MULTICS TECHNICAL BULLETIN MTB-597

To: MTB Distribution

From: Gary M. Palter

Date: 3 September 1982

Subject: MR10.1 Multics Mail System Extensions

This MTB presents the planned extensions to the Multics mail system to be included in the MR10.1 release.

Please direct any comments on this proposal to the author

by electronic mail to:

Palter.Multics on either MIT or System-M

by the forum teleconferencing system to the meeting:

`>udd>Multics>Palter>forums>Mail_System` on System-M

or by the U.S. Postal service to:

Honeywell Information Systems, CISL
575 Technology Square
Cambridge, Mass. 02139

Inter-System Mail

As part of the ARPA network TCP/IP conversion effort, the present mail system network support is generalized to allow communication with users on arbitrary networks. This support will not be documented in MR10.1, however, as several areas such as real network host table management tools and a standard inter-Multics mailer will not be available until MR10.2.

In order to provide this support, separate changes will have to be made to to mail_system_ (for Executive Mail), the Extended Mail Facility, and Emacs RMAIL. The mlsys_mailer_ interface developed over the last six months by Charlie Hornig will be merged into mail_system_; the Extended Mail Facility and Emacs RMAIL will, however, continue to call mlsys_mailer_ directly. In MR10.2, the Extended Mail Facility and Emacs RMAIL will be converted to use mail_system_ directly and the mlsys_mailer_ interface will be deleted.

Conversion of the Extended Mail Facility to the New Subsystem Utilities

The read_mail and send_mail subsystems are upgraded to use the new subsystem utilities installed in MR10. Use of these new utilities provide numerous enhancements to both subsystems including:

ABBREVIATION PROCESSING

The user can request that abbreviations be expanded on read_mail and send_mail request lines. Through the use of new command line control arguments — *-abbrev (-ab)*, *-no_abbrev (-nab)*, and *-profile path* -- the user can specify whether abbreviation processing is enabled or disabled when entering the subsystem and can also specify the profile used to look up abbreviation definitions. If expansion is enabled and a profile is not specified, the same profile used at Multics command level will be used within the subsystem.

A new request, abbrev (ab), is also provided which allows the user to enable or disable abbreviation processing and to change profiles once within the subsystem.

For example, the user may define an *rdm* abbreviation to enter read_mail with expansion enabled using the profile *mail_system.profile* in the home directory as follows:

```
.ab rdm do "read_mail -abbrev -profile [hd]>mail_system
     &rf1"
```

ADDITIONAL STANDARD REQUESTS

The following new requests, supplied as part of the subsystem utilities, are available in read_mail and send_mail:

exec_com (ec):
This request executes a segment containing subsystem requests. The full capabilities of the Multics exec_com processors (versions one and two) are available. The read_mail subsystem uses the suffix *rdmec* and the send_mail subsystem uses the suffix *sdmec* for their exec_com segments to avoid

confusion with exec_com's that are intended for execution at Multics command level. Additionally, the subsystems will search for the exec_com using the *mail_system* search list. The default content of this search list is:

```
-working_dir
>udd>[user project]>[user name]>[user name].mlsys
```

`do, if, answer:`

These requests are identical to the do, if, and answer commands available at Multics command level except that they execute request lines rather than Multics command lines. These requests are invaluable in the creation of abbreviations within the subsystem.

`ready (rdy):`

This request prints a Multics ready message.

`ready_on (rdn), ready_off (rdf):`

These requests control whether a ready message will be printed after the execution of each request line. By default, neither read_mail nor send_mail print ready messages.

`subsystem_name, subsystem_version:`

These requests return the name and version of the current subsystem, respectively. These requests are invaluable in abbreviations which are shared by multiple subsystems or which must know whether certain features of a subsystem are present. For example, a *quit* abbreviation may be defined as follows which supplies *-force* only when in the read_mail subsystem:

```
.ab quit do """quit"" [if [e equal
        [subsystem_name] read_mail] -then -force]
        &rf1"
```

IMPROVED SELF-DOCUMENTATION FACILITIES

The "?" request in read_mail and send_mail is changed to print a multi-columnar list of the requests available in the subsystem.

