

7 March 1982

Multics Technical Bulletin

MTB-574

From: Jay Pattin
To: MTB Distribution
Date: 7 March 1982
Subject: The forum interactive meeting subsystem.

This MTB describes the current implementation of forum and touches upon several issues which must be dealt with in the near future. There will be a second MTB published about a month after this one which will describe enhancements to the forum subsystem.

Please direct any comments or questions to the author:

By Multics mail at MIT or System-M to:

Pattin.PDO at MIT
Pattin.Multics at System-M

Or to the forums:

>libraries>forum>forum_dir>conprob at MIT
>udd>Demo>dbmt>con>conprob at System-M

Multics Project internal working documentation. Not to be distributed outside the project without the consent of the author or the Director of Multics System Development.

INTRODUCTION

The forum interactive meeting subsystem is intended to facilitate information flow between Multics users. A forum may be thought of as a cross between a meeting and a bulletin board. Announcements may be entered in a forum and be seen by all the participants, questions may be posed, opinions proffered, and discussions held; with the added benefit that a record of all transactions is maintained.

Forum allows users to create forums and specify who may participate in them, enter transactions and reply to previously-entered transactions, be notified when another user has entered a transaction, and determine what portion of the proceedings have been seen by individual participants.

Forum enables the effective distribution of information from the people who have the knowledge, to the people who need the information, and overcomes the barriers imposed by space, time, and number of participants. It is not meant to replace electronic mail, for there is no substitute for a secure and flexible system for one-to-one or one-to-few communication. It is rather designed to establish a wider community of users, sharing their knowledge and keeping them well-informed on issues of interest to all.

The forum subsystem is a modified version of the continuum subsystem developed several years ago at USGS. It is nearly compatible with the version 3 continuum developed by Mike Auerbach, and is almost exactly the same as the version 4 continuum developed by Spencer Love, Jay Pattin, and Jeff Schiller. The major change, other than naming, is that forum is based on the `ssu_` subsystem utilities instead of the older `subsystem_utils_` package.

Continuum has been in use on System-M for some time and on MIT for almost a year. During this time, it has had wide use on many different applications including as a bulletin board substitute, bug-reporting mechanism, and development and design meetings.

CURRENT STATUS OF FORUM

Currently, forum version 1.0 has been placed in the experimental libraries on both MIT and System-M. This version includes most of what is documented in this MTB. The version of forum which is herein documented, tentatively called version 1.1, has been written and tested to some extent. It will be installed in EXL on both sites as soon as the new version of the subsystem utilities is installed.

UNRESOLVED ISSUES

There are several issues which must be dealt with before forum can be installed. These include the location of the central forum directory, compatibility and incompatibility with version 4 continuum, the ring of

execution of the forum kernel, and security and access control issues.

The forum subsystem maintains the pathname of a directory called the central forum directory where site-wide public forums such as Bulletin_Board are kept. Currently there is no well-defined location for this directory. On MIT it is >libraries>forum>forum_dir and on System-M it is >udd>Demo>dbmt>con. neither of these is particularly acceptable. There have been suggestions made that it be a directory off the root or a subdirectory of >sc1.

Forum is only slightly different from the version of continuum that has been used since the summer of 1981. However, there are some major user-visible changes, though these are almost completely confined to naming. Since continuum was never installed software, we need not worry so much about name changes. The names of all the commands and several of the requests have been changed. In addition, forum uses the "forum" search path to locate forums instead of the "continuum" search path. Instructions on how to convert from continuum to forum are given in the info segment con_to_forum_conversion, which resides in >ex1>forum>info on both sites.

The other major change concerns notifications. Sending notifications of new transactions in continuum was extremely expensive, because it attempted to send a message to every user with the notify switch on, without checking to see if they were logged in. In order to alleviate the cost associated with entering a transaction, forum will require that the forum_accept_notifications command be given in order to receive notifications. This command will leave an event channel in the meeting which will be used to determine if the user is logged in before attempting to do a send_message_express.

Forums and continuum meetings are completely compatible. The forum subsystem may be used to access meetings created and used by the continuum subsystem and vice-versa. Forum may not work correctly with pre-version 4 continuum, but it is thought that any version 3 user community is insignificant, if it exists. However, for many reasons it is desirable to move the existing user community to forum as soon as possible. This will help make the transition between methods of receiving notifications easier, and will keep users from encountering some of the uncorrected bugs in continuum.

The forum kernel currently resides in ring 3. Installed inner-ring software seems to be restricted to ring 1, so forum will probably have to be moved into ring 1. When this is done, all existing forums on MIT and System-M should also be moved into ring 1.

The issues of security and access to forum have been of major interest in recent weeks. The questions that have been raised are: Should the ability to create forums be restricted, and if so, how; and should the ability to add participants to forums be restricted and,

7 March 1982

again, by what mechanism should it be restricted? The arguments in favor of restricting access are that site administrators would otherwise have insufficient control over who passed what information to whom through forum. The arguments against access control are that there does not seem to be any effective, general model which can be used to control access, that it would severely complicate the use of forum at sites that do not care about the security issues, and that no built-in security could ever be effective.

Since there is no consensus on what should be done as far as enforcing security, nothing has been. In accordance with the wishes of System-M administrators, there is a separate gate at that site containing the chairman functions. This is not planned to be part of forum as it is proposed in this MTB.

It is clear that some form of control is needed, but there does not seem to be any better form of control than the discretion of the forum chairmen and participants. With this in mind, forum will attempt to provide users with sufficient information to allow them to decide how secure the forum is, and what information should be divulged in it. A -eligible control argument has been added to the list_users command/request to show the user who may attend the forum, and all requests which use the pathname of the forum now say where the forum actually is, rather than displaying the path of the link used to locate it. A new chairman request, set_message, has been added to allow the chairman of a forum to store a message in the forum which will be printed when a user enters a forum and each time a transaction is entered.

I will not attempt to touch upon this issue further here. There is currently a very active discussion of these issues, and they will be presented in the second forum MTB.

FUTURE ENHANCEMENTS

Future enhancements to the forum subsystem will be the subject of a further MTB to be published in the near future. A quick summary of things currently on the drawing board follows.

Read-only participants:

Each forum will have a bit telling whether new attendees will be read-only participants. This bit may be set by the forum_create command and a new forum_(un)make_read_only command. The chairman may re/set the read-only switch for individuals. All current participants will be grandfathered into being allowed to write.

Garbage collection:

A chairman command to garbage-collect deleted transactions will be written. It is possible that this may also want to garbage-collect

deleted participants, but first we have to invent deleted participants. The garbage-collection of forums could also be made to store the attendees in alphabetical order, to allow for a binary-search lookup instead of a linear search. This would be a substantial performance improvement.

Addition of a forum_priv_ gate.

It has become evident that a gate is necessary which will allow SysMaint/SysAdmin personnel to invoke chairmanly powers on forums whose chairman is no longer present.

Improvements to the transaction specifier parser.

The transaction specifier parser should be re-written to support constructs similar to what the read_mail parser does. This includes selection by date, multiple usernames and regexps connected by logical operators, and regexps attached to keywords (fref/foo/ = first on chain containing foo).

Restructuring of forums.

For many reasons, the format of forums should be changed. Forums should be MSF's. This would allow better access control and virtually unlimited growth of the proceedings segment among other good things.

Deletion of one's own transactions.

Expand the rules allowing deletion of transactions to at least allow participants to delete a newly-entered incorrect transaction.

DOCUMENTATION.

The remainder of this document consists of the online info segments describing the forum subsystem and its subroutine interface. The first section covers the forum commands.

03/06/82 forum

Syntax: forum {forum_name} {-control_args}

Function: Enters the forum interactive meeting system.

Arguments:

forum_name

is the name or pathname of the forum to be entered immediately upon invoking forum. If a pathname is specified, it identifies the forum to enter. Otherwise, forum searches for the forum_name by using the "forum" search list. See "Notes on Search List:" below.

Control arguments:

-abbrev, -ab

specifies that abbreviation processing should be done by the forum request processor. If the -profile argument is not given, the user's default profile segment will be used.

-brief, -bf

suppresses some messages from forum and shortens others.

-forum forum_name

enters the forum whose entryname or pathname is forum_name. See the description of the forum_name argument above.

-input_fill, -ifi

specifies that transactions will be filled before being entered into the proceedings of a forum. See "Notes on filling" below for a discussion of filling. (Default)

-line_length N, -ll N

cause filling (fill request and default filling) to be done with this line length. (Default is 72)

-list {trans_spec}, -ls {trans_spec}

causes forum to list the specified transactions before entering the request loop. If no transaction specifier is given, all the transactions in the forum will be listed.

-long, -lg

use long messages from forum. (Default)

-no_abbrev, -nab

Specifies that abbreviation processing is not to be done by the forum request processor. (Default)

-no_input_fill, -nif

specifies that transactions will not be filled before being entered into the proceedings of a forum. See "Notes on filling" below for a discussion of filling.

- no_output_fill, -nof
specifies that transactions will not be filled before being printed on the user's terminal. See "Notes on filling" below for a discussion of filling. (Default)
- no_prompt
do not prompt in the request loop.
- output_fill, -ofi
specifies that transactions be filled before being printed or written. See "Notes on filling" below for a discussion of filling.
- prompt prompt_string
set the request loop prompt to prompt_string. (Default is "forum:")
- profile profile_path
Specifies that abbreviation processing is to be done, and that the profile named by profile_path is to be used. If this control argument is given, the -abbrev argument need not be.
- quit
tells forum to process the initial request line and then return without entering the request loop. If the initial request line is aborted, the request loop will be entered.
- request string, -rq string
execute the requests in string before entering the request loop.

Notes:

Request lines use () for iteration, "" for quoting, and [] to invoke forum active requests, listed below ("List of active requests").

Any request line which begins with ".." will be passed directly to the Multics command processor with the leading ".." stripped off. Consequently, any reference to an active function will be evaluated by the Multics command processor. The 'execute' (e) forum request may also make use of active strings via the square brackets ([]), but these will be evaluated as active requests by forum before the command line is passed to the Multics command processor.

Notes on filling:

Transactions may be filled out to a given line length on both input and output. By default, all transactions are filled to a line length of 72 upon entry. The -input_fill (-if) and -no_output_fill (-nof) control arguments are the default. These control arguments specify a global attribute for the invocation of forum, which may be overridden by control arguments associated with the various requests for inputting and outputting transactions (such as the -no_fill control argument to the 'print' request).

List of requests:

Only the most important aspects of the requests are documented here. For further information on these requests, use the forum 'help' request.

? lists the available forum requests and active requests.

