
     waisindex          w a i s : / / hostport <18> / database <19> [ ?
                        search <20> ]

     waisdoc            w a i s : / / hostport <21> / database <22> /
                        wtype <23> / digits <24> / path <25>

     groupart           * | group <26> | article <27>

<ref.number>, Back, <RETURN> for more, Quit, or Help:
list


      HYPERTEXT REFERENCES :=

     [1]        Addressing.html

     [2]        #47

     [3]        #22

     [4]        #22

     [5]        #20

     [6]        #1

     [7]        #51

     [8]        #92

     [9]        #55

     [10]        #72

     [11]        #105

     [12]        #3

     [13]        #9

     [14]        #11

     [15]        #43

     [16]        #9

     [17]        #90

     [18]        #3

     [19]        #95

     [20]        #11

     [21]        #3

     [22]        #95
     [23]        #102

     [24]        #24

     [25]        #9

     [26]        #87

     [27]        #85

     [28]        #81

     [29]        #87


 <ref.number>, Back, <RETURN> for more, Quit, or Help:
 help


 WWW Line Mode Browser:     COMMANDS AVALIABLE Version 0.14

     You are reading document:
     'http://info.cern.ch/hypertext/WWW/Addressing/BNF.html'

     <RETURN>       Produces the next page of the remaining text.
     TOP            Returns to the first page of the present document.
     LIST           Produces a list of hypertext references
                    which have been accumulated from the text
                    already shown on the screen.
     <number>       Select a referenced document by number
                    (from 1 to 29)
     RECALL         Gives a list of the previous nodes
                    visited.
     RECALL <number>
                    Returns to a previously visited document
                    numbered in the recall list.
     HOME           Returns to the starting node
     BACK           Moves back to the previous node
     Q              Quits the program.

 <ref.number>, Back, <RETURN> for more, Quit, or Help:
 recall


            HISTORY OF PREVIOUS NODES :-


       1)       CERN Information
       2)       The World Wide Web project
       3)       Summary -- /WWW
       4)       The HTTP Protocol As Implemented In W3
       5)       (No title)
       6)       The HTTP Protocol As Implemented In W3
       7)       BNF -- /Addressing


 <ref.number>, Back, <RETURN> for more, Quit, or Help:
 recall 5
```

HYPERTEXT TRANSFER PROTOCOL

See also: Why a new protocol? <1> , Other protocols used <2>

This is a list of the choices made and features needed in a hypertext transfer protocol. See also the HTTP protocol as currently implemented <3>.

UNDERLYING PROTOCOL

There are various distinct possible bases for the protocol - we can choose

Something based on, and looking like, an Internet protocol. This has the advantage of being well understood, of existing implementations being all over the place. It also leaves open the possibility of a universal FTP/HTTP or NNTP/HTTP server. This is the case for the current HTTP.

Something based on an RPC standard. This has the advantage of making it easy to generate the code, that the parsing of the messages is done automatically, and that the transfer of binary data is efficient. It has the disadvantage that one needs the RPC code to be available on all platforms. One would have to chose one (or more) styles of RPC. Another disadvantage may be that existing RPC systems are not efficient at transferring large quantities of text over a stream protocol unless (like DD-OC-RPC) one has a let-out and can access the socket directly.

Something based on the OSI stack, as is Z39.50. This would have to be run over TCP in the internet world.

Current HTTP uses the first alternative, to make it simple to program, so that it will catch on: conversion to run over an OSI stack will be simple as the structure of the messages is well defined.

IDEMPOTENT ?

Another choice is whether to make the protocol idempotent or not. That is, does the server need to keep any state informat about the client? (For example, the NFS protocol is idempotent, but the FTP and NNTP protocols are not.) In the case of FTP the state information consists of

<ref.number>, Back, <RETURN> for more, Quit, or Help:

are not.) In the case of FTP the state information consists of authorisation, which is not trvial to establish every time but could be, and current directory and transfer mode which are basically trivial. The propsed protocol IS idempotent.

This causes, in principle, a problem when trying to map a non-dempotent system (such as library search systems which stored "result sets" on behalf of the client) into the web. The problem is that to use them in an idempotent way requires the re-evaluation of the intermediate result sets at each query. This can be solved by the gateway intelligently caching result sets for a reasonable time.

REQUEST: INFORMATION TRANSFERRED FROM CLIENT

Parameters below, however represented on the network, are given in upper case, with parameter names in lower case. This set assumes a model of format negociation in which in which the client says what he can take, and the server decides what to give him. One imagines that each function would return a status, as well as information specified below.

When running over a byte stream protocol, SGML would be an encoding possibility (as well as ASN/1 etc).

GET document name Please transfer a named document back. Transfer the results back in a standard format or one which I have said I can accept. The reply includes the format. In practice, one may want to transfer the document over the same link (a la NNTP) or a different one (a la FTP). There are advantages in each technique. The use of the same link is standard, with moving to a different link by negociation (see PORT <4>).

SEARCH keywords Please search the given index document for all items with the given word combination, and transfer the results back as marked up hypertext. This could elaborate to an SQL query. There are many advantages in making the search criterion just a subset of the document name space.

