SCREENPRINTS 1992/SLACVM/

SLAC ARCHIVES COLL. OC-072
SERIES 2 SUBSERIES 2
BOX 7 FOLDER 1

1992

Index of /archive/1992/SLACVM

```
     Name                    Last modified       Size  Description
------------------------------------------------------------------
     Parent Directory        31-Jan-1997 09:49     -
     spicell/                22-Jul-1998 02:16     -
     www.console             12-Feb-1992 08:06    49k
     www.console.readme      12-Aug-1997 17:05     2k
     www/                    25-Jan-1997 22:47     -
     wwwtest/                25-Jan-1997 22:47     -
------------------------------------------------------------------
```

Apache/1.3.12 Server at www.slac.stanford.edu Port 80

About www.console File

This file, WWW CONSOLE, is a spooled console that Joan M. Winters captured during her very early use of the SLACVM WWW linemode browser, presumably on 12 Feb 1992. The file has been on a winters minidisk since then, possibly all that time on the winters 400 minidisk. The WWW CONSOLE file was copied 12 Aug 1997 to the UNIX AFS production Web space at /afs/slac/www/archive/1992/SLACVM/www.console to preserve it after SLACVM goes away.

The spooled console shows possibly her first use of WWW. The SLACVM WWW command goes to the CERN home page, not to the SLAC Home Page (which apparently did not yet exist). SLAC SPIRES is link number 15 on the CERN home page. In addition, she explored various parts of the CERN Web that described WWW and tried out WWW browser subcommands including HELP.

The WWW session terminated with an abend. Some aspects of her VM/the SLAC VM environment standard at the time were shown upon re-IPL. It is likely that the WWW invoking EXEC resided on the BEBO 191 minidisk (belonging to Bebo White).

Output from XSHOW LISTFILE WWW CONSOLE * command issued on SLACVM 12 Aug 1997 showing attributes of the file follows:

```
FILENAME FILETYPE FM FORMAT LRECL      RECS     BLOCKS  DATE     TIME     LABEL
WWW      CONSOLE  M0 V         82      1196         26  2/12/92 16:06:09 JCW400
```

From the file 59F HIST1992 showing the installation history on the standard system minidisk 59F, it seems that Bebo first installed the WWW invoking EXEC, browser, and HELP file in March. Here's the extract:

```
26Mar92 14:58:56 BEBO     OWN     WWW     *
26Mar92 14:59:40 BEBO     ADD     WWW     EXEC     A  BEBO     0191
  Interface exec to WWW
26Mar92 15:00:21 BEBO     ADD     WWW     MODULE   A  BEBO     0191
  WWW browser (with CERN front page)
26Mar92 15:01:08 BEBO     ADD     WWW     HELPCMS  A  BEBO     0191
  help file for WWW
```

By Joan M. Winters
12 Aug 1997

```
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 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
```

About www.console File

This file, WWW CONSOLE, is a spooled console that Joan M. Winters captured during her very early use of the SLACVM WWW linemode browser, presumably on 12 Feb 1992. The file has been on a winters minidisk since then, possibly all that time on the winters 400 minidisk. The WWW CONSOLE file was copied 12 Aug 1997 to the UNIX AFS production Web space at /afs/slac/www/archive/1992/SLACVM/www.console to preserve it after SLACVM goes away.

The spooled console shows possibly her first use of WWW. The SLACVM WWW command goes to the CERN home page, not to the SLAC Home Page (which apparently did not yet exist). SLAC SPIRES is link number 15 on the CERN home page. In addition, she explored various parts of the CERN Web that described WWW and tried out WWW browser subcommands including HELP.

The WWW session terminated with an abend. Some aspects of her VM/the SLAC VM environment standard at the time were shown upon re-IPL. It is likely that the WWW invoking EXEC resided on the BEBO 191 minidisk (belonging to Bebo White).

Output from XSHOW LISTFILE WWW CONSOLE * command issued on SLACVM 12 Aug 1997 showing attributes of the file follows:

```
FILENAME FILETYPE FM FORMAT LRECL       RECS     BLOCKS   DATE      TIME     LABEL
WWW      CONSOLE  M0 V         82       1196         26  2/12/92  16:06:09  JCW400
```

From the file 59F HIST1992 showing the installation history on the standard system minidisk 59F, it seems that Bebo first installed the WWW invoking EXEC, browser, and HELP file in March. Here's the extract:

```
26Mar92 14:58:56 BEBO    OWN     WWW     *
26Mar92 14:59:40 BEBO    ADD     WWW     EXEC     A  BEBO     0191
  Interface exec to WWW
26Mar92 15:00:21 BEBO    ADD     WWW     MODULE   A  BEBO     0191
  WWW browser (with CERN front page)
26Mar92 15:01:08 BEBO    ADD     WWW     HELPCMS  A  BEBO     0191
  help file for WWW
```

By Joan M. Winters
12 Aug 1997

SCREENPRINTS 1992 / SLACVM / [illegible] / [illegible]

Index of /archive/1992

Name	Last modified	Size	Description
Parent Directory	25-Oct-2000 21:14	-	
SLACVM/	22-Jul-1998 01:46	-	

Apache/1.3.12 Server at www.slac.stanford.edu Port 80

Index of /archive/1992/SLACVM

Name	Last modified	Size	Description
Parent Directory	31-Jan-1997 09:49	-	
spicell/	22-Jul-1998 02:16	-	
www.console	12-Feb-1992 08:06	49k	
www.console.readme	12-Aug-1997 17:05	2k	
www/	25-Jan-1997 22:47	-	
wwwtest/	25-Jan-1997 22:47	-	

Apache/1.3.12 Server at www.slac.stanford.edu Port 80

Index of /archive/1992/SLACVM/spicell

Name	Last modified	Size	Description
Parent Directory	22-Jul-1998 01:46	-	
191/	22-Jul-1998 01:53	-	
192/	22-Jul-1998 03:12	-	

Apache/1.3.12 Server at www.slac.stanford.edu Port 80

Index of /archive/1992/SLACVM/spicell/191

```
     Name                    Last modified       Size  Description
--------------------------------------------------------------------
     Parent Directory        22-Jul-1998 02:16      -
     rl0498/                 22-Jul-1998 01:51      -
     unix-commands           22-Jul-1998 01:53     1k
--------------------------------------------------------------------
```

