File created: 24-Feb-2024 11:57:21 **{WMEDLEY}<library>lafite>LAFITE-UNIXMAIL.;4**

edit by: rmk

changes to: (VARS LAFITE-UNIXMAILCOMS)

previous date: 24-Feb-2024 11:35:24 {WMEDLEY}<library>lafite>LAFITE-UNIXMAIL.;3

Read Table: XCL

Package: INTERLISP

Format: XCCS

```
(RPAQQ LAFITE-UNIXMAILCOMS
       ((DECLARE\: DOEVAL@COMPILE DONTCOPY (FILES (SOURCE)
                                                  LAFITE-DECLS LAFITE-NSMAIL)
              (RECORDS UNIXMAILBOX UNIXMAILFILEINFO UNIXMAILPARSE))
        (DECLARE\: DONTEVAL@LOAD DOCOPY (FILES (SYSLOAD)
                                               LAFITE))
        (ALISTS (LAFITEMODELST UNIX))
        ;; JDS 4/6/97:  CHANGE TRANSMIT SMTP INTERACTION TO put <> around mail-from name, which SMTP seems to require.
        ;; These variables control how mail is sent and received.  UNIXMAIL.SEND.MODE controls whether the SMTP stream is opened via
        ;; process-stream (PROCESS) or TCP socket (TCP).  UNIXMAIL.RECEIVE.MODE controls whether mail is received through the Berkeley
        ;; mailer (MAILER) or by reading the spool file directly (SPOOL).  PROCESS and MAILER can only be done under an emulator; TCP and
        ;; SPOOL will also work on D-machines (but may need other library packages like TCP or NFS).
        (INITVARS (UNIXMAIL.SEND.MODE 'PROCESS)
               (UNIXMAIL.RECEIVE.MODE 'SPOOL))
        ;; List used by \UNIXMAIL.AUTHENTICATE to construct the MAILSERVEROPS list
        (VARS UNIXMAIL.MSOPS.LIST)
        ;; These variables control filenames, hostnames, etc. They default to NIL, meaning they are not used or the mailer will try and figure them out
        ;; itself
        (INITVARS UNIXMAIL.SEND.HOST UNIXMAIL.DOMAIN.NAME (UNIXMAIL.SEND.PROCESS '("/usr/bin/mconnect"
                                                                                    "/usr/etc/mconnect"))
               (UNIXMAIL.RECEIVE.PROCESS "/usr/ucb/mail -N")
               UNIXMAIL.SPOOL.FILE
               (UNIXMAIL.DONT.RECEIVE.STATUS "")
               (UNIXMAIL.WRAP.LINES T)
               (UNIXMAIL.WRAP-LIMIT 72)
               (UNIXMAIL.TABWIDTH 8))
        ;; Functions used to receive mail
        (FNS UNIX.POLLNEWMAIL UNIX.NEXTMESSAGE UNIXMAILER.OPENMAILBOX UNIXMAILER.RETRIEVEMESSAGE
             UNIXMAILER.CLOSEMAILBOX UNIXSPOOL.OPENMAILBOX UNIXSPOOL.RETRIEVEMESSAGE UNIXSPOOL.CLOSEMAILBOX)
        ;; Functions used to send mail
        (FNS UNIX.FLUSH.STREAM UNIX.RETRIEVE.LINE \\UNIXMAIL.SEND \\UNIXMAIL.SEND.WRAPLINES \\SMTP-DUMP
             \\UNIXMAIL.SEND.PARSE \\UNIXMAIL.CHECK.ABORT \\UNIXMAIL.MUNG.RECIPIENTS \\UNIXMAIL.SMTP
             \\UNIXMAIL.SMTP.FLUSH \\UNIXMAIL.CHANGE.MODE)
        ;; This returns multiple-values, so it's a CL:LAMBDA (what the heck).
        (FUNCTIONS \\UNIXMAIL.SMTP.TCP.STREAMS)
        ;; Other functions Lafite uses and needs Unix equivalents for
        (FNS \\UNIXMAIL.AUTHENTICATE \\UNIXMAIL.LOGIN \\UNIXMAIL.PARSENAMES \\UNIXMAIL.MAKEANSWERFORM
             \\UNIXMAIL.MESSAGE.FROM.SELF.P \\UNIXMAIL.MESSAGE.P \\UNIXMAIL.REALADDRESS \\UNIXMAIL.FQNAME
             \\UNIXMAIL.FIXMICROSOFT)
        ;; This is a stub needed by the TEdit-uuencode strategy; if we ever decide on a reasonable way to do this and make it part of Lafite, this may
        ;; go away
        (P (MOVD? 'NILL 'UNIX.UUDECODE.IF.NEEDED))
        ;; Hack to install easy interface for line wrapping.  I should really put line-wrapping into Lafite as a whole.
        (P (CL:WHEN (CAR (NLSETQ (EDITE LAFITESENDINGMENUITEMS '(F \\SENDMSG.CHANGE.MODE))))
               ;; The CONS below insures that Lafite notices you've changed LAFITESENDINGMENUITEMS, so the menu will get updated.
               (SETQ LAFITESENDINGMENUITEMS (EDITE (CONS (CAR LAFITESENDINGMENUITEMS)
                                                         (CDR LAFITESENDINGMENUITEMS))
                                                   '(CHANGE \\SENDMSG.CHANGE.MODE TO \\UNIXMAIL.CHANGE.MODE)))))
        (PROP FILETYPE LAFITE-UNIXMAIL)))

(DECLARE\: DOEVAL@COMPILE DONTCOPY

(FILESLOAD (SOURCE)
       LAFITE-DECLS LAFITE-NSMAIL)

(DECLARE\: EVAL@COMPILE

(RECORD UNIXMAILBOX (UMSTREAM UMNUMBERS UMNEXT UMLOCKSTREAM))

(RECORD UNIXMAILFILEINFO (UMFNAME UMFTIME))

(RECORD UNIXMAILPARSE (UNIXMAILSUBJECT UNIXFROM UNIXTO UNIXOTHER FORMATTED? UNIXBODY))
)
```

```
)

(DECLARE\: DONTEVAL@LOAD DOCOPY

(FILESLOAD (SYSLOAD)
       LAFITE)
)

(ADDTOVAR LAFITEMODELST (UNIX 3 \\UNIXMAIL.SEND.PARSE \\UNIXMAIL.SEND \\UNIXMAIL.MAKEANSWERFORM
                              \\UNIXMAIL.AUTHENTICATE \\UNIXMAIL.MESSAGE.P \\UNIXMAIL.MESSAGE.FROM.SELF.P
                              \\UNIXMAIL.LOGIN))
```

;; JDS 4/6/97: CHANGE TRANSMIT SMTP INTERACTION TO put <> around mail-from name, which SMTP seems to require.

;; These variables control how mail is sent and received. UNIXMAIL.SEND.MODE controls whether the SMTP stream is opened via process-stream
;; (PROCESS) or TCP socket (TCP). UNIXMAIL.RECEIVE.MODE controls whether mail is received through the Berkeley mailer (MAILER) or by
;; reading the spool file directly (SPOOL). PROCESS and MAILER can only be done under an emulator; TCP and SPOOL will also work on D-machines
;; (but may need other library packages like TCP or NFS).

```
(RPAQ? UNIXMAIL.SEND.MODE 'PROCESS)

(RPAQ? UNIXMAIL.RECEIVE.MODE 'SPOOL)
```

;; List used by \UNIXMAIL.AUTHENTICATE to construct the MAILSERVEROPS list

```
(RPAQQ UNIXMAIL.MSOPS.LIST ((MAILER UNIX.POLLNEWMAIL UNIXMAILER.OPENMAILBOX UNIX.NEXTMESSAGE
                                    UNIXMAILER.RETRIEVEMESSAGE UNIXMAILER.CLOSEMAILBOX)
                            (SPOOL UNIX.POLLNEWMAIL UNIXSPOOL.OPENMAILBOX UNIX.NEXTMESSAGE
                                   UNIXSPOOL.RETRIEVEMESSAGE UNIXSPOOL.CLOSEMAILBOX)))
```

;; These variables control filenames, hostnames, etc. They default to NIL, meaning they are not used or the mailer will try and figure them out itself

```
(RPAQ? UNIXMAIL.SEND.HOST NIL)

(RPAQ? UNIXMAIL.DOMAIN.NAME NIL)

(RPAQ? UNIXMAIL.SEND.PROCESS '("/usr/bin/mconnect" "/usr/etc/mconnect"))

(RPAQ? UNIXMAIL.RECEIVE.PROCESS "/usr/ucb/mail -N")

(RPAQ? UNIXMAIL.SPOOL.FILE NIL)

(RPAQ? UNIXMAIL.DONT.RECEIVE.STATUS "")

(RPAQ? UNIXMAIL.WRAP.LINES T)

(RPAQ? UNIXMAIL.WRAP-LIMIT 72)

(RPAQ? UNIXMAIL.TABWIDTH 8)
```

