Caucus E-mail Interface
Installation and Usage Guide

Last Revised: 16 December 1996

1. INTRODUCTION AND PURPOSE

The Caucus e-mail interface package adds a "listserv" or mailing-list like capability to the existing Caucus conferencing system.

With this optional package, you can extend the use of your Caucus conferencing system to people who have only e-mail access to the Internet.

When this package is installed and enabled, each Caucus conference organizer can specify a list of e-mail addresses that may participate in that conference. New material (items and responses) are automatically sent to those participants, via e-mail, on a regular basis.

Those users may in turn contribute to the conference by simply replying to those messages. The replies are automatically placed in the proper conference and item.

2. INSTALLATION

2.1 E-mail Kit Contents.

The Caucus E-mail link kit is contained in a file called "email.tar" that may be downloaded directly from the Screen Porch web site, at http://screenporch.com.

The email.tar file contains this README file, the 'einstall' installation script, and a compressed kit file, kit.t.Z.

2.2 Create the Caucus Mailer userid ("caumail").

Create a Unix userid that is dedicated to handling the e-mail for this interface. (The 'root' user or system administrator must do this.)
A good name for this userid is "caumail", although any userid will work.
(Do not use the regular "caucus" userid for this account. This must be a separate userid that is only used for this purpose.)

This userid must be able to use the Unix 'crontab' utility.

2.3 Install the software.

Login to the id you created in step 2.2. Do NOT install the software as root! Download or copy the email.tar file to the home directory of that id. Type:

```
tar xvf email.tar
./einstall
```

Follow the instructions that are displayed.

Initially the e-mail updates will be sent out once per day.
This may be changed by examining and modifying the contents of the "crontab" listing for this userid.

2.4 Connect the e-mail link to Caucus.

To finish the installation, you must "connect" the Caucus e-mail link software with your regular Caucus installation.

Login to the "caucus" userid. From this id, run the script called "copysweb" that is located in the Caucus Mailer userid's home directory. (For example, if the Caucus Mailer userid's home directory is /home/caumail, then type "/home/caumail/copysweb".)

Now edit the file CML/SP31/Local/switch.i, and change the definition of the "mail_out" variable to be the Caucus Mailer userid (for example, "caumail".)

3. CONFERENCE ORGANIZER INSTRUCTIONS

To allow e-mail users to participate in a conference, the conference organizer must do two things from the "customize" page:

3.1 Include the Caucus Mailer userid in your conference.

Add the Caucus Mailer userid from step 2.2 to the list of users included in your conference. This only needs to be done once.

3.2 Add individual e-mail users.

For each e-mail user that is participating, add their e-mail address to the "Section IV: E-mail participants" box at the bottom of the customize page.

Note that this must be the address that appears on mail sent FROM the user. Caucus uses the entries in the E-mail participants box for two purposes: to determine who to send mail TO, and to control who mail will be accepted FROM.

This is somewhat subtle point. A person with simple "mail to" address may actually have a longer "from" address. You MUST use the "from" address. (Some people may also have multiple e-mail aliases that all point to their "real" e-mail address. In either case, you must always use the "from" address that appears in their replies.)

To remove an e-mail participant, simply delete their address from the box. (There is no way to "rename" an e-mail participant to another e-mail address.)

4. E-MAIL PARTICIPANT INSTRUCTIONS

When an e-mail participant is added to a conference (in step 3.2), they will receive the entire contents of the conference as e-mail. Each item will appear (with all of its responses) as one message. The subject heading of the message begins with "::Caucus", and then shows the conference name, item number, response numbers, and item title.

Thereafter, as new items and responses are added to the conference, e-mail participants will receive regular updates (typically daily). All new responses to an item will be delivered as one message. Each new item (with its responses so far) will be delivered as one message.

An e-mail participant may add a response by simply replying to the appropriate message. A reply to a particular message will be posted as a response to that item.

E-mail participants may post HTML responses, by making the first word of their response be "<HTML>". (It must be followed by a space or a return.)