Apache/1.3.12 Server at www.slac.stanford.edu Port 80

Index of /archive/1992/SLACVM/spicell/191/rl0498

```
    Name                    Last modified       Size  Description
-----------------------------------------------------------------
    Parent Directory        22-Jul-1998 01:53      -
    README                  21-Jul-1998 18:44    23k
    lasting.globalv         30-May-1990 02:11     1k
    load.map                12-Dec-1991 10:03     1k
    profile.exec            12-Dec-1991 10:06     2k
    profile.xedit           30-Nov-1989 06:49     7k
    spice191.backup         21-Jul-1998 18:28     2k
    spicell.conslast        11-Dec-1991 16:34     1k
    spicell.lastnews        11-Dec-1991 16:33     1k
    spicell.netlog          20-Dec-1991 03:30     1k
    winters.filelist        21-Jul-1998 18:49     1k
    winters.jcwl19          21-Jul-1998 18:44     9k
    winters.ls              22-Jul-1998 01:51     1k
-----------------------------------------------------------------
```

Apache/1.3.12 Server at www.slac.stanford.edu Port 80

File /afs/slac.stanford.edu/www/archive/1992/SLACVM/spicell/191/rl0498/README

On 22 Jul 1998 restored currently existing backup copy #16 (on tape cartridge RL0498) of userid SPICELL minidisk 191 via the CMS RECOVER command to WINTERS virtual reader, and from there to WINTERS 294 (a SPACE TEMPDISK) using a "RECEIVE / (OLDDATE" command within RDRLIST. The VM backup tape was taken 1/7/92.

Then copied all the RECOVERed files plus five files on the recovery and migration process from SLACVM to UNIX. Used an NFS mount?? in UNIX of the SLACVM WINTERS 294 minidisk, followed by a UNIX copy of all those files to the subdirectory /afs/slac.stanford.edu/www/archive/1992/SLACVM/spicell/191/rl0498. (A summary of UNIX commands is in file unix-commands.)

SPICELL 191 may have been used in very early development of the experimental WWW system at SLAC. SPICELL 191 is not an "INSTALL" disk.

More specifically, from the CMS RECOVER list of backup tapes on 7/22/98 selected:

```
16 ) SPICELL 0191 dumped to file 90 of tape RL0498 on 01/07/92 at 19:32:12
```

See file spice191.backup for the entire list of currently existing backups (created from XSHOW capture of RECOVER command output).

RECOVER submitted VM batch job JCW119 from userid WINTERS on 7/22/98. The job was run shortly afterwards. Its batch console output is in file winters.jcw119.

There were a total of ?? files on the SLACVM SPICELL 191 minidisk of 1/7/92. In inverse chronological order since last update, here are the files (from "FILELIST (STATS" command output*):

```
    Filename Filetype Fm Format LRECL    Records     Blocks   Date     Time

???
```

* As defined in "HELP CMS FILELIST", the STATS option produces the following output:

```
    STAts
        lists the following information about the specified files:

          o File identifier or directory identifier
          o Format and logical record length of the file
          o Number of records and number of blocks in the file
          o Date and time of last update.

        See the "Examples" for a sample display using the STATS option.
```

To find fairly often the author(s) of one of these executable files:

o Look in comments near the top (or bottom) for an EXEC's "owner", the person(s) with primary responsibility for the file at the time the file was created or most recently updated before this backup was made.

For a list of all the files recovered plus three files created related to the recovery plus a record of the WINTERS CMS environment in which the recovery occurred (in file CMS PRESERVE), see winters.filelist (created by SAVEing the output of a FILELIST command). A list of all the files on the backup tape (which seems to be the same?? as all the files recovered) is in the batch job output winters.jcwl19.

For a list of all the files moved to UNIX (which should include all the files in the winters.filelist file), see winters.ls (which also lists the unix-commands file, plus).

In summary, for more details about the recovery from VM backup and move from SLACVM to SLAC UNIX, see the following files:

```
README  [created in SLACVM about entire migration, named README RL0498S]
   cms.preserve

   spice191.backup   ??
   winters.jcwl19
   winters.filelist

   unix-commands  [created in UNIX]
      winters.ls
```

See also files in subdirectory /afs/slac/www/archive/SLACVM for generally applicable information.

