Mills, R. G.

D R A F T

To: Distribution
From: T. H. Van Vleck
Date: 2/6/69
Subject: CTSS Resource-management Documentation

The enclosed is a draft of the documentation on the operation of the CTSS resource-management procedures. Your comments will be appreciated.

R. G. Mills (2)
T. H. Van Vleck (2)
J. R. DeCourcey
R. G. Hart
R. Roach
P. R. Bos
D. A. Anderson
M. M. Jones

This document is intended as a guide to the inner workings of the resource-management tools used on CTSS for system programmers and accounting personnel. It assumes that the reader is familiar with CTSS, and that he has some programming knowledge. In particular, complete understanding of the system and effective use of this document will require reference to the program listings in many cases.

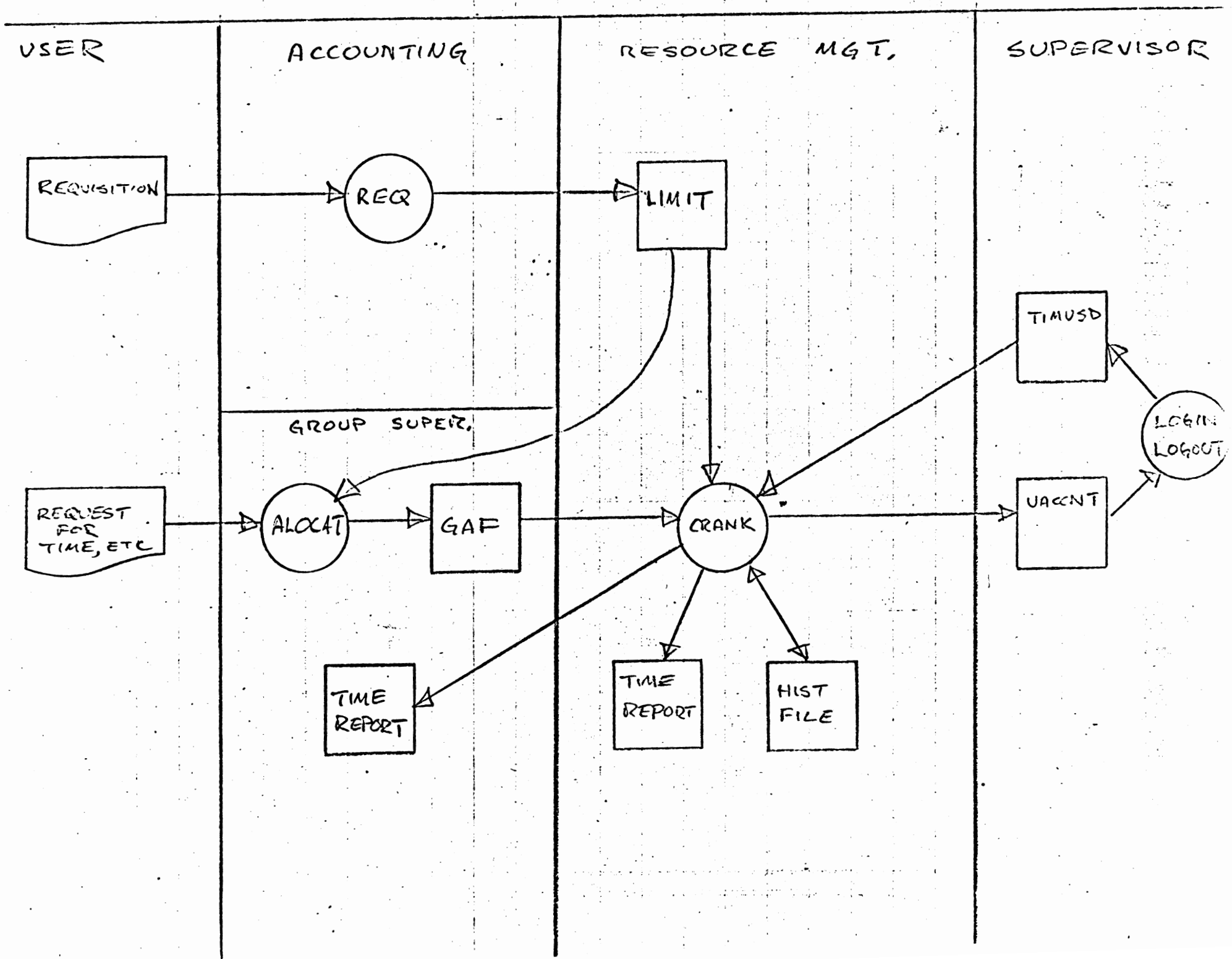
The subsystem of programs used to control CTSS use and to account for this use consists of over 130 programs. These programs fall into four categories:

1. CTSS supervisor - The "hooks" which the accounting system attaches to, the Core-A accounting routines and the LOGIN and LOGOUT commands are the most obvious, but many other system routines contribute also.

2. Central management - The programs which actually modify and record system behavior. The programs of the CRANK runcom, the monitoring programs, and many special tools are included here.

3. Group allocation and suballocation - These programs provide for the delegation of responsibility for rationing resources. ALOCAT and other programs used by the Group Supervisors fall here.

4. Central accounting - Interface to the MIT financial people, the users, and the central management. The billing and requisition-management programs fall under this heading.


USER
ACCOUNTING
RESOURCE MGT.
SUPERVISOR
REQUISITION
REQ
LIMIT
TIMUSD
GROUP SUPER.
LOGIN
LOGOUT
REQUEST FOR TIME, ETC
ALOCAT
GAF
CRANK
UACCNT
TIME REPORT
TIME REPORT
HIST FILE

## DATA BASES

One way to describe the resource-management subsystem is to speak of a set of important data bases, surrounded by programs which create and update them. The data bases are kept as disk files on CTSS. (One could argue that the tables kept in core A are also data bases, modified by the supervisor, LOGIN, and LOGOUT.)

The most important data files are the following:

UACCNT TIMACC - Master resource-allocation file. This file controls when and how a user gains access to the system. Users' time and space allocations, identification, privileges, and passwords are kept here. Two copies of UACCNT are kept; the "real" one, used by LOGIN, resides in M1416 CMFLO2. The "CRANK" copy resides in the main administrative directory, M1416 385. Editing and changing of UACCNT is usually done by changing the copy in 385, checking to make sure that it will be okay, and then using the program UDONLY to update the new UACCNT into M1416 CMFLO2. A third copy of UACCNT is kept in M1416 385 under the name VACCNT TIMACC in case of editing mistakes or disk failures.

M.F.D. (FILE) - The system Master File Directory. All user file directories are files in the MFD. When a new user is added, MODUA creates a directory for him. When a user is deleted, MODUA tries to delete him, but does not succeed unless the directory is empty. If MODUA fails to delete a directory, it stays on the system until deleted by hand. Such a directory is called a "sleeper".

HIST FILE - This file contains entries for every user who has been on the system at any time in the year. For each user, totals are kept itemizing his time and storage use by day, week, month, and year. Totals of charges, itemized by category, are kept for each problem number.

GRPXX GAF - The Group Allocation File for each administrative group. The format is almost the same as that of UACCNT TIMACC. This file represents the wishes of the group allocator concerning distribution of his group's resources.

GRPXX LIMIT - This file contains resource limits, plus a record of the current requisition for each problem number in the group.

Temporary files

| | | Created by | Was |
|---|---|---|---|
| OLD | UACCNT | UDONLY | VACCNT |
| UACCNT | OLD | MODUA | UACCNT |
| OLD | FILE | ATIME | HIST |
| REQH | FILE | HRQ | HIST |
| UACCNT | SORTED | SORTUA | Temp |
| NUACCT | TIMACC | MODUA | Temp |
| UA | TIMACC | KLUDGE | Temp |
| M1416 | TIMACC | UDONLY | Temp |
| NEW | FILE | ATIME | Temp |

## Daily Operation of the CRANK

The only daily operation required for the operation of the CRANK is to check to make sure that everything ran all right. The QED program DAY QED has been written to simplify this. The daily operations reduce to the following steps:

log in

type QED DAY

read the output

if it is all right, type DELETE $$$FIB OUTPUT and log out

otherwise, follow the instructions in the section on errors.