E-mail participants may post new items by replying to any message (from the relevant conference), and changing the subject field to remove the item and response numbers. (I.e., the subject field should just contain the "::Caucus" and the conference name.) On most mailers, this can easily be accomplished by simply backspacing over the subject until the conference name is reached.

The first line of the message will be used as the item title. If the first word of the title is "<HTML>", then the entire item text will be treated as HTML.

5. APPEARANCE OF E-MAIL POSTINGS IN A CONFERENCE

Items and responses posted by e-mail participants look just the same as entries made by regular Caucus users.

The only exception is the name of the participant. The name will appear as plain text, typically followed by their e-mail address, shown "blued" as a link. (Since Caucus doesn't know anything else about them, only the e-mail address is active.)

Conference organizers can delete or edit the participants' items or responses in the usual way.

6. POTENTIAL PROBLEMS

There is one known problem, having to do with e-mail replies.

For many mailers, when a user replies to an e-mail message, the content of the original message is made part of the reply, with a "> " before each line (to distinguish it from the reply proper).

The Caucus e-mail package understands this syntax, and strips all such lines from the text before adding it as a response.

However, some mailers use other methods of marking the lines from the "original text". As these methods are identified, those lines should also be stripped out! (Otherwise a potentially exponential growth may apply, as replies to replies to replies etc. get posted in the conference.)

See the section in the file import.cml in the Caucus e-mail package for more information about how to accomplish this stripping.