22 Jul 1998   Joan M. Winters

```
GONE    †ACTIVE—INACTIVE
```

```
DMSLIO201W The following names are undefined:
IBMBLIIA
```

```
/* SLAC new-user PROFILE EXEC                                        */
/* This file is installed as PROFILE EXEC on the 191 disk of each   */
/* new VM account.  Executed at logon, commands in the PROFILE      */
/* help establish an initial working environment for the account.   */
/*                                                                   */
/* DON'T ERASE THIS FILE.  If the commands below don't meet your    */
/* needs, the file may be edited and replaced.                       */
/*                                                                   */
/* Master copy of new-user PROFILE maintained by VM userid           */
/* BILLIE, SCS Computing Services                                    */
/* Revision History: 0  11 March   1988, first working version       */
/*                   1  23 October 1989, XA-Conversion changes       */
/*                   2  21 Feb.    1990, XA-Conversion - Removed     */
/*                                       Set ACNT & Set BLIP         */
/*                   3  23 Feb.    1990, Removed ref. to FORTMOD2    */
/*                                                                   */
/* *************************************************************** */

  Trace R                 /* Don't echo these commands to the console */

                          /*  Following things not done in batch      */
  If ^ XFLAG(BATCH) then do

    QCONSOLE              /*  Determine and set up for terminal type  */
    Pull . contype .
    If contype = 'TTY'     then EXEC ASCII NOQTERM NOCLEAR
    Else if contype = 3270 then EXEC 3270  NOQTERM NOCLEAR RESET

                          /*  Only do news/mail if there's
                              really a terminal there.                */
    If contype ^= 'DISC' then do
      DROPBUF 0           /*  Clear stack buffers before news & mail  */
    End                   /*  contype ^= 'DISC'                       */
   End                    /*  processing for non-BATCH                */
/* C compiler stuff */
'GIME C370'

/* SET UP TCP ACCESS */
/* "GIME TCPMNT 582" tyh */
"GIME TCPAPPL "
"GLOBAL TXTLIB IBMLIB CMSLIB EDCBASE COMMTXT VSPASCAL AMPLANG"
'QCONSOLE'
  pull . cons .

  if cons = 'DISC' then do
     'EXEC RUNDAEMO'
  end

Return
```

```
    &TRACE OFF
 * This profile was set up for D. Phillips, by T. Beeman 12/29/81
 * LAST MODIFIED, 3/5/84 BY L. ADDIS FOR GROUP $LI
    &IF &N = 0 &EXIT
 *
 *  FIND THE OPTIONS LIST (IF ONE EXISTS) AND BREAK IT OFF FOR LATER
    &PAREN = &LOCATION OF ( &ARGSTRING
    &IF &PAREN = 0 &GOTO -FINDASTER
       &OPTIONS = &SUBSTR OF &ARGSTRING &PAREN *
       &PAREN = &PAREN - 1
       &ARGSTRING = &SUBSTR OF &ARGSTRING 1 &PAREN
       &STACK LIFO &ARGSTRING
       &READ ARGS
 *
 *  OPTIONS ARE GONE.  USE HEURISTICS TO DETERMINE FILE NAME.
 -FINDASTER
    &AST = &LOCATION OF * &ARGSTRING
    &IF &AST = 0  &IF &N >= 2  &GOTO -LOAD     (NO SEARCHING IS NEEDED)
    &IF .&2 = . &ARGS &1 * *
    &COMMAND DESBUF
    &COMMAND SET CMSTYPE HT
    &COMMAND LISTX &1 &2 &3 ( STACK FIFO FORMAT  SYSTEM
    & = &RC
    &COMMAND SET CMSTYPE RT
    &IF & ^= 0 &GOTO -NOTFOUND         (NOTHING CAME BACK FROM LISTX)
*------------------------------------------------------------------------
    &COMMAND SENTRIES
    &N = &RC
    &IF &N = 1 &GOTO -JUSTONE           (ONE FILENAME IN STACK - EDIT IT)
*------------------------------------------------------------------------
*
*  THE STACK CONTAINS MORE THAN ONE ENTRY.  UNSTACK AND PUT THEM INTO
*  THE ARRAY(I) VARIABLES.
   &TYPE Multiple files match given filename:
   &I = 1
   &LOOP -ENDLOOP &N
      &READ STRING &LINE
      &TYPE &I &LINE
      &ARRAY&I = &SUBSTR OF &LINE 1 20
-ENDLOOP &I = &I + 1
*------------------------------------------------------------------------
*  THE USER CAN TYPE UP TO 10 NUMBERS IN ANY ORDER, OR ALL.  THE FILES
*  WILL BE BROUGHT INTO THE EDITING RING IN THE ORDER SPECIFIED, WITH
*  THE FINAL FILE BEING CURRENT.
*  AN EMPTY LINE RESPONSE DEFAULTS TO THE FIRST FILE IN THE LIST.
*
-ASKWANTS
   &TYPE Which one(s) do you want? (Nos., A(ll), Q(uit), or R(edisplay))
   &READ VAR &W0 &W1 &W2 &W3 &W4 &W5 &W6 &W7 &W8 &W9 &W10
   &Q = &SUBSTR OF .&W0 1 2
   &IF &Q = .Q &GOTO -QUIT
*------------------------------------------------------------------------
   &IF .&W0 = . &W0 = 1
   &IF &Q ^= .R &GOTO -CHEKFORALL   (NO REDISPLAY - KEEP CHECKING)
      &I = 1
      &LOOP -ENDREDIS &N
         &TYPE &I &ARRAY&I
-ENDREDIS &I = &I + 1
      &GOTO -ASKWANTS
*------------------------------------------------------------------------
```

```
 -CHEKFORALL
    &IF &Q ^= .A &GOTO -WANTLIST      (USER GAVE US A LIST)
       &STACK &ARRAY1
       &READ ARGS
       &I = 2
       &LOOP -ENDALL &N
          &STACK XEDIT &ARRAY&I
 -ENDALL  &I = &I + 1
       &STACK XEDIT &ARRAY1
       &GOTO -LOAD                       (DONE MESSING AROUND - LOAD THE EDITOR)
 *-------------------------------------------------------------------------
 *
 *  UNWRAP THE W(I) LIST AND STACK 'XEDIT' SUBCOMMANDS FOR THE EDITOR TO READ
 *  ONCE IT IS RUNNING.  THEY WILL CREATE THE RING FOR THE USER.
 -WANTLIST
    &TYPEW0 = &TYPE OF &W0
    &IF &TYPEW0 = NUM &GOTO -FIRSTOK
       &TYPE '&W0 ' is invalid.  (Must be an integer, ALL, or QUIT.)
       &GOTO -ASKWANTS
 -FIRSTOK
    &STACK &ARRAY&W0
    &READ ARGS
    &I = 1
    &LOOP -ENDWANTS 10
    &IF .&W&I = . &GOTO -ENDWANTS
       &STACK XEDIT &ARRAY&W&I
 -ENDWANTS &I = &I + 1
    &STACK XEDIT &ARRAY&W0
    &GOTO -LOAD
 *-------------------------------------------------------------------------
 *
 *  EXACTLY ONE FILE MATCHED THE SPECIFIED LISTX PATTERN.  STACK A MESSAGE
 *  TELLING THIS TO THE USER.  IT WILL BE IN THE DISPLAY AREA OF A 3270.
 -JUSTONE
    &READ ARGS
    &STACK COMMAND MSG One file found: &1 &2 &3
    &GOTO -LOAD
 *-------------------------------------------------------------------------
 -NOTFOUND
    &TYPE FILE &1 &2 &3 NOT FOUND
 -QUIT
    &ARGS $DUMMY$$ $DUMMY$$ S
    &QUITFLAG = YES
 -LOAD
    LOAD &1 &2 &3 &OPTIONS