. identifies forum with version number, gives forum_name if attending, gives count of new, total, last, and current transactions, and number of lines in the unprocessed transaction.

forum_list {forum_names} {-control_args},
 fls {forum_names} {-control_args}
 prints a list of selected forums and information about them.

goto forum_name, g forum_name
 enters the user into the forum_name forum.

help {STR}
 prints information about request names or topics. A list of available topics is produced by the list_help request.

list {trans_specs} {-control_args}, ls {trans_specs} {-control_args}
 prints a summary of the specified transactions.

list_help {topics}, lh {topics}
 Prints a list of available info segments whose names include a topic string.

list_requests {-control_args}, lr {-control_args}
 prints information about forum requests.

list_users {-control_args}, lsu {-control_args}
 prints information about specified participants in a forum.

print {trans_specs} {-control_args}, pr {trans_specs} {-control_args}
 prints selected transactions from a forum.

quit {-control_arg}, q {-control_arg}
 exits forum.

reply {trans_spec} {-control_args}, rp {trans_spec} {-control_args}
 enters/builds a new transaction in a forum which has as its subject a reference to some other transaction in the form "Re: <some other subject>", AND which will be logically linked to the transaction specified by trans_spec.

reset {trans_spec} {-control_args}, rs {trans_spec} {-control_args}
 resets the user's current or highest seen transaction index to the specified transaction.

subject {strings}, sj {strings}
 Print or modify the subject of an unprocessed transaction.
 If strings are supplied, they are catenated together to become the new subject. If no strings are supplied, the current subject is printed.

talk {-control_args}
 enters/builds a new transaction in a forum forum.

write {trans_specs} {-control_args}, w {trans_specs} {-control_args}
 write selected transactions to a segment.

abbrev {-control_args}, ab {-control_args}
 turns abbreviation processing on or off and changes profile segments.

answer STR {-control_args} request_line
 supply an answer to a question asked by a request.

apply STRs, ap STRs
 places the unprocessed transaction into a temporary segment, concatenates all the STRs with the pathname, and passes the result to the Multics command processor. The temp segment is then read back in as the unprocessed transaction.

chairman {forum_name} {-control_arg}, cm {forum_name} {-control_arg}
 prints the user_id (Person_id.Project_id) of the forum's chairman, or changes the chairman

current_forum {-control_args}, cf {-control_args}
 prints the name of the current forum.

delete trans_specs, dl trans_specs
 allows the chairman to delete specified transactions from the proceedings.

do {request_line} {args}, do {-control_args}
 substitutes args into the request_line and passes the result to the forum request processor. -control_args can be -nogo to suppress execution -long (-lg) to display expanded line before execution, and more.

enter {-control_args}, en {-control_args}, send {-control_args}
 enters the unprocessed transaction into the proceedings of a forum.

execute STRs, e STRs
 executes STRs as a Multics command line after evaluating forum active requests. As an active request, return the result of evaluating strings as an Multics active string.

fill {-control_args}, fi {-control_args}
 reformats transaction text to fit in a given line length.

forum_dir, fd
prints the pathname of the central forum directory.

if EXPR -then request_line {-else request_line}
conditionally execute a request.

qedx, qx
invokes the qedx editor on the unprocessed transaction.

ted
invokes the ted editor on the unprocessed transaction.

set_message {-control_args}
sets a greeting message for the forum.

set_notify_off {forum} {-control_args}, snf {forum} {-control_args}
turns off the notify indicator for a user in a forum. The notify indicator is off by default.

set_notify_on {forum} {-control_args}, snn {forum} {-control_args}
turns on notify indicator for this forum. The means that the user will get an interactive message every time a new transaction is entered in the forum.

set_participate_off {forum_name} {-control_args},
spf {forum_name} {-control_args}
removes the user as a participant of a forum.

set_participate_on forum_name {-control_args},
spn forum_name {-control_args}
restores the user as a participant of a forum.

subsystem_name
prints the name of the subsystem. ("forum").

subsystem_version
prints the current version of forum.

List of active requests:

chairman {forum_name}, cm {forum_name}
returns the person_id.project_id of forum chairman.

current_forum {-control_args}, cf {-control_args}
returns the name of the current forum.

forum_dir, fd
returns absolute pathname of central forum directory.

do {request_string} {args}

returns expanded request string.

execute STRs, e STRs

invokes Multics active function within forum request line.

forum_list, fls

returns names of forums which have new transactions.

list_users {-control_args}, lsu {-control_args}

return names of participants matching given conditions.

subject

returns subject of unprocessed transaction.

subsystem_name

returns the name of the subsystem. ("forum").

subsystem_version

returns the current version of forum.

Transaction specifiers:

Transaction specifiers are used to identify transactions. They are used as arguments to the forum requests that act on transactions. In addition, there are several requests and active requests that return information about transaction numbers. Use the forum request 'help trans_specs' for more information on transaction specifiers.

Notes on search list:

The forum command uses the "forum" search path. For more information about search paths, see the descriptions of the search facility commands in the MPM Commands and Active Functions manual, AG92.

Type:

psp forum

to see what the current "forum" search list is. The default search list is composed of the "central forum directory". This directory can be obtained by using the forum_dir command/active function or the forum_dir request/active request from within forum itself. The central forum directory contains the forums which have been set up for public exposure. Private forums may be established in any directory desired with the 'forum_create' command. In order to allow forum to locate such private forums, the user may include the directory pathname of the private forum in the "forum" search list.

Locating Private Meetings:

In order to allow forum to locate forums which do not reside in the central forum directory, it is necessary to manipulate the "forum" search list. This can be accomplished in one of three ways:

- 1) Enter forum, and use the request
..asp forum private_forum_dir_path
- 2) Before entering forum, use the Multics command:
ssp forum [fd] private_forum_dir_path
- 3) Use the pathname of the private forum with all
forum requests as in
forum >udd>proj>fd>private

7 March 1982

MTB-574

01/23/82 forum_dir, fd

Syntax: fd

Syntax as an active function: [fd]

Function:

Prints/returns the absolute pathname of the central forum directory.

Notes:

The central forum directory is the location for site-approved public forums. This directory is included in the default forum search path.

The command:

```
add_search_paths forum [forum_dir]
```

may be used to add the central directory to the forum search path.

01/23/82 forum_list, fls

Syntax: fls {forum_names} {-control_args}

Syntax as an active function: [fls {-control_args}]

Function:

Prints a list of selected forums on the user's terminal.
For each forum selected, information about the names of the forum and per-user forum attributes are listed.

Arguments:

forum_names

are optional forum_names. If any are supplied, information about only the specified forum_names is printed. The star convention is supported for forum_names. If no forum_names are given, information for all forums found in the 'forum' search path will be printed.

Control arguments:

-absolute_pathname, -absp

Causes the absolute pathname of a forum to be printed. The default is to list the long and short forum names only.

-all, -a

specifies that information about all forums be printed. The default is to print information about the forums in which the user is a participant only.

-brief, -bf

suppresses the message 'No forums have changed' which is printed by default if the -changes control argument is used and no forums have changed.

-chairman {username}, -cm {username}

specifies that information about forums of which username is chairman be printed. If username is not given, the user's personid will be used.

-changes, -chg

specifies that information about forums in which the user is a participant and in which new transactions have been entered be printed.

-count, -ct

prints out the number of new transactions for a forum in which the user is participating. This control argument may not be used if con_list_forums is invoked as an active function.

-eligible, -elig

specifies that information about all forums in which the user is eligible to participate be printed. (Default)

- exclude forum_names, -ex forum_names
excludes the forums identified by forum_names from the output list. This control argument is designed to be used when a sturname has been supplied as the forum_names in order to exclude the specified forums from being selected. The default (-include) is to select all forums which match the specified sturname.
- header, -he
print the forum header. This is the default, unless the -changes control argument was supplied.
- include forum_names, -incl forum_names
specifies that the forums identified by forum_names not be excluded from the output list. This control argument is intended to be used when a sturname has been specified for selecting forums. It is the default.
- inhibit_error, -ihe
specifies that warning messages for such as things as bad forum format and errors encountered while search for forums are not to be printed.
- long, -lg
causes the message 'No forums have changed' to be printed if the -changes control argument was used and no forums in which the user is a participant had changed. (Default)
- no_header, -nhe
suppress printing of the forum header.
- no_inhibit_error, -nihe
specifies that all warning messages be printed. (Default)
- notify, -nt
specifies that only forums in which the user has the notify flag set be listed.
- participating, -part
specifies that only forums in which the user is participating be listed. The default is to print information about all forums which the user is eligible to participate in.
- user_access_id
specifies that the user specified by access_id be used in determining participation, eligibility, and changes attributes. access_id is in the form Person_id.Project_id.tag, where any of the components can be the character '*'. If a component is omitted, it is assumed to be '*'.

-verbose, -vb

prints the chairman's userid and current and last transactions for each forum. May not be used with -changes.

List of flags:

The output from this command may include flags which have the following interpretation:

- e eligible flag: indicates that the user may participate in the forum.
- p participant flag: indicates that the user is a participant in forum (i.e., has 'gone to' the forum at least once).
- r removed flag: indicates that the user has removed himself from participation in the forum.
- n notify flag: indicates that the user has turned on the notify flag in the forum (i.e., that the user has requested online notification when new transactions are entered in the forum).
- c change flag: indicates that new transactions have been entered in the proceedings of the forum since the user last attended this forum.

Note that the flags corresponding to the selection criteria are not shown (i.e., if the user is selecting forums to which he is eligible, the 'e' flag is not printed).

02/12/82 forum_list_users, flsu

Syntax: flsu {forum_name} {-control_args}

Syntax as an active function: [flsu {forum_name} {-control_args}]

Function:

Prints a list of selected participants of a forum. For each participant selected, the person_id, project_id, current transaction number (last one seen), the date/time last attended, and several flags are listed. As an active function, it returns a list of person_ids.

Arguments:

forum_name

is the name or pathname of the forum for which participants are to be listed. Either this argument or the -forum control argument, but not both, must be given.

Control arguments:

-all, -a

specifies that all participants be listed, including those who have been 'removed' from the meeting.

-attending, -at

specifies that only participants who are currently attending this meeting be listed.

-eligible, -elig

prints a list of users and projects that are eligible to attend the forum.

-forum forum_name

specifies that participants of the 'forum_name' forum are to be listed. forum_name is either a name or a pathname of a forum.

-header, -he

print the header. (Default)

-no_header, -nhe

Suppresses printing of the name_list header.

-notify, -nt

specifies that only participants with the notify flag on should be listed.

-seen transaction_number

specifies that only participants who have read the specified transaction should be listed.

-totals, -tt

specifies that only the total number of participants selected be printed.

-unseen transaction_number

specifies that only participants who have not read the specified transaction should be listed.

-user person_ids

specifies that only information about the named participants be given. All arguments following -user until the next control argument are taken as person_ids.

Note:

The displayed flags have the following meanings: "r" indicates that the participant has been removed from the forum, and is therefore no longer a participant, and the "n" flag means that the user has the "notify flag" turned on.

7 March 1982

MTB-574

01/23/82 forum_create, fcr

Syntax: fcr

Function:

Creates a new forum, with the user as chairman.

Notes:

This command engages the user in a dialogue, prompting her for all information necessary for the creation of a new forum, including long and short forum names, participant information, and the first transaction in the forum.

01/23/82 forum_delete, fd1

Syntax: fd1 forum_name

Function:
deletes a forum.

Argument:
forum_name
is the pathname or entryname of the forum to be deleted. Pathnames contain greater-than (>) or less-than (<) characters. If a pathname is specified, it identifies the forum to be adjusted. Otherwise, fd1 searches for the forum_name by using the "forum" search path.

Note:
Only the chairman of a forum may delete that forum.

7 March 1982

MTB-574

01/23/82 forum_add_participant, fapt

Syntax: fapt forum_name person_id

Function:

Adds the person identified by person_id as a participant of the forum forum_name meeting.