```
<ref.number>, Back, <RETURN> for more, Quit, or Help:
4
```

(No title)

HYPERTEXT TRANSFER PROTOCOL

See also: Why a new protocol? <1> , Other protocols used <2>

This is a list of the choices made and features needed in a hypertext transfer protocol. See also the HTTP protocol as currently implemented <3>.

UNDERLYING PROTOCOL

There are various distinct possible bases for the protocol - we can choose

Something based on, and looking like, an Internet protocol. This has the advantage of being well understood, of existing implementations being all over the place. It also leaves open the possibility of a universal FTP/HTTP or NNTP/HTTP server. This is the case for the current HTTP.

Something based on an RPC standard. This has the advantage of making it easy to generate the code, that the parsing of the messages is done automatically, and that the transfer of binary data is efficient. It has the disadvantage that one needs the RPC code to be available on all platforms. One would have to chose one (or more) styles of RPC. Another disadvantage may be that existing RPC systems are not efficient at transferring large quantities of text over a stream protocol unless (like DD-OC-RPC) one has a let-out and can access the socket directly.

Something based on the OSI stack, as is Z39.50. This would have to be run over TCP in the internet world.

Current HTTP uses the first alternative, to make it simple to program, so that it will catch on: conversion to run over an OSI stack will be simple as the structure of the messages is well defined.

IDEMPOTENT ?

Another choice is whether to make the protocol idempotent or not. That is, does the server need to keep any state informat about the client? (For example, the NFS protocol is idempotent, but the FTP and NNTP protocols

are not.) In the case of FTP the state information consists of
<ref.number>, Back, <RETURN> for more, Quit, or Help:

are not.) In the case of FTP the state information consists of authorisation, which is not trvial to establish every time but could be, and current directory and transfer mode which are basically trivial. The propsed protocol IS idempotent.

This causes, in principle, a problem when trying to map a non-dempotent system (such as library search systems which stored "result sets" on behalf of the client) into the web. The problem is that to use them in an idempotent way requires the re-evaluation of the intermediate result sets at each query. This can be solved by the gateway intelligently caching result sets for a reasonable time.

REQUEST: INFORMATION TRANSFERRED FROM CLIENT

Parameters below, however represented on the network, are given in upper case, with parameter names in lower case. This set assumes a model of format negociation in which in which the client says what he can take, and the server decides what to give him. One imagines that each function would return a status, as well as information specified below.

When running over a byte stream protocol, SGML would be an encoding possibility (as well as ASN/1 etc).

GET document name — Please transfer a named document back. Transfer the results back in a standard format or one which I have said I can accept. The reply includes the format. In practice, one may want to transfer the document over the same link (a la NNTP) or a different one (a la FTP). There are advantages in each technique. The use of the same link is standard, with moving to a different link by negociation (see PORT <4>).

SEARCH keywords — Please search the given index document for all items with the given word combination, and transfer the results back as marked up hypertext. This could elaborate to an SQL query. There are many advantages in making the search criterion just a subset of the document name space.