    &IF &RC ^= 0 &EXIT &RC
    &IF .&QUITFLAG ^= .YES  &GOTO -DEFAULTS
       &STACK LIFO QUIT
       &EXIT &RC
COMMAND QUIT
*
*  NOW WE HAVE A FILE.  SET UP THE DEFAULTS.
*
-DEFAULTS
   TRANSFER TERMINAL FN FT FM RECFM LRECL SIZE LINE
   &READ VAR &TERM &FN &FT &FM &RECFM &LRECL &SIZE
   &IF &TERM = TYPEWRITER &GOTO -TYPER
*-------------------------------------------------------------------------
*
*  DISPLAY TERMINAL HANDLING
```

```
   SET NULL ON
   SET ENTER IGNORE COMMAND CURSOR CMDLINE 1 PRIORITY 10
   SET CASE M RESPECT
 * SET CMDLINE TOP
   SET NUMBER ON
 * MAKE THE CURRENT LINE IN XEDIT ABOUT 1/3 DOWN FROM TOP
 * TRANSFER LSCREEN
 * &READ VARS &NLINL
 * &CURLIN = &NLINL + 4
 * &CURLIN = &DIV OF &CURLIN 3
 * SET CURLINE ON &CURLIN
 * &KURLIN = &CURLIN - 3
 * DOWN &KURLIN
 * &MAXLIN = &NLINL - 2
 * &IF &CURLIN >= &MAXLIN   &SKIP 4
     SET CURLINE ON 10
 *   &SCLLIN = &CURLIN + 1
     SET SCALE ON 11
 *   SET SCALE ON &SCLLIN
 *   SET SCALE OFF
 * &SKIP 1
 *   SET SCALE OFF
 * IF THE LRECL IS > 80 THEN TRY TO PUT AS MUCH ON THE SCREEN AS POSIBLE
   &IF &LRECL LE 80 &GOTO -SKIPWIDE
 *     SET PREFIX OFF
       VERIFY OFF 1 *
-SKIPWIDE
 *   &IF &FT = EXEC SET PREFIX ON
 *   &IF &FT = XEDIT SET PREFIX ON
 *   &IF &FT = SCRIPT SET PREFIX ON
 *   &IF &FT = TEX SET PREFIX ON
  SET PREFIX ON
  &GOTO -REJOIN
*--------------------------------------------------------------------------
*
*  ASCII TERMINAL HANDLING
-TYPER
  &TYPE &FN &FT &FM (&SIZE LINES), &RECFM &LRECL
  SET COLPTR OFF
  SET MACRO ON
*--------------------------------------------------------------------------
*
*  CONTROL REJOINS HERE FOR BOTH TERMINAL TYPES
*  CONVENIENT SYNONYMS:  A (AGAIN),  TA (TYPE ALL)
*  HANDLING OF SPECIAL FILETYPES
-REJOIN
   &IF &TERM = DISPLAY SET SCALE ON
   &IF &FT = PASCAL SET CASE M I

   &IF &FT = LISTING SET VERIFY 2 *
   &IF &IF = FILE SET VERIFY 2 *
   &IF &FT = MYCARDS SET CASE UPPER
   &IF &FT = MYCARDS SET LRECL 80
   &IF &FT = SASLOG SET VERIFY 2 *
   &IF &FT = FORTRAN SET CASE UPPER
   &IF &FT = XEDIT SET CASE UPPER
   &IF &FT = EXEC SET CASE MIXED
   &IF &FT = SCRIPT SET CASE M I
   &IF &FT = TEX SET CASE M I
   &IF &FT = MYCARDS SET SERIAL OFF
 *  SET SYN A 1 =
 *  SET SYN ; 1 :
 *  SET SYN TA 2 -* T*

 *  SET MY SYNONYMS
    SET SYN BOTTOM   1 BOTTOM
    SET SYN BACKWARD 4 BACKWARD
    SET SYN LWYLBUR 1 RSMLIST
    SET SYN PWYLBUR 1 RSMLIST
 *  SET SYNONYM LIST 1 RSMLIST
    SET SYN LOCATE  2 LOCATE
    SET SYN POINT   2 SET POINT
 *  SET MY PREFIX SYNONYMS
    SET PREFIX SYN B P
    SET PREFIX SYN L /
    SET PREFIX SYN R "
    SET SYN SCRIPT 2 XSCRIPT
    SET TABS 1 5 10 15 20 25 30 35 40 50 60 70 80
 *-----------------------------------------------------------------------
 *
 *  THE PF KEYS ARE DEFINED FOR MOVING QUICKLY AROUND IN A FILE.  SCHEMATICALLY
 *  THEY ARE:
 *
 * MOVE PAPER FORWARD ----->    A LINE     10 LINES     A PAGE
 *
 * MOVE PAPER      BACKWARD---    A LINE    1 IE    RAE          -
 *
 * MOVE CURSOR TO THE ----->    PREV WD    NEXT WD      END OF LINE
 *
 *                                                      CURRENT LINE &
 *                                                      CURRENT COLUMN
   SET PF1 ?
   SET PF4  TOP
   SET PF5  BOTTOM
   SET PF6  TABKEY
   SET PF7 INSCROLL B
   SET PF8 INSCROLL F
   SET PF9  SPLTJOIN
   SET PF10 MACRO ENDLINE
   SET PF11 CURSOR CO

   SET PF12 FILE
   SET PF13 VER 1 70
   SET PF14 VER 51 120
   SET PF15 VER 91 150
   SET PF16 U10
*  SET PF17 *I
*  SET PF18 TOP
*  SET PF19 MACRO ENDLINE
   SET PF20 RIGHT 20
   SET PF21 LEFT 20
SLACPREF
&IF &TERM = DISPLAY SHOWPF ON

   &EXIT 0
```

```
__________________________ RECOVER ____________________________
Searching ...
*** Note: SPICELL is not now a valid VM userid ***

There are 29 copies of SPICELL 191 on backup tapes.
They are:

1 ) SPICELL 0191 dumped to file 207 of tape RL2746 on 10/03/95 at 18:22:00
2 ) SPICELL 0191 dumped to file 210 of tape RL1813 on 07/04/95 at 18:05:01
3 ) SPICELL 0191 dumped to file 221 of tape RL1565 on 04/04/95 at 20:02:12
4 ) SPICELL 0191 dumped to file 218 of tape RL2991 on 01/03/95 at 21:09:07
5 ) SPICELL 0191 dumped to file 277 of tape RL3151 on 10/04/94 at 18:24:37
6 ) SPICELL 0191 dumped to file 217 of tape RL1810 on 07/06/94 at 17:53:47
7 ) SPICELL 0191 dumped to file 137 of tape RL3319 on 04/06/94 at 00:14:06
8 ) SPICELL 0191 dumped to file 161 of tape RL3137 on 01/06/94 at 03:21:58
9 ) SPICELL 0191 dumped to file 322 of tape RL2982 on 10/06/93 at 04:34:49
10 ) SPICELL 0191 dumped to file 24 of tape RL2709 on 07/06/93 at 15:24:44
11 ) SPICELL 0191 dumped to file 35 of tape RL1750 on 04/06/93 at 13:28:30
12 ) SPICELL 0191 dumped to file 45 of tape RL2516 on 01/05/93 at 14:53:27
13 ) SPICELL 0191 dumped to file 51 of tape RL1534 on 10/07/92 at 00:38:18
14 ) SPICELL 0191 dumped to file 50 of tape RL1373 on 07/07/92 at 18:42:04
15 ) SPICELL 0191 dumped to file 65 of tape RL1782 on 04/07/92 at 19:00:29
16 ) SPICELL 0191 dumped to file 90 of tape RL0498 on 01/07/92 at 19:32:12
17 ) SPICELL 0191 dumped to file 89 of tape RL0749 on 10/02/91 at 01:11:20
18 ) SPICELL 0191 dumped to file 230 of tape RL1202 on 07/03/91 at 07:12:02
19 ) SPICELL 0191 dumped to file 456 of tape RL0784 on 04/03/91 at 06:08:25
20 ) SPICELL 0191 dumped to file 118 of tape RL2272 on 01/03/91 at 04:38:31
21 ) SPICELL 0191 dumped to file 33 of tape RL2122 on 10/03/90 at 15:01:58
22 ) SPICELL 0191 dumped to file 34 of tape RL1997 on 07/03/90 at 13:30:12
23 ) SPICELL 191 dumped to file 3 of tape RL1872 on 04/03/90 at 19:16:38
24 ) SPICELL 191 dumped to file 51 of tape RL1645 on 01/02/90 at 17:14:51
25 ) SPICELL 191 dumped to file 9 of tape RL1474 on 10/03/89 at 18:12:59
26 ) SPICELL 191 dumped to file 52 of tape RL1313 on 07/05/89 at 21:54:23
27 ) SPICELL 191 dumped to file 13 of tape RL1109 on 04/05/89 at 11:56:31
28 ) SPICELL 191 dumped to file 91 of tape RL0586 on 01/03/89 at 22:48:03
29 ) SPICELL 191 dumped to file 21 of tape SL0153 on 10/04/88 at 10:19:38

Which one of the above do you wish to restore?
( 1 - 29 , <CR>=none )
```

```
APSNEWS APSNEWS 52
SLCNEWS SLCNEWS 995
LEPNEWS LEPNEWS 101
```