Arguments:

forum_name

is the pathname or entryname of the forum to be adjusted. Pathnames contain greater-than (>) or less-than (<) characters. If a pathname is specified, it identifies the forum to be adjusted. Otherwise, fapt searches for the forum_name by using the "forum" search list.

person_id

is the person_id of the participant to be added.

Note:

The user invoking this command must be the chairman of the forum_name forum.

01/23/82 forum_add_project, fapj

Syntax: fapj forum_name project_id

Function:

Allows users from the project identified by project_id to become participants of the forum_name forum.

Arguments:

forum_name

is the pathname or entryname of the forum to be adjusted. Pathnames contain greater-than (>) or less-than (<) characters. If a pathname is specified, it identifies the forum to be adjusted. Otherwise, fapj searches for the forum_name by using the "forum" search list.

project_id

is the project_id for the project to be admitted to the forum.

Note:

The user invoking this command must be the chairman of the forum_name forum.

7 March 1982

MTB-574

01/23/82 forum_make_public, fmp

Syntax: fmp forum_name

Function:

opens the forum_name forum to public participation.

Argument:

forum_name

is the pathname or entryname of the forum to be adjusted. Pathnames contain greater-than (>) or less-than (<) characters. If a pathname is specified, it identifies the forum to be adjusted. Otherwise, fmp searches for the forum_name by using the "forum" search list.

Note:

The user invoking this command must be the chairman of the forum_name forum.

01/23/82 forum_remove_participant, frpt

Syntax: frpt forum_name person_id

Function:

removes the person identified by person_id as a participant of the forum_name forum.

Arguments:

forum_name

is the pathname or entryname of the forum to be adjusted. Pathnames contain greater-than (>) or less-than (<) characters. If a pathname is specified, it identifies the forum to be adjusted. Otherwise, frpt searches for the forum_name by using the "forum" search list.

person_id

is the person_id of the user to be removed from participation in the forum.

Note:

The user invoking this command must be the chairman of the forum_name forum.

7 March 1982

MTB-574

01/23/82 forum_remove_project, frpj

Syntax: frpj forum_name project_id

Function:

removes all participants who are currently allowed participation by virtue of their project participation, not by virtue of their person_id, from the forum_name forum.

Arguments:

forum_name

is the pathname or entryname of the forum to be adjusted. Pathnames contain greater-than (>) or less-than (<) characters. If a pathname is specified, it identifies the forum to be adjusted. Otherwise, frpj searches for the forum_name by using the "forum" search list.

project_id

is the project_id of the project whose members are to be removed from participation if they lack explicit person_id participation rights.

Note:

The user invoking this command must be the chairman of the forum_name forum.

01/23/82 forum_unmake_public, fump

Syntax: fump forum_name

Function:

removes all participants from the forum_name forum who are participating by virtue of the public attribute. All users who are not participating because of project_id or person_id rights are removed.

Argument:

forum_name

is the pathname or entryname of the forum to be adjusted. Pathnames contain greater-than (>) or less-than (<) characters. If a pathname is specified, it identifies the forum to be adjusted. Otherwise, fump searches for the forum_name by using the "forum" search list.

Note:

The user invoking this command must be the chairman of the forum_name forum.

03/01/82 forum_accept_notifications, fant

Syntax: fant {-control_args}

Function:

Enables reception of interactive messages notifying the user of new transactions being entered in forums. Notifications will be enabled for all forums in the "forum" search list for which the user has set the "notify" flag.

Control arguments:

-inhibit_error, -ihe

Suppresses error messages about being unable to access forums in the search list. (Default)

-no_inhibit_error, -nihe

Print all error messages.

Notes:

Type "help forum" for a description of the "forum" search list and the meaning and manipulation of the notify flag.

03/01/82 forum_refuse_notifications, frnt

Syntax: frnt {-control_args}

Function:

Disables reception of interactive messages notifying the user of new transactions being entered in forums. Notifications will be disabled for all forums in the "forum" search list.

Control arguments:

-inhibit_error, -ihe

Suppresses error messages about being unable to access forums in the search list. (Default)

-no_inhibit_error, -nihe

Print all error messages.

Notes:

Type "help forum" for a description of the "forum" search list and the meaning and manipulation of the notify flag.

7 March 1982

MTB-574

FORUM REQUESTS

The next section covers all the requests available from within the forum subsystem. This includes requests provided by the `ssu_subsystem` utilities, whose documentation is as provided by `ssu_` and thus contains references to subsystems other than forum, such as `read_mail`.

01/24/82 Standard subsystem request: ?

Syntax: ?

Function:

prints a list of requests available in this subsystem.

Notes:

There may be other requests available in the subsystem which are not listed by this request. Such requests may be listed using the request:

```
list_requests -all
```

01/22/82 forum request: .

Syntax: .

Function:

Prints information about the current version of forum, the forum the user is attending (if any) and the unprocessed transaction (if any). The format of the output is:

```
forum 1.0: 0 new, 220 last, 220 current.  
Attending the >udd>Company>forums>excursions forum.  
12 lines unprocessed; subject: Re: picnic
```

If the user is attending a forum, the number of new (unseen) transactions, the transaction number of the last transaction, and the number of the current transaction are printed. If the user has built a transaction but not yet entered it into a forum, the subject and number of lines in that transaction are printed.

7 March 1982

MTB-574

01/24/82 Standard subsystem request: abbrev, ab

Syntax: ab {-control_args}

Function:

controls abbreviation processing within the subsystem.

Control Arguments:

-off

specifies that abbreviations are not to be expanded.

-on

specifies that abbreviations should be expanded. (Default)

-profile PATH

specifies that the segment named by PATH is to be used as the profile segment; the suffix ".profile" is added to PATH if not present. The segment named by PATH must exist.

Notes:

Most subsystems which support abbreviation processing provide command line control arguments (-abbrev, -no_abbrev, -profile) to specify the initial state of abbreviation processing within the subsystem. For example, a Multics abbreviation could be defined to invoke the forum subsystem with a default profile as follows:

```
.ab fm do "forum -abbrev -profile [hd]>forum &rf1"
```

If invoked with no arguments, this request will enable abbreviation processing within the subsystem using the profile that was last used in this subsystem invocation. If abbreviation processing was not previously enabled, the profile in use at Multics command level is used; this profile is normally [home_dir]>Person_id.profile.

See MPM Commands for a description of abbreviation processing.

01/24/82 Standard subsystem request: answer

Syntax: answer STR {-control_args} request_line

Function:

provides preset answers to questions asked by another request.

Arguments:

STR

is the desired answer to any question. If the answer is more than one word, it must be enclosed in quotes. If STR is -query, the question is passed on to the user. The -query control argument is the only one that can be used in place of STR.

request_line

is any subsystem request line. It can contain any number of separate arguments (i.e., have spaces within it) and need not be enclosed in quotes.

Control Arguments:

-brief, -bf

suppresses printing (on the user's terminal) of both the question and the answer.

-call STR

evaluates the active string STR to obtain the next answer in a sequence. The active string is constructed from subsystem active requests and Multics active strings (using the subsystem's execute active request). The outermost level of brackets must be omitted (ie: "forum_list -changed") and the entire string must be enclosed in quotes if it contains request processor special characters. The return value "true" is translated to "yes", and "false" to "no". All other return values are passed as is.

-match STR

answers only questions whose text matches STR. If STR is surrounded by slashes (/), it is interpreted as a qedx regular expression. Otherwise, answer tests whether STR is literally contained in the text of the question. Multiple occurrences of -match and -exclude are allowed (see Notes below). They apply to the entire request line.

-exclude STR, -ex STR

passes on, to the user or other handler, questions whose text matches STR. If STR is surrounded by slashes (/), it is interpreted as a qedx regular expression. Otherwise, answer tests whether STR is literally contained in the text of the question. Multiple

occurrences of `-match` and `-exclude` are allowed (see Notes below). They apply to the entire request line.

`-query`

skips the next answer in a sequence, passing the question on to the user. The answer is read from the `user_i/o` I/O switch.

`-then STR`

supplies the next answer in a sequence.

`-times N`

gives the previous answer (`STR`, `-then STR`, or `-query`) `N` times only (where `N` is an integer).

Notes:

Answer provides preset responses to questions by establishing an on unit for the condition `command_question`, and then executing the designated request line. If any request in the request line calls the `command_query_subroutine` (described in the MPM Subroutines) to ask a question, the on unit is invoked to supply the answer. The on unit is reverted when the answer request returns to subsystem request level. See "List of System Conditions and Default Handlers" in the MPM Reference Guide for a discussion of the `command_question` condition.

If a question is asked that requires a yes or no answer, and the preset answer is neither "yes" nor "no", the on unit is not invoked.

The last answer specified is issued as many times as necessary, unless followed by the `-times N` control argument.

The `-match` and `-exclude` control arguments are applied in the order specified. Each `-match` causes a given question to be answered if it matches `STR`, each `-exclude` causes it to be passed on if it matches `STR`. A question that has been excluded by `-exclude` is reconsidered if it matches a `-match` later in the request line. For example, the request line:

```
answer yes -match /fortran/ -exclude /fortran_io/ -match
/^fortran_io/
```

answers questions containing the string "fortran", except that it does not answer questions containing "fortran_io", except that it does answer questions beginning with "fortran_io".

01/22/82 forum request: apply, ap

Syntax: ap line

Function:

places the text of the unprocessed transaction into a temporary segment, catenates the pathname of this segment to the end of `command_line`, executes the resulting Multics command line, and updates the transaction text from the contents of the temporary segment after execution.

Arguments:

`command_line`

is the Multics command line to which is catenated the pathname of the temporary segment before execution.

Notes:

The supplied command line need not be enclosed in quotes. However, if there are (), [], or ''s in the command line which should be processed by the Multics command processor, they should be enclosed in quotes to prevent processing by continuum's request processor.

This request may be used to edit the transaction with an editor other than `qedx` or `ted`. For example, the request

`apply emacs`

will invoke the emacs editor on the transaction.

7 March 1982

MTB-574

01/22/82 forum request: chairman, cm

Syntax: cm {meeting} {-control_args}

Syntax as an active request: [cm {meeting}]

Function:

Prints/returns the user_id (Person_id.Project_id) of the chairman of the specified meeting, or changes the chairman of the current meeting. If no arguments are given, the name of the chairman of the current meeting is printed.

Argument:

meeting

is the name or pathname of a forum meeting.

Control Arguments:

-force, -fc

tells forum not to ask if the chairman should be changed.

-set new_chairman

changes the chairman of the current meeting. new_chairman must be of the form Person_id.Project_id. Only the current chairman may change the chairman.

01/22/82 forum request: current_forum, cf

Syntax: cf {-control_args}

Syntax as an active request: [cf {-control_args}]

Function:

Prints or returns the entryname or pathname of the forum that the user is currently attending.

Control Arguments:

-absolute_pathname, -absp

specifies that the full pathname of the forum should be printed.

-entry, -et

specifies that only the entryname portion of the name of the current forum be printed. (Default)

7 March 1982

MTB-574

01/22/82 forum request: delete, dl

Syntax: dl trans_specs

Function:

Allows the chairman of a meeting to logically remove specified transactions from a meeting.