----------END OF README FILE----------

```
#!/bin/sh
#
#---EINSTALL.  Install Caucus 3.1 e-mail link components.
#
#   Invoked as:
#      einstall
#
#   Parameters:
#
#   Purpose:
#      Install and set up files needed by Caucus 3.1 e-mail link programs.
#
#   How it works:
#
#   Known bugs:
#
#   History of revisions:
#: CR 12/16/96 21:37 First release version.
#------------------------------------------------------------------------

echo
echo "Beginning Caucus 3.1 E-mail link installation.  If you encounter"
echo "an error, or need to interrupt the installation, you may run"
echo "this script as many times as necessary."
echo

#---Do not let user run this as root!
echo hi >.superuser.junk
if ls -l .superuser.junk | grep root >/dev/null ; then
   echo "Do not run this installation script as root or superuser."
   echo "See the file README for instructions."
   echo
   rm -f .superuser.junk
   exit  0
fi
rm -f .superuser.junk

#---Unpack the rest of the files.
zcat kit.t.Z | tar xf -
chmod 644 *.cml *.i

#---Determine where the standard mailer is.
mailer=`which mailx 2>/dev/null | (read a b; echo $a)`
if test "x$mailer" = "x" -o "x$mailer" = "xno"; then
   mailer=`which mail 2>/dev/null | (read a b; echo $a)`
fi

while test "x$mailer" = "x" -o "x$mailer" = "xno"; do
   echo
   echo "Caucus needs to run the 'mail' or 'mailx' (or equivalent)"
```

```
   echo "program in order to import e-mail."
   echo
   echo "What is the full path name of that program? "
   read mailer

   if test "x$mailer" != "x"; then
      if test ! -e $mailer; then
         mailer=
      fi
   fi
done


#---Get the caucus userid and home directory.
while true; do
   echo
   echo -n "What is the name of the Caucus userid? "
   read caucus
   if test "x$caucus" != "x"; then
      break
   fi
done
while true; do
   echo
   echo "What is the full pathname of the home directory of the"
   echo -n "Caucus userid '$caucus' ? "
   read homedir
   if test "x$homedir" != "x"; then
      break
   fi
done

#---Create the sweba.conf file.
chmod 700 mksweba.conf
./mksweba.conf $caucus $homedir >sweba.conf
chmod 755 sweba.conf

#---Determine a good temporary directory.
echo
echo "The Caucus e-mail link occasionally needs some temporary file"
echo "space.  Typically, temporary files are placed in /tmp.  But some"
echo "host run very 'tight' /tmp directories, which can cause problems."
echo "On your host, /tmp has the free space shown:"
echo
df /tmp
echo
echo "If your host has plenty of free /tmp space, answer 'y' to the question"
echo "below.  If there is some doubt, answer 'n'."

while true; do
```

```
  echo
  echo -n "Use /tmp for temporary files? [y/n] "
  read answer
  if test "x$answer" = "xy"; then
    temp=/tmp
    break
  fi
  if test "x$answer" = "xn"; then
    temp=$HOME/TEMP
    if test ! -d $temp; then
      mkdir     $HOME/TEMP
    fi
    chmod 777 $HOME/TEMP
    break
  fi
done


#---Write the mail import script:
echo "$HOME/fixhome"                                          >import
echo "(echo \"p *\";  echo \"d *\";  echo \"q\") | $mailer >mailtext" >>import
echo "chmod 644 mailtext"                                     >>import
echo "$HOME/sweba $HOME/sweba.conf import.cml mailtext $temp $mailer" >>import
chmod 700 import

#---Write 'sendup' script, to export mail.
echo "rm -f $temp/caumail*"                                   >sendup
echo "$HOME/sweba $HOME/sweba.conf sendup.cml $temp $mailer"  >>sendup
chmod 700 sendup

#---Write 'import_export', to do both.
echo "$HOME/import"   >import_export
echo "$HOME/sendup"  >>import_export
chmod 700 import_export

#---Write 'copysweb' script, and prepare to run it.
echo "cp \$HOME/SWEB/sweba  $HOME/sweba"   >copysweb
echo "chmod 4711 $HOME/sweba"             >>copysweb
chmod 755 copysweb
chmod 777 $HOME

#---Write 'fixhome', to fix up $HOME after using copysweb.
echo "a=\`ls -ld $HOME | (read a b; echo \$a)\`"   >fixhome
echo "if test \"x\$a\" = \"xdrwxrwxrwx\"; then"   >>fixhome
echo "   chmod 711 $HOME"                         >>fixhome
echo "fi"                                      >>fixhome
chmod 700 fixhome

#---Write and submit the crontab_file, to automatically run import_export.
if test ! -f crontab_file; then
```

```
   (cat crontab_0;
    echo;
    echo "0 5 * * * $HOME/import_export") >crontab_file
fi

echo
if crontab crontab_file; then
   echo "The Caucus E-mail kit has been successfully unpackaged and"
   echo "the files put in place."
   echo
   echo "To complete the installation, please read perform the"
   echo "instructions in section 2.4 of the README file."
else
   echo "This userid is not authorized to run 'crontab'.  Please"
   echo "contact your system administrator and request crontab"
   echo "authorization, and then run this installation script "
   echo "when that has been done."
fi
echo

#-------------------------END OF README FILE--------------------------
```