CMS59F 2513

```
Note QSPIRES  MAIL     A  disc from QSPIRES  at SLACVM   on 12/20/91 11:30:29
Note QSPIRES  MAIL     A  disc from QSPIRES  at SLACVM   on 12/20/91 11:30:29
```

```
WINTERS  JCW119   A1 F     132     73     3  7/22/98  1:44:29 SPC2
README   RL0498S  A1 F     255     91     6  7/22/98  1:44:01 SPC2
SPICE191 BACKUP   A1 V      75     40     1  7/22/98  1:28:39 SPC2
SPICELL  NETLOG   A0 V      77      2     1 12/20/91 11:30:29 SPC2
PROFILE  EXEC     A2 V      69     50     1 12/12/91 18:06:33 SPC2
LOAD     MAP      A5 F     100      2     1 12/12/91 18:03:45 SPC2
SPICELL  CONSLAST A1 F      80      3     1 12/12/91  0:34:30 SPC2
SPICELL  LASTNEWS A1 V      12      1     1 12/12/91  0:33:19 SPC2
LASTING  GLOBALV  A1 V      24      1     1  5/30/90  9:11:23 SPC2
PROFILE  XEDIT    A1 V      79    235     2 11/30/89 14:49:13 SPC2
```

```
*************************************************************************
*                                                                       *
* Start execution of JCW119 for WINTERS, class S.                       *
* Execution begins at 01:29:32 on 22 Jul 1998 in BATCH08A at SLACVM.    *
* Submitted at        01:29:22 on 22 Jul 1998 for WINTERS at SLACVM.    *
* Running CURCMS, mode XA, storage = 4096K, time limit = 2 minutes.     *
* Max print = 10K lines, max punch = 100K cards.                        *
*                                                                       *
*************************************************************************

-----> Execute "R$COV191"

** RECOVER SPICELL 191 to TDISK **
DASD 0555 DEFINED
TAPE 0F13 ATTACHED TO BATCH08A 0181
TAPE 0181 ON DEV  0F13 3490 R/W SUBCH = 000D
MNTREQ100I 01:29:34 Mounting volume RL0498 on device 181, SL NoRing
MNTVER107I 01:30:10 Remounting volume RL0498 on device 181; intervention required
MNTVER101I 01:30:37 Volume RL0498 on device 181 has been verified
Rewind complete
RESTORING LXA191
DATA DUMPED   01/07/92 AT 19.32.07  GMT FROM LXA191 RESTORED TO SCRATCH
INPUT CYLINDER EXTENTS      OUTPUT CYLINDER EXTENTS
        START     STOP       START     STOP
        0000      0000       0000      0000
END OF RESTORE
END OF JOB
01:32:07  * MSG FROM VMSETUPA: VMSCAN001I VMSETUP REQUEST FOR WINTERS JCW119 WILL BE
TAPE 0181 DETACHED BY VMSETUPA
CARD DUMP LASTING GLOBALV Z1
PUN FILE 0625 SENT TO   WINTERS  RDR AS  7295 RECS 0002 CPY  001 Y NOHOLD NOKEEP
CARD DUMP LOAD MAP Z5
PUN FILE 0626 SENT TO   WINTERS  RDR AS  7296 RECS 0003 CPY  001 Y NOHOLD NOKEEP
CARD DUMP PROFILE EXEC Z2
PUN FILE 0627 SENT TO   WINTERS  RDR AS  7297 RECS 0023 CPY  001 Y NOHOLD NOKEEP
CARD DUMP PROFILE XEDIT Z1
PUN FILE 0628 SENT TO   WINTERS  RDR AS  7298 RECS 0090 CPY  001 Y NOHOLD NOKEEP
CARD DUMP SPICELL CONSLAST Z1
PUN FILE 0629 SENT TO   WINTERS  RDR AS  7299 RECS 0003 CPY  001 Y NOHOLD NOKEEP
CARD DUMP SPICELL LASTNEWS Z1
PUN FILE 0630 SENT TO   WINTERS  RDR AS  7300 RECS 0002 CPY  001 Y NOHOLD NOKEEP
CARD DUMP SPICELL NETLOG Z0
PUN FILE 0631 SENT TO   WINTERS  RDR AS  7301 RECS 0004 CPY  001 Y NOHOLD NOKEEP
FILENAME FILETYPE FM FORMAT LRECL       RECS     BLOCKS   DATE      TIME     LABEL
LASTING  GLOBALV  Z1 V         24          1          1  5/30/90   9:11:23 LXA191
LOAD     MAP      Z5 F        100          2          1 12/12/91 18:03:45 LXA191
PROFILE  EXEC     Z2 V         69         50          2 12/12/91 18:06:33 LXA191
PROFILE  XEDIT    Z1 V         79        235          4 11/30/89 14:49:13 LXA191
SPICELL  CONSLAST Z1 F         80          3          1 12/12/91  0:34:30 LXA191
SPICELL  LASTNEWS Z1 V         12          1          1 12/12/91  0:33:19 LXA191
SPICELL  NETLOG   Z0 V         77          2          1 12/20/91 11:30:29 LXA191
-----> Batch return code = 0.

-----> System cleanup $BCLEAN.

*************************************************************************
*                                                                       *
* End execution of JCW119 for WINTERS at 01:32:08 on 22/07/98.          *
*                                                                       *
* Actual CPU time (mm:ss[.th]):      Virtual = 0:00.38 , Total = 0:00.60 *
* I/O counts:  SIO=000387 RDR-000000 PRT-000086 PCH-000204               *
*                                                                        *
* Space used on A-disk at job end:  48K                                  *
*                                                                        *
* I/O counts on units still available                                    *
* Spool units                                                            *
* 0009:    66    000C:      0    000D:      9    000E:      0            *
* 00AE:     0                                                            *
* Dasd units                                                             *
* 0190(S):    34    0191(A):    98    019E(Y):    37    0555(Z):    32   *
* 059E(T):    12    059F(U):     6    05A0(V):    27    05A1(R):    18   *
**************************************************************************
```

```
total 108
-r--r--r--   1 winters  sf        23296 Jul 21 18:44 README
-r--r--r--   1 winters  sf           25 May 30 1990  lasting.globalv
-r--r--r--   1 winters  sf          202 Dec 12 1991  load.map
-r--r--r--   1 winters  sf         2212 Dec 12 1991  profile.exec
-r--r--r--   1 winters  sf         7206 Nov 30 1989  profile.xedit
-r--r--r--   1 winters  sf         2430 Jul 21 18:28 spice191.backup
-r--r--r--   1 winters  sf          243 Dec 11 1991  spicell.conslast
-r--r--r--   1 winters  sf           13 Dec 11 1991  spicell.lastnews
-r--r--r--   1 winters  sf          156 Dec 20 1991  spicell.netlog
-r--r--r--   1 winters  sf         1080 Jul 21 18:49 winters.filelist
-r--r--r--   1 winters  sf         9709 Jul 21 18:44 winters.jcwl19
-rw-r--r--   1 winters  sf            0 Jul 22 01:51 winters.ls
```