Arguments:

trans_specs

are transaction specifiers which determine which transactions will be deleted. Use the forum request "help trans_specs" for information on transaction specifiers. At least one transaction must be specified with this request.

Notes:

Only the chairman of a meeting may delete transactions. Deleted transactions will be ignored by all requests which process transactions. Deleted transactions may be restored by using the "retrieve" request.

01/24/82 Standard subsystem request: do

Syntax: do request_string {args}
or:
do -control_args

Syntax As An Active Request: [do "request_string" args]

Function:

expands a request line by substituting the supplied arguments into the line before execution. As an active request, returns the expanded request_string rather than executing it.

Arguments:

request_string
is a request line in quotes.

args
are character string arguments that replace parameters in request_string.

Control Arguments:

These control argument set the mode of operation of the do request:

- long, -lg
the expanded request line is printed before execution.
- brief, -bf
the expanded request line is not printed before execution. (Default)
- nogo
the expanded request line is not passed on for execution.
- go
the expanded request line is passed on for execution. (Default)
- absentee
an any_other handler is established which catches all conditions and aborts execution of the request line without aborting the process.
- interactive
the any_other handler is not established. (Default)

List Of Parameters:

Any sequence beginning with & in the request line is expanded by the do request using the arguments given on the request line.

&l
is replaced by argl. l must be a digit from 1 to 9.

&(l)
is also replaced by argl. l can be any value, however.

&ql
is replaced by argl with any quotes in argl doubled. l must be a digit from 1 to 9.

&q(l)
is also replaced by argl with any quotes doubled. l can be any value.

&rl
is replaced by argl surrounded by a level quotes with any contained quotes doubled. l must be a digit from 1 to 9.

&r(l)
is also replaced by a requoted argl. l can be any value.

&fl
is replaced by all the arguments starting with argl. l must be a digit from 1 to 9.

&f(l)
is also replaced by all the arguments starting with argl. l can be any value.

&qfl
is replaced by all the arguments starting with argl with any quotes doubled. l must be a digit from 1 to 9.

&qf(l)
is also replaced by all the arguments starting with argl with quotes doubled. l can be any value.

&rl
is replaced by all the arguments starting with argl. Each argument is placed in a level of quotes with contained quotes doubled. l must be a digit from 1 to 9.

&rf(l)
is also replaced by all the arguments starting with argl, requoted. l can be any value.

&&

is replaced by an ampersand.

&

is replaced by a 15 character unique string. The string used is the same everywhere & appears in the request line.

&n

is replaced by the actual number of arguments supplied.

&f&n

is replaced by the last argument supplied.

01/24/82 Standard subsystem request: execute, e

Syntax: e LINE

Syntax As An Active Request: [e LINE]

Function:

executes the supplied line as a Multics command line. As an active request, evaluates a Multics active string and returns the result to the subsystem request processor.

Arguments:

LINE

is the Multics command line to be executed or the Multics active string to be evaluated. It need not be enclosed in quotes.

Notes:

The recommended method to execute a Multics command line from within a subsystem is the ".." escape sequence. The execute request is intended as a means of passing information from the subsystem to the Multics command processor.

All (), [], and ''s in the given line are processed by the subsystem request processor and not the Multics command processor. This fact permits the passing of the values of subsystem active requests to Multics commands when using the execute request or, when using the execute active request, to Multics active functions for further manipulation before returning the values to the subsystem request processor for use within a request line.

Examples:

The read_mail request line:

```
e mbla [mailbox]
```

can be used to list the ACL of the mailbox being read by the current invocation of read_mail. The read_mail request line:

```
write all [e strip_entry [mailbox]]
```

may be used to to write the ASCII representation of all messages in the mailbox into a segment in the working directory whose entry name is the same as that of the mailbox with the "mbx" suffix changed to "mail".

01/22/82 forum request: fill, fi

Syntax: fi {-control-args}

Function:

reformats the text of the unprocessed transaction to fit within a given line length. It is also used to control whether or not the transaction will be filled after it is entered.

Control arguments:

-line_length N, -ll N

is the width to be used when reformatting the text. (Default-- either the value specified by the -line_length argument to the forum command or 72 if this argument was not given). The line length given must be between 10 and 136.

-off

causes this transaction to not be filled by default when printed or written.

-on

causes this transaction to be filled by default when printed or written.

01/22/82 forum request: forum_list, fls

Syntax: fls {forum_names} {-control_args}

Syntax as an active request: [lsm {-control_args}]

Function:

prints a list of selected forum on the user's terminal. For each forum selected, information about the names of the forum and per-user forum attributes are listed.

Arguments:

forum_names

are optional forum_names. If any are supplied, information about only the specified forum_names is printed. The star convention is supported for forum_names. If no forum_names are given, information for all forums found in the 'forum' search path will be printed.

Control arguments:

-absolute_pathname, -absp

Causes the absolute pathname of a forum to be printed. The default is to list the long and short forum names only.

-all, -a

specifies that information about all forums be printed. The default is to print information about the forums in which the user is a participant only.

-brief, -bf

suppresses the message 'No forums have changed' which is printed by default if the -changes control argument is used and no forums have changed.

-chairman {username}, -cm {username}

specifies that information about forums of which the user is chairman be printed. If username is not given, the user's personid will be used.

-changes, -chg

specifies that information about forums in which the user is a participant and in which new transactions have been entered be printed.

-count, -ct

prints out the number of new transactions for a forum in which the user is participating. This control argument may not be used if forum_list is invoked as an active request.

-eligible, -elig

specifies that information about all forums in which the user is

eligible to participate be printed. (Default)

- exclude forum_names, -ex forum_names
excludes the forums identified by forum_names from the output list. This control argument is designed to be used when a starname has been supplied as the forum_names in order to exclude the specified forums from being selected. The default (-include) is to select all forums which match the specified starname.
- header, -he
print the header. This is the default, unless the -changes control argument was supplied.
- include forum_names, -incl forum_names
specifies that the forums identified by forum_names not be excluded from the output list. This control argument is intended to be used when a starname has been specified for selecting forums. It is the default.
- inhibit_error, -ihe
specifies that warning messages for such as things as bad forum format and errors encountered while search for forums are not to be printed.
- long, -lg
causes the message 'No forums have changed' to be printed if the -changes control argument was used and no forums in which the user is a participant had changed. (Default)
- no_header, -nhe
suppress printing of the header.
- no_inhibit_error, -nihe
specifies that all warning messages be printed. (Default)
- notify, -nt
specifies that only forums in which the user has the notify flag set be listed.
- participating, -part
specifies that only forums in which the user is participating be listed. The default is to print information about all forums which the user is eligible to participate in.
- user_access_id
specifies that the user specified by access_id be used in determining participation, eligibility, and changes attributes. access_id is in the form Person_id.Project_id.tag, where any of the components can be the character '*'. If a component is omitted, it is assumed to be '*'.
- verbose, -vb
prints the chairman's userid and current and last transactions for

each forum. May not be used with -changes.

List of flags:

The output from this command may include flags which have the following interpretation:

- e eligible flag: indicates that the user may participate in the forum.
- p participant flag: indicates that the user is a participant in forum (i.e., has 'gone to' the forum at least once).
- r removed flag: indicates that the user has removed himself from participation in the forum.
- n notify flag: indicates that the user has turned on the notify flag in the forum (i.e., that the user has requested online notification when new transactions are entered in the forum).
- c change flag: indicates that new transactions have been entered in the proceedings of the forum since the user last attended this forum.

Note that the flags corresponding to the selection criteria are not shown (i.e., if the user is selecting forums to which he is eligible, the 'e' flag is not printed).

01/22/82 forum request: forum_dir, fd

Syntax: fd

Syntax as an active request: [fd]

Function:

Prints/returns the absolute pathname of the central forum directory.

Notes:

The central forum directory is the location for site-approved public forum forums. This directory is included in the default forum search path. The forum request

e add_search_paths forum [forum_dir]

may be used to add the central directory to the forum search path.

7 March 1982

MTB-574

01/22/82 forum request: enter, en, send

Syntax: en {-control_args}

Function:

Enter an unprocessed transaction, which was created either by escaping to the forum request loop via \fq from 'talk' mode, or built using the qedx or ted requests.

Control Arguments:

-brief, -bf

Suppress the message "Transaction [nnnn] entered in <forum_name> forum.

-long, -lg

Print a message indicating that the transaction was successfully entered. (Default).

-forum forum_name

Enter the transaction into the proceedings of the forum_name forum. The default is to enter the transaction into the forum the user is currently attending. forum_name may be the name or pathname of a forum. This control argument may not be used if the transaction was built using the reply request.

01/22/82 forum request: goto, go, g

Syntax: g {forum_name} {-control_arg}

Function:

The goto request is used to enter a forum, or to switch from one forum to another. This request requires access to the forum_name forum.

Argument:

forum_name

is the pathname or entryname of an established forum. If a pathname is specified, it identifies the forum directly. Otherwise, the "forum" search list will be used in an attempt to locate the forum. The '.control' suffix is optional. Either this argument or the "-forum" control argument must be given.

Control argument:

-forum forum_name

enters the forum whose pathname or entryname is forum_name. See the description of the forum_name argument above.

01/24/82 Standard subsystem request: if

Syntax: if EXPR -then LINE1 {-else LINE2}

Syntax As An Active Request: [if EXPR -then STR1 {-else STR2}]

Function:

conditionally executes one of two request lines depending on the value of an active string. As an active request, returns one of two character strings to the subsystem request processor depending on the value of an active string.

Arguments:

EXPR

is the active string which must evaluate to either "true" or "false". The active string is constructed from subsystem active requests and Multics active strings (using the subsystem's execute active request).

LINE1

is the subsystem request line to execute if EXPR evaluates to "true". If the request line contains any request processor characters, it must be enclosed in quotes.

STR1

is returned as the value of the if active request if the EXPR evaluates to "true".

LINE2

is the subsystem request line to execute if EXPR evaluates to "false". If omitted and EXPR is "false", no additional request line is executed. If the request line contains any request processor characters, it must be enclosed in quotes.

STR2

is returned as the value of the if active request if the EXPR evaluates to "false". If omitted and the EXPR is "false", a null string is returned.

01/22/82 forum request: list, ls

Syntax: ls {trans_specs} {-control_args}

Function:

Print the transaction number, author, subject, and date/time of specified transactions on the user's terminal.

Arguments:

trans_specs

are transaction specifiers which determine the transactions to be listed. Use the forum request "help trans_specs" for more information on transaction specifiers. If no transaction specifiers are supplied, all transactions are listed.

Control arguments:

-fill, -fi

causes filling of listed transactions in order to obtain the line count.

-header, -he

prints the header. (Default)

-initial

only lists transactions at the beginning of transaction chains.

-no_fill, -nfi

suppresses filling of transactions in order to obtain the line count. (Default).

-forum forum_name

indicates that the specified transactions are to be selected from the forum_name forum. The default is to select transactions from the current forum.

-no_header, -nhe

suppresses printing of the header.

-no_update

causes the list request to not update the current transaction index.

-reverse, -rv

causes transactions to be listed in the reverse order that they were given in the transaction specifier.

-update

causes the current transaction index to be set to the index of the first transaction listed unless the current transaction is among those listed. (Default)

7 March 1982

MTB-574

01/24/82 Standard subsystem request: list_requests

Syntax: lr {STRs} {-control_args}

Function:

prints a brief description of selected subsystem requests.