Produced In Native Format

No Image Available

```
# Crontab file for Caucus E-mail link id.

# Each line beginning with a "#" is a comment and is ignored.

# Each non-comment line contains a single entry.  Each entry has 6 fields,
# separated from the others by a space.  The first five fields are
# numbers which determine the time or times at which the command at the end
# of the line should be run.  Each field must be a valid number for
# that field or a comma-separated list of numbers.  There must *not* be
# spaces on either side of the comma: the space is a field separator.
# (The last field, which is a Unix command, can have spaces in it.)

# The first field is the minute during the hour that the command at the end
# of the line should be run.  It can be a number from 0-59.
# The second field is the hour of the day; it can be a number from 0-23.
# The third field is the day of the month; it can be a number from 1-31.
# The fourth field is the month of the year; it can be a number from 1-12.
# The fifth field is the day of the week; it can be a number from 0-6.
#  (Sunday=0, Saturday=6)

# Any of the first five fields may have a '*' in them, which means "any
# valid number".
```

```
#
#---IMPORT.CML.    Caucus CML script: import conference updates from e-mail.
#
#   Invoked as:
#      sweba sweba.conf import mailtext temp mailer
#
#   Parameters:
#      sweba        The Caucus stand-alone CML interpreter program
#      sweba.conf   The configuration file for sweba
#      import.cml   This file.  (Must be relative to CML_Path in sweba.conf!)
#      mailtext     (arg 1) full pathname of file containing incoming mail
#      temp         (arg 2) full pathname of temporary file directory
#      mailer       (arg 3) full pathname of e-mail client
#
#   Purpose:
#      Import.cml is a CML script for the Caucus stand-alone CML
#      interpreter, sweba.  It imports e-mail from conference
#      members that is in a known format (specifically, as a reply
#      to conference e-mail generated by sendup.cml), and posts
#      the replies as responses to the appropriate items and
#      conferences.
#
#   How it works:
#      Each conference has an e-mail list that is controlled by the
#      switch "mail_out", and by the Caucus interface file custom.cml.
#
#      Typically import.cml is run on a regular basis, (e.g. once a day)
#      by the 'cron' utility or some other automatic scheme.
#
#      Import scans the incoming e-mail, plucks the conference and
#      item number off of the subject field, strips out any
#      'original' text, and adds what remains to the appropriate
#      item.
#
#   Known bugs:
#      Different mail clients use different ways of indicating
#      'original', or replied-to, text.  As new formats are
#      discovered, import.cml should be extended to skip over them
#      as well.
#
#   History of revisions:
#: CR  8/21/96 17:14 First release version.
#-------------------------------------------------------------------------

set mailer $arg(3)

if $empty ($arg(2))
   "Error: no temporary file directory specified.
   return
end
```

```
set tmp $arg(2)

#---If this id is not registered, "create" it as a person.
if $not ($my_exist())
  set ignore $set_my_name (Caucus Mailer)
end

#---Open input mail text file, and verify it exists.
set in $open ($arg(1) r)
if $equal ($(in) 0)
  "
  "Cannot read mail text file '$arg(1)'.
  "
  return
end

#---Force all added responses to be considered "new" immediately.
set ignore $set_my_text (1)

#---Use a simple state machine to process each line of the input.
#   0 => ignore text
#   1 => processing mail header
#   2 => reading text of message.
set state 0

while $readln ($(in) line)

  #---If line starts with "From", begin processing header (state 1)
  set word1 $word (1 $(line))
  if $and ($not_equal ($charval(0 $(line)) 32) \
          $or ($equal ($(word1) From)  $equal($(word) From:) ))
    if $equal ($(state) 2)
      include import.i
    end
    set state 1
  end

  #---Add text lines to the accumulating response.
  if $equal ($(state) 2)

    #---Here's where the stripping of "original text" of
    #   a replied-to message occurs.  This code handles
    #   the traditional "> " before each line of original
    #   text.
    #
    #   If you add code to this section, please inform
    #   us (at roth@screenporch.com) what code you added
    #   and why, so we may add other stripping techniques
    #   to future versions of this script.
```

```
    set strip 0
    if $and ($equal ($charval(0 $(line)) 62)  \
            $equal ($charval(1 $(line)) 32) )
      set strip 1
    end

    if $not($(strip))
      set text $(text)$(line)$newline()
    end
  end

  #---Parsing message header....
  if $equal ($(state) 1)

    #---Pluck off "From:" field.
    if $equal ($(word1) From:)
      set from $rest (2 $(line))
    end

    #---Pluck off the "subject" field.  If not "::Caucus:", skip rest of msg.
    if $equal ($(word1) Subject:)
      set state 0
      if $equal ($word (3 $(line)) ::Caucus:)
        set subject $rest (4 $(line))
        set state 1
      end