<ref.number>, Back, <RETURN> for more, Quit, or Help:
4

(No title)

HYPERTEXT TRANSFER PROTOCOL

See also: Why a new protocol? <1> , Other protocols used <2>

This is a list of the choices made and features needed in a hypertext transfer protocol. See also the HTTP protocol as currently implemented <3>.

UNDERLYING PROTOCOL

There are various distinct possible bases for the protocol - we can choose

Something based on, and looking like, an Internet protocol. This has the advantage of being well understood, of existing implementations being all over the place. It also leaves open the possibility of a universal FTP/HTTP or NNTP/HTTP server. This is the case for the current HTTP.

Something based on an RPC standard. This has the advantage of making it easy to generate the code, that the parsing of the messages is done automatically, and that the transfer of binary data is efficient. It has the disadvantage that one needs the RPC code to be available on all platforms. One would have to chose one (or more) styles of RPC. Another disadvantage may be that existing RPC systems are not efficient at transferring large quantities of text over a stream protocol unless (like DD-OC-RPC) one has a let-out and can access the socket directly.

Something based on the OSI stack, as is Z39.50. This would have to be run over TCP in the internet world.

Current HTTP uses the first alternative, to make it simple to program, so that it will catch on: conversion to run over an OSI stack will be simple as the structure of the messages is well defined.

IDEMPOTENT ?

Another choice is whether to make the protocol idempotent or not. That is, does the server need to keep any state informat about the client? (For example, the NFS protocol is idempotent, but the FTP and NNTP protocols are not.) In the case of FTP the state information consists of

<ref.number>, Back, <RETURN> for more, Quit, or Help:
2

Protocol Specifications relevant to HyperText

RELEVANT PROTOCOLS

The WorldWideWeb <1> system can pick up information from many information sources, using existing protocols. Among these are file and news transfer protocols.

FILE TRANSFER

The file transfer protocol currently most used for accessing fairly stable public information over a wide area is "Anonymous FTP". This means the use of the internet File Transfer Protocol without authentication. As the WWW project currently operates for the sake of public information, anonymous FTP is quite appropriate, and WWW can pick up any information provided by anonymous FTP. FTP is defined in RFC 959 <2> which includes material from many previous RFCs. (See also: file address syntax <3>). See also the prospero project and the shift <4> project, for more powerful file access systems.

NETWORK NEWS

The "Network News Transfer Protocol" (NNTP) is defined in RFC 977 <5> by Kantor and Lampsley. This allows transient news information in the USENET news format to be exchanged over the internet. The format of news articles is defined in RFC 850, Standard for Interchange of USENET Messages <6> by Mark Horton. This in turn refers to the standard RFC 822 <7> which defines the format of internet mail messages. (See news address syntax <8>)

SEARCH AND RETRIEVE

The WWW project defines its own protocol for information transfer, which allows for negotiation on representation. This we call HTTP, for HyperText Transfer Protocol <9> .See also HyperText Transfer Format <10> , and the HTTP address syntax) <11>

Whilst the HTTP <12> protocol provides an index search function, another common protocol for index search is Z39.50, and the version of it used by the WAIS <13> project. (See also the WAIS-WWW gateway <14>).

<End>