Arguments:

STRs

specifies the requests to be listed. Any request with a name containing one of these strings is listed unless -exact is used in which case the request name must exactly match one of these strings.

Control Arguments:

-all, -a

includes undocumented and unimplemented requests in the list of requests eligible for matching the STR arguments.

-exact

lists only those requests one of whose names exactly match one of the STR arguments.

Notes:

If no STRs are given, all requests are listed.

When matching STRs with request names, a request name is considered to match a STR only if that STR is at the beginning or end of a word within the request name. Words in request names are bounded by the beginning and end of the request name and by the characters period (.), hyphen (-), underscore (_), and dollar sign (\$).

Examples:

The request line:

```
list_requests list
```

will match requests named list, list_users, and forum_list but will not match a request named prelisting.

02/12/82 forum request: list_users, lsu

Syntax: lsu {-control_args}

Syntax as an active request: [lsu {-control_args}]

Function:

Prints a list of selected participants of a forum meeting. For each participant selected, the person_id, project_id, current transaction number (last one seen), the date/time last attended, and several flags are listed. As an active request, it returns a list of person_ids.

Control arguments:

-all, -a

specifies that all participants be listed, including those who have been 'removed' from the forum.

-attending, -at

specifies that only participants who are currently attending this forum be listed.

-eligible, -elig

prints a list of users and projects that are eligible to attend the forum.

-forum forum_name

specifies that participants of the 'forum_name' forum are to be listed. The default is to list participants of the current forum.

-header, -he

print the header. (Default)

-no_header, -nhe

Suppresses printing of the name_list header.

-notify, -nt

specifies that only participants with the notify flag on should be listed.

-seen transaction_number

specifies that only participants who have read the specified transaction should be listed.

-totals, -tt

specifies that only the total number of participants selected be printed.

-unseen transaction_number

specifies that only participants who have not read the specified transaction should be listed.

-user person_ids
specifies that only information about the named participants be given. All arguments following -user until the next control argument are taken as person_ids.

Note:

The displayed flags have the following meanings: "r" indicates that the participant has been removed from the forum, and is therefore no longer a participant, and the "n" flag means that the user has the "notify flag" turned on.

01/22/82 forum request: print, pr, p

Syntax: p {trans_specs} {-control_args}

Function:

Prints specified transactions on the terminal.

Arguments:

trans_specs

are transaction specifiers which determine the transactions to be printed. Use the forum request "help trans_specs" for more information on transaction specifiers. If no transaction specifiers are supplied, the unprocessed transaction will be printed if there is one, otherwise the current transaction will be printed.

Control arguments:

-fill, -fi specifies that transactions are to be filled before being printed.

-initial

selects only transactions which are at the beginning of transaction chains.

-no_fill, -nfi

suppresses filling of transactions. (Default)

-forum forum_name

indicates that the specified transactions are to be selected from the forum_name forum. The default is to select transactions from the current forum.

-reverse, -rv

causes transactions to be printed in the reverse order that they were given in the transaction specifier.

7 March 1982

MTB-574

01/22/82 forum request: retrieve, rt

Syntax: rt trans_specs

Function:

Allows the chairman to retrieve specified transactions that were previously deleted with the 'delete' request.

Arguments:

trans_specs

are transaction specifiers which determine the transactions to be retrieved. Use the forum request "help trans_specs" for more information on transaction specifiers. Note: regular expressions and personids may not be used as transaction specifiers with this request.

01/22/82 forum request: qedx, qx

Syntax: qx

Function:

Invokes the "qedx" editor to edit the text of the unprocessed transaction or to build a new transaction.

Notes:

It is not necessary to issue a "w" request within "qedx" to reflect changes in the transaction to forum. The "qedx" request "1,\$dr" may be used to restore the original text of the transaction to the buffer while editing.

7 March 1982

MTB-574

01/22/82 forum request: quit, q

Syntax: q {-control_arg}

Function:

exits the forum subsystem.

Control argument:

-force, -fc

causes forum not to ask if the user really wants to quit if there is an unprocessed transaction.

01/24/82 Standard subsystem request: ready, rdy

Syntax: rdy

Function:

prints a ready message.

Notes:

The Multics `general_ready` command may be used to change the format of the ready message printed by this request and also after execution of request lines if the `ready_on` request is used. Type:

```
.. help general_ready
```

for more information on the available formats.

The default ready message gives the time of day and the amount of CPU time and page faults used since the last ready message was typed.

7 March 1982

MTB-574

01/24/82 Standard subsystem request: ready_off, rdf

Syntax: rdf

Function:

turns off printing of ready messages after execution of each subsystem request line.

01/24/82 Standard subsystem request: ready_on, rdn

Syntax: rdn

Function:

turns on printing of ready messages after execution of each subsystem request line.

01/22/82 forum request: reply, rp

Syntax: rp {transaction_specifier} {-control_args}

Function:

Allows a participant to build, and optionally enter into the proceedings of a forum, a transaction which is in reply to an earlier transaction.

Argument:

transaction_specifier

is the transaction specifier of the transaction to be replied to. If this argument is not given, the current transaction is replied to.

Control arguments:

-brief, -bf

suppresses printing of the message indicating that the transaction was successfully entered.

-fill, -fi

specifies that the transaction should be filled after the user has exited input mode. This is the default, unless the -input_fill (-ifi) control argument was given with the invocation of forum. The default fill width is 72, but the user can specify the fill width via the -line_length NN (-ll NN) control argument to the forum command.

-input_file pathname, -if pathname

specifies that the segment identified by pathname be entered into the proceedings. The default is to input the transaction from the terminal.

-long, -lg

print a message indicating that the transaction was successfully entered. (Default).

-forum forum_name

enters the transaction into the proceedings of the forum_name forum. The default is to enter the transaction into the proceedings of the forum the user is currently attending. forum_name may be a forum name or a pathname.

-no_fill, -nfi

specifies that the transaction not be filled to the default fill width then entered. The default is to fill the transaction, unless the -no_input_fill (-nifi) control argument was specified when forum was invoked. The fill width defaults to 72, unless specified differently via the -line_length NN (-ll NN) control argument was specified when forum was invoked.

-no_request_loop

causes the transaction to be entered into the forum without first entering the forum request loop. (Default)

-request_loop

causes the forum request loop to be entered before the transaction is entered in the forum.

-subject subject_string, -sj subject_string

Specifies that subject_string be used as the subject, avoiding the prompt for subject. The default is to prompt the user for the subject. If subject_string contains whitespace, it must be quoted.

-terminal_input, -ti

Specifies that the transaction be input from the user's terminal. (Default)

Subject processing:

Each transaction entered into the proceedings of a forum requires a subject field. To use the subject of an earlier transaction, the user should simply type in the transaction number in response to the "Subject:" prompt or with after the -subject control argument.

Terminal input:

Unless the user makes use of the -input_file control argument, the transaction will be built by taking lines from the user's terminal. There are three methods of exiting from terminal input mode:

line consisting of the single character period (.)

Enters the transaction as is, returns to forum request level.

\f

Exits terminal input mode and invokes 'qedx' editor on transaction.

\fq

Exits terminal input mode and enters forum request loop.

Notes:

A transaction which was not entered via the period (.) exit from terminal input mode may be entered from forum request level via the 'enter' request. Type 'help enter' from forum request level for details.

01/22/82 forum request: reset, rs

Syntax: rs {trans_specs} {-control_arg}

Function:

Resets the user's "current" or "highest transaction seen" index to the specified transaction number.

Argument:

trans_specs

is a transaction specifier which determines the value of the transaction number that the index will be reset to. Only one transaction specifier may be given. Type "help trans_specs" for information on transaction specifiers.

Control arguments:

-current

Specifies that the "current" index is to be adjusted. (Default)

-highest

Specifies that the "highest transaction seen" index is to be adjusted.

Notes:

The "current transaction index" is a logical pointer to a transaction in a meeting. Many forum requests, including print and write, use the value of this index if no transaction specifiers are supplied. It is initially set to the value of the "highest transaction seen" when the user enters a meeting. The "highest transaction seen" index is a logical pointer to the highest transaction that the user has printed or written to a segment. This index is used when determining which transactions are 'new'.

03/06/82 forum request: set_message

Syntax: set_message {-control_args}

Function:

Allows the chairman to set a message which will be printed each time a user enters the forum or enters a transaction.

Control arguments:

-delete, -dl

specifies that the existing message be deleted.

-fill, -fi

specifies that the message should be filled after the user has exited input mode. This is the default, unless the -no_input_fill (-nifi) control argument was given with the invocation of forum. The default fill width is 72, but the user can specify the fill width via the -line_length NN (-ll NN) control argument to the forum command.

-input_file pathname, -if pathname

specifies that the segment identified by pathname be entered into the proceedings. The default is to input the message from the terminal.

-no_fill, -nfi

specifies that the message not be filled to the default fill width when entered. The default is to fill the transaction, unless the -no_input_fill (-nifi) control argument was specified when forum was invoked. The fill width defaults to 72, unless specified differently via the -line_length NN (-ll NN) control argument was specified when forum was invoked.

-no_request_loop

causes the message to be entered into the forum without first entering the forum request loop. (Default)

-request_loop

causes the forum request loop to be entered before the message is entered in the forum.

-terminal_input, -ti

Specifies that the message be input from the user's terminal. (Default)

Terminal input:

Unless the user makes use of the -input_file control argument, the message will be built by taking lines from the user's terminal. There are three methods of exiting from terminal input mode:

line consisting of the single character period (.)

Enters the message as is, returns to forum request level.

7 March 1982

MTB-574

\f

Exits terminal input mode and invokes 'qedx' editor on message.

\fq

Exits terminal input mode and enters forum request loop.

Notes:

A message which was not entered via the period (.) exit from terminal input mode may be entered from forum request level via the 'enter' request. Type 'help enter' from forum request level for details.

01/22/82 forum request: set_notify_off, snf

Syntax: snf {forum_name} {-control_argument}

Function:

The 'snf' request turns off the 'notify' flag for a participant. If the notify flag is set, the participant will receive an interactive message when a new transaction is entered in the forum.

Argument:

forum_name

indicates that the 'notify' indicator for this forum is to be reset. The default is to turn off the notify flag for the current forum.

Control argument:

-user person_id

specifies that the notify flag for user person_id is to be reset. Only the chairman may reset another user's flag.

01/22/82 forum request: set_notify_on, snn

Syntax: snn {forum_name} {-control_argument}

Function:

This request turns on the 'notify' flag for a participant. If the notify flag is set, the participant will receive an interactive message when a new transaction is entered in the forum.

Argument:

forum_name

indicates that the 'notify' indicator for this forum is to be set. The default is to turn on the notify flag for the current forum.

Control argument:

-user person_id

specifies that the notify flag for user person_id is to be set. Only the chairman may set another user's flag.

01/22/82 forum request: set_participate_off, spf

Syntax: spf {forum_name} {-control_argument}

Function:

Removes a user from the list of active participants of a forum.

Argument: forum_name

removes a user from the 'forum_name' forum. The default is to remove the user from the current forum.

Control argument:

-user person_id

specified that user person_id is to be removed. Only the chairman of a forum may remove a user other than himself.