All that DAY QED does is to delete uninteresting lines from the CRANK's $$$FIB output file, and then print what's left.

```
p crank ru

*THIS IS THE IPSC ADMINISTRATIVE UITLITY -- THE 'CRANK'
*
ADMIN PB 20
*
R DATE HEAD
R RECUR CRANK 30 DAY 0001
R RECUR ASCY 30 DAY 0500 M1416 2962 NVLECK
R RECUR ASCY 30 DAY 2200 M1416 2962 NVLECK
*
*CAPTUR GETS 'CURENT GAFS', COMBINES + CHECKS AGAINST LIMS
R CAPTUR (AT) C0116 CMFL01 ALLNO MASH
R SORTUA UACCNT
*
* MODUA MERGES CORRECTIONS
R MODUA YES
*
* DISK USAGE, TIME RECORDS
R SWEEP
COPY 2 TIMUSD TIMACC
COPY 2 DISKED TIMACC
R IFDAY1 RUNCOM RESETT
R ATIME
R HRQ
R NEWTR1
R IFDAY1 RUNCOM MTH.TR
*
* IF USERS OUT OF TIME, CUT OFF
CALL TRFILE DAILY SUMMRY
R CUT
RQUEST PRINT DAILY SUMMRY
*
R MFDCHK
*
R UDONLY
*
R DEL UACCNT GAF
R TUT
*
ADMIN PB 0
*
*END
```

# A TRIP THROUGH THE CRANK

The CRANK runcom is executed once a day, right after midnight, on FIB. Its functions are to:

1. Make any changes in allocations requested by Group Supervisors.
2. Update accounting files and charge users for storage and access.
3. Cut users off the system if they have spent all their funds or allowed their requisitions to expire.
4. Prepare reports describing the time and funds used by users.

ADMIN PB 20

The CRANK runs at 20% in order to keep from locking time-accounting files for a long time.

R DATE HEAD

R RECUR CRANK 20 DAY 0001

R RECUR ASCY 30 DAY 0500 M1416 2962 MVLEK

R RECUR ASCY 30 DAY 2200 M1416 2962 NVLECK

The CRANK prints out the date into the output file, and then reschedules itself to run again the next day. It also sets up the ASCII editor.

R CAPTUR ALLNO MASH

CAPTUR picks up the file CURENT GAFS from C116 CMFL01. This file is a list of all groups which wish changes made by MODUA. The GAF's mentioned in CURENT GAFS are read in, checked for legal entries and for limits, and combined into the file UACCNT GAF.

R SORTUA UACCNT

SORTUA sorts the entries within each group in UACCNT GAF.

R MODUA YES

MODUA merges UACCNT GAF into UACCNT TIMACC and writes a new copy of UACCNT. Groups with no change are simply copied. If a group is found in UACCNT GAF, it is compared with the entries in UACCNT TIMACC. MODUA will create the file directories for users to be added, and attempt to delete the file directory of a user being deleted. It also makes all requested changes in time allotments.

If CURENT GAFS is empty, there are no changes necessary. CAPTUR will say "No modifications" to indicate this. SORTUA will not find UACCNT GAF and will print an error message which can be ignored. MODUA will do nothing to UACCNT if it does not find UACCNT GAF.

R SWEEP

SWEEP reads UACCNT TIMACC and goes over the whole system, looking at the record quotas and usage for each user. It writes a file called DISK USAGE which contains all the usage information for later use. SWEEP also sets record quotas for all users except those in Group 5.

COPY 2 DISKED TIMACC
COPY 2 TIMUSD TIMACC

These commands pick up the latest versions of the system time-accounting files which contain the time used by each user from the system file, M1416 CMFLO2.

R 1FDAY 1 RUNCOM RESETT

If it is the first of the month, RESETT will zero all used-time information. We have just made a copy which we will use to bill the last day of the month.

R ATIME

ATIME updates HIST FILE. It reads HIST FILE, DISK USAGE, TIMUSD TIMACC, and DISKED TIMACC, and it writes a new copy of HIST FILE. Additions and deletions are noted, and users have their usage of CTSS recorded and charged. The total amount spent by each problem number is updated in the account trailer at the end of each problem-number group in HIST.