The list_requests (lr) request is added to both subsystems to provide the functionality of the old "?" request. It produces a list of valid requests with a brief summary of each request. Additionally, the list_requests request accepts request name topics and lists those requests which match those topics. For example, the request line:

```
list_requests list
```

in send_mail will print the brief summary of the list_help, list_original, and list_requests requests.

The help request is extended to accept most control arguments accepted by the Multics help command. In particular, the *-brief* control argument can be used to produce a summary of any read_mail or send_mail request which includes the request's syntax line, arguments, and control arguments. In addition, the help request is changed to explain how to obtain additional online information when it is invoked with no arguments.

Enhancements to read_mail

The read_mail subsystem is upgraded by converting it to use the new subsystem utilities and by correcting many outstanding problems.(1) The following enhancements are also included in MR10.1:

* Mailbox specification heuristic:

  The definition of a non-control argument on the read_mail command line is changed to the following:

  STR

  is first interpreted as *-mailbox STR;* if no mailbox is found, this specification is then interpreted as *-save STR;* if no savebox is found, this specification is then interpreted as *-user STR.*

  This definition is chosen rather than the one used for an address(2) to make it simpler for a user to reference mailboxes and saveboxes in the working directory. This definition has been employed in the EXL version of read_mail for several years and has proved quite useful.

* print_header request:

  The print_header request prints the header of the selected messages. It is intended as a replacement for the *-header_only* control argument available in the print request. The *-header_only* control argument is now undocumented but will be retained for at least one release to allow users to convert their exec_coms (if any). The print_header request accepts the *-long (-lg)* and *-brief (-bf)* control arguments to control the amount of information displayed from the header; the default is *-long.* print_header also accepts all of the message selection and disposition control arguments accepted by the print request (see below).

* apply request:

  The apply request executes an arbitrary command line on a segment in the process directory containing the header and text of a message. For example, the following request line will issue an output request for the current message:

  ```
  apply "do """copy &1 ===; eor &1 -dl"""""
  ```

  Due to the lack of appropriate mail system primitives, the apply request can not be used to actually modify the actual message in the mailbox.

---

(1) The complete list of problems corrected by and suggestions implemented in this release will be included as part of the MCR to install the MR10.1 mail system. However, it is interesting to note that over one hundred trouble reports will be closed by this installation.

(2) The definition of a non-control argument used when parsing addresses in read_mail and send_mail is:

STR

if STR contains either "<" or ">", it is interpreted as *-mailbox STR;* otherwise, it is interpreted as *-user STR*

- New request name abbreviations:
  The *first* request now has the short name *f*, the *last* request now has the short name *l*, the *current* request now has the short name *c*, the *forward* request now has the short name *for* in addition to *fwd*, the *print* request now has the short name *p* in addition to *pr*, and the *delete* request now has the short name *d* in addition to *dl*.

- New command line control arguments:
  The following control arguments are now recognized on the read_mail command line:

  -count, -ct
  prints the number of messages read from the mailbox before entering the request loop. This is the default.

  -no_count, -nct
  does not print the message count.

  -acknowledge, -ack
  acknowledges messages which request acknowledgement. This is the default.

  -no_acknowledge, -nack
  does not acknowledge any messages.

- Elimination of the *-all* control argument:
  Three new control arguments -- *-include_deleted (-idl)*, *-only_deleted (-odl)*, and *-only_non_deleted (-ondl)* -- are added to all requests to replace the *-all* control argument. The now obsolete *-all* control argument caused a request to operate on deleted messages in addition to non-deleted messages. However, the choice of *-all* for this control argument caused considerable confusion as it is too similar to the *all* message specifier which selects all non-deleted messages in the mailbox. The *-all* control argument is undocumented but will be retained for one release to allow users to convert to the new control arguments. The descriptions of the new control arguments follow:

  -include_deleted, -idl
  includes all messages in the mailbox whether or not they have been deleted when processing any message_specifiers to determine which messages to process.

  -only_deleted, -odl
  includes only those messages which have been deleted. This is the default for the retrieve request.