;; Functions used to receive mail

```
(DEFINEQ
```

(UNIX.POLLNEWMAIL

```
  (LAMBDA (ADDRESS REGISTEREDNAME CREDENTIALS MAILSERVER)        ; Edited 19-May-99 10:34 by rmk:
                                                                 ; Edited 24-Oct-90 00:04 by jrb:
```

;; We have mail iff our mail spool file (either the value of UNIXMAIL.SPOOL.FILE or /usr/spool/mail/<SHORTUSERNAME>) exists and its date is
;; later than the last time we got our Unix mail. In relentlessly hackish use of the existing MAILSERVER structure, MAILPORT holds a
;; UNIXMAILFILEINFO which remembers the name of our mail file and when we last looked at it.

```
    (LET (X (FILEINFO (OR (|fetch| (MAILSERVER MAILPORT) |of| MAILSERVER)
                          (|replace| (MAILSERVER MAILPORT) |of| MAILSERVER
                             |with| (|create| UNIXMAILFILEINFO
                                              UMFNAME _ (OR UNIXMAIL.SPOOL.FILE (CL:CONCATENATE 'STRING
                                                                                   "{UNIX}/usr/spool/mail/"
                                                                                   (|fetch| (LAFITEMODEDATA
                                                                                              SHORTUSERNAME)
                                                                                      |of| *LAFITE-MODE-DATA*)))
                                              UMFTIME _ 0)))))
          (AND (CL:PROBE-FILE (|fetch| (UNIXMAILFILEINFO UMFNAME) |of| FILEINFO))
               (SETQ X (GETFILEINFO (|fetch| (UNIXMAILFILEINFO UMFNAME) |of| FILEINFO)
                                    'LENGTH))
               (IGREATERP X 0)
               (SETQ X (GETFILEINFO (|fetch| (UNIXMAILFILEINFO UMFNAME) |of| FILEINFO)
                                    'IWRITEDATE))
               (IGREATERP X (|fetch| (UNIXMAILFILEINFO UMFTIME) |of| FILEINFO))))))
```

(UNIX.NEXTMESSAGE

```
  (LAMBDA (MAILBOX)                                              ; Edited  5-Jan-89 18:18 by bane
    (CAR (|fetch| UMNEXT |of| MAILBOX))))
```

(UNIXMAILER.OPENMAILBOX

```
  (LAMBDA (ADDRESS REGISTEREDNAME CREDENTIALS MAILSERVER)        ; Edited 15-Oct-90 20:31 by jrb:
```

```
    ;; A Unix "mailbox" is a process-stream talking to /usr/ucb/mail
    (|if| (OR (|fetch| (MAILSERVER NEWMAILP) |of| MAILSERVER)
              (UNIX.POLLNEWMAIL ADDRESS REGISTEREDNAME CREDENTIALS MAILSERVER))
        |then| (LET* ((MSTREAM (CREATE-PROCESS-STREAM UNIXMAIL.RECEIVE.PROCESS))
                      (UMBOX (|create| UNIXMAILBOX
                                       UMSTREAM _ MSTREAM
                                       UMNUMBERS _ NIL
                                       UMNEXT _ NIL))
                      NUMBERS)
                     ;; Get it in condition to be talked to
                     (CL:FORMAT MSTREAM "set screen=10000~%set prompt= ~%")
                     (BLOCK 1000)
                     (UNIX.FLUSH.STREAM MSTREAM)
                     ;; OK, get it to print the headers followed by a line with a strange character on it (char code 254)
                     (CL:FORMAT MSTREAM "h~%echo ")
                     (PRINTCCODE 254 MSTREAM)
                     (PRINTCCODE (CHARCODE NEWLINE)
                           MSTREAM)
                     ;; Before we really get rolling, scream if UNIXMAIL.DONT.RECEIVE.STATUS isn't a string
                     (OR (STRINGP UNIXMAIL.DONT.RECEIVE.STATUS)
                         (ERROR "UNIXMAIL.DONT.RECEIVE.STATUS isn't a string" UNIXMAIL.DONT.RECEIVE.STATUS))
                     ;; Headers look like this:
                     ;; <space><status-char> other junk like size, subject...
                     (|until| (EQ (READCCODE MSTREAM)
                                  254)
                        |do| (|if| (NULL (STRPOS (CL:READ-CHAR MSTREAM)
                                                 UNIXMAIL.DONT.RECEIVE.STATUS))
                                   |then| (|push| NUMBERS (READ MSTREAM)))
                             (UNIX.FLUSH.STREAM MSTREAM (CHARCODE NEWLINE))
                        |finally| (UNIX.FLUSH.STREAM MSTREAM (CHARCODE NEWLINE)))
                     (|if| NUMBERS
                         |then| (SETQ NUMBERS (DREVERSE NUMBERS))
                                (|replace| UMNUMBERS |of| UMBOX |with| NUMBERS)
                                (|replace| UMNEXT |of| UMBOX |with| NUMBERS)
                                (|create| OPENEDMAILBOX
                                          MAILBOX _ UMBOX
                                          PROPERTIES _ (LIST '\#OFMESSAGES (LENGTH NUMBERS)))
                          |else| (CL:FORMAT MSTREAM "x~%")          ; Empty; close the mail process
                                                                    ; and remember that we've checked on the mailbox.
                                                                    ; HACK!  Depends on \LAFITE.RETRIEVEMESSAGES binding
                                                                    ; of MAILSERVER
                                (|replace| (UNIXMAILFILEINFO UMFTIME) |of| (|fetch| (MAILSERVER MAILPORT) |of| MAILSERVER)
                                   |with| (IDATE))
                                'EMPTY))
      |else| 'EMPTY)))
```

(UNIXMAILER.RETRIEVEMESSAGE

```
  (LAMBDA (MAILBOX MSGOUTFILE)                                      ; Edited  5-Sep-90 22:58 by jrb:
    (LET ((MSTREAM (|fetch| UMSTREAM |of| MAILBOX))
          (OUTSTART (GETFILEPTR MSGOUTFILE)))
         ;; The UMCOUNT in the MAILBOX is the number of the message we're about to read in.  The echo command below makes the message
         ;; text be followed by a line starting with the character 254; you'll never see that in a message that has gone over an SMTP channel
         ;; (guaranteed 7-bit chars).
         (CL:FORMAT MSTREAM "p ~d~%echo " (|pop| (|fetch| UMNEXT |of| MAILBOX)))
         (PRINTCCODE 254 MSTREAM)
         (PRINTCCODE (CHARCODE NEWLINE)
               MSTREAM)
         (UNIX.FLUSH.STREAM MSTREAM (CHARCODE NEWLINE))             ; Throw away "Message 1:" line
         (|while| (UNIX.RETRIEVE.LINE MSTREAM MSGOUTFILE) |do| NIL)
         ;; This could use a little error-handling, of course... perhaps a LOT of error handling.
         ;; Check out the tail of the message and uudecode it if it's an encoded message
         (UNIX.UUDECODE.IF.NEEDED MSGOUTFILE OUTSTART))))
```

(UNIXMAILER.CLOSEMAILBOX

```
  (LAMBDA (MAILBOX FLUSH?)                                          ; Edited  5-Sep-90 23:01 by jrb:
    (LET ((MSTREAM (|fetch| UMSTREAM |of| MAILBOX)))
         ;; If FLUSH?, first clean out the mailbox
         (|if| FLUSH?
             |then| (CL:FORMAT MSTREAM "d~{ ~D~}~%" (|fetch| UMNUMBERS |of| MAILBOX)))
         ;; Then close it up
         (CL:FORMAT MSTREAM "q~%")
         (CLOSEF MSTREAM)
         ;; Twiddle a second to let the mailer run, then remember what time we closed the mailbox
         (BLOCK 1000)
```

```
        ;; HACK!  Depends on \LAFITE.RETRIEVEMESSAGES binding of MAILSERVER
        (|replace| (UNIXMAILFILEINFO UMFTIME) |of| (|fetch| (MAILSERVER MAILPORT) |of| MAILSERVER) |with| (IDATE)))))
```