UNIX Command Summary with an Extract of Output

Commands used to copy files RECOVERed from CMS Backup to WINTERS 294 (a SPACE TEMPDISK), from that SLACVM minidisk on to UNIX AFS WWW archive space:

```
   /afs/slac.stanford.edu/www/archive/1993/SLACVM/www/191/rl0498
```

The commands (almost always) and output were cut and pasted from the xterm (or aixterm) window where the work was executed. Text in square brackets ([ ]) is commentary.

```
pwd
/afs/slac.stanford.edu/www/archive/1992/SLACVM/spicell/191/rl0498

date
Wed Jul 22 01:49:13 PDT 1998

vmnfs refresh winters.294
[SLACVM.SLAC.STANFORD.EDU]
Disk WINTERS.294 refreshed.

cp -p /nfs/slacvm/ascii/winters.294/* .

chmod a-x *

mv readme.rl0498s README

ls -l > winters.ls
671 cassandra /191/rl0498> chmod u-w winters.ls

 wc winters.ls
      13     110     825 winters.ls
673 cassandra /191/rl0498> wc winters.filelist
      10     130    1080 winters.filelist


[Dashing along, omitted generating CMS PRESERVE...]
```

Note the date of this file, unix-commands, has been updated since the file winters.ls was created. README is also likely to be updated, too.