R HRQ

HRQ reads the file TODAY RQST to pick up miscellaneous changes requested by use of REQ ADJ, REQ DEBIT, etc. It reads and rewrites HIST FILE if any changes have been requested. The contents of TODAY RQST are kept in a file HIST RQST for use at the end of the month. If TODAY RQST is empty, HRQ prints "no requests" and does nothing.

R NEWTR1

A time report is created for each group. These reports are written through links in M1416 385.

R IFDAY1 RUNCOM MTHTR

If it is the first at the month, a monthly time report is written.

CALL TRFILE DAILY SUMMRY
R CUT

CUT reads the HIST and LIMIT files, and makes a new copy of UACCNT TIMACC which has special flags set if a user is out of funds or past his termination date. CUT writes DAILY SUMMRY, a file which summarizes the CTSS charges this month.

R KLUDGE

KLUDGE rewrites UACCNT and sets up special entries for all users of the ESL display. It gets this information from the file KLUDGE FILE, maintained by the ESL administration.

R MFDCHK

This program makes sure that all file directories which should exist do exist and that no others do. It prints a line for each discrepancy.

R UDONLY

UDONLY updates the new copy of UACCNT into M1416 CMFLO2, and makes a backup copy called VACCNT TIMACC.

R DEL UACCNT GAF
REQUEST PRINT DAILY SUMMRY

R TUT

ADMIN PB O

Cleanup

Utility Operations on Master Files

Several situations may require hand modification of the administrative files. A system crash or program malfunction may mess up the data files, or a user may get into an emergency situation where he needs more time right away, or the files may just need cleaning up. There is no easy way to describe all these situations and the action required for each, so a description of the most useful tools will be given, followed by several examples showing their use. See the write-ups of individual programs for more information.

MFDCHK checks for discrepancies between the MFD and UACCNT. "Sleeper" directories are noted, and so are missing directories. MFDCHK should be run after you have edited UACCNT and before putting it into effect, to make sure you haven't clobbered UACCNT or some user entry in it. It is also a good idea to run an MFDCHK after any system crash in which directories may have been lost. Optional arguments to MFDCHK, especially "ASK1," enable you to clean out "sleepers".

EDUA is used to edit UACCNT TIMACC.

Example: Suppose an emergency comes up, and some user needs 20 minutes move on shift 1 right away -- he can't wait until the CRANK runs. You type

```
R EDUA UACCNT
  C PROB PROG T1 +20 *
  FILE
R UDONLY
```

All users in GRPO5, operations, are given time only by this method.

There is no GRPO5 GAF.

HED is used to edit HIST FILE.

Example: Suppose you wish to set the "charges this requisition" field to zero for a user. Type

```
R HED
  F PROB ACCT.. C SP O * P FILE
```

Example: How to take a particular problem number and make it a <u>new</u> group. Say we wish to take problem number T100 from GRPO1 and make it GRPO2.

1. RIPP UACCNT T100

```
RENAME T0100 TGAF GRPO2 GAF
TAUT MAKE C116 9002 25
CALL MOVFIL GRPO2 GAF C116 9002
CALL LINK GRPO2 GAF C116 9002
TAUT UNLOCK GRPO2
CALL LINK GRPO2 TIMRPT C116 9002
CALL ATTACH C116 9002
CALL LINK GROUP LIMIT M1416 385 ** 104
COMFIL
MOVE PROTO LIMIT GRPO2
REQ L 2 GN 2 ALPG 9002 NAME *
REQ NEW 2 T100
REQ D 1 T100
```

2. Now for the one tricky part. We must insert the group separator for GRPO2 in UACCNT after the last user in GRPO1. Say his name is SMITH.

```
ED UACCNT TIMACC
    F SMITH
    N
    I
    GRPO2
    (space)

    FILE
```

3. Now we run MODUA twice, to delete T100 from GRPO1 and then to add it to GRPO2.

```
CAPTUR ALLNO (GRP) GRPO1
SORTUA UACCNT
MODUA NO
DEL UACCNT GAF
CAPTUR (GRP) GRPO2
MODUA NO
DEL UACCNT GAF
```

(notice that the second time, SORTUA is not needed, since the GAF is already in sort.)