```
<ref.number>, Back, Quit, or Help:
9

                                   The HTTP Protocol As Implemented In W3

                         HTTP AS IMPLEMENTED IN WWW

   This document defines the Hypertext Transfer protocol  (HTTP) as currently
   implemented by the WorldWideWeb <1> initaitive software. This is a subset
   of the proposed <2> full HTTP protocol.  No client profile information is
   transferred with the query. Future HTTP protocols will be back-compatible
   with this protocol.

   The protocol  uses the normal internet-style telnet protocol style on a
   TCP-IP link. The following describes how a client acquires a (hypertext)
   document from an HTTP server, given an HTTP document address <3> .
 CONNECTION
   The client makes a TCP-IP connection to the host using the domain name <4>
   or IP number <5> , and the port number <6>  given in the address.

   During development, the default HTTP TCP port number is 2784 -- this will
   change when an official port number is allocated.

   The server accepts the connection.

   Note: HTTP currently runs over TCP, but could run over any
   connection-oriented service.   The interpretation of the protocol below in
   the case of a sequenced packet service (such as DECnet(TM) or ISO TP4) is
   that that the request should be one TPDU, but the repose may be many.
 REQUEST
   The client sends a document request consisting of a line of ASCII
   characters terminated by a CR LF (carriage return, line feed) pair. A
   well-behaved server will not require the carriage return character.

   This request consists of the word "GET", a space, the document address <7>
   , omitting the "http:, host and port parts when they are the coordinates
   just used to make the connection. (If a gateway is being used, then a full
   document address may be given specifying a different naming scheme).

   The search functionality of the protocol lies in the ability of the
   addressing syntax to describe a search on a named index <8> .

<ref.number>, Back, <RETURN> for more, Quit, or Help:

   A search should only be requested by a client when the index document
   itself has been descibed as an index using the  ISINDEX tag <9> .
 RESPONSE
   The response to a simple GET request is a message in hypertext mark-up
   language ( HTML <10> ). This is a byte stream of ASCII characters.

   Lines shall be delimited by an optional carriage return followed by a
   mandatory line feed chararcter. The client should not assume that the
   carriage return will be present.  Lines may be of any length. Well-behaved
   servers should retrict line length to 80 characters excluding the CR LF
   pair.

   The format of the message is HTML - that is, a trimmed SGML document. Note
   that this format allows for menus and hit lists to be returned as
   hypertext. It also allows for plain ASCII text to be returned following
   the  PLAINTEXT tag <11> .

   The message is terminated by  the closing of the connection by the server.

   Well-behaved clients will read the entire document as fast as possible.
   The client shall not wait for user action (output paging for example)
   before reading the whole of the document.  The server may impose a timeout
   of the order of 15 seconds on inactivity.

   Error responses are supplied in human readable text in HTML syntax. There
   is no way to distinguish an error response from a satisfactory response
   except for the content of the text.
 DISCONNECTION
   The TCP-IP connection is broken by the server when the whole document has
   been transferred.

   The client may abort the transfer by breaking the connection before this,
   in which case the server will not record any error condidtion.

   Requests are idempotent <12> .  The server need not store any information
   about the request after disconnection.


<ref.number>, Back, Quit, or Help:
recall


             HISTORY OF PREVIOUS NODES :-


       1)       CERN Information
       2)       The World Wide Web project
       3)       Summary -- /WWW
       4)       The HTTP Protocol As Implemented In W3
       5)       (No title)
       6)       The HTTP Protocol As Implemented In W3
       7)       BNF -- /Addressing
       8)       (No title)
       9)       Protocol Specifications relevant to HyperText
      10)       The HTTP Protocol As Implemented In W3



<ref.number>, Back, Quit, or Help:
recall 9
                              Protocol Specifications relevant to HyperText

                           RELEVANT PROTOCOLS

   The WorldWideWeb <1> system can pick up information from many information
   sources, using existing protocols. Among these are file and news transfer
   protocols.
FILE TRANSFER
   The file transfer protocol currently most used for accessing fairly stable
   public information over a wide area is "Anonymous FTP". This means the use
```

```
    of the internet File Transfer Protocol without authentication. As the WWW
    project currently operates for the sake of public information, anonymous
    FTP is quite appropriate, and WWW can pick up any information provided by
    anonymous FTP. FTP is defined in RFC 959 <2> which includes material from
    many previous RFCs. (See also:  file address syntax <3> ). See also the
    prospero project and the shift <4> project, for more powerful file access
    systems.
 NETWORK NEWS
    The "Network News Transfer Protocol" (NNTP) is defined in RFC 977 <5> by
    Kantor and Lampsley. This allows transient news information in the USENET
    news format to be exchanged over the internet. The format of news articles
    is defined in RFC 850, Standard for Interchange of USENET Messages <6> by
    Mark Horton. This in turn refers to the standard RFC 822 <7> which defines
    the format of internet mail messages. (See news address syntax <8> )
 SEARCH AND RETRIEVE
    The WWW project defines its own protocol for information transfer, which
    allows for negotiation on representation. This we call HTTP, for HyperText
    Transfer Protocol <9> .See also  HyperText Transfer Format <10> , and the
    HTTP address syntax ) <11>
    Whilst the HTTP <12> protocol provides an index search function, another
    common protocol for index search is Z39.50, and the version of it used by
    the WAIS <13> project.  (See also the WAIS-WWW gateway <14> ).