22 Jul 1998   Joan M. Winters

[illegible] 1414

Index of /archive/1992/SLACVM/spicell/192

```
     Name                      Last modified       Size  Description
------------------------------------------------------------------------
     Parent Directory          22-Jul-1998 02:16     -
     rl1414/                   22-Jul-1998 03:04     -
     rl1559/                   22-Jul-1998 03:39     -
------------------------------------------------------------------------
```

Apache/1.3.12 Server at www.slac.stanford.edu Port 80

Index of /archive/1992/SLACVM/spicell/192/rl1414

Name	Last modified	Size	Description
Parent Directory	22-Jul-1998 03:12	-	
README	21-Jul-1998 19:30	10k	
a.c	16-Oct-1991 11:13	1k	
a.errs	16-Oct-1991 11:15	1k	
a.exec	01-Nov-1991 07:13	1k	
a.text	16-Oct-1991 11:15	1k	
binlist.html	06-Dec-1991 13:11	1k	
binlist.index	06-Dec-1991 13:00	1k	
bsdtypes.h	15-Oct-1991 11:01	3k	
cc2.exec	30-Oct-1991 06:18	1k	
cc_all.exec	14-Oct-1991 08:29	1k	
cmod.exec	04-Nov-1991 03:35	4k	
cms.preserve	06-Nov-1991 09:06	1k	
cpobj.rmap	16-Oct-1991 11:46	1k	
default.html	06-Dec-1991 13:15	1k	
dload.exec	01-Nov-1991 07:15	1k	
edcxv.c	16-Oct-1991 11:08	1k	
edcxv.text	05-Dec-1991 08:03	1k	
fearch.orig	06-Nov-1991 04:04	5k	
fget.exec	06-Dec-1991 12:45	1k	
fget.orig	06-Nov-1991 09:08	5k	
findgate.c	06-Nov-1991 09:05	7k	
findgate.c2	14-Oct-1991 08:28	7k	
findgate.orig	04-Nov-1991 03:30	7k	
findgate.slacvm	06-Nov-1991 05:57	7k	
findgate.text	05-Dec-1991 08:02	9k	
fsearch.exec	09-Dec-1991 10:10	1k	
fsearch.save	06-Nov-1991 06:06	5k	
htdaemon.c	15-Oct-1991 11:10	16k	
htdaemon.c2	15-Oct-1991 08:45	16k	

```
htdaemon.module      05-Dec-1991 08:03   128k
htdaemon.text        05-Nov-1991 06:37    14k
httcp.c              16-Oct-1991 10:04     7k
httcp.h              14-Oct-1991 08:47     2k
httcp.text           05-Nov-1991 06:38     6k
htutils.h            14-Oct-1991 08:47     2k
load.map             05-Dec-1991 07:55     1k
make.exec            19-Sep-1991 07:06     1k
mycc.exec            16-Oct-1991 11:28     1k
nicfoun1.exec        24-Sep-1991 04:01     5k
q.q                  16-Oct-1991 10:22     4k
run.exec             19-Sep-1991 07:19     1k
rundaemo.exec        06-Nov-1991 09:03     1k
setup.exec           05-Nov-1991 06:57     1k
spice192.backup      21-Jul-1998 18:58     1k
spires.html          09-Dec-1991 09:20     1k
spires.index         12-Dec-1991 07:59     1k
string.h             16-Oct-1991 10:07     6k
tcp.h                14-Oct-1991 08:47     5k
terryh.tyh239        14-Nov-1991 10:43    11k
unix-commands        22-Jul-1998 03:08     4k
winters.filelist     21-Jul-1998 19:57     5k
winters.jcw120       21-Jul-1998 19:11    25k
winters.ls           22-Jul-1998 03:04     3k
```

Apache/1.3.12 Server at www.slac.stanford.edu Port 80

File /afs/slac.stanford.edu/www/archive/1992/SLACVM/spicell/192/rl1414/README

On 22 Jul 1998 restored currently existing backup copy #16 (on tape cartridge RL1414) of userid SPICELL minidisk 192 via the CMS RECOVER command to WINTERS virtual reader, and from there to WINTERS 294 (a SPACE TEMPDISK) using a "RECEIVE / (OLDDATE" command within RDRLIST. The VM backup tape was taken 1/8/92.

Then copied all the RECOVERed files plus five files on the recovery and migration process from SLACVM to UNIX. Used an NFS mount?? in UNIX of the SLACVM WINTERS 294 minidisk, followed by a UNIX copy of all those files to the subdirectory /afs/slac.stanford.edu/www/archive/1992/SLACVM/spicell/192/rl1414. (A summary of UNIX commands is in file unix-commands.)

SPICELL 192 may have been used in very early development of the experimental WWW system at SLAC. SPICELL 192 is not an "INSTALL" disk.

More specifically, from the CMS RECOVER list of backup tapes on 7/22/98 selected:

```
16 ) SPICELL 0192 dumped to file 51 of tape RL1414 on 01/08/92 at 01:02:19
```

See file spice192.backup for the entire list of currently existing backups (created from XSHOW capture of RECOVER command output).

RECOVER submitted VM batch job JCW120 from userid WINTERS on 7/22/98. The job was run shortly afterwards. Its batch console output is in file winters.jcw120.

After the recovered files were RECEIVEd, QUERY DISK A showed:

```
LABEL  VDEV M  STAT   CYL TYPE BLKSIZE  FILES  BLKS USED-(%) BLKS LEFT  BLK TOTAL
SPC294 294  A   R/W    20 3390 4096       50       144-04        3456       3600
```