4. Now we move T100's history to GRPO2.

```
HED
  F T100 * SA T100 P FILE
```

(This creates T100 FILE. Now suppose T123 is the last problem number in GRPO1.)

```
ED T100 FILE
```

C / GRPO1 /GRPO2 / 999

FILE

HED

F T123 ACCT.. A F T100 P FILE

5. Now we create the group allocator number for GRPO2.

EDUA UACCNT

F C116 GRPO1

A C116 GRPO2 9002 25 20 1055

FILE

6. This should do it, except for:

a) deleting the requisition for T100 from GRPO1 LIMIT (use ED).

b) Fixing up the storage limits for GRPO1 and GRPO2 by using REQL. Presumably a GAFSUM of GRPO2 GAF should be done to find how much to move, but there may be policy agreements which intervene.

c) Fixing up the GRPO2 LIMIT header with REQ L to set the group allocator name and phone, group supervisor name and phone, and so forth.

d) a UDONLY

e) Linking up GRPO2 LIMIT in C116 9000

This description includes, as special cases, creating a new group and moving a bunch of users from one group to another.

Recovery Procedures

If the CRANK's output shows any error messages or stops in the middle because CTSS crashed, manual intervention may be necessary to make it run right again.

PRFIB should show the CRANK scheduled for 00:01 tomorrow. If it does not, use RECUR to set it up by typing

R RECUR CRANK 30 DAY 0001

Next, do a LISTF (LONG) (FILE) to see what files are present. The creation dates of temporary files produced by the CRANK will help determine when the CRANK died and how far it got. Do not LOGOUT until you rename all temporary files and change them to O mode.

First, consider the easy case where the CRANK bombed the same day you discovered it. Try to determine where in the run can it died by examining the output and the dates on the following files:

| | | |
|---|---|---|
| UACCNT | TIMACC | CUT, MODUA |
| UACCNT | OLD | MODUA |
| OLD | UACCNT | CUT |
| HIST | FILE | ATIME, HRQ |
| OLD | FILE | ATIME |
| REQH | FILE | HRQ |
| DISK | USAGE | SWEEP |
| GRPXX | TIMRPT | TR1 |
| UACCNT | GAF | CAPTUR, SORTUA |

Check for error messages in the output, and if any are present, determine why. If there is a problem with the files, correct it.

If MODUA did not run, run it. This may require "forcing" notifications for several groups, and running CAPTUR and SORTUA.

If ATIME did not run, type "R ATIME". Note that a user who has been on all month should have the day of the month minus 1 in the NDAYS field of his HIST entry if ATIME ran. (You can check this with HED). ATIME should not be run twice in one day, since it will then overcharge storage and access. Typing ATIME TIME

will cause ATIME to do only times and add/delete processing.

If CUT did not run, type "R CUT". If it did run but there was trouble, you may wish to type "R CUT NOMAIL" to avoid sending mail to all users. This is useful if you wish to make a new requisition take effect immediately.

If SWEEP did not run, it should be run after MODUA and before ATIME. SWEEP sets user disk quotas from UACCNT, so it may be useful to run it after a forced MODUA.

There is no harm in running the following programs several times in one day:

SWEEP
CUT NOMAIL
CAPTUR
MODUA
TR1
ATIME TIME

If the CRANK has not turned for several days, it may be necessary to cause ATIME to "catch up" on storage and access charges. To catch up n days, type

R ATIME N

## Error Messages

MODUA

1. Failed to delete PROB PROG

   User being deleted by Group Supervisor had files in his directory. The directory is not deleted. This is an expected message.

2. Failed to attach to PROB PROG-DELMF

   System error. Check to see if PROB PROG is in MFD by typing

   LISTF (ATT) PROB PROG

   If it is there, okay. Otherwise, use TAUT to create the directory.

3. Failed to return - DELMF

   System error. M1416 385 was lost.

4. GRPXX GAF too big for UG
   Max size is 200.

   Over 200 users in GRPXX. Remove users or reassemble MOD FAP and reload MODUA.

5. GRPXX in UACCNT too big for UT
   Max size is 200.

   Same as 4.

6. UACCNT ends before GRPXX GAF

   The group header entry in UACCNT for GRPXX is missing. Edit it in and rerun MODUA.

7. GAF out of sort. US = PROB1, UT = PROB2.

   Usually because SORTUA was omitted before MODUA. Try again.

8. Fatal MODUA error.

   A file-system error message follows.