(UNIXSPOOL.OPENMAILBOX

```
(LAMBDA (ADDRESS REGISTEREDNAME CREDENTIALS MAILSERVER)        ; Edited 10-Feb-2000 12:03 by rmk:
                                                               ; Edited 10-Feb-2000 12:02 by rmk:
                                                               ; Edited 11-Mar-99 16:09 by rmk:
                                                               ; Edited 11-Mar-99 16:08 by rmk:
  (IF (OR (FETCH (MAILSERVER NEWMAILP) OF MAILSERVER)
          (UNIX.POLLNEWMAIL ADDRESS REGISTEREDNAME CREDENTIALS MAILSERVER))
      THEN (LET (MSTREAM UMBOX NUMBERS LOCKSTREAM)
                (BIND WRITEDATE (LOCKFILE _ (PACK* "{UNIX}" UNIXMAIL.SPOOL.FILE ".lock"))
                      (TRYNUM _ 0) UNTIL (NLSETQ (SETQ LOCKSTREAM (OPEN LOCKFILE :DIRECTION :OUTPUT
                                                                        :IF-EXISTS :ERROR)))
                  DO (IF (NULL (SETQ WRITEDATE (CAR (NLSETQ (GETFILEINFO LOCKFILE 'IWRITEDATE)))))
                         THEN ;; Error on writedate means file doesn't exist, go around immediately to acquire the lock.
                              (SETQ TRYNUM 0)
                       ELSEIF (IGREATERP (- (IDATE)
                                            WRITEDATE)
                                         300)
                         THEN                                  ; Delete and try again
                              (SETQ TRYNUM 0)
                              (DELFILE LOCKFILE)
                       ELSE (ADD TRYNUM 1)                     ; File still exists and was recently modified; wait and try again
                            (CL:WHEN (EQ TRYNUM 4)
                                (LAB.PROMPTPRINT MAILFOLDER "Unix mailbox file is locked, can't open")
                                (ERROR!))
                            (DISMISS (TIMES TRYNUM 5000))))
                (PRINTOUT LOCKSTREAM "0")
                (CLOSEF LOCKSTREAM)                            ; Close the lock file but use the closed stream later for the
                                                               ; DELFILE.
               ;; Note:  The THROUGH must come before the EOL, otherwise the file reverts back to CR.
               (SETQ MSTREAM (OPENSTREAM (FETCH (UNIXMAILFILEINFO UMFNAME) OF (FETCH (MAILSERVER MAILPORT)
                                                                                      OF MAILSERVER))
                                         'INPUT NIL '((TYPE TEXT)
                                                      (EXTERNALFORMAT :THROUGH)
                                                      (EOL LF))))
               (SETQ UMBOX (CREATE UNIXMAILBOX
                                   UMSTREAM _ MSTREAM
                                   UMNUMBERS _ NIL
                                   UMNEXT _ NIL
                                   UMLOCKSTREAM _ LOCKSTREAM))
               ;; Merrily scan the spool file remembering where all the messages start; there had better be at least one.  All messages in Unix
               ;; spool files start with the character sequence "(OR beginning-of-file Newline)From "; this sequence is guaranteed to occur
               ;; nowhere else but at the start of messages.
               (IF (ZEROP (FILEPOS "From " MSTREAM))
                   THEN (PUSH NUMBERS 0)
                        ;; The (CONCAT...) stuff below is to avoid having a string with a LF character in it; the file package/reader/printer
                        ;; have been known to EOL-translate such characters in strings inappropriately.
                        (BIND POS WHILE (SETQ POS (FILEPOS (CONSTANT (CONCAT (CHARACTER (CHARCODE LF))
                                                                             "From "))
                                                                   MSTREAM))
                           DO (PUSH NUMBERS (ADD1 POS))
                              (READCCODE MSTREAM))
                        (SETQ NUMBERS (DREVERSE NUMBERS))
                        (REPLACE UMNUMBERS OF UMBOX WITH NUMBERS)
                        (REPLACE UMNEXT OF UMBOX WITH NUMBERS)
                        (SETFILEPTR MSTREAM 0)
                        (CREATE OPENEDMAILBOX
                                MAILBOX _ UMBOX
                                PROPERTIES _ (LIST '\#OFMESSAGES (LENGTH NUMBERS)))
                 ELSE (CL:UNLESS (ZEROP (GETEOFPTR MSTREAM))
                         (LAB.PROMPTPRINT MAILFOLDER "Mail spool file is not in Unix format: "
                                 (FETCH (UNIXMAILFILEINFO UMFNAME) OF (FETCH (MAILSERVER MAILPORT)
                                                                             OF MAILSERVER))
                                 ": "))
                      (CLOSEF MSTREAM)
                      'EMPTY))
    ELSE 'EMPTY)))
```

(UNIXSPOOL.RETRIEVEMESSAGE

```
(LAMBDA (MAILBOX MSGOUTFILE)                                   ; Edited 10-Mar-99 08:59 by rmk:
                                                               ; Edited 10-Mar-99 08:57 by rmk:
                                                               ; Edited 10-Mar-99 08:55 by rmk:
                                                               ; Edited 10-Mar-99 08:54 by rmk:
                                                               ; Edited 10-Mar-99 08:44 by rmk:
                                                               ; Edited 26-Feb-99 11:25 by rmk:
  (LET ((MSTREAM (|fetch| UMSTREAM |of| MAILBOX))
        (OUTSTART (GETFILEPTR MSGOUTFILE)))
       ;; The numbers in the UMNEXT of the mailbox are file positions in the spool file of the start of each message, so to get a message, just
```

```
        ;; COPYCHARS from the start of the current message to the start of the next one.
        ;;
        ;; NOTE, however, that a message in a Unix mailbox begins with a "From " line which should not be copied.
        (LET ((MSTART (|pop| (|fetch| UMNEXT |of| MAILBOX)))
              (MEND (OR (CAR (|fetch| UMNEXT |of| MAILBOX))
                        (GETEOFPTR MSTREAM))))
             ;; Confirm and skip the From line.
             (CL:UNLESS (EQ (CHARCODE F)
                            (READCCODE MSTREAM))
                 (ERROR "Not a valid Unix mail-spool file" (FULLNAME MSTREAM)))
             (UNTIL (MEMB (BIN MSTREAM)
                          (CHARCODE (LF CR))))
             (COPYCHARS MSTREAM MSGOUTFILE (GETFILEPTR MSTREAM)
                    MEND))
        ;; This could use a little error-handling, of course... perhaps a LOT of error handling.
        ;; Check out the tail of the message and uudecode it if it's an encoded message
        (UNIX.UUDECODE.IF.NEEDED MSGOUTFILE OUTSTART))))
```

(UNIXSPOOL.CLOSEMAILBOX

```
  (LAMBDA (MAILBOX FLUSH?)                                    ; Edited 10-Mar-99 08:45 by rmk:
                                                              ; Edited 15-Oct-90 20:46 by jrb:
    (LET ((MSTREAM (|fetch| UMSTREAM |of| MAILBOX)))
         ;; If FLUSH?, nuke the spool file
         (|if| FLUSH?
             |then| (SETFILEINFO (|fetch| (UNIXMAILFILEINFO UMFNAME) |of| (|fetch| (MAILSERVER MAILPORT) |of| MAILSERVER)
                                 )
                           'LENGTH 0)
                  ;; HACK!  Depends on \LAFITE.RETRIEVEMESSAGES binding of MAILSERVER
                  (|replace| (UNIXMAILFILEINFO UMFTIME) |of| (|fetch| (MAILSERVER MAILPORT) |of| MAILSERVER)
                     |with| (IDATE)))
         ;;  In any event, close the mailbox stream
         (CLOSEF MSTREAM)
         (DELFILE (FULLNAME (FETCH UMLOCKSTREAM OF MAILBOX))))))

)

;; Functions used to send mail

(DEFINEQ
```

(UNIX.FLUSH.STREAM

```
  (LAMBDA (STREAM CHAR)                                       ; Edited 13-Sep-90 15:58 by jrb:
    ;; Just vacuum out the stream until you see CHAR (if it's NIL, read until EOF)
    ;; If CHAR is supplied, stream must not be at EOF
    (|if| CHAR
        |then| (|until| (OR (NOT (READP STREAM))
                            (EQ (READCCODE STREAM)
                                CHAR))
                  |do| NIL)
      |else| (|until| (NOT (READP STREAM)) |do| (READCCODE STREAM)))
    STREAM))
```

(UNIX.RETRIEVE.LINE

```
  (LAMBDA (MSTREAM MSGOUTFILE)                                ; Edited 18-Sep-89 14:39 by jrb:
    ;; Copies a line of text from MSTREAM to MSGOUTFILE except if that line starts with a strange character (charcode 254; see
    ;; UNIX.RETRIEVEMESSAGE). Returns NIL on seeing such a line
    (BLOCK)                                                   ; This looks like a good place...
    (LET ((CHAR (READCCODE MSTREAM)))
         (|if| (EQ CHAR 254)
             |then| (SETQ CHAR NIL))
         ;; When we get here, if CHAR is non-NIL, it needs to be printed to MSGOUTFILE
         (|if| CHAR
             |then| (PRINTCCODE CHAR MSGOUTFILE)
                    (|until| (EQ CHAR (CHARCODE NEWLINE)) |do| (PRINTCCODE (SETQ CHAR (READCCODE MSTREAM))
                                                                            MSGOUTFILE))
                    T))))
```

