To: Distribution

From: Steve Herbst

Subject: New message facility

Date: August 1, 1975

The commands and subroutines documented in this MTB are part of a facility for sending and receiving mail and interprocess messages stored in ring 1 mailboxes. The mail command and net mail facility have already been modified to use ring 1 mailboxes.

The extensions documented here can be divided into three categories:

1. Commands to handle secure interprocess messages (one-line messages with wakeups). These commands were proposed in MTB-085, "New Message Commands". Extended access to send a wakeup to the owner of a mailbox is separate from extended access to add a message. Two types of extended access control two classes of wakeup.

2. A subroutine interface to send all types of messages including mail. Switches instruct the subroutine send_mail_ whether to send a wakeup with a message and which class of wakeup to send. The recipient determines the class of a wakeup by looking at the accompanying message.

3. A system mail table containing, among other things, the pathname of a default mailbox for each user. The entry point mail_table_$lookup obtains this pathname given a user's person id or alias. Using this subroutine, the send command can send given a single name as a destination. A set of subroutines and commands enable system administrators to modify the table. A user can request a particular default mailbox or he can ask not to be listed at all.

Other new features mentioned in the documentation are the acknowledgement of sent mail and messages (send command) and the deferring of wakeup messages until ready time (accept and defer commands).

The documented programs are all to be installed at once. Users will be encouraged to delete their Person.con_msgs segments. To ease the transition to new mailboxes, two of the old message commands will be modified for compatibility when the new ones are installed. The print_messages command will print new-style messages as well as old, and the send_message command will send the new way.

The mail command and have_mail active function will read interprocess messages as well as mail.

Possible future extensions to the new mail/message facility are:

1. A better command interface for sending mail.

2. Wakeups with mail. A mail wakeup is a special kind that just causes the notice "MAIL" to be printed on the recipient's terminal.

3. Logging of correspondence. Messages in a mailbox are already in chronological order. Logging requires that received messages are not deleted and that sent messages are added to the sender's mailbox.

Documentation comprises the remainder of this MTB. Contact Steve Herbst, Multics project with questions and comments.

Name: send

The send command sends one-line messages to a specified user on a specified project. Messages are placed in the mailbox:

>udd>Project_id>Person_id>Person_id.mbx

There are two classes of messages, normal and urgent. If the recipient is accepting the class of message that is being sent and the sender has appropriate extended access to the mailbox (w for normal, u for urgent), each line that is sent is printed immediately on the recipient's terminal preceded by the sender's name and project.

Usage:

send destination -control_args- -message-

where:

1. destination is either of the form Person_id.Project_id or of the form Name where Name is to be looked up in the system data base mail_table.

2. control_args are among the following:

-acknowledge, -ack	Have the recipient return a standard acknowledgement when he reads the message. The acknowledgement states the recipient, time sent and time read of the original message.
-brief, -bf	Do not print an error message if insufficient access to print immediately or if the recipient is not accepting messages.
-urgent, -ur	Send an urgent message. Default is to send a normal message.

3. message is an optional message comprising the remainder of the command line. Words in the message that begin with - are assumed to be control_args. If there is no message on the command line, send enters input mode and accepts lines from the terminal. Each line is sent after it is typed. Input mode is terminated by a line consisting solely of a period.

Notes:

Parentheses, quotes, brackets, and semicolons in the send command line have their usual command language interpretation. For example, the command line:

```
send Fred.Boss Testing complete; installation soon.
```

sends:

```
Testing complete
```

and the command processor prints: "Segment installation not found." The command line

```
send Herman So long (for now anyway)
```

sends three lines:

```
So long for
So long now
So long anyway
```

In each of these examples, the sender could have enclosed the intended message in quotes.

The user can receive messages while he is in input mode. He can therefore carry on a conversation with a single invocation of the command.

Name: accept

The accept command initializes or reinitializes a user's mailbox for accepting messages sent by the send command. If the mailbox:

>udd>Project_id>Person_id>Person_id.mbx

does not exist, it is created. When normal or urgent messages are accepted, these messages are printed on the user's terminal as soon as they are sent. Messages can be printed only at ready time if desired. See also the defer, send and messages commands.

Usage:

accept -control_args-

where control_args are among the following:

-ready, -rdy Print messages only at ready time.

-urgent, -ur Accept only urgent messages. Default is to accept both normal and urgent.

Note:

In order to send a message that prints immediately on the user's terminal, a sender must have appropriate extended access to his mailbox. This is:

w - normal messages
u - urgent messages

See the mbx_set_acl and related commands in this document.

Name: defer

The defer command prevents messages sent by the send command from printing on the user's terminal. Instead, these messages are saved in the user's mailbox and can be printed using the messages command. See also the send and accept commands.