      if $equal ($word (2 $(line)) ::Caucus:)
        set subject $rest (3 $(line))
        set state 1
      end
    end

    #---Empty line indicates start of actual message text.
    if $empty ($(word1))
      set state 2
      set text $newline()
    end
  end

end
set ignore $close($(in))

if $equal ($(state) 2)
  include import.i
end
#-------------------------------------------------------------------------
```

```
#
#---IMPORT.I   Add the current message as a response.
#
#-----------------------------------------------------------------

#---Break up the Subject: line to get the conference name & # and item #.
set subject $replace (: 32  $(subject))
set cname   $lower($word (1 $(subject)))
set inum          $word (2 $(subject))
set cnum    $word (1 $cl_list ($(cname)))

#---Remove <>'s and tabs from address field, it messes up HTML!
set from  $replace (  < 32 $(from))
set from  $replace (  > 32 $(from))
set from  $replace ( 09 32 $(from))
#set from  $word (1 $(from))

#---Determine the desired text type (0=wordwrap, 2=HTML).
set word1 $lower($word (1 $(text)))
set prop  0
if $equal ($(word1) <html>)
  set prop 2
  set text $rest (2 $(text))
end


#---Valid message?  (Conference number found, text was non-empty?)
if $and ($not_equal ($(cnum) 0)  $not_empty($(text)) )

  #---Is the sender's e-mail address in the list of e-mail participants?
  set email  $lower($conf_var ($(cnum) email))
  set ok 0
  for word in $(from)
    if $tablefind ($lower($(word)) $(email))
      set ok 1
      break
    end
  end

  #---If so, add it to the conference.
  if $(ok)
    set ignore $ad_author($(from))

    if $empty($(inum))
      set ignore $ad_item  ($(cnum) $(prop) $(text))
    end
    else
      set ignore $ad_resp  ($(cnum) $(prop) $(inum)_$(text))
    end
  end
```

```
  #---If not, send it back to the author and complain.
  else
    set ignore $output($(tmp)/caucomplain 0644)
    "You are not permitted to participate in the $upper($(cname)) conference
    "from the address:
    "  $(from)
    "
    "If you have any questions, please contact the organizer
    "at: $user_var ($co_org($(cnum)) e-mail)
    "
    "------ Your text was: ----------------------------
    "$(text)
    set ignore $output()

    #---Remove ()'s from e-mail address, and send complaint.
    set from $replace (40 32 $(from))
    set from $replace (41 32 $(from))
    set ignore $shell($(mailer) <$(tmp)/caucomplain -s "No permission" $(from))
  end
end
#-------------------------------------------------------------------
```

```
#!/bin/sh
#
#---MKSWEBA.CONF.   Build the default sweba.conf configuration file.
#
#   Input arguments:
#     $1   Caucus userid
#     $2   Caucus home directory
#
#   Output: to stdout
#
#-----------------------------------------------------------------

echo "#"
echo "#---SWEBA.CONF      Sweb configuration file."
echo "#"
echo "# This is the Caucus Stand-Alone CML interpreter configuration file."
echo "# It contains parameters which must be set for your host and server."
echo "#"
echo ""
echo "#============================================================="
echo "# Caucus_ID is the userid that owns the Caucus data files."
echo "# You MUST set this to the appropriate id for your system."
echo "# (You may use the userid name or number.)"
echo ""
echo "Caucus_ID    $1"
echo ""
echo "#============================================================="
echo "# Caucus_Path is the full pathname of the directory"
echo "# where the Caucus programs and data files are installed."
echo "# You MUST set this to the proper pathname for your system."
echo ""
echo "Caucus_Path  $2"
echo ""
echo "#============================================================="
echo "# CML_Path is the full pathname of the directory which"
echo "# contains the CML scripts.  You MUST change this to the"
echo "# proper path for your system."
echo ""
echo "CML_Path     $HOME"
echo ""
echo "#============================================================="
echo "# Caucus_Lib is the full Unix pathname of the Caucus \"Library\" area."
echo "# Files uploaded by Caucus users are placed in directories"
echo "# under this library."
echo "#"
echo ""
echo "Caucus_Lib   $2/public_html/LIB"
echo ""
echo "#============================================================="
echo "# Disk_Format defines the character set in use for the data in"
```

```
echo "# the Caucus data files.  Choices are: ascii, iso8859, euc, and"
echo "# sjis.  (The last two are the most common japanese language"
echo "# character sets, and are only supported by the Japanese license"
echo "# option.)"
echo ""
echo "Disk_Format  ascii"
echo ""
echo "#================================================================="
echo "# Browser_Format defines which character set (same choices as"
echo "# Disk_Format) the browser is expecting to see.  Sweb automatically"
echo "# handles conversion between different character sets, to the extent"
echo "# possible."
echo ""
echo "Browser_Format ascii"
echo ""
echo "#==================End of sweba.conf============================="

#-----------------------------------------------------------------
```