(\\UNIXMAIL.SEND

```
  (LAMBDA (MSG PARSE EDITORWINDOW ABORTWINDOW)                ; Edited  4-Jul-99 21:26 by rmk:
                                                              ; Edited  2-Apr-99 15:44 by rmk:
                                                              ; Edited  6-Apr-97 17:27 by sybalsky:mv:envos
    ;; The strategy here is to talk to an SMTP server and throw the message at it.
```

```
(|if| (AND (TEDIT.FORMATTEDFILEP MSG)
           (EQ (|fetch| (UNIXMAILPARSE FORMATTED?) |of| PARSE)
               'TEDIT))
     |then| (\\LAFITE.SEND.FAIL EDITORWINDOW "UNIX mode can't send raw TEdit; try TEdit-UUencoded")
   |else|
    (\\UNIXMAIL.CHECK.ABORT ABORTWINDOW)
    (CL:MULTIPLE-VALUE-BIND (SMIN SMOUT)
        (\\UNIXMAIL.SMTP.TCP.STREAMS)
      (LET
       ((PWINDOW (AND EDITORWINDOW (GETPROMPTWINDOW EDITORWINDOW)))
        (RECIPIENTS (CL:REMOVE-DUPLICATES (FOR R IN (|fetch| (UNIXMAILPARSE UNIXTO) |of| PARSE)
                                             COLLECT (\\UNIXMAIL.REALADDRESS R))
                          :TEST
                          (FUNCTION STRING-EQUAL)))
        RESULT)
       (|if| (NULL SMIN)
           |then| (\\LAFITE.SEND.FAIL EDITORWINDOW (CL:FORMAT NIL "Couldn't open SMTP stream because ~s" SMOUT))
           )
       (CL:FLET
        ((GIVE-UP-AND-DIE (&REST WHY)
                ;; Close up the SMTP streams, send up a flare for the user and return NIL
                (IGNORE-ERRORS (CLOSEF SMIN)
                       (CLOSEF SMOUT))
                (\\LAFITE.SEND.FAIL EDITORWINDOW (CL:APPLY #'CL:FORMAT NIL WHY))
                NIL))
         ;; Get the connection ready to talk to
         (|for| I |from| 1 |to| 5 |while| (SETQ RESULT (\\UNIXMAIL.SMTP.FLUSH SMIN)) |do| NIL)
         (|if| RESULT
             |then| (GIVE-UP-AND-DIE "Couldn't open SMTP stream because ~a" RESULT)
           |else| (|if| PWINDOW
                      |then| (CLEARW PWINDOW)
                           (CL:FORMAT PWINDOW "Delivering ~:[~;formatted ~]to ~D recipient~:P"
                                  (|fetch| (UNIXMAILPARSE FORMATTED?) |of| PARSE)
                                  (LENGTH RECIPIENTS)))
                  (|if| (SETQ RESULT (CL:CATCH 'SMTP-LOST
                                       ;; First, announce who we are
                                       (\\UNIXMAIL.SMTP SMIN SMOUT (CL:FORMAT NIL "HELO ~a~%" (UNIX-GETPARM
                                                                                  "HOSTNAME")))
                                       ;; Then send who it's from
                                       (\\UNIXMAIL.SMTP SMIN SMOUT (CL:FORMAT NIL "MAIL FROM: <~a>~%"
                                                                          (\\UNIXMAIL.FQNAME (|fetch| (
                                                                                  LAFITEMODEDATA
                                                                                  SHORTUSERNAME
                                                                                  )
                                                                                |of|
                                                                                 *LAFITE-MODE-DATA*
                                                                                    ))))
                                       ;; Then the recipients
                                       (|for| R |in| RECIPIENTS |do| (\\UNIXMAIL.SMTP SMIN SMOUT (CL:FORMAT NIL
                                                                                  "RCPT TO:
                                                                                  <~a>~%" R)))
                                       ;; Print a '.' to show we're this far
                                       (AND PWINDOW (|printout| PWINDOW '\.))
                                       ;; Then the message itself; the SMTP response here is a 300-range number meaning "Lay it on me"
                                       (\\UNIXMAIL.SMTP SMIN SMOUT "DATA~%" 300 399)
                                       ;; First hose out the other fields
                                       (CL:FORMAT SMOUT "From: ~a~%" (\\UNIXMAIL.FQNAME (|fetch| (LAFITEMODEDATA
                                                                                  SHORTUSERNAME)
                                                                                |of| *LAFITE-MODE-DATA*)
                                                                          ))
                                       (|for| F |in| (|fetch| (UNIXMAILPARSE UNIXOTHER) |of| PARSE)
                                          |do| (CL:FORMAT SMOUT "~a: ~a~%" (CAR F)
                                                    (CADR F))
                                          |finally| (|if| (|fetch| (UNIXMAILPARSE FORMATTED?) |of| PARSE)
                                                      |then| (CL:FORMAT SMOUT "Format: ~a~%" (|fetch| (UNIXMAILPARSE
                                                                                  FORMATTED?)
                                                                                |of| PARSE)))
                                                (TERPRI SMOUT))
                                       ;; Print a '.' to show we're this far
                                       (AND PWINDOW (|printout| PWINDOW '\.))
                                       (|if| (TEDIT.FORMATTEDFILEP MSG)
                                           |then| (SELECTQ (|fetch| (UNIXMAILPARSE FORMATTED?) |of| PARSE)
                                                      (TEDIT (GIVE-UP-AND-DIE "UNIX mode can't send raw TEdit; try
                                                                  TEdit-UUencoded"))
                                                      (TEDIT-UUENCODE ;; Nuke the header in a copy of the msg
                                                           (SETQ MSG (COPYTEXTSTREAM MSG))
                                                           (TEDIT.DELETE MSG 1 (|fetch| (UNIXMAILPARSE UNIXBODY)
```

```
                                                                |of| PARSE))
                                                    (UNIX.TEDIT-UUENCODE.MESSAGE MSG SMOUT))
                                                (NIL ;; Strip TEdit formatting
                                                     (\\SMTP-DUMP (LAFITE.MAKE.PLAIN.TEXTSTREAM
                                                                   MSG
                                                                   (|fetch| (UNIXMAILPARSE UNIXBODY)
                                                                      |of| PARSE))
                                                            SMOUT))
                                                (GIVE-UP-AND-DIE "Send failed due to strange Format: ~A"
                                                       (|fetch| (UNIXMAILPARSE FORMATTED?) |of| PARSE)))
                                          |else| (\\SMTP-DUMP MSG SMOUT))
                                     (|if| (OPENP SMOUT)
                                        |then| (\\UNIXMAIL.CHECK.ABORT ABORTWINDOW SMIN SMOUT)
                                               (\\UNIXMAIL.SMTP SMIN SMOUT "~%.~%")
                                               ;; Print a '.' to show we're this far
                                               (AND PWINDOW (|printout| PWINDOW '\.))
                                               (\\UNIXMAIL.SMTP SMIN SMOUT "QUIT~%")
                                               NIL
                                        |else| (BLOCK 1000)
                                               554)))
                               |then| (GIVE-UP-AND-DIE "SMTP error is ~a" RESULT)
                              |else| (\\UNIXMAIL.CHECK.ABORT ABORTWINDOW SMIN SMOUT)
                                     (IGNORE-ERRORS (CLOSEF SMIN)
                                            (CLOSEF SMOUT))
                                     (LENGTH RECIPIENTS)))))))))
```

(**UNIXMAIL.SEND.WRAPLINES**

```
(LAMBDA (INSTREAM OUTSTREAM)                                    ; Edited  4-Jul-99 21:14 by rmk:
                                                                ; Edited  4-Jul-99 21:11 by rmk:
  (LET ((BUF (CL:MAKE-ARRAY (IPLUS UNIXMAIL.WRAP-LIMIT 8)
                    :ELEMENT-TYPE
                    'CL:CHARACTER))
        (BPTR 0)
        (BLENGTH 0)
        (LASTWS NIL)
        C)
       (CL:FLET ((ADDCHAR (C)
                         (CL:SETF (CL:CHAR BUF BPTR)
                                  C)
                         (CL:INCF BPTR)
                         (CL:INCF BLENGTH))
                 (DUMP-BUFFER NIL (CL:DOTIMES (I BPTR)
                                          (CL:WRITE-CHAR (CL:CHAR BUF I)
                                                 OUTSTREAM))
                         (TERPRI OUTSTREAM)
                         (SETQ BPTR 0 BLENGTH 0)))
              (|while| (SETQ C (CL:READ-CHAR INSTREAM NIL NIL))
                 |do| (CASE C
                           (#\Newline
                              (DUMP-BUFFER)
                              (SETQ LASTWS NIL))
                           (#\.
                              (|if| (EQ 0 BPTR)
                                 |then| (ADDCHAR C))
                              (ADDCHAR C))
                           (#\Space
                              (SETQ LASTWS BPTR)
                              (ADDCHAR C))
                           (#\Tab
                              (CL:INCF BLENGTH (CL:1- UNIXMAIL.TABWIDTH))
                              (SETQ BLENGTH (ITIMES UNIXMAIL.TABWIDTH (CL:FLOOR BLENGTH UNIXMAIL.TABWIDTH)))
                              (ADDCHAR C))
                           (CL:OTHERWISE (ADDCHAR C)))
                      (|if| (IGREATERP BLENGTH UNIXMAIL.WRAP-LIMIT)
                          |then| (|if| LASTWS
                                    |then| (SETQ C (IDIFFERENCE (IDIFFERENCE BPTR LASTWS)
                                                           1))
                                           (SETQ BPTR LASTWS)
                                           (DUMP-BUFFER)
                                           (|if| (EQ (CL:CHAR BUF (CL:INCF LASTWS))
                                                     #\.)
                                               |then| (CL:SETF (CL:CHAR BUF (CL:DECF LASTWS))
                                                               #\.)
                                                      (CL:INCF C))
                                           (|for| I |from| 0 |to| (IDIFFERENCE C 1) |as| BPTR |from| LASTWS
                                              |do| (CL:SETF (CL:CHAR BUF I)
                                                            (CL:CHAR BUF BPTR)))
                                           (SETQ BPTR C BLENGTH C LASTWS NIL)
                                    |else| (DUMP-BUFFER)
                                           (SETQ LASTWS NIL)))
                 |finally| (DUMP-BUFFER)))))
```