  -only_non_deleted, -ondl
  include only those messages which have not been deleted. This is the default for all requests other than retrieve.

- Extended message selection:
  New control arguments are added to the list, print, print_header, delete, and retrieve requests to allow selection of messages by date, author, recipient,

and/or subject. These selection facilities can be used with other requests by use of the list active request. For example, the request line:

```
log [list -before 10/1/81 -from Palter.Multics
     -subject /mail/]
```

logs all messages sent by the user Palter.Multics before October 1981 whose subject contains the string *mail*.

* Enhancements to the reply request:

  Several improvements have been made to the reply request including:

  * Improved *Replying to* prompt:

    The message printed by reply now lists as many of the recipients as will fit on a single line rather than simply stating how many recipients would receive the reply. For example:

    ```
    Replying to Palter.Multics,
         Sibert.PDO, and 3 others.
    ```

  * Improved *-include_original* action:

    The Date, From, and Subject fields of the original message are now included in the reply along with the original text when the *-include_original* control argument is used.

  * Improved interaction with send_mail:

    The send_mail created to compose the reply message is created with the same state of abbreviation processing and the same profile as the read_mail invocation in which the reply request is given. In addition, if send_mail is exited without sending the reply, the reply request will refuse to honor the *-delete* control argument.

  * New reply control arguments:

    The following control arguments are added to the reply request:

    -prompt STR

    specifies the prompt to be used by the send_mail created to compose the reply.

    -no_prompt

    specifies that the send_mail created to compose the reply will not prompt for request lines if it enters the request loop.

    -include_self, -is

    allows a copy of the reply to be sent to the person composing the reply if the request determines that such a copy should be sent from use of the *-include_authors* or *-include_recipients* control arguments.

    -no_include_self, -nis

    specifies that a copy of the reply only be sent to the person composing the reply if explicitly requested by the use of the *-to* or *-cc* control arguments. This is the default. This default allows the user to create a reply abbreviation which automatically logs the reply without receiving an extra copy whenever *-include_recipients* is specified.

- New read_mail control arguments:
  The *-line_length N (-ll N)*, *-indent N (-ind N)*, *-include_self (-is)*, and *-no_include_self (-nis)* control arguments are added to the read_mail command to set default values for any use of the reply request within that invocation of read_mail.

* Protection from accidental deletion of messages:
  The user is now queried if he attempts to delete a message which hasn't been listed, printed, saved, or written. This change protects the user from accidently deleting newly arrived messages without having first examined them. The new *-force (-fc)* control argument to the delete request may be used to suppress this query.

* Enhancement to message acknowledgement:
  The message sent when acknowledging a message now includes the subject of the original message if present.

* Minimal video system support:
  The print request now issues the *reset_more* control order after printing each message. This change allows users of the video system to easily abort the printing of a single message when printing several messages.

Enhancements to send_mail

The send_mail subsystem is upgraded by converting it to use the new subsystem utilities and by correcting many outstanding problems. The following enhancements are also included in MR10.1:

* Changes to message filling:
  Several changes are made to send_mail's filling of the message. These changes provide compatibility with filling in forum and are believed to present a more user-friendly interface.

  * Change in default state of filling:
    The default for terminal input is changed to *-fill;* the default for file input is left unchanged as *-no_fill.* The majority of messages typed by a user on the terminal are simple text. Such messages should be filled automatically for the user so that he does not need to worry about entering overlength lines. When inputting the message from a file, however, the user has probably already preformatted the message and would be upset if it were automatically filled.

- New times for filling:
  If enabled, filling takes place after exiting qedx during initial input rather than before entering qedx. The prior behavior often made qedx requests fail as the message in the editor's buffer was formatted differently from what was on the user's screen. In addition, filling, if enabled, now occurs automatically after execution of any qedx or apply request. Two new control arguments -- *-fill (-fi)* and *-no_fill (-nfi)* -- are added to the qedx and apply requests to allow the user to control the automatic filling. Of course, filling, if enabled, still occurs after the user types "." to terminate initial input of the message without entering qedx.