There were a total of 47 files on the SLACVM SPICELL 192 minidisk of 1/8/92. In inverse chronological order since last update, here are the files (from "FILELIST (STATS" command output*):

```
Filename Filetype Fm Format LRECL   Records    Blocks   Date      Time

SPIRES   INDEX    A1 V         43       12          1 12/12/91 15:59:13 SPC2
FSEARCH  EXEC     A1 V         40        7          1 12/09/91 18:10:32 SPC2
SPIRES   HTML     A1 V         74        9          1 12/09/91 17:20:32 SPC2
DEFAULT  HTML     A1 V         81       13          1 12/06/91 21:15:48 SPC2
BINLIST  HTML     A1 V         73       13          1 12/06/91 21:11:53 SPC2
BINLIST  INDEX    A1 F         80        8          1 12/06/91 21:00:22 SPC2
FGET     EXEC     A2 V         60       16          1 12/06/91 20:45:45 SPC2
HTDAEMON MODULE   A1 V      65535        5         32 12/05/91 16:03:14 SPC2
EDCXV    TEXT     A1 F         80        9          1 12/05/91 16:03:06 SPC2
FINDGATE TEXT     A1 F         80      113          3 12/05/91 16:02:49 SPC2
LOAD     MAP      A1 F        100        2          1 12/05/91 15:55:46 SPC2
TERRYH   TYH239   A1 F        132       87          3 11/14/91 18:43:33 SPC2
FGET     ORIG     A2 V         78      132          2 11/06/91 17:08:04 SPC2
CMS      PRESERVE A1 V         65       30          1 11/06/91 17:06:02 SPC2
FINDGATE C        A1 V         78      245          2 11/06/91 17:05:49 SPC2
RUNDAEMO EXEC     A1 V         33        5          1 11/06/91 17:03:09 SPC2
```

```
FSEARCH  SAVE      A1 V        76        123       2 11/06/91 14:06:50 SPC2
FINDGATE SLACVM    A1 V        78        245       2 11/06/91 13:57:17 SPC2
FEARCH   ORIG      A1 V        76        125       2 11/06/91 12:04:44 SPC2
SETUP    EXEC      A1 V        62          4       1 11/05/91 14:57:21 SPC2
HTTCP    TEXT      A1 F        80         77       2 11/05/91 14:38:12 SPC2
HTDAEMON TEXT      A1 F        80        183       4 11/05/91 14:37:48 SPC2
CMOD     EXEC      A2 V        68        203       2 11/04/91 11:35:12 SPC2
FINDGATE ORIG      A1 V        78        245       2 11/04/91 11:30:12 SPC2
DLOAD    EXEC      A1 V        64          4       1 11/01/91 15:15:36 SPC2
A        EXEC      A1 V        68         10       1 11/01/91 15:13:32 SPC2
CC2      EXEC      A1 V        58          8       1 10/30/91 14:18:05 SPC2
CPOBJ    RMAP      A1 F        80          3       1 10/16/91 18:46:59 SPC2
MYCC     EXEC      A1 V        63         51       1 10/16/91 18:28:29 SPC2
A        ERRS      A1 V        49          1       1 10/16/91 18:15:35 SPC2
A        TEXT      A1 F        80          5       1 10/16/91 18:15:35 SPC2
A        C         A1 F        80          5       1 10/16/91 18:13:27 SPC2
EDCXV    C         A1 F        80          5       1 10/16/91 18:08:43 SPC2
Q        Q         A1 F        80         52       2 10/16/91 17:22:58 SPC2
STRING   H         A2 F        80         75       2 10/16/91 17:07:43 SPC2
HTTCP    C         A1 V        79        282       2 10/16/91 17:04:54 SPC2
HTDAEMON C         A1 V        85        639       5 10/15/91 18:10:19 SPC2
BSDTYPES H         A1 V        70        120       1 10/15/91 18:01:30 SPC2
HTDAEMON C2        A1 V        85        639       5 10/15/91 15:45:38 SPC2
HTTCP    H         A1 V        78         94       1 10/14/91 15:47:38 SPC2
TCP      H         A1 V        91        180       2 10/14/91 15:47:28 SPC2
HTUTILS  H         A1 V        79         85       1 10/14/91 15:47:21 SPC2
CC_ALL   EXEC      A1 V        81         13       1 10/14/91 15:29:29 SPC2
FINDGATE C2        A1 V        78        245       2 10/14/91 15:28:41 SPC2
NICFOUN1 EXEC      A2 V        78        125       2  9/24/91 11:01:53 SPC2
RUN      EXEC      A1 V        74         21       1  9/19/91 14:19:57 SPC2
MAKE     EXEC      A1 V        51          6       1  9/19/91 14:06:07 SPC2
```

```
* As defined in "HELP CMS FILELIST", the STATS option produces
the following output:

       STAts
           lists the following information about the specified files:

             o File identifier or directory identifier
             o Format and logical record length of the file
             o Number of records and number of blocks in the file
             o Date and time of last update.

           See the "Examples" for a sample display using the STATS option.

N.B.:  The oldest SLAC pages probably shown to the world re the
       Fri, 13 Dec 91 17:55:53 GMT+0100 announcement
       to www-interest@cernvax.cern.ch and www-talk@cernvax.cern.ch,
       Cc: pfkeb@kaon.slac.stanford.edu (Paul Kunz):

>There is an experimental W3 server for the SPIRES High energy Physics
>preprint
>database, thanks to Terry Hung, Paul Kunz and Louise Addis of SLAC.  It's
>only just
>been put up, so don't expect perfection.   With the w3 line mode browser,
>follow a
>link to it from our home page, then type for example
>
>        K FIND AUTHOR KUNZ
>
```

```
>the "FIND" is necessary at the moment, though it may change later.
>
>        - Tim
>
>Paul Kunz wrote a few days ago:-
>
>   "The SLAC Library maintainer of SPIRES databases, Louise Addis, is
>absolutely
>delighted.   She will ask for a permanent VM service machine and finish
>off the
>polishing.   Things are really moving now."
>
>   "By the way, we certainly have the impression that accessing SPIRES
>from www on
>a UNIX machine is faster than using a terminal logged into SLACVM.   Even
>a real
>3278 terminal is not as fast.   Actually, accessing CERNVM FIND via www seems
>faster than logging into cernvm and doing the same command as well."
```

I haven't found any ownership noted in the files. The people involved were probably Terry Hung, Paul Kunz, and Louise Addis as the email says.

For a list of all the files recovered plus three files created related to the recovery plus a record of the WINTERS CMS environment in which the recovery occurred (in file CMS PRESERVE), see winters.filelist (created by SAVEing the output of a FILELIST command). A list of all the files on the backup tape (which seems to be the same as all the files recovered) is in the batch job output winters.jcwl20.

For a list of all the files moved to UNIX (which should include all the files in the winters.filelist file), see winters.ls (which also lists the unix-commands file, plus).

In summary, for more details about the recovery from VM backup and move from SLACVM to SLAC UNIX, see the following files:

```
README  [created in SLACVM about entire migration, named README RL1414]
   cms.preserve

   spice192.backup   ??
   winters.jcwl20
   winters.filelist

   unix-commands  [created in UNIX]
      winters.ls
```

See also files in subdirectory /afs/slac/www/archive/SLACVM for generally applicable information.

22 Jul 1998   Joan M. Winters

```
main (argc)
  int argc;
{
  printf("here here");
}
```

```
Warning! line 5: Symbol 'printf' assumed external
```