(**SMTP-DUMP**

```
(LAMBDA (INSTREAM OUTSTREAM)                                    ; Edited  4-Jul-99 21:20 by rmk:
                                                                ; Edited  4-Jul-99 21:17 by rmk:
    ;; In both wrapped and unwrapped cases, we have to treat specially lines beginning with '.': the '.' must be doubled when talking to the SMTP
    ;; server.
    (|if| UNIXMAIL.WRAP.LINES
        |then| (\\UNIXMAIL.SEND.WRAPLINES INSTREAM OUTSTREAM)
      |else| (LET ((STARTC (GETFILEPTR INSTREAM))
                   ENDC)
                  (WHILE (SETQ ENDC (FILEPOS (CONSTANT (CONCAT (CHARACTER (CHARCODE NEWLINE))
                                                               "."))
                                               INSTREAM STARTC))
                     DO (COPYCHARS INSTREAM OUTSTREAM STARTC (ADD1 ENDC))
                        (PRIN1 ".." OUTSTREAM)
                        (SETQ STARTC (IPLUS ENDC 2))
                     FINALLY (COPYCHARS INSTREAM OUTSTREAM STARTC (GETEOFPTR INSTREAM)))))))
```

(\\UNIXMAIL.SEND.PARSE

```
(LAMBDA (MSG EDITORWINDOW)                                      ; Edited 30-Jun-99 23:54 by rmk:
                                                                ; Edited 28-Feb-99 19:27 by rmk:
                                                                ; Edited 26-Feb-99 23:59 by rmk:
                                                                ; Edited 20-Sep-91 16:59 by jrb:
    ;; Do some obvious checks here and build a \UNIXMAILPARSE for UNIXMAIL.SEND to munch on
    (PROG ((MSGFIELDS (\\LAFITE.PREPARE.SEND MSG EDITORWINDOW))
           SENDINGFORMAT HEADEREOF FROMFIELD RECIPIENTS OTHERSTUFF SUBJECT)
          (|if| MSGFIELDS
              |then| (|if| (EQ (CAAR MSGFIELDS)
                               'EOF)
                           |then| (SETQ HEADEREOF (CADR (|pop| MSGFIELDS))))
                     (|for| PAIR |in| MSGFIELDS |do| (SELECTQ (CAR PAIR)
                                                              (|Date| (\\SENDMESSAGEFAIL EDITORWINDOW "User-supplied Date
                                                                      not allowed"))
                                                              (|Sender| (\\SENDMESSAGEFAIL EDITORWINDOW "User-supplied
                                                                        Sender not allowed"))
                                                              ((TO T\o |cc|)
                                                                   (SETQ RECIPIENTS (NCONC RECIPIENTS (
                                                                                                      \\UNIXMAIL.PARSENAMES
                                                                                                       (CDR PAIR))))
                                                                   (|push| OTHERSTUFF PAIR))
                                                              (|From| (SETQ FROMFIELD (\\UNIXMAIL.PARSENAMES (CDR PAIR)))
                                                                      (|push| OTHERSTUFF PAIR))
                                                              (|Format| (SETQ SENDINGFORMAT (CADR PAIR)))
                                                              (|Subject| (SETQ SUBJECT (CADR PAIR))
                                                                         (|push| OTHERSTUFF PAIR))
                                                              (|push| OTHERSTUFF PAIR)))
                     (|if| (NULL RECIPIENTS)
                         |then| (\\SENDMESSAGEFAIL EDITORWINDOW "No recipients!")
                                (RETURN NIL))
                     (|if| (NULL SENDINGFORMAT)
                         |then| (SETQ SENDINGFORMAT (OR (\\LAFITE.CHOOSE.MSG.FORMAT MSG HEADEREOF EDITORWINDOW)
                                                        (RETURN))))
                     (RETURN (|create| UNIXMAILPARSE
                                       UNIXMAILSUBJECT _ SUBJECT
                                       UNIXFROM _ FROMFIELD
                                       UNIXTO _ RECIPIENTS
                                       UNIXOTHER _ OTHERSTUFF
                                       UNIXBODY _ HEADEREOF
                                       FORMATTED? _ (SELECTQ SENDINGFORMAT
                                                             ((NIL TEXT)
                                                                 NIL)
                                                             SENDINGFORMAT)))))))
```

(\\UNIXMAIL.CHECK.ABORT

```
(LAMBDA (ABORTWINDOW SMIN SMOUT)                                ; Edited  2-Apr-99 14:37 by rmk:
                                                                ; Edited  2-Apr-99 14:36 by rmk:
                                                                ; Edited  2-Apr-99 13:52 by rmk:
                                                                ; Edited  2-Apr-99 12:31 by rmk:
                                                                ; Perhaps the Abort button was pushed?
    (BLOCK)
    (CL:WHEN (AND ABORTWINDOW (WINDOWPROP ABORTWINDOW 'ABORT))
        (CL:WHEN SMIN
            (IGNORE-ERRORS (CLOSEF SMIN)
                   (CLOSEF SMOUT)))
        (ERROR!))))
```

(\\UNIXMAIL.MUNG.RECIPIENTS

```
(LAMBDA (RECIPIENTS)                                            ; Edited 11-Mar-99 16:38 by rmk:
    ;; Coerces addresses from XNS formats to Internet formats, needed for answering messages that arrived via XNS.
    (FOR R COLPOS IN RECIPIENTS
       COLLECT (IF (NOT (SETQ COLPOS (STRPOS ":" R)))
                   THEN ;; Either a vanilla name or an Internet address, mailer will handle it
```

```
                    (IF (STRPOS ".XEROX" R -6 NIL T NIL UPPERCASEARRAY)
                        THEN                                    ; domain was stripped out of unix address
                             (SETQ R (CONCAT R ".com")))
               ELSEIF (STRPOS "@" R)
                  THEN ;; A converted internet address, fix up last domain.  foo@fum.fie:Xerox, foo@fum.fie:SMTP:Xerox, foo@fum:fie all go
                       ;; to foo@fum.fie
                    (IF (STRPOS ":XEROX" R -6 NIL T NIL UPPERCASEARRAY)
                        THEN ;; Strip Xerox
                             (SETQ R (SUBSTRING R 1 -7))
                             (IF (IGREATERP COLPOS (NCHARS R))
                               ELSEIF (STRPOS ":SMTP" R -5 T NIL UPPERCASEARRAY)
                                  THEN ;; Another colon, followed by SMTP
                                       (SETQ R (SUBSTRING R 1 -6))
                                ELSE ;; Another colon not followed by SMTP
                                     (RPLCHARCODE R COLPOS (CHARCODE \.)))
                      ELSE ;; No Xerox, last segment is domain
                           (RPLCHARCODE R COLPOS (CHARCODE \.)))
               ELSE ;; A pure NS address:  foo:bar:bas -> foo.bar@baz.xerox.com, foo:bar:xerox -> foo.bar@xerox.com
                    (LET ((FOO (SUBSTRING R 1 (SUB1 COLPOS)))
                          (BAR (SUBSTRING R (ADD1 COLPOS)
                                          (SUB1 (OR (SETQ COLPOS (STRPOS ":" R (ADD1 COLPOS)))
                                                    0))))
                          (BAZ (AND COLPOS (SUBSTRING R (ADD1 COLPOS)))))
                         ;; Spaces in FOO and BAR go to underscore
                         (FOR I C FROM 1 WHILE (SETQ C (NTHCHARCODE FOO I))
                            WHEN (EQ C (CHARCODE SPACE)) DO (RPLCHARCODE FOO I (CHARCODE _)))
                         (CL:WHEN BAR
                             (FOR I C FROM 1 WHILE (SETQ C (NTHCHARCODE BAR I))
                                WHEN (EQ C (CHARCODE SPACE)) DO (RPLCHARCODE BAR I (CHARCODE _))))
                         ;; Spaces in BAZ disappear
                         (CL:WHEN (AND BAZ (STRPOS " " BAZ))
                             (FOR I C (NEWI _ 1)
                                  (NEWBAZ _ (ALLOCSTRING (NCHARS BAZ))) FROM 1
                                WHILE (SETQ C (NTHCHARCODE BAZ I)) UNLESS (EQ C (CHARCODE SPACE))
                                DO (RPLCHARCODE NEWBAZ NEWI C)
                                   (ADD NEWI 1)
                                FINALLY (SETQ BAZ (SUBSTRING NEWBAZ 1 (SUB1 NEWI)
                                                             (CONSTANT (CONCAT))))))
                         (IF (STREQUAL (L-CASE BAZ)
                                       "xerox")
                             THEN (SETQ BAZ NIL))
                         (SETQ R (IF BAZ
                                     THEN (CONCAT FOO "." BAR "@" BAZ ".xerox.com")
                                   ELSE (CONCAT FOO "." BAR "@xerox.com")))))
          R)))
```