Usage:

```
defer -control_args-
```

where control_args are among the following:

-ready, -rdy Defer messages only until each ready time.

-urgent, -ur Defer urgent as well as normal messages. Default is to defer only normal messages.

Name: messages

The messages command prints any messages sent by the send command that were received while the user was not accepting messages.

This command also deletes any messages that were printed before, either by the messages command or when they were sent.

Usage:

```
messages -path- -control_args-
```

where:

1. path is the optional pathname of a mailbox.

2. control_args are among the following:

-all, -a	Print all messages, including those that were printed before, and do not delete any messages.
-last, -lt	Print only the latest message received.
-urgent, -ur	Print only urgent messages.

Entry: send_mail_

This subroutine sends one message, optionally accompanied by a wakeup, to a specified user. Two classes of wakeup, normal and urgent, can accompany a message.

Usage:

```
dcl send_mail_ entry(char(*),char(*),ptr,fixed bin(35));

call send_mail_ (destination, message, info_ptr, code);
```

where:

1. destination is a destination of the form Person.Project, or of the form Name where Name is to be looked up in the system's mail table. (Input)

2. message is the text of the message to be sent. (Input)

3. info_ptr is a pointer to the following structure:

```
dcl 1 send_mail_info aligned,
      2 version fixed bin(17),
      2 sent_from char(32) aligned,
      2 switches,
        3 normal_wakeup bit(1) unal,
        3 urgent_wakeup bit(1) unal,
        3 always_add bit(1) unal,
        3 query bit(1) unal,
        3 pad bit(32) unal init("0"b);
```

sent_from — is additional information about the sender, for example, a terminal ID or network host.

normal_wakeup — if ON, means a normal wakeup message. If this bit is on, urgent_wakeup must be off.

urgent_wakeup — if ON, means an urgent wakeup message. If this bit is on, normal_wakeup must be off.

always_add — if ON, says if no wakeup could be sent and no query was asked, add the message anyway.

query — if ON, says if a normal wakeup cannot be sent but the recipient is accepting urgent wakeups, issue the query "Is it urgent?" If the sender answers yes to this query, an urgent wakeup is sent. Otherwise, the message is not sent.

4. code is a standard system status code, for example:

error_table_$name_not_found
if destination had to be looked up in mail_table and was not found there.
error_table_$noentry
if the recipient's mailbox does not exist.
error_table_$moderr
if the caller does not have sufficient extended access to add a message.
error_table_$wakeup_denied
if no wakeup could be sent, either because of insufficient access or because the recipient is deferring all wakeup messages.
error_table_$invalid_channel
if the recipient is not logged in or has not initialized for accepting messages.
error_table_$action_not_performed
if the caller answered no to the query "Is it urgent?"

Entry: mailbox_$accept_wakeups_index

This entry point manipulates information in the mailbox header to allow and defer normal and urgent message wakeups.

Usage:

```
dcl mailbox_$accept_wakeups_index entry(fixed bin,
                fixed bin(71),bit(36),fixed bin(35));

call mailbox_$accept_wakeups_index (mbx_index,
                        channel_id, switches, code);
```

where:

1. mbx_index is the index of a mailbox. (Input)

2. channel_id is the id of an event-call channel created by the caller. (Input)

3. switches are: (Input)

   normal_wakeup is ON if normal wakeups are to be allowed.

   urgent_wakeup is ON if urgent wakeups are to be allowed. If normal_wakeup is on, urgent_wakeup must be on.

4. code is a standard status code, probably zero.

Entry: mailbox_$wakeup_add_index

This entry points adds a message to a specified mailbox and sends a wakeup to the owner of the mailbox.

Usage:

```
dcl mailbox_$wakeup_add_index entry(fixed bin,ptr,
            fixed bin,bit(36),fixed bin(71),fixed bin(35));

call mailbox_$wakeup_add_index (mbx_index, msg_ptr,
                        msg_bitcnt, switches, id, code);
```

where:

1. mbx_index is the index of a mailbox. (Input)

2. msg_ptr is a pointer to the message to be added. (Input)

3. msg_bitcnt is the length of the message in bits. (Input)

4. switches are: (Input)

   normal_wakeup is ON if a normal wakeup is to be sent.

   urgent_wakeup is ON if an urgent wakeup is to be sent.

   always_add is ON if the message is to be added to the mailbox regardless of whether a wakeup could be sent.

6. code is a standard status code, for example:

   error_table_$action_not_performed
   when sending a normal wakeup if the recipient is accepting urgent but not normal wakeups. This error code effectively means, "Try urgent."
   error_table_$moderr
   if insufficient access to add a message.
   error_table_$wakeup_denied
   if unable to send an urgent wakeup.
   error_table_$invalid_channel
   if the recipient is not logged in or has not initialized for accepting message wakeups.
   error_table_$no_info
   if the sender of a wakeup is not allowed to know what has taken place because the recipient has higher AIM authorization than him.