Note:

This request removes one from active participation in a forum, and can be reverted by the 'set_participate_on' ('spn') request. Removing oneself from participation in a forum is a convenient way to ignore a forum which is of no interest, but in which one has participation rights by default.

7 March 1982

MTB-574

01/22/82 forum request: set_participate_on, spn

Syntax: spn forum_name {-control_argument}

Function:

Reinstates a user as a participant of a forum.

Argument:

forum_name

is the name of the forum in which the user wants to be reinstated.

Control argument:

-user person_id

specified that user person_id is to be reinstated as a participant.
Only the chairman of a forum may reinstate a user other than himself.

01/22/82 forum request: subject, sj

Syntax: sj {new_subject} {-control_args}

Syntax as an active request: [sj]

Function:

changes the subject of the unprocessed transaction or prints/returns the subject of the unprocessed transaction.

Arguments:

new_subject

is a string formed by concatenating the non-control arguments. It becomes the subject of the unprocessed transaction.

Control arguments:

-default

changes the subject to the subject of the transaction that the unprocessed transaction is in reply to. This control argument may only be used with replies and may not be used if new_subject is given.

-subject, -sj

specifies that the subject begins with the next argument. Useful if the first character of the subject is '- '.

Notes:

When given no arguments, the subject request prints or returns the current subject of the unprocessed transaction, if any. An unprocessed transaction is a transaction that has been "built" by the user, either by exiting the talk request with the \fq request, or by using the qedx or ted requests; but has not yet been entered into a forum.

7 March 1982

MTB-574

02/24/82 Standard subsystem request: subsystem_name

Syntax: subsystem_name

Syntax as an active request: [subsystem_name]

Function:

prints the name of the subsystem; as an active request, returns the name of the subsystem.

02/24/82 Standard subsystem request: subsystem_version

Syntax: subsystem_version

Syntax as an active request: [subsystem_version]

Function:

prints the version number of the subsystem; as an active request,
returns the version number of the subsystem.

01/22/82 forum request: talk, t

Syntax: t {-control_args}

Function:

Allows a participant to build, and optionally enter into the proceedings of a forum, a new transaction.

Control arguments:

-brief, -bf

suppresses the message that says that the transaction was successfully entered.

-fill, -fi

specifies that the transaction should be filled after the user has exited input mode. This is the default, unless the -input_fill (-ifi) control argument was given with the invocation of forum. The default fill width is 72, but the user can specify the fill width via the -line_length NN (-ll NN) control argument to the forum command.

-input_file pathname, -if pathname

specifies that the segment identified by pathname be entered into the proceedings. The default is to input the transaction from the terminal.

-long, -lg

print a message indicating that the transaction was successfully entered. (Default).

-forum forum_name

enters the transaction into the proceedings of the forum_name forum. The default is to enter the transaction into the proceedings of the forum the user is currently attending. forum_name may be a forum name or a pathname.

-no_fill, -nfi

specifies that the transaction not be filled to the default fill width when entered. The default is to fill the transaction, unless the -no_input_fill (-nifi) control argument was specified when forum was invoked. The fill width defaults to 72, unless specified differently via the -line_length NN (-ll NN) control argument was specified when forum was invoked.

-no_request_loop

causes the transaction to be entered into the forum without first entering the forum request loop. (Default)

-request_loop

causes the forum request loop to be entered before the transaction is entered in the forum.

- subject subject_string, -sj subject_string
Specifies that subject_string be used as the subject, avoiding the prompt for subject. The default is to prompt the user for the subject. If subject_string contains spaces, it must be quoted.
- terminal_input, -ti
Specifies that the transaction be input from the user's terminal.
(Default)

Subject processing:

Each transaction entered into the proceedings of a forum requires a subject field. To use the subject of an earlier transaction, the user should simply type in the transaction number in response to the "Subject:" prompt or after the -subject control argument.

Terminal input:

Unless the user makes use of the -input_file control argument, the transaction will be built by taking lines from the user's terminal. There are three methods of exiting from terminal input mode:

- line consisting of the single character period (.)
Enters the transaction as is, returns to forum request level.
- \f
Exits terminal input mode and invokes 'qedx' editor on transaction.
- \fq
Exits terminal input mode and enters forum request loop.

Notes:

A transaction which was not entered via the period (.) exit from terminal input mode may be entered from forum request level via the 'enter' request. Type 'help enter' from forum request level for details.

7 March 1982

MTB-574

01/22/82 forum request: ted

Syntax: ted

Function:

Invokes the "ted" editor to edit the text of the unprocessed transaction or to build a new transaction.

Notes:

The transaction need not be written out with the 'w' request in order to make changes in the text.

02/16/82 forum request: write, w

Syntax: w {trans_specs} {-control_args}

Function:

Write the selected transactions into a segment.

Arguments:

trans_specs

are transaction specifiers which determine the transactions to be written. Use the forum request "help trans_specs" for more information on transaction specifiers. If no transaction specifiers are supplied, the current transaction is written.

Control arguments:

-brief, -bf

suppresses the message about how many transactions were written.

-extend

specifies that the selected transactions be written be added to the end of a segment. (Default)

-fill, -fi

specifies that all transactions be filled to the terminal's width when written. The default is to fill only transactions which were marked as unfilled when entered.

-forum forum_name

specifies that transactions be selected from the forum_name forum. The default is to select transactions from the current forum.

-initial

only writes transactions which are at the beginning of transaction chains.

-line_length LENGTH, -ll LENGTH

specifies that transactions are to be filled to LENGTH characters. (Default = 80)

-long, -lg

print the message about how many transactions were written. (Default)

-no_fill, -nfi

specifies that transactions not be filled to the terminal's width when written. The default is to fill only transactions which were marked as unfilled when entered.

-no_separator, -no_sep

specifies that transactions are to be written out with no separation between them.

- pathname <pathname>, -pn <pathname>
specifies that the transactions selected be written to the segment identified by <pathname>. If the entryname portion of <pathname> does not contain the ".trans" suffix, it will be assumed.
- reverse, -rv
causes transactions to be written in the reverse order that they were given in the transaction specifier.
- separator string, -sep string
writes string to the output segment after each transaction. string is an ioa_control string of up to 32 characters. (Default is "^|".)
- truncate, -tc
specifies that the segment be truncated before the transactions are written to it.

Notes:

If no pathname is given, the selected transactions are written to a segment in the working directory with the entryname forum_name.trans. The user's current transaction index will be set to the last transaction written.

ADDITIONAL DOCUMENTATION

The following information describes the transaction specifiers used to reference individual transactions or groups of transactions within a forum. These specifiers are recognized by all the forum requests which deal with individual transactions, such as list, print, and delete.

01/22/82 forum transaction specifiers

Transaction specifiers are used to refer to transactions in a forum. They are composed of transaction numbers, keywords, the arithmetic operators + and -, person_ids, and qedx-type regular expressions. Specifiers are used as arguments to the forum requests that operate on transactions. In addition, the keywords may be used as requests and active requests to return transaction numbers.

Transaction numbers and keywords:

Transaction numbers are integers assigned by forum when it enters a transaction into the proceedings. Transactions may be linked together (through the use of the 'reply' request) and may be addressed using this linking information. There are several keywords which can be used to refer to transactions symbolically:

unprocessed, u
refers to the unprocessed transaction.

first, f
refers to the first transaction in the proceedings.

last, l
refers to the last transaction in the proceedings.

next, n
refers to the transaction immediately after the current transaction.

previous, p
refers to the transaction immediately before the current transaction.

current, c
refers to the transaction last printed, written, or reset to.

all, a
refers to all transactions (equivalent to first:last).

firstref, fref
refers to the first transaction in the current chain of transactions.

lastref, lref
refers to the last transaction in the current chain of transactions.

nextref, nref
refers to the next transaction in the current chain of transactions.

priorref, pref
 refers to the previous transaction in the current chain of transactions.

restref, rref
 refers to all remaining transactions in the current transaction chain.

allref, aref
 refers to all transactions in the current chain of transactions.

new
 refers to transactions which the user has not yet seen.

All keywords except for 'unproc', 'all', 'new', 'rref' and 'allref' may be used in place of actual transaction numbers, and may be combined with either of the simple operators '+' or '-', as in '1-4' 'nref+1' and so on. Leading '+' and '-' are used to reference from the current transaction, '-4' is the same as 'current-4'. A range of transactions may be specified by two transaction numbers or keywords separated by a colon (:), as in 'first:last', 'c-3:nref' and so on.

Usage of keywords as requests:

All of the keywords except for 'unprocessed' may be used as requests and active requests. They return a list of the transaction numbers that the keyword represents. All of the keywords that use reference chains and the keywords 'next' and 'previous' may be given an argument to use as the 'current' transaction. Thus, 'aref 4' would return a list of all transactions which reference transaction 4.

Regular expressions and person_ids:

Simple regular expressions are character strings enclosed in slashes (/). The control arguments '-subject' and '-text' may be given immediately before a regexp. They specify that matching is to be done only against the subject or text respectively. By default, matching is done against both the subject and the text. For example:

```
/FORTRAN/
```

specifies all transactions which contain the string 'FORTRAN'. The null regexp "/" signifies that last regular expression used.

person_ids:

A person_id can be supplied in order to indicate that only transactions authored by the person identified by the person_id be selected.

SUBROUTINE INTERFACE

All functions performed by the forum subsystem are available to user programs through the gate forum_. Documentation for all entries in forum_ is included here.

Name: forum_\$create_forum

This entry is used to create forums. The caller becomes the chairman of the new forum.

Usage:

```
dc1 forum_$create_forum entry (char (*), char (*), fixed bin (35));  
call forum_$create_forum (dirname, entryname, status);
```

where:

dirname (Input)

The name of the directory where the forum is to be created.

entryname (Input)

The name of the forum to be created. The ".control" suffix is required. The entryname of the forum exclusive of ".control" can only have 20 characters because of the length of the ".proceedings" suffix.

status (Output)

A standard system error code or a code in forum_error_table_.

Name: forum_\$delete_forum

This entry is used to delete forums.

Usage:

```
dcl forum_$delete_forum (char (*), char (*), fixed bin (35));  
call forum_$delete_forum entry (dirname, entryname, status);
```

where

dirname (Input)

The name of the directory containing the forum or a link to the ".control" segment.

entryname (Input)

The entryname of the forum. The ".control" suffix must be given. A link may have up to 24 characters of forum name, but the forum segment can only have 20 because of the length of the ".proceedings" suffix. If a link, the suffix of the target segment primary name will be checked. The directory pathname for the proceedings segment and the forum name for the proceedings segment are taken from the target of the link, not from the pathname supplied.

status (Output)

A standard system error code or a code in forum_error_table_. The code forum_error_table_\$chairman_only will be returned if the caller is not the chairman of the forum.

Name: forum_\$open_forum

This entry opens a forum. It returns the forum_idx used in many of the other calls to forum_.

Usage:

```
dcl forum_$open_forum entry (char (*), char (*), fixed bin,  
    fixed bin (35));
```

```
call forum_$open_forum (dirname, entryname, forum_idx, status);
```

where

dirname (Input)
As above.

entryname (Input)
As above.

forum_idx (Output)
This number can be used for subsequent calls to to forum that take a forum index (most interesting operations). All coexistent openings of the same forum in the same process will return the same number.

status (Output)
If this number is not zero the forum could not be opened. It is a standard file system error code or a code in forum_error_table_.