      <End>




<ref.number>, Back, Quit, or Help:
10

                                                  HyperText Mark-up Language

                                   HTML

    The WWW <1> system uses marked-up text to represent a hypertext document
    for transmision over the network. The hypertext mark-up language is an
    SGML <2> format. This defines the basic syntax used. The particular
    language, the set of tags and the rules about their use, and their
    significance is not part of the SGML standard. There being no standard on
    this, we have adopted a set which seems sensible. We call them HTML --
    hypertext markup language. HTML is not an alternative to SGML, it is a
    particular format within the SGML rules (an SGML "DTD"). HTML parsers
    should ignore tags which they do not understand, and ignore attributes
    which they do not understand of tags which they do understand.

   See also:

       The tags <3>          A list of the tags used in HTML with their
                             significance.

       Example <4>           A file containing a variety of tags used for test
                             purposes, and its source text <5>.

 DEFAULT TEXT
    Unless otherwise defined by tags, text is transmitted as a stream of
    lines. The division of the stream of characters into lines is arbitrary,
    and only made in order to allow the text to be passed through systems
    which can only handle text with a limited line length. The recommended
    line length for transmission is 80 characters. The division into lines has
    no significance (except in the case of  example sections <6> and PLAINTEXT
    <7> ) apart from indicating a word end. Line breaks between tags have no
    significance.

      <End>




 <ref.number>, Back, Quit, or Help:
 4

                                             Hypertext HTML formatting example

                                 TEST DATASET

    This is an example bit of hypertext - compare  the formatted version with
    the original HTML source <1>. Let's try introducing an initial paragrpah
    between the H1 and the H2 headings.

    This file was typed into WWW (on the NeXT) and saved.
 INTRODUCTION
    This file contains a test set of HTML mark-up, as a test of hypertext
    browsers and an example of the syntax of the tags. See also:

        An arbitrary news article <2>

        The newsgroup on hypertext <3>

        More details about the WWW project. <4>

    That is the end of the list.
 SOME ANCHORS
    Anchors come in two forms: whole nodes or parts of nodes. The line mode
    browser can't currently (Nov 91)  jump to a part of a node: it always
    jumps to the top.
  Leading to whole nodes
    Here is an anchor which leads to the VM FIND command. <5> Note the nested
    highlighting (hp2) within the anchor. If you want to click on this, <6>
    you will go to the system default page.
  Leading to anchors within nodes
    Now THIS <7> leads to anchor #2 in this file, and if you want to click on
    THIS, <8> you should go to the system default page with anchor #2
    selected. Now the word "destination" is a named destination anchor,
    connected to the word "source" <9> . Clicking on the destination shouldn't
    do anything. Selecting the source should lead to the destination.

    Now let's go through the limited set of markup tags which we accept. The
    title, "Hypertext HTML formatting example" was between TITLE tags. "Test
    Dataset" at the top of this page was a Level One Heading (H1). The quick


list
recall

?????
```

```
quit
hx

hx
CMS

CMS
jkljkl
SCSABN: invalid FCB (   6) at X'0000CB', processing terminated
SCSABN: recursive abend during abend recovery
HCPGIR450W CP entered; disabled wait PSW 00060000 15E4E3E2
IPL CMS
SLAC XA CMS 5.6 11/01/91 15:32


TEST disk for testing files prior to installing on the U or Y disks.

CMS29E ( 59C R ) RO
CMS5A5 ( 1A7 P ) RO

660 USERS, 295 LOGGED, 365 DISC, 0 DIALED, 0 NET
AVGPROC-100% 03
XSTORE-000320/SEC MIGRATE-0006/SEC
XSTORE MDC READS-000169/SEC WRITES-000009/SEC HIT RATIO-099%
STORAGE-031% PAGING-0020/SEC STEAL-000%
Q0-00000(00000)                           DORMANT-00642
Q1-00003(00000)           E1-00000(00000)
Q2-00002(00000) EXPAN-002 E2-00000(00000)
Q3-00017(00000) EXPAN-002 E3-00000(00000)
PROC 0001-100%          PROC 0000-100%
PROC 0002-100%


Conference LEPNEWS has 45 new items.
Conference PUBEXECS has 73 new items.
Conference XA-ASM has 8 new items.
Conference XA-FORT has 15 new items.