* Interaction of *-input_file* and *-request_loop*:
  Use of the *-input_file* control argument now implies the *-request_loop* control argument. In this way, the user is given the opportunity to fill or otherwise edit the message before sending it. This change again makes send_mail compatible with forum.

* Interaction with read_mail reply:
  Six new requests are added to send_mail which are only available within a send_mail that was created by the read_mail reply request. These new requests, listed below, allow the user to examine or manipulate the original message(s) which he is answering. Additionally, these requests accept read_mail message specifiers to allow the user to possibly examine other messages which might be relevant to the reply he is composing. The new requests are:

  print_original (pro):
  prints the message(s) being answered.

  print_original_header (prohe):
  prints the header of the message(s) being answered.

  list_original (lso):
  summarizes the message(s) being answered.

  log_original (logo):
  saves the message(s) being answered into the user's logbox.

  save_original (svo):
  saves the message(s) being answered into an arbitrary savebox.

  write_original (wo):
  writes the message(s) being answered into an ASCII segment.

* New request name abbreviations:
  The *print* request now has the short name *p* in addition to *pr* and the *apply* request now has the short name *app*.

Enhancements to print_mail

Many outstanding problems in the print_mail command are corrected in MR10.1. The following enhancements are also included in MR10.1:

- Mailbox specification heuristic:
  The heuristic applied to a non-control argument on the print_mail command line is changed to be the same as the one described above in the section on enhancements to read_mail.

- program_interrupt handler:
  The print_mail command now has a program_interrupt handler which repeats the *Delete this message?* query. This handler permits a user to interrupt the printing of a long message and go directly to the query for the message as soon as he decides to delete the message.

- ? response:
  The user may now answer "?" to the *Delete this message?* query to obtain a list of valid responses to the query.

- Improved acknowledgements:
  The message sent when acknowledging a message now includes the subject of the original message if present.

- New control arguments:
  The following control arguments are now accepted by print_mail. The *-long* and *-no_list* control arguments are the inverse of the already implemented *-brief* and *-list* control arguments and are added to allow overriding non-standard defaults set by abbreviations.

  `-long, -lg`
  prints the long form of the greeting message. This is the default.

  `-no_list, -nls`
  does not print a summary of the messages before printing the first message. This is the default.

  `-acknowledge, -ack`
  acknowledges messages which request acknowledgement. This is the default.

  `-no_acknowledge, -nack`
  does not acknowledge any messages.

  `-header, -he`
  prints the entire header associated with each message. This is the default.

  `-brief_header, -bfhe`
  prints an abbreviated form of the header associated with each message.

-own

causes print_mail to only print those messages in the mailbox which were sent by the user of print_mail.

Enhancements to have_mail

The have_mail active function is upgraded to accept the same mailbox specifications as the read_mail and print_mail commands. These specifications include the mailbox specification heuristic described above in the section on enhancements to read_mail and the *-mailbox, -user, -save,* and *-log* control arguments. This change insures that have_mail will not check a different mailbox than the one read_mail and print_mail would read if used in exec_com statements such as:

```
&if [have_mail &rl] &then print_mail &rl
```

New control arguments -- *-interactive_messages (-im), -no_interactive_messages (-nim), -mail (-ml),* and *-no_mail (-nml)* -- are provided to specify which types of messages in the mailbox should be counted when determining if there is something in the mailbox. The default is to check for both normal mail and interactive messages. This default causes have_mail to return *true* whenever print_mail would find messages in the mailbox; by using the *-nim* control argument, the user can cause have_mail to return *true* only when read_mail would decide that the mailbox is not empty.

A new command, have_messages, is added which is identical to have_mail except that, by default, it only checks for the presence of interactive messages in the mailbox.

Documentation

Extensive revisions will have to be made to the *Level 68 Mail System User's Guide,* order number CH23-00, to reflect the changes to print_mail, read_mail, and send_mail described above. All of the input needed to make these changes and to revise the write-ups in *MPM Commands* are contained in the info segments available in the three directories

```
>exl>mail_system_dir>info
>exl>mail_system_dir>info>read_mail, and
>exl>mail_system_dir>info>send_mail
```

on both MIT and System M.