Caucus E-mail Interface
Installation and Usage Guide

Last Revised: 16 December 1996

1. INTRODUCTION AND PURPOSE

The Caucus e-mail interface package adds a "listserv" or mailing-list like capability to the existing Caucus conferencing system.

With this optional package, you can extend the use of your Caucus conferencing system to people who have only e-mail access to the Internet.

When this package is installed and enabled, each Caucus conference organizer can specify a list of e-mail addresses that may participate in that conference. New material (items and responses) are automatically sent to those participants, via e-mail, on a regular basis.

Those users may in turn contribute to the conference by simply replying to those messages. The replies are automatically placed in the proper conference and item.

2. INSTALLATION

2.1 E-mail Kit Contents.

The Caucus E-mail link kit is contained in a file called "email.tar" that may be downloaded directly from the Screen Porch web site, at http://screenporch.com.

The email.tar file contains this README file, the 'einstall' installation script, and a compressed kit file, kit.t.Z.

2.2 Create the Caucus Mailer userid ("caumail").

Create a Unix userid that is dedicated to handling the e-mail for this interface. (The 'root' user or system administrator must do this.) A good name for this userid is "caumail", although any userid will work. (Do not use the regular "caucus" userid

for this account. This must be a separate userid that is only used for this purpose.)

This userid must be able to use the Unix 'crontab' utility.

2.3 Install the software.

Login to the id you created in step 2.2. Do NOT install the software as root! Download or copy the email.tar file to the home directory of that id. Type:

```
tar xvf email.tar
./einstall
```

Follow the instructions that are displayed.

Initially the e-mail updates will be sent out once per day. This may be changed by examining and modifying the contents of the "crontab" listing for this userid.

2.4 Connect the e-mail link to Caucus.

To finish the installation, you must "connect" the Caucus e-mail link software with your regular Caucus installation.

Login to the "caucus" userid. From this id, run the script called "copysweb" that is located in the Caucus Mailer userid's home directory. (For example, if the Caucus Mailer userid's home directory is /home/caumail, then type "/home/caumail/copysweb".)

Now edit the file CML/SP31/Local/switch.i, and change the definition of the "mail_out" variable to be the Caucus Mailer userid (for example, "caumail").

3. CONFERENCE ORGANIZER INSTRUCTIONS

To allow e-mail users to participate in a conference, the conference organizer must do two things from the "customize" page:

3.1 Include the Caucus Mailer userid in your conference.

Add the Caucus Mailer userid from step 2.2 to the list of users included in your conference. This only needs to be done once.

3.2 Add individual e-mail users.

For each e-mail user that is participating, add their e-mail address to the "Section IV: E-mail participants" box at the bottom of the customize page.

Note that this must be the address that appears on mail sent **from** the user. Caucus uses the entries in the E-mail participants box for two purposes: to determine who to send mail **to**, and to control who mail will be accepted **from**.

This is somewhat subtle point. A person with simple "mail to" address may actually have a longer "from" address. You **must** use the "from" address. (Some people may also have multiple e-mail aliases that all point to their "real" e-mail address. In either case, you must always use the "from" address that appears in their replies.)

To remove an e-mail participant, simply delete their address from the box. (There is no way to "rename" an e-mail participant to another e-mail address.)

4. E-MAIL PARTICIPANT INSTRUCTIONS

When an e-mail participant is added to a conference (in step 3.2), they will receive the entire contents of the conference as e-mail. Each item will appear (with all of its responses) as one message. The subject heading of the message begins with "::Caucus", and then shows the conference name, item number, response numbers, and item title.

Thereafter, as new items and responses are added to the conference, e-mail participants will receive regular updates (typically daily). All new responses to an item will be delivered as one message. Each new item (with its responses so far) will be delivered as one message.