FILES: 0058 RDR, 0001 PRT,   NO PUN
Ready;
vmstatus restore
Disk S(190) could not be accessed
Ready;
qdiskid
Label     CCU  Mode Stat Owner     Vaddr

TMP193    193  A    R/W  Tdisk     -
JCW198    198  B    R/W  WINTERS   0198
JCW192    192  D    R/W  WINTERS   0192
BEB191    195  E    R/O  BEBO      0191
JCW194    194  J    R/W  WINTERS   0194
JCW400    400  M    R/W  WINTERS   0400
JCW191    191  N    R/W  WINTERS   0191
MAIL      1A6  O    R/O  RENATA    0195
CMS5A5    1A7  P    R/O  XAMAINT   05A5
CMS29E    59C  R    R/O  MAINT     028E
MNT190    190  S    R/O  XAMAINT   0190
CMS59E    59E  T    R/O  $TDISK    049E
CMS59F    59F  U    R/O  $UDISK    049F
CMS19E    19E  Y/S  R/O  $YDISK    029E
```

```
JCW223    223  Z    R/W  WINTERS   0223
Ready;
l
CMS       PRESERVE A1
FAQ       MAIL     A0
Ready;
qdiskid
Label     CCU  Mode Stat Owner     Vaddr

TMP193    193  A    R/W  Tdisk     -
JCW198    198  B    R/W  WINTERS   0198
JCW192    192  D    R/W  WINTERS   0192
BEB191    195  E    R/O  BEBO      0191
JCW194    194  J    R/W  WINTERS   0194
JCW400    400  M    R/W  WINTERS   0400
JCW191    191  N    R/W  WINTERS   0191
MAIL      1A6  O    R/O  RENATA    0195
CMS5A5    1A7  P    R/O  XAMAINT   05A5
CMS29E    59C  R    R/O  MAINT     028E
MNT190    190  S    R/O  XAMAINT   0190
CMS59E    59E  T    R/O  $TDISK    049E
CMS59F    59F  U    R/O  $UDISK    049F
CMS19E    19E  Y/S  R/O  $YDISK    029E
JCW223    223  Z    R/W  WINTERS   0223
Ready;
q cons
CONS 0009 ON LDEV L041F   TERM START
     0009 CL T NOCONT NOHOLD COPY 001    READY FORM STANDARD
     0009 TO WINTERS  RDR DIST BIN_048   FLASHC 000 DEST OFF
     0009 FLASH       CHAR       MDFY       0 FCB
     0009 3215   NOEOF OPEN 1131 NOKEEP NOMSG NONAME
Ready;
sp cons close
```

[illegible] Annotated

```
*****************************************************************************************************************************************
**  VM/ESA    R1.1 SERVICE LEVEL 0000 ************** 11/20/96 ** 09:33:16 ** PRINTER 042E **** CPU SERIAL  74657  CPU MODEL  9021 **