(**UNIXMAIL.SMTP**

```
(LAMBDA (SIN SOUT STRING LO HI)                                 ; Edited  6-Apr-97 17:30 by sybalsky:mv:envos
```

;; Very dumb protocol handler; shows STRING to STREAM and reads the response number, returning T or NIL depending on the range of the
;; result. Currently we only accept 200-range answers (completions)

```
(LET (RESULT)
     (CL:FORMAT SOUT STRING)
     (FORCEOUTPUT SOUT)
     (|if| (SETQ RESULT (\\**UNIXMAIL.SMTP.FLUSH** SIN LO HI))
         |then| (CL:THROW 'SMTP-LOST RESULT))))
```

(**UNIXMAIL.SMTP.FLUSH**

```
(LAMBDA (STREAM LO HI)                                          ; Edited  6-Jul-99 15:16 by rmk:
                                                                ; Edited  4-Jul-99 12:55 by rmk:
                                                                ; Edited  8-Mar-99 13:08 by N.H.Briggs
```

;; STREAM is about to throw an SMTP exchange at us; get the number and shake hands appropriately

```
(LET (RESULT ERRORTYPE CHAR)
     ;; Flush any continuation result codes
     (|while| (EQUAL "-" (SUBSTRING (FOR RPT FROM 1 TO 10 DO
                                     ;; Try again 10 times on end-of-file error, hoping that the error comes from a timing glitch.  Both
                                     ;; RESULT and the value of the loop will be NIL for any other kind of error and for the 10th end-of-file,
                                     ;; which will be reported as the error type.
                                     (CL:MULTIPLE-VALUE-SETQ (RESULT ERRORTYPE)
                                            (IGNORE-ERRORS (RATOM STREAM)))
                                     (CL:WHEN RESULT (RETURN RESULT))
                                REPEATWHILE (EQ 'END-OF-FILE (TYPENAME ERRORTYPE)))
                                4 4))
        DO (UNIX.FLUSH.STREAM STREAM (CHARCODE NEWLINE)))
```

```
          (|if| (NOT (SMALLP RESULT))
              |then| (UNIX.FLUSH.STREAM STREAM (CHARCODE NEWLINE))
                     (CL:FORMAT NIL "Not an SMTP number:~S" (OR ERRORTYPE RESULT))
            |else| (|if| (OR (IGREATERP RESULT (OR HI 299))
                             (ILESSP RESULT (OR LO 200)))
                       |then| (SETQ RESULT (CL:FORMAT NIL "~a: " RESULT))
                              (|until| (OR (NOT (READP STREAM))
                                           (EQ (SETQ CHAR (READCCODE STREAM))
                                               (CHARCODE NEWLINE)))
                                 |do| (SETQ RESULT (CONCAT RESULT (CHARACTER CHAR))))
                              RESULT
                     |else| (UNIX.FLUSH.STREAM STREAM (CHARCODE NEWLINE))
                            NIL)))))
```