Name: forum_\$close_forum

This entry closes an open forum.

Usage:

```
dcl forum_$close_forum entry (fixed bin, fixed bin (35));
```

```
call forum_$close_forum (forum_idx, status);
```

where

forum_idx (Input/Output)

As returned by forum_\$open_forum. This argument will be set to zero if the forum is successfully closed.

status (Output)

If this argument is nonzero, an error occurred while closing. This argument may be of some interest whilst debugging, but rarely if ever is it interesting enough to have a properly working program print out.

Name: forum_\$forum_add_name

This entry is used to add a name to the forum. The name is added to both the control and proceedings segment.

Usage:

```
dcl forum_$forum_add_name entry (fixed bin, char (*),
    fixed bin (35));
```

```
call forum_$forum_add_name (forum_idx, new_name, status);
```

where

forum_idx (Input)
As above.

new_name (Input)
The new name of the forum. The ".control" suffix must be given. The forum name can only have 20 characters because of the length of the ".proceedings" suffix.

status (Output)
A standard system error code or a code in forum_error_table_. The code forum_error_table_\$chairman_only will be returned if the caller is not the chairman of the forum.

Name: forum_\$forum_delete_name

This entry deletes a name from the control and proceedings segments of a forum.

Usage:

```
dcl forum_$forum_delete_name entry (fixed bin, char (*),
    fixed bin (35));
```

```
call forum_$forum_delete_name (forum_idx, old_name, status);
```

where

forum_idx (Input)
As above.

old_name (Input)
The name to be deleted from the forum. The ".control" suffix is required.

status (Output)
A standard system error code or a code in forum_error_table_. The code forum_error_table_\$chairman_only will be returned if the caller is not the chairman of the forum.

Name: forum_\$set_forum_acl

This entry is used to add or change an acl term on a forum. ACL terms may only be specified for User.* or *.Project, terms of the form User.Project are not allowed.

Usage:

```
dcl forum_$set_forum_acl entry (fixed bin, char (*), bit (1) aligned,
    bit (1) aligned, fixed bin (35));
```

```
call forum_$set_forum_acl (forum_idx, access_name, user_sw, set_sw,
    status);
```

where

forum_idx (Input)
As above.

access_name (Input)
The person_id or project_id whose acl is to be added or deleted.
Access_name may not contain a ".".

user_sw (Input)
This switch should be on if access_name is a person_id, and off if it is a project_id.

set_sw (Input)
This switch should on if access is to be added, and off if it is to be deleted.

status (Output)
A standard system error code or a code in forum_error_table_.

Name: forum_\$list_forum_acl

This entry is used to list the ACL on the control segment of the forum.

Usage:

```
dcl forum_$list_forum_acl entry (char (*), char (*), ptr, ptr,  
    fixed bin, fixed bin (35));
```

```
call forum_$list_forum_acl (dirname, pathname, area_ptr, acl_ptr,  
    acl_count, status);
```

where

dirname (Input)
As above.

entryname (Input)
As above.

area_ptr (Input)
points to an area where the segment_acl structure may be allocated.

acl_ptr (Output)
points to the allocated segment_acl structure. This structure is described in the MPM description of the hcs_\$add_acl_entries subroutine.

acl_count (Output)
is the number of acl entries in the segment_acl structure.

status (Output)
is a standard Multics error code.

Name: forum_\$get_message

This entry is used to read out the greeting message set by the chairman using the forum_\$set_message entry.

Usage:

```
dcl forum_$get_message entry (fixed bin, char (*), fixed bin (35));
```

```
call forum_$get_message (forum_idx, message, status);
```

where

forum_idx (Input)
As above.

message (Output)
The chairman-set message, this variable should be at least 256 characters long.

status (Output)
A standard system error code or a code in forum_error_table_. The code forum_error_table_\$no_message will be returned if the set_message entry has never been called.

Name: forum_\$set_switch

This entry is used to change several switches associated with participants in a forum.

Usage:

```
dcl forum_$set_switch entry (char (*), char (*), char (*), char (*),
    bit (1) aligned, fixed bin (35));
```

```
call forum_$set_switch (dirname, entryname, person_id, switch_name,
    switch_setting, status);
```

where

dirname (Input)
As above.

entryname (Input)
As above.

person_id (Input)
The username of the user on whose behalf you are setting the switch. The project_id should not be given. Starnames are not permitted. If this is blank, the current user is assumed. Only the chairman of a forum may set switches for persons other than herself.

switch_name (Input)
This can be "participating", "notify", or "listening".

switch_setting (Input)
The new switch setting. The usual boolean interpretation applies.

status (Output)
This is a standard file system error code or a code in forum_error_table_. If the code is zero, the switch was changed. If the code is forum_error_table_\$switch_not_changed, the old setting was the same as the new setting. Otherwise the operation failed.

Name: forum_\$set_switch_idx

This entry is the same as set_switch, except that a forum_idx is used instead of a pathname.

Usage:

```
dcl forum_$set_switch_idx entry (fixed bin, char (*), char (*),  
    bit (1) aligned, fixed bin (35));
```

```
call forum_$set_switch_idx (forum_idx, person_id, switch_name,  
    switch_setting, status);
```

where

forum_idx	(Input)
As above.	
person_id	(Input)
As above.	
switch_name	(Input)
As above.	
switch_setting	(Input)
As above.	
status	(Output)
As above.	

Name: forum_\$set_event_channel

This entry is used to set the event channel used to send wakeups when interesting forum events occur.

Usage:

```
dcl forum_$set_event_channel entry (char (*), char (*),
    fixed bin (71), fixed bin (35));
```

```
call forum_$set_event_channel entry (dirname, entryname,
    event_channel, status);
```

where

dirname (Input)

As above.

entryname (Input)

As above.

event_channel (Input)

The event channel which should receive wakeups from other forum users in the same forum.

status (Output)

If not zero, it is a standard error_table_ or forum_error_table_ code telling why this failed.

Notes:

Currently, 2 types of wakeups are sent by forum. The first one is a "are-you-there" wakeup sent merely to determine if the process is still there. The wakeup message in this case is 72-bits of 0. The other wakeup is sent when a transaction is entered while you are attending the forum. This message consists of one word containing the attendee_idx of the user who entered the transaction and one word containing the uid of the forum.

Name: forum_\$set_event_channel_idx

This entry is the same as set_event_channel except that a forum_idx is used to identify the forum instead of a pathname.

Usage:

```
dcl forum_$set_event_channel_idx entry (fixed bin, fixed bin (71),  
    fixed bin (35));
```

```
call forum_$set_event_channel_idx entry (forum_idx, event_channel,  
    status);
```

where

forum_idx	(Input)
As above.	
event_channel	(Input)
As above.	
status	(Output)
As above.	

Name: forum_\$forum_info

This entry is used to obtain interesting data about a forum.

Usage:

```
dcl forum_$forum_info entry (char (*), char (*), char (*),
    fixed bin (71), ptr, fixed bin (35));
```

```
call forum_$forum_info (dirname, entryname, access_name, access_time,
    forum_info_ptr, status);
```

where

dirname (Input)
As above.

entryname (Input)
As above.

access_name (Input)
The Person.Project.Tag for which access is to be calculated. If blank, this is done (more cheaply) for the calling user. The per-user information returned is for the Person portion of access_name, or the current user.

access_time (Input)
Changes information is calculated from the specified date-time. Usually, "clock ()" should be passed.

forum_info_ptr (Input)
A pointer to a copy of the structure in forum_info.incl.pl1. The version number must be set.

status (Output)
The code forum_error_table_\$not_eligible will be returned if the current user does not have access to obtain information about transaction counts and changes, but the chairman name is still valid in this case.

Name: forum_\$forum_info_idx

This entry is the same as forum_info except that a forum_idx is used to identify the forum instead of a pathname.

Usage:

```
dcl forum_$forum_info_idx entry (fixed bin, char (*), fixed bin (71),  
ptr, fixed bin (35));
```

```
call forum_$forum_info_idx (forum_idx, access_name, access_time,  
forum_info_ptr, status);
```

where

forum_idx As above.	(Input)
access_name As above.	(Input)
access_time As above.	(Input)
forum_info_ptr As above.	(Input)
status As above.	(Output)

Name: forum_\$list_users

This entry is used to extract information about the participants in a forum.

Usage:

```
dcl forum_$list_users entry (char (*), char (*), ptr, ptr,  
    fixed bin (35));
```

```
call forum_$list_users (dirname, entryname, area_ptr,  
    name_list_info_ptr, fixed bin (35));
```

where

dirname (Input)
As above.

entryname (Input)
As above.

area_ptr (Input)
A pointer to an area where the name_list_info structure may be allocated.

name_list_info_ptr (Output)
A pointer to the structure declared in forum_name_list_info.incl.pl1.

status (Output)
A standard system error code or a code in forum_error_table_.

Name: forum_\$list_users_idx

This entry is the same as list_users except that a forum_idx is used to identify the forum instead of a pathname.

Usage:

```
dcl forum_$list_users_idx entry (fixed bin, ptr, ptr,  
    fixed bin (35));
```

```
call forum_$list_users_idx (forum_idx, area_ptr, name_list_info_ptr,  
    status);
```

where

forum_idx	(Input)
As above.	
area_ptr	(Input)
As above.	
name_list_info_ptr	(Output)
As above.	
status	(Output)
As above.	

Name: forum_\$set_last_seen_idx

This entry is used to set the last_seen_idx for a user. This index denotes the highest transaction that the user has accessed.

Usage:

```
dcl forum_$set_last_seen_idx (fixed bin, fixed bin, bit (1) aligned,
    fixed bin (35));
```

```
call forum_$set_last_seen_idx (forum_idx, last_seen_trans_idx,
    force_switch, status);
```

where

forum_idx (Input)
As above.

last_seen_trans_idx (Input)
The index of the transaction to be marked as the most recent seen by this user.

force_switch (Input)
If this switch is on, the last_seen_idx for this user is always set to last_seen_trans_idx. If it is off, the maximum of last_seen_trans_idx and the current last seen for this user is used.

status (Output)
A standard system error code or a code in forum_error_table_.

Name: forum_\$get_forum_path

This entry is used to determine the actual name of the forum instead of the name of the link used to access it.

Usage:

```
dcl forum_$get_forum_path entry (char (*), char (*), char (*),  
char (*), fixed bin (35));
```

```
call forum_$get_forum_path (dirname, entryname, forum_dirname,  
forum_entryname, status);
```

where

dirname (Input)

As above.

entryname (Input)

As above.

forum_dirname (Output)

The directory where the forum resides, as returned by hcs_\$get_link_target.

forum_entryname (Output)

The entryname of the control segment of the forum, as returned by hcs_\$get_link target.

status (Output)

A standard system error code or a code in forum_error_table_.

Name: forum_\$get_forum_path_idx

This entry is the same as get_forum_path except that a forum_idx is used to identify the forum instead of a pathname.