BIN_048   WINTERS   FILE NAME/TYPE=     WWWTEST   LASTNEWS     ORIGINID= WINTERS                VV        VV MM        MM    SLAC
BIN_048   WINTERS                                                                               VV        VV MMM      MMM  Computing
BIN_048   WINTERS   CREATION DATE/TIME= 11/19/96  19:39:00     SYSID=    SLACVM                 VV        VV MM M    M MM   Services
BIN_048   WINTERS                                                                               VV        VV MM  M  M  MM
BIN_048   WINTERS   COPY=   001   RECS= 0719    333333333   8888888    6666666                   VV      VV  MM   M M   MM
BIN_048   WINTERS                               333333333  888888888  666666666                   VV    VV   MM    M    MM
BIN_048   WINTERS   CLASS=  A     SPID= 6685    33      33 88      88 66      66                   VV  VV    MM         MM
BIN_048   WINTERS                                       33 88      88 66                    EEEEEEEEEEEEEV VSSSSSSSSSSS      MAAAAA
BIN_048   WINTERS   FORM=   STANDARD                  3333  8888888   66666666                 EE           VVSS     MM  SS      AA   AA
BIN_048   WINTERS                                   333333  8888888   666666666                EE            VSS     MM         AAM    AA
BIN_048   WINTERS   DEST=   OFF                         33 88      88 66      66               EE             SS               AA       AA
BIN_:48   WINTERS                               33      33 88      88 66      66               EEEEEEEEEEE     SSSSSSSSSSS   AAAAAAAAAAAAA
BIN_^48   WINTERS   CHAR=   TN5T                333333333  888888888  666666666                EE                       SS   AA          AA
                                                 3333333    8888888    6666666                 EE                       SS   AA          AA
                                                                                               EE             SS        SS   AA          AA
                                                                                               EEEEEEEEEEEEE   SSSSSSSSSSS   AA          AA


:AG DATA:


WW       WW   IIIIII    N       NN  TTTTTTTT    EEEEEEEEE  RRRRRRRR    SSSSSSS
WW       WW   IIIIII    NN      NN  TTTTTTTT    EEEEEEEEE  RRRRRRRRR  SSSSSSSSS
WW       WW     II      NNN     NN     TT       EE         RR     RR  SS     SS
WW       WW     II      NNNN    NN     TT       EE         RR     RR  SS
WW       WW     II      NN NN   NN     TT       EEEEEE     RRRRRRRRR  SSSSSSSS
WW       WW     II      NN  NN  NN     TT       EEEEEE     RRRRRRRR    SSSSSSSS
WW   W   WW     II      NN   NNNN      TT       EE         RR  RR            SS
WW  WWW  WW     II      NN    NNN      TT       EE         RR   RR    SS     SS
 WWW WWW      IIIIII    NN      NN     TT       EEEEEEEEE  RR    RR   SSSSSSSSS
  W   W       IIIIII    NN       N     TT       EEEEEEEEE  RR     RR   SSSSSSS

WW       WW  WW       WW  WW       WW  TTTTTTTT    EEEEEEEEE   SSSSSSS   TTTTTTTT
WW       WW  WW       WW  WW       WW  TTTTTTTT    EEEEEEEEE  SSSSSSSSS  TTTTTTTT
WW       WW  WW       WW  WW       WW     TT       EE         SS     SS     TT
WW       WW  WW       WW  WW       WW     TT       EE         SS            TT
WW       WW  WW       WW  WW       WW     TT       EEEEEE     SSSSSSSS      TT
WW       WW  WW       WW  WW       WW     TT       EEEEEE      SSSSSSSS     TT
WW   W   WW  WW   W   WW  WW   W   WW     TT       EE                SS     TT
WW  WWW  WW  WW  WWW  WW  WW  WWW  WW     TT       EE         SS     SS     TT
 WWW WWW      WWW WWW      WWW WWW        TT       EEEEEEEEE  SSSSSSSSS     TT
  W   W        W   W        W   W         TT       EEEEEEEEE   SSSSSSS      TT

_BBBBBBBB     IIIIII    N       NN                  00000          4     8888888
 BBBBBBBBB    IIIIII    NN      NN                 0000000        44    888888888
 BB      BB     II      NNN     NN                00     00      444    88     88
 BB      BB     II      NNNN    NN                00     00     4444    88     88
 BBBBBBBB       II      NN NN   NN                00     00    44 44     8888888
 BBBBBBBB       II      NN  NN  NN                00     00   44  44     8888888
 BB      BB     II      NN   NNNN                 00     00  444444444  88     88
 BB      BB     II      NN    NNN                 00     00  444444444  88     88
_BBBBBBBBB    IIIIII    NN      NN                 0000000        44    888888888
-BBBBBBBB     IIIIII    NN       N                  00000         44     8888888
```

Oldest
a backup of WWWTEST 191
on tape RL0631   4/7/92   11:18:37:tw