(**UNIXMAIL.CHANGE.MODE**

```
  (LAMBDA (WINDOW TEXTSTREAM MENU ITEM)
    (SELECTQ (MENU (|create| MENU
                             TITLE _ "Mail mode"
                             ITEMS _ `(("Change Mode" '\\SENDMSG.CHANGE.MODE "Change the mail protocol used to send
                                                                              this message")
                                       (,(|if| UNIXMAIL.WRAP.LINES
                                              |then| "Line wrap Off"
                                            |else| "Line wrap On")
                                        'UNIXMAIL.WRAP.LINES
                                        ,(|if| UNIXMAIL.WRAP.LINES
                                              |then| "Send message as is"
                                            |else| "Insert newlines to make message lines shorter than
                                                    UNIXMAIL.WRAP-LIMIT")))))
           (\\SENDMSG.CHANGE.MODE
                (\\SENDMSG.CHANGE.MODE WINDOW TEXTSTREAM MENU ITEM))
           (UNIXMAIL.WRAP.LINES
                (SETQ UNIXMAIL.WRAP.LINES (NOT UNIXMAIL.WRAP.LINES))
                (\\SENDMESSAGE.PROMPT WINDOW (|if| UNIXMAIL.WRAP.LINES
                                                   |then| "Line wrapping is on"
                                                 |else| "Line wrapping is off")))
           NIL)
        ;; Exit with error so that the window is restored to previous state
        (ERROR!)))

)

;; This returns multiple-values, so it's a CL:LAMBDA (what the heck).

(CL:DEFUN \\**UNIXMAIL.SMTP.TCP.STREAMS** () ; Edited 27-Feb-99 13:55 by rmk:
    ;; Opens two streams representing the input and output streams of an SMTP TCP connection.  On failure return NIL and a string describing the
    ;; failure.
    (SELECTQ UNIXMAIL.SEND.MODE
        (PROCESS (|if| (EQ (MACHINETYPE)
                           'MAIKO)
                       |then| ;; UNIXMAIL.SEND.PROCESS can be a list of possibilities because the process may be in different places in
                               ;; different operating systems (e.g. solaris vs. sunos).  If the first one doesn't exist at this time, we search the
                               ;; remaining ones and move the first one we find to the beginning of the list for next time.  This could be done as an
                               ;; AFTERSYSOUTFORMS, but easy enough just to do it here.
                               (LET ((S (CREATE-PROCESS-STREAM
                                         (CONCAT (IF (NLISTP UNIXMAIL.SEND.PROCESS)
                                                     THEN UNIXMAIL.SEND.PROCESS
                                                   ELSEIF (INFILEP (PACKFILENAME 'HOST 'DSK 'BODY (CAR
                                                                                                     UNIXMAIL.SEND.PROCESS
                                                                                                           )))
                                                     THEN (CAR UNIXMAIL.SEND.PROCESS)
                                                   ELSE (FOR P IN (CDR UNIXMAIL.SEND.PROCESS)
                                                           WHEN (INFILEP (PACKFILENAME 'HOST 'DSK 'BODY P))
                                                           DO (SETQ UNIXMAIL.SEND.PROCESS (CONS P (DREMOVE P
                                                                                                     UNIXMAIL.SEND.PROCESS
                                                                                                                )))
                                                              (RETURN P)))
                                                 (IF UNIXMAIL.SEND.HOST
                                                     THEN (CONCAT " " UNIXMAIL.SEND.HOST)
                                                   ELSE ""))))))
                                     (CL:VALUES S S))
                             |else| (CL:VALUES NIL "this MACHINETYPE can't do Unix process-streams; change
                                                    UNIXMAIL.SEND.MODE")))
        (SOCKET (|if| (EQ (MACHINETYPE)
                          'MAIKO)
                      |then| (LET ((S (OPENTCPSTREAM (OR UNIXMAIL.SEND.HOST (UNIX-GETPARM "HOSTNAME"))
                                                     25)))
                               (CL:VALUES S S))
                    |else| (LET ((S (TCP.OPEN UNIXMAIL.SEND.HOST 25 NIL 'ACTIVE 'INPUT T)))
                             (|if| S
                                 |then| (CL:VALUES S (TCP.OTHER.STREAM S))
                               |else| (CL:VALUES NIL "TCP.OPEN failed; check your Lisp TCP configuration")))))
        (ERROR "Unrecognized UNIXMAIL.SEND.MODE:" UNIXMAIL.SEND.MODE)))
```

```
;; Other functions Lafite uses and needs Unix equivalents for

(DEFINEQ

(\\UNIXMAIL.AUTHENTICATE
  (LAMBDA NIL                                                    ; Edited  2-Apr-99 15:33 by rmk:
                                                                 ; Edited  2-Apr-99 15:25 by rmk:
                                                                 ; Edited 10-Mar-99 18:00 by N.H.Briggs
    ;; No authentication really necessary (we're depending on underlying Unix to keep us from doing illegal things), so just return an appropriate
    ;; LAFITEMODEDATA

    ;; Unfortunately, all the various versions of the username have to be the short one, because Lafite insists they all be acceptable mailing addresses,
    ;; in particular FULLUSERNAME... *sigh*...

    (LET ((FQNAME (\\UNIXMAIL.FQNAME (UNIX-USERNAME))))
         (|create| LAFITEMODEDATA
                LAFITEOPS _ (FASSOC 'UNIX LAFITEMODELST)
                FULLUSERNAME _ (CONCAT (UNIX-FULLNAME)
                                       " <" FQNAME ">")
                UNPACKEDUSERNAME _ FQNAME
                SHORTUSERNAME _ (UNIX-USERNAME)
                MAILSERVERS _ (LIST (|create| MAILSERVER
                                            MAILSERVERNAME _ "Unix mail"
                                            MAILSERVEROPS _ (OR (CDR (FASSOC UNIXMAIL.RECEIVE.MODE
                                                                             UNIXMAIL.MSOPS.LIST))
                                                                (ERROR "Not a known Unix mail receive mode"
                                                                       UNIXMAIL.RECEIVE.MODE)))))))))

(\\UNIXMAIL.LOGIN
  (LAMBDA NIL                                                    ; Edited  4-Jan-89 13:29 by bane
    (|if| (EQ (MACHINETYPE)
              'MAIKO)
        |then| (\\LAFITE.GET.USER.DATA 'UNIX NIL T)
               (\\LAFITE.WAKE.WATCHER)
      |else| (PROMPTPRINT "No Unix mode; this isn't Maiko"))))

(\\UNIXMAIL.PARSENAMES
  (LAMBDA (FIELD)                                                ; Edited  7-Mar-99 23:31 by rmk:
                                                                 ; Edited  7-Mar-99 23:28 by rmk:
                                                                 ; Edited  4-Jan-89 15:41 by bane
    ;; Just returns a list of the unquoted, unparenthesized comma-seperated fields in this string

    (FOR F RESULT INSIDE FIELD
       DO (FOR I C SEGMENT (START _ 1) FROM 1
             DO (SELCHARQ (SETQ C (NTHCHARCODE F I))
                          ((\, NIL)
                               (SETQ SEGMENT (CL:STRING-TRIM "       " (SUBSTRING F START (SUB1 I))))
                               (PUSH RESULT (\\UNIXMAIL.FIXMICROSOFT SEGMENT))
                               (CL:UNLESS C (RETURN))
                               (SETQ START (ADD1 I)))
                          (\" (FOR OLD I FROM (ADD1 I) BY 1 WHILE (SETQ C (NTHCHARCODE F I))
                                 WHEN (EQ C (CHARCODE \")) DO (RETURN I) FINALLY
                                                                 ; ran off the end
                                                                       (SETQ I (SUB1 I))))
                          (\( (FOR OLD I (PDEPTH _ 1) FROM (ADD1 I) BY 1 WHILE (SETQ C (NTHCHARCODE F I))
                                 DO (SELCHARQ C
                                              (\( (ADD PDEPTH 1))
                                              (\) (IF (EQ PDEPTH 1)
                                                      THEN (RETURN I)
                                                    ELSE (ADD PDEPTH -1)))
                                              NIL)
                                 FINALLY                         ; ran off the end
                                         (SETQ I (SUB1 I))))
                          NIL))
       FINALLY (RETURN (\\UNIXMAIL.MUNG.RECIPIENTS (DREVERSE RESULT))))))

(\\UNIXMAIL.MAKEANSWERFORM
  (LAMBDA (MSGDESCRIPTORS MAILFOLDER)                            ; Edited 11-Oct-91 09:13 by jrb:
    ;;  Code borrowed liberally from GV.MAKEANSWERFORM

    (LET ((MSGFIELDS (\\LAFITE.PARSE.MESSAGE MAILFOLDER (OR (CAR (LISTP MSGDESCRIPTORS))
                                                            MSGDESCRIPTORS)))
          SUBJECT FROM DATE SENDER REPLYTO TO CC OLDFROM NEWTO NEWCC)
                                                                 ; get the fields from the file
         (|for| PAIR |in| MSGFIELDS |do| (SELECTQ (CAR PAIR)
                                        (|Subject| (SETQ SUBJECT (CADR PAIR)))
                                        (|Sender| (SETQ SENDER (CADR PAIR)))
                                        (|From| (SETQ FROM (CADR PAIR)))
                                        (|Date| (SETQ DATE (CADR PAIR)))
                                        (|Reply-to| (SETQ REPLYTO (CDR PAIR)))
                                        ((T\o TO)
                                            (SETQ TO (CDR PAIR)))
                                        (|cc| (SETQ CC (CDR PAIR)))
```

```
                                        NIL))
    ;; first parse the strings into recipients.  Need to find the sender's registry in order to get the registry defaults correct for its recipients.
    (COND
       (SENDER                                                  ; Sender is a mail address, and has the official registry
               (SETQ OLDFROM (\\UNIXMAIL.PARSENAMES SENDER))    ; Elements are of the form (prettyname gvname . registry)
               (|if| FROM
                   |then|                                       ; Now that we have a source of official registry (we hope), parse
                                                                ; the From field with reference to it.
                         (SETQ OLDFROM (\\UNIXMAIL.PARSENAMES FROM))))
       (FROM                                                    ; Have to parse the From field before we can get its registry
               (SETQ OLDFROM (\\UNIXMAIL.PARSENAMES FROM))))
    (|if| (NULL OLDFROM)
        |then| (LAB.PROMPTPRINT MAILFOLDER T "Warning: message has no FROM field"))
    (AND TO (SETQ TO (\\UNIXMAIL.PARSENAMES TO)))
    (AND CC (SETQ CC (\\UNIXMAIL.PARSENAMES CC)))
    (SETQ NEWTO (OR (AND REPLYTO (SETQ REPLYTO (\\UNIXMAIL.PARSENAMES REPLYTO)))
                    OLDFROM))
    (SETQ NEWCC (CL:SET-DIFFERENCE (COND
                                      (REPLYTO                  ; Reply goes only to this address, so the only cc is to self
                                             (MKLIST (|fetch| (LAFITEMODEDATA SHORTUSERNAME) |of|
                                                                                    *LAFITE-MODE-DATA*
                                              )))
                                      (T                        ; By default CC everyone who received the original message and
                                                                ; to whom we are not directly replying already
                                         (APPEND TO (CL:SET-DIFFERENCE CC TO :TEST #'STRING-EQUAL))))
                        NEWTO :TEST #'STRING-EQUAL))
    (LAFITE.FILL.IN.ANSWER.FORM SUBJECT (|if| (AND (OR (NULL REPLYTO)
                                                       (EQUAL REPLYTO OLDFROM))
                                                   (NULL (CDR NEWCC))
                                                   (OR (NULL NEWCC)
                                                       (STRING-EQUAL (CAR NEWCC)
                                                              (|fetch| (LAFITEMODEDATA SHORTUSERNAME)
                                                                  |of| *LAFITE-MODE-DATA*))))
                                              |then|            ; Replying only to sender (and maybe self), so just say "your"
                                                                ; instead of "Joe Bob Smith <JBSmith.pa>'s"
                                                   NIL
                                              |else| FROM)
                DATE NEWTO NEWCC #'LA.PRINT.COMMA.LIST))))
```

(**UNIXMAIL.MESSAGE.FROM.SELF.P**

```
(LAMBDA (MSG)                                                   ; Edited  6-Jul-99 15:22 by rmk:
                                                                ; Edited  2-Apr-99 15:27 by rmk:
                                                                ; Edited  6-Dec-88 15:41 by bane
```

;; For the moment, we send stuff with our SHORTUSERNAME only in the FROM field

```
(OR (STRING-EQUAL (\\UNIXMAIL.FQNAME (|fetch| (LAFITEMSG FROM) |of| MSG))
                  (\\UNIXMAIL.FQNAME (|fetch| (LAFITEMODEDATA SHORTUSERNAME) |of| *LAFITE-MODE-DATA*)))
    (STRING-EQUAL (\\UNIXMAIL.REALADDRESS (|fetch| (LAFITEMSG FROM) |of| MSG))
                  (\\UNIXMAIL.FQNAME (|fetch| (LAFITEMODEDATA SHORTUSERNAME) |of| *LAFITE-MODE-DATA*))))
```

(**UNIXMAIL.MESSAGE.P**

```
(LAMBDA (MSG)                                                   ; Edited  6-Dec-88 15:39 by bane
```

;; We're guessing here; basically if it doesn't look like an NS message, say maybe.

```
(AND (NOT (STRPOS ":" (|fetch| (LAFITEMSG FROM) |of| MSG)))
     '?))
```

(**UNIXMAIL.REALADDRESS**

```
(LAMBDA (R)                                                     ; Edited  1-Jul-99 00:00 by rmk:
                                                                ; Edited 30-Jun-99 23:57 by rmk:
                                                                ; Edited  2-Apr-99 15:30 by rmk:
```