Usage:

```
dcl forum_$get_forum_path_idx entry (fixed bin, char (*), char (*),
    fixed bin (35));
```

```
call forum_$get_forum_path_idx (forum_idx, forum_dirname,
    forum_entryname, status);
```

where

forum_idx (Input)
As above.

forum_dirname (Output)
The name of the directory where the forum was actually found, as returned by hcs_\$fs_get_path_name.

forum_entryname (Output)
The entryname of the control segment of the forum, as returned by hcs_\$fs_get_path_name.

status (Output)
is a standard file system error code or forum_error_table_\$invalid_forum_idx.

Name: forum_\$change_chairman

This entry is used to change the chairman of a forum.

Usage:

```
dcl forum_$change_chairman entry (char (*), char (*), char (*),
    fixed bin (35));

call forum_$change_chairman (dirname, entryname, chairman, status);
```

where

dirname (Input)
As above.

entryname (Input)
As above.

chairman (Input)
The name of the new chairman of the forum, in the form
Person.Project.

status (Output)
A standard system status code or a code from forum_error_table_. The
code forum_error_table_\$chairman_only will be returned if the caller
is not the current chairman.

Name: forum_\$change_chairman_idx

This entry is the same as change_chairman except that a forum_idx is used to identify the forum instead of a pathname.

Usage:

```
dcl forum_$change_chairman_idx entry (fixed bin, char (*),
    fixed bin (35));
```

```
call forum_$change_chairman_idx (forum_idx, chairman, status);
```

where

forum_idx (Input)
As above.

chairman (Input)
As above.

status (Output)
As above, except that forum_error_table_\$invalid_forum_idx may be returned.

Name: forum_sender_trans

This entry is used to enter a transaction into a forum.

Usage:

```
dcl forum_sender_trans entry (fixed bin, char (*), fixed bin,  
    char (*), bit (1) aligned, fixed bin, fixed bin (35));
```

```
call forum_sender_trans (forum_idx, text, previos_trans, subject,  
    filled_sw, trans_num, status);
```

where

forum_idx (Input)
As above.

text (Input)
The text of the transaction to be entered.

previous_trans (Input)
The index of the transaction that this transaction is in reply to, or
0 if it is the start of a new chain.

subject (Input)
The subject of the transaction to be entered.

filled_sw (Input)
Should be on if the transaction should not be filled on output.

trans_num (Output)
The index of the transaction entered, if successful.

status (Output)
A standard system error code or a code in forum_error_table_.

Name: forum_\$read_trans

This entry is used to read a transaction out from the inner ring.

Usage:

```
dcl forum_$read_trans entry (fixed bin, fixed bin, ptr, ptr,  
    fixed bin (35));
```

```
call forum_$read_trans (forum_idx, trans_idx, area_ptr,  
    forum_user_trans_ptr, status);
```

where

forum_idx (Input)
As above.

trans_idx (Input)
The index of the transaction to be read.

area_ptr (Input)
A pointer to an area where the forum_user_trans structure may be allocated.

forum_user_trans_ptr (Output)
A pointer to the structure declared in forum_user_trans.incl.pl1.

status (Output)
A standard system error code or a code in forum_error_table_. The code forum_error_table_\$trans_deleted will be returned if the deleted switch is set for the transaction but it has not been garbage collected. The code forum_error_table_\$trans_reaped will be returned if the transaction has been garbage collected.

Name: forum_\$trans_ref_info

This entry is used to obtain information about an individual transaction.

Usage:

```
dcl forum_$trans_ref_info entry (fixed bin, fixed bin, fixed bin,  
    fixed bin, bit (1) aligned, fixed bin (35));  
  
call forum_$trans_ref_info (forum_idx, trans_idx, prev_ref, next_ref,  
    deleted_sw, status);
```

where

forum_idx	(Input)
As above.	
trans_idx	(Input)
The index of the transaction for which information is desired.	
prev_ref	(Output)
The index of the preceding transaction in the same transaction chain.	
next_ref	(Output)
The index of the next transaction in the same transaction chain.	
deleted_sw	(Output)
This switch is on if the transaction has been deleted.	
status	(Output)
A standard system error code or a code in forum_error_table_.	

Name: forum_\$set_delete_sw

This entry is used to delete or retrieve a transaction.

Usage:

```
dcl forum_$set_delete_sw entry (fixed bin, fixed bin,  
    bit (1) aligned, fixed bin (35));
```

```
call forum_$set_delete_sw (forum_idx, trans_idx, delete_sw, status);
```

where

forum_idx (Input)
As above.

trans_idx (Input)
The index of the transaction whose delete switch is to be changed.

delete_sw (Input)
The switch should be on to delete the transaction, and off to retrieve the transaction.

status (Output)
A standard system error code or a code in forum_error_table_. It it will be forum_error_table_\$chairman_only if the caller is not the chairman of the forum.

Name: forum_\$forum_limits

This entry is used to obtain various pieces of information that needs to be known about an open forum.

Usage:

```
dcl forum_$forum_limits entry (fixed bin, fixed bin, fixed bin,
    fixed bin, fixed bin, bit (36) aligned, fixed bin (35));
```

```
call forum_$forum_limits (forum_idx, last_seen, first_trans,
    last_trans, new_trans_count, flags, status);
```

where

forum_idx (Input)

As above.

last_seen (Output)

The index of the most recent transaction that has been seen, as set by forum_\$set_last_seen_idx.

first_trans (Output)

The index of the oldest undeleted transaction in the forum, or 0 if the forum has no undeleted transactions.

last_trans (Output)

The index of the most recent undeleted transaction in the forum, or 0 if the forum has no undeleted transactions.

new_trans_count (Output)

The number of undeleted transactions more recent than the highest-seen transaction.

flags (Output)

This collection of switches tells the access on the forum and whether the user is the chairman. See forum_flags.incl.pll for a description of the flags.

status (Output)

A standard system error code or a code in forum_error_table_.

Name: forum_\$convert_attendee_idx

This entry is used to convert an attendee_idx into a user name.

Usage:

```
dcl fo-um_$convert_attendee_idx entry (fixed bin, fixed bin,  
    char (*), fixed bin (35));
```

```
call forum_$convert_attendee_idx (forum_idx, attendee_idx, user_name,  
    status);
```

forum_idx (Input)
As above.

attendee_idx (Input)
The index of a user in the forum, as placed in forum ipc messages.

user_name (Output)
The name of the user with the specified attendee_idx, in the form
Person_id.Project_id.

status (Output)
A standard system error code or a code in forum_error_table_.

Name: forum_\$validate_uid

This entry is used to determine if a given forum uid corresponds to a given forum.

Usage:

```
dcl forum_$validate_uid entry (fixed bin, bit (36) aligned,  
    fixed bin (35));
```

```
call forum_$validate_uid (forum_idx, forum_uid, status);
```

forum_idx (Input)

As above.

forum_uid (Input)

The uid of the forum identified by forum_idx, as placed in forum ipc messages.

status (Output)

A standard system error code or a code in forum_error_table_. The code forum_error_table_\$incorrect_uid will be returned if the supplied uid is not the uid of the forum identified by forum_idx.

forum_mgr_ Internal Interfaces.

The following internal interfaces also exist. They are to be called by other ring 3 procedures only. The caller needn't change the validation level, since all necessary validation level changes for initiating and terminating forum segments and determining access are performed by these entrypoints.

Name: forum_mgr_\$get_ptrs

Most inner ring forum interfaces should use a forum index to identify the forum. The correspondence between these identifiers and useful pointers is mapped by this entrypoint. Pass the index by reference; no need to copy, and get back the information you need to do your work. The forum is also locked by this entrypoint. To unlock the forum, call forum_mgr_\$unlock. You should have a cleanup handler to do this in case of random lossage, and also check all exit points in your program, because it is very important that the forum never be left locked when control is returned to the user ring. You should also never return pointers to the forum to the user ring, or other information like other processes' event channels or process_ids. Please also do not terminate the pointers returned by this interface.

Usage:

```
dcl forum_mgr_$get_ptrs entry (fixed bin, ptr, ptr, fixed bin,
    fixed bin (35));
```

```
call forum_mgr_$get_ptrs (forum_idx, forum_control_ptr,
    proceedings_ptr, attendee_idx, status);
```

where

forum_idx (Input)

As above. This interface is for validation of the forum index and getting the internal per-opening information. NOTE: it locks the forum. The forum MUST be unlocked before returning to the user.

forum_control_ptr (Output)

A pointer to the base of the control segment. See forum_structures.incl.pl1. DO NOT TERMINATE THIS POINTER.

proceedings_ptr (Output)

A pointer to the base of the proceedings segment. See forum_structures.incl.pl1 DO NOT TERMINATE THIS POINTER.

attendee_idx (Output)

The index in the array forum_control.attendee of the current user.

status (Output)

As above.

Name: forum_mgr_\$unlock

The following entrypoint is for cleaning up after forum_mgr_\$get_ptrs. The following sequence or an equivalent should be used in all code that accepts forum indexes as input:

```
forum_control_ptr = null ();
on cleanup call forum_mgr_$unlock (forum_control_ptr);
call forum_mgr_$get_ptrs (P_forum_idx, forum_control_ptr,
    proceedings_ptr, attendee_idx, status);
if status ^= 0 then do;
    P_status = status;
    return;
end;
```

It accepts a null pointer. It traps faults. There is no error code because nothing can go wrong that doesn't kill your process, and that would be due to random damage in the environment, not this module.

Usage:

```
dcl forum_mgr_$unlock entry (ptr);
call forum_mgr_$unlock (forum_control_ptr);
```

where

```
forum_control_ptr (Input)
As above.
```

Name: forum_mgr_\$initiate

There is also an interface for dealing with forums that are not already open. It optionally locks the forum. Otherwise it performs essentially all the checks that forum_\$open provides. However, it can return with a zero code and proceedings pointer = null. In this case, the user has sufficient access to find out the name of the chairman and the names of the participants. Absolutely no information whatsoever should be returned about the transactions. Entrypoints that return no or partial information in this case should return forum_error_table_\$not_eligible after calling forum_mgr_\$terminate. This interface is intended to be called with a user ring validation level. It can be called at the forum ring validation level, but the caller probably shouldn't have its privileges enabled. It is also intended that you pass the character string arguments from the user ring by reference without copying so that truncation can be detected.

Usage:

```
dcl forum_mgr_$initiate entry (char (*), char (*), bit (1) aligned,
    ptr, ptr, fixed bin (35));
```

```
call forum_mgr_$initiate (dirname, entryname, lock_switch,
    forum_control_ptr, proceedings_ptr, status);
```

where

dirname	(Input)
As above.	
entryname	(Input)
As above.	
lock_switch	(Input)
If this switch is "1"b, the forum will have been locked on a successful return. Otherwise, not.	
forum_control_ptr	(Output)
As above.	
proceedings_ptr	(Output)
As above.	
status	(Output)
As above.	

Name: forum_mgr_\$terminate

The following is for cleaning up after forum_mgr_\$initiate. It unlocks the forum if it is locked. It traps faults. It eats null pointers. It hacks its own validation level. It nulls its arguments. There is no error code because nothing can go wrong that doesn't kill your process, and that is an act of God and the initializer, not this module.

Usage:

```
dcl forum_mgr_$terminate entry (ptr, ptr);
```

```
call forum_mgr_$terminate (forum_control_ptr, proceedings_ptr);
```

where

```
forum_control_ptr  (Input/Output)  
  As above
```

```
proceedings_ptr   (Input/Output)  
  As above.
```