An e-mail participant may add a response by simply replying to the appropriate message. A reply to a particular message will be posted as a response to that item.

E-mail participants may post HTML responses, by making the first word of their response be "<HTML>". (It must be followed by a space or a return.)

E-mail participants may post new items by replying to any message (from the relevant conference), and changing the subject field to remove the item and response numbers. (I.e., the subject field should just contain the "::Caucus" and the conference name.) On most mailers, this can easily be accomplished by simply backspacing over the subject until the conference name is reached.

The first line of the message will be used as the item title. If the first word of the title is "<HTML>", then the entire item text will be treated as HTML.

5. APPEARANCE OF E-MAIL POSTINGS IN A CONFERENCE

Items and responses posted by e-mail participants look just the same as entries made by regular Caucus users.

The only exception is the name of the participant. The name will appear as plain text, typically followed by their e-mail address, shown "blued" as a link. (Since Caucus doesn't know anything else about them, only the e-mail address is active.)

Conference organizers can delete or edit the participants' items or responses in the usual way.

6. POTENTIAL PROBLEMS

There is one known problem, having to do with e-mail replies.

For many mailers, when a user replies to an e-mail message, the content of the original message is made part of the reply, with a "> " before each line (to distinguish it from the reply proper).

The Caucus e-mail package understands this syntax, and strips all such lines from the text before adding it as a response.

However, some mailers use other methods of marking the lines from the "original text". As these methods are identified, those lines should also be stripped out! (Otherwise a potentially exponential growth may apply, as replies to replies to replies etc. get posted in the conference.)