;; Finds the true address inside R. We look for angle brackets outside of double-quotes. If that doesn't work, send the whole string with parenthetic
;; material replaced by white space.

```
(FOR I C (STARTTEXT _ 1)
     SEGMENTS FROM 1 WHILE (SETQ C (NTHCHARCODE R I))
   DO (SELCHARQ C
             (<  ;; If we find a top-level angle bracket, we're done
                 (RETURN (\\UNIXMAIL.FQNAME (CL:STRING-TRIM "  " (SUBSTRING R (ADD1 I)
                                                                    (SUB1 (OR (STRPOS ">" R (ADD1 I))
                                                                              0)))))))
             (\" (CL:UNLESS (EQ I STARTTEXT)
                    (PUSH SEGMENTS (LIST 'TEXT STARTTEXT (SUB1 I))))
                 (PUSH SEGMENTS (LIST 'QUOTES I (FOR OLD I FROM (ADD1 I) BY 1
                                                   WHILE (SETQ C (NTHCHARCODE R I))
                                                   WHEN (EQ C (CHARCODE \")) DO (RETURN I)
                                                   FINALLY    ; ran off the end
                                                          (SETQ I (SUB1 I)))))
                 (SETQ STARTTEXT (ADD1 I)))
             (\( (CL:UNLESS (EQ I STARTTEXT)
                    (PUSH SEGMENTS (LIST 'TEXT STARTTEXT (SUB1 I))))
                 (PUSH SEGMENTS (LIST 'PARENS I (FOR OLD I (PDEPTH _ 1) FROM (ADD1 I) BY 1
```

```
                                                    WHILE (SETQ C (NTHCHARCODE R I))
                                                    DO (SELCHARQ C
                                                           (\( (ADD PDEPTH 1))
                                                           (\) (IF (EQ PDEPTH 1)
                                                                   THEN (RETURN I)
                                                                 ELSE (ADD PDEPTH -1)))
                                                           NIL)
                                                    FINALLY     ; ran off the end
                                                          (SETQ I (SUB1 I)))))
                    (SETQ STARTTEXT (ADD1 I)))
                NIL)
        FINALLY (RETURN (\\UNIXMAIL.FQNAME (IF (NULL SEGMENTS)
                                               THEN R
                                             ELSE (CL:STRING-TRIM
                                                    "     "
                                                    (CONCATLIST (FOR S IN SEGMENTS
                                                                   COLLECT (SELECTQ (CAR S)
                                                                                    ((TEXT QUOTES)
                                                                                         (SUBSTRING R (CADR S)
                                                                                                    (CADDR S)))
                                                                                    (PARENS " ")
                                                                                    R)))))))))
```

(\\UNIXMAIL.FQNAME

```
(LAMBDA (NAME)                                          ; Edited  2-Apr-99 15:24 by rmk:
                                                        ; Edited  2-Apr-99 15:23 by rmk:
                                                        ; Edited  2-Apr-99 15:21 by rmk:
    ;; Returns the fully qualified version of name
    (IF (OR (STRPOS "@" NAME)
            (NULL UNIXMAIL.DOMAIN.NAME)
            (EQ 0 (NCHARS NAME)))
        THEN NAME
      ELSE (CONCAT NAME "@" UNIXMAIL.DOMAIN.NAME))))
```

(\\UNIXMAIL.FIXMICROSOFT

```
(LAMBDA (STRING)                                        ; Edited  7-Mar-99 16:13 by rmk:
                                                        ; Edited  7-Mar-99 16:10 by rmk:
                                                        ; Edited  7-Mar-99 16:10 by rmk:
                                                        ; Edited  7-Mar-99 16:09 by rmk:
                                                        ; Edited  7-Mar-99 16:08 by rmk:
                                                        ; Edited  7-Mar-99 16:05 by rmk:
                                                        ; Edited  7-Mar-99 16:03 by rmk:
                                                        ; Edited  7-Mar-99 16:02 by rmk:
                                                        ; Edited  7-Mar-99 15:54 by rmk:
                                                        ; Edited  7-Mar-99 15:53 by rmk:
                                                        ; Edited  7-Mar-99 15:52 by rmk:
                                                        ; Edited  7-Mar-99 15:18 by rmk:
                                                        ; Edited  7-Mar-99 15:15 by rmk:
    (SETQ STRING (CL:STRING-TRIM "       " STRING))
    ;; Try to simplify goofy addresses from Microsoft mailer.  "'foo <abc>'" "<abc>" goes to foo <abc>.  Foo is quoted if it contains commas
    (DO                                                 ; Strip leading blanks
        (SELCHARQ (CHCON1 STRING)
             ((SPACE TAB)
                  (GNC STRING))
             (RETURN)))
    (DO                                                 ; Strip trailing blanks
        (SELCHARQ (NTHCHARCODE STRING -1)
             ((SPACE TAB)
                  (GLC STRING))
             (RETURN)))
    (LET ((HASQUOTE (EQ (CHARCODE \")
                        (CHCON1 STRING)))
          SECONDQUOTE FQBRK SQBRK FBRK SBRK QUOTEDSTRING BRKSTRING)
         (IF (AND HASQUOTE (SETQ SECONDQUOTE (STRPOS "\"" STRING 2))
                  (SETQ FBRK (STRPOS "<" STRING (ADD1 SECONDQUOTE)))
                  (SETQ SBRK (STRPOS ">" STRING (ADD1 FBRK))))
             THEN (SETQ QUOTEDSTRING (CL:STRING-TRIM "  " (SUBSTRING STRING 2 (SUB1 SECONDQUOTE)
                                                                    (CONSTANT (CONCAT)))))
                  (SETQ BRKSTRING (CL:STRING-TRIM "     " (SUBSTRING STRING (ADD1 FBRK)
                                                                    (SUB1 SBRK)
                                                                    (CONSTANT (CONCAT)))))
                  (CL:UNLESS (STRPOS "@" BRKSTRING)
                      (SETQ BRKSTRING (CONCAT BRKSTRING "@parc.xerox.com")))
                  (IF (AND (SETQ FQBRK (STRPOS "<" QUOTEDSTRING))
                           (SETQ SQBRK (STRPOS ">" QUOTEDSTRING (ADD1 FQBRK)))
                           (STRING-EQUAL BRKSTRING (SUBSTRING QUOTEDSTRING (ADD1 FQBRK)
                                                              (SUB1 SQBRK)
                                                              (CONSTANT (CONCAT)))))
                      THEN ;; The quoted string has a bracketed substring that matches the outer string, so that part of the quoted string is
                           ;; redundant.
                           (CL:WHEN (AND (EQ (CHARCODE \')
                                             (CHCON1 QUOTEDSTRING))
```

```
                                        (EQ (CHARCODE \')
                                            (NTHCHARCODE QUOTEDSTRING -1)))
                                    (GNC QUOTEDSTRING)
                                    (GLC QUOTEDSTRING)
                                    (SETQ QUOTEDSTRING (CL:STRING-TRIM "    " QUOTEDSTRING)))
                                ;; Chop bracket stuff out of quoted string
                                (SETQ QUOTEDSTRING (CL:STRING-TRIM "        " (CONCAT (OR (SUBSTRING QUOTEDSTRING 1
                                                                                           (SUB1 FQBRK)
                                                                                           (CONSTANT (CONCAT))
                                                                                           )
                                                                                       "")
                                                                                  (OR (SUBSTRING QUOTEDSTRING
                                                                                                 (ADD1 SQBRK)
                                                                                                 -1
                                                                                                 (CONSTANT (CONCAT)))
                                                                                      ""))))
                                (IF (STRPOS "," QUOTEDSTRING)
                                    THEN (CONCAT "\"" QUOTEDSTRING "\" " (SUBSTRING STRING FBRK SBRK))
                                  ELSE (CONCAT QUOTEDSTRING " " (SUBSTRING STRING FBRK SBRK)))
                            ELSEIF (AND (EQ (CHARCODE \')
                                            (CHCON1 QUOTEDSTRING))
                                        (EQ (CHARCODE \')
                                            (NTHCHARCODE QUOTEDSTRING -1))
                                        (STRING-EQUAL BRKSTRING (SUBSTRING QUOTEDSTRING 2 -2 (CONSTANT (CONCAT)))))
                              THEN ;; The quoted string is 'foo@parc.xerox.com' and the (possibly extended) bracketed string is the same.  We can
                                   ;; just return the original bracketed string
                                   (SUBSTRING STRING (ADD1 FBRK)
                                          (SUB1 SBRK))
                            ELSE STRING)
                      ELSE STRING))))

)

;; This is a stub needed by the TEdit-uuencode strategy; if we ever decide on a reasonable way to do this and make it part of Lafite, this may go away

(MOVD? 'NILL 'UNIX.UUDECODE.IF.NEEDED)

;; Hack to install easy interface for line wrapping.  I should really put line-wrapping into Lafite as a whole.

(CL:WHEN (CAR (NLSETQ (EDITE LAFITESENDINGMENUITEMS '(F \\SENDMSG.CHANGE.MODE))))
    ;; The CONS below insures that Lafite notices you've changed LAFITESENDINGMENUITEMS, so the menu will get updated.
    (SETQ LAFITESENDINGMENUITEMS (EDITE (CONS (CAR LAFITESENDINGMENUITEMS)
                                              (CDR LAFITESENDINGMENUITEMS))
                                        '(CHANGE \\SENDMSG.CHANGE.MODE TO \\UNIXMAIL.CHANGE.MODE))))

(PUTPROPS LAFITE-UNIXMAIL FILETYPE :COMPILE-FILE)
```

FUNCTION INDEX

VARIABLE INDEX

RECORD INDEX

PROPERTY INDEX