Name: mail_table_

This module manages the data base mail_table, which contains information useful for sending mail and dprinting listings to users.

Entry: lookup

This entry point returns information from mail_table given a user's registered person id or alias as a lookup name.

Usage:

```
dcl mail_table_$lookup entry
                        (char(*), ptr, fixed bin(35));

call mail_table_$lookup (name, infop, code);
```

1. name is a registered person id or a registered alias. (Input)

2. infop is a pointer to the following structure in which information is returned:

```
dcl 1 mail_table_info aligned,
      2 version fixed bin(17),
      2 person char (22),
      2 alias char (8),
      2 mbx char (168),
      2 ds char (12),
      2 he char (64);
```

3. code is a standard status code, usually either zero or error_table_$name_not_found. (Output)

Entry: add

This entry point adds a user to mail_table.

Usage:

```
dcl mail_table_$add entry (ptr, fixed bin(35));

call mail_table_$add (infop, code);
```

1. infop is a pointer to the above mail_table_info structure. (Input)

2. code is a standard status code. error_table_$namedup indicates that an entry for person already exists and the caller must use the delete or update entry point to modify it. (Output)

Entry: delete

This entry point deletes a user from mail_table.

Usage:

```
dcl mail_table_$delete entry point (char(*), code);

call mail_table_$delete (name, code);
```

1. name is a registered person id or alias. (Input)

2. code is a standard status code. (Output)

Entry: update

This entry updates selective information in mail_table for a particular user.

Usage:

```
dcl mail_table_$update entry
                        (char(*), ptr, fixed bin(35));

call mail_table_$update (name, infop, code);
```

1. name is a registered person id or alias. (Input)

2. infop is a pointer to the above mail_table_info structure. (Input)

3. code is a standard status code. (Output)

Notes

If a field in the mail_table_info structure is null, that field is not updated in mail_table.

Entry: mail_table_exists_

The system data base mail_table contains information useful for sending mail and messages and dprinting listings to users. This entry point, intended to be called by the answering service, makes sure that both mail_table and its associated hash table mail_table.ht exist and if not, creates them from the PNT.

Usage:

```
dcl mail_table_exists_ entry
                       (ptr, bit(*), fixed bin(35));

call  mail_table_exists_ (ansp, switches, segname, code);
```

1. ansp is a pointer to the answer table. (Input)

2. switches (Output):

made_one	mail_table and mail_table.ht were created.
old_one	there was an old mail_table. This switch is on only when made_one is on.
new_one	a copy of mail_table exists because there is no access to write the original.
fatal_error	a fatal error is reflected in code, for example when the PNT is the wrong version.

3. segname is the name of the segment referred to by code if code is non-zero and new_one is off. If new_one is on, segname is the name of the copy. (Output)

4. code is a standard status code.

Name: mail_table_lookup, mtl

The mail_table_lookup command returns information from the system data base mail_table given a user's registered person id or alias.

Usage:

```
mail_table_lookup name -control_arg-
```

1. name is a registered person id or alias.

2. control_arg can be -all or -a to print the following information:

Person	registered person id
Alias	registered alias
Mailbox	pathname of default mailbox
Destination	default dprint destination
Header	default dprint header

The default is to print only Mailbox.

Name: mail_table_add, mta

The administrative command mail_table_add adds a user to the system data base mail_table.

Usage:

mail_table_add person alias -control args-

1. person is a registered person id, maximum of 22 characters.

2. alias is the registered alias, maximum of 8 characters.

3. control_args are among the following:

-mailbox path -mbx path	path is the pathname of a default mailbox. If the suffix mbx is not present, it is assumed.
-destination string -ds string	string is a default destination for the dprint command, no longer than 16 characters.
-header string -he string	string is a default header for the dprint command, no longer than 64 characters.

Note:

Any fields not specified by control arguments are set to null.

Name: mail_table_delete, mtd

The administrative command mail_table_delete removes a user from the system data base mail_table.

Usage:

```
mail_table_delete name
```

1. name is a registered person id or alias.

Note:

An error message is printed if there is no entry for name.

Name: mail_table_update, mtu

The administrative command mail_table_update updates information in the system data base mail_table for a particular user.

Usage:

```
mail_table_update name -control args-
```

1. name is a registered person id or alias.

2. control_args are among the following:

-mailbox path -mbx path	path is the pathname of a default mailbox. If the suffix mbx is not present, it is assumed.
-destination string -ds string	string is a default destination for the dprint command, no longer than 12 characters.
-header string -he string	string is a default header for the dprint command, no longer than 64 characters.

Notes:

Any fields not specified by control arguments are not updated.

This command cannot replace the person id or alias.