See the section in the file import.cml in the Caucus e-mail package for more information about how to accomplish this stripping.

```
#
#---SENDITEMS.I    E-mail selected items to address list.
#
#   Arguments:
#     inc(1)    Name of variable containing triplet item list of items to send
#     inc(2)    Name of variable containing e-mail address list
#     inc(3)    Mark selected responses as "seen"?
#     cname     Name of conference

#---For each selected item...
for vi_cnum vi_inum vi_rnum in $($inc(1))

   #---Redirect the output for this particular item to a temp file.
   set ignore $output ($(tmp)/caumail.$(vi_cnum).$(vi_inum) 0744 )

   #---Actually display the selected item/response text.
   set rlast  $it_resps($(vi_cnum) $(vi_inum))
   include writeitem.i  $(vi_rnum) $(rlast)

   #---Close the temp file.
   set ignore $output()

   #---Prepare the e-mail subject header (with no "s, please!)
   set range  $(vi_rnum)-$(rlast)
   if $equal ($(vi_rnum) $(rlast))
     set range $(vi_rnum)
   end
   set subject ::Caucus: $(cname): $(vi_inum):$(range) \
                       $re_title($(vi_cnum) $(vi_inum) 0)
   set subject $replace (" ' $(subject))

   #---Send the mail to each person in the list.
   for address in $($inc(2))
     set ignore $silent ($(mailer) \
                         <$(tmp)/caumail.$(vi_cnum).$(vi_inum) \
                         -s "$(subject)" $(address); sleep 2)
   end

   #---Finally, mark this item as completely 'seen'.
   if $inc(3)
     set ignore $set_it_seen( $(vi_cnum) $(vi_inum) $(rlast))
   end
end

#---Remove temp files.
 set ignore $shell (rm -f $(tmp)/caumail.$(vi_cnum).*)

#-------------------------------------------------------------
```

```
#
#---SENDUP.CML.   Caucus CML file: send conference updates by e-mail.
#
#   Invoked as:
#      sweba sweba.conf sendup.cml temp mailer
#
#   Parameters:
#      sweba        The Caucus stand-alone CML interpreter program
#      sweba.conf   The configuration file for sweba
#      sendup.cml   This file.  (Must be relative to CML_Path in sweba.conf!)
#      temp         (arg 1) Full pathname of directory for temporary files
#      mailer       (arg 2) Full pathname of e-mail client
#
#
#   Purpose:
#      Sendup.cml is a CML script for the Caucus stand-alone CML
#      interpreter, sweba.  It sends the text of any new material in
#      selected conferences to the users in the subscription
#      list, via e-mail.  (It also sends *all* the text of a
#      conference to new e-mail subscribers to an on-going conference.)
#
#   How it works:
#      Each conference has an e-mail list that is controlled by the
#      switch "mail_out", and by the Caucus interface file custom.cml.
#
#      Typically sendup.cml is run on a regular basis, (e.g. once a day)
#      by the 'cron' utility or some other automatic scheme.
#
#      For each person that has been added to the e-mail list (since
#      the last time sendup.cml was run), ALL of the items in that
#      conference are sent to the person, one item per message.
#
#      For everyone else on the e-mail list, only new material is
#      sent.  (All new responses to an item are sent as one message.)
#
#   Known bugs:
#
#   History of revisions:
#: CR  8/20/96 17:14 First version.
#: CR  9/03/96 14:00 Beta version.
#: CR 12/16/96 23:12 First release version.
#-------------------------------------------------------------------------

set mailer $arg(2)

if $empty ($arg(1))
   "Error: no temporary file directory specified.
   return
end
set tmp $arg(1)
```

```
#---If this id is not registered, "create" it as a person.
if $not ($my_exist())
  set ignore $set_my_name (Caucus Mailer)
end

#---For each conference that has an e-mail list...
for cnum in $cl_list()

  #---Make sure that the e-mail master id is a member (if possible).
  if $not ($it_member($(cnum)))
    set ignore $it_join($(cnum))
  end

  set email $conf_var($(cnum) email)
  if $and ($it_member($(cnum))  $not_empty ($(email)) )

    #---People who were just added to the e-mail list get ALL
    #   material; everyone else just gets NEW material.
    set cname      $upper1($cl_name($(cnum)))
    set old_email  $conf_var ($(cnum) old_email)
    set ignore $set_conf_var ($(cnum) old_email $(email))
    set get_new
    set get_all
    for address in $(email)
      if $tablefind ($(address) $(old_email))
        set get_new $(get_new) $(address)
      end
      else
        set get_all $(get_all) $(address)
      end
    end
    set markseen 1


    #---Handle those that get (just) the new material.
    if $not_empty ($(get_new))
      set ilist $it_listrnew($(cnum)) $it_listiunseen($(cnum)) \
                                $it_listinew($(cnum))
      include senditems.i ilist get_new 1
      set markseen 0
    end


    #---Handle those that get all material.
    if $not_empty ($(get_all))

      #---Prepare a list of all visible items.
      set ilist
      count inum 1 $it_last($(cnum))
```

```
            if $it_visib ($(cnum) $(inum))
               set ilist $(ilist) $(cnum) $(inum) 0
            end
         end

         include senditems.i ilist get_all $(markseen)
      end

   end
end

#-------------------------------------------------------------------------
```

```
#
#---WRITEITEM.I      Write item text to output
#
#   See senditems.i, which includes this file.
#
#-------------------------------------------------------------------------

"Caucus: $(cname) Item $(vi_inum):
"$re_title($(vi_cnum)  $(vi_inum) 0)
"

set between_responses - - - - - - - - - - -
set show_id 1
set rstart  $inc(1)
set rstop   $inc(2)

count response $(vi_rnum) $(rstop)

   if $re_exists($(vi_cnum) $(vi_inum) $(response))

      #---Response header
      if $(show_id)
         set uid ($re_owner())
      end
      if $equal ($(response) 0)
         "$re_time() $re_author() $(uid)
      end
      else
         "$(vi_inum):$(response)) $re_time%16() $re_author() $(uid)
      end

      #---Response text
      if $equal ($re_prop() 2)
         "$reval($protect($re_text()))
      end
      else
         if $equal ($re_prop() 1)
            "$re_text()
         end
         else
            "$re_text()
         end
      end

      if $not_equal ($(response) $(rstop))
         "$(between_responses)
      end
   end
end
```

#--