   Try to correct it and rerun.

ATIME

1. Fatal ATIME error, TIMUSD. G. XXXX

   Too many entries in TIMUSD TIMACC, perhaps because of garbage in the file.

   If the file is okay, reassemble TIMAIN FAP, reload ATIME.

2. DISKED too big.

   Same as 1 with DISKED TIMACC.

3. HPROB no users. At UPROB UPROG UGRP

   Sorting error in HIST or UACCNT, or program bug near EOF.

4. HPROB no trailer. At UPROB UPROG UGRP

   Same as 3.

5. Extra time PROB PROG GRP T1 DE

   This usually arises due to bad entries in DISKED TIMACC. These lines are printed for usage which cannot be charged to any user. If PROB PROG exists, there is trouble in disk editor or LOGOUT.

6. No usage for PROB PROG GRP - XPROB XPG XG

   Sort error between DISK USAGE and HIST. Could be because of two entries in UACCNT for PROB PROG, or because DISK USAGE was not created properly.

7. Extra usage for PROB PROG GRP - XPROB XPG XG

   Like 6.

8. Fatal ATIME error.

   A file-system error message tells what happened.

SWEEP

1. NO DIRECTORY PROB PROG GRP

   File directory missing. Create it with TAUT, or edit UACCNT to remove the entry.

CAPTUR

1. GRPXX code XXXX LIM = XXXX

   Notification ignored

   A group has allocated more than its quota of resources, and so will not be allowed to make changes. The second line will not appear and changes will go through if "MASH" is not specified on the CAPTUR command line.

2. GRPXX gaf missing.

The GAF cannot be found, and the notification will be ignored. Check to see that there is a link in M1416 385 to the GAF. It may be necessary to recreate the GAF for the group allocator from UACCNT by using RIP. Call the allocator.

3. GRPXX bad CODE word n

The entry will be typed out after this message. CAPTUR attempts to fix this error but may have to drop the entry. See document on CAPTUR for more information. If the GAF is garbaged badly, contact the allocator.

4. GRPXX GAF ED'd.

The group leader has used ED on his GAF. This is dangerous. Have a talk with him.

CUT

1. GRPXX LMIT too big

The LIMIT file is over 10 records. Delete old entries or recompile CUT.

2. PROB missing in GRPXX

No limit-file entry for PROB was found in GRPXX LIMIT. Use REQ to see if it is there. If so, rerun CUT and the problem should disappear.

3. SHIFT entry

The LIMIT-file entry has been shifted over because of date termination on one requisition. Probably you will need to use REQ ADJ to adjust the spent-field of the problem number by the amount spent on the old requisition.

4. Date gap PROB GRP D1 D2

One requisition has ended but the next has not begun. The problem number has been cut off the system. Fix with REQ and rerun CUT if necessary.

5. No mail to PROB PRG

Usually record-quota overflow. Maybe you should call the user.

6. UA eof before HIST eof
H = PB PG GP U = GP PB PG

sort error, usually. A HIST-file entry for an active user has no UACCNT entry. Repair files and rerun.

7. GRPXX PROB not in UACCNT

   Cannot find UACCNT entry. See 6.

8. Fatal CUT error

   Correct the file-system error which follows, and rerun.

DSCREP

1. *ERR PB PG MYT = X1 SIGM = X2 SP = X3 SY = X4

   Internal discrepancy in HIST. X1 should equal X2. X1 is computed by adding individual user charges. X2 is the total in the trailer. Edit HIST to repair this problem, using HED.

2. UC = X1 FE = X2 PROB PROG GRP

   Internal discrepancy in HIST

   X1 is the total computed by DSCREP, while X2 is the HIST figure. Edit HIST with HED to repair.

```
p $$$fib output
```

Example of CRANK's output showing a real crash

```
 M1416   385 LOGGED IN  01/22/69  008.4 FROM (FIB)
 CRANK STARTED
 M1416   385  ON  (FIB)  ST PB       M1416   385  TO      20

 M1416   385  MILLS TST7B8 (FIB)  0008.9 Wednesday, January 22, 1969

job  CRANK for  M1416   385  MILLS scheduled at 01/23 0001.0    30
job   ASCY for  M1416  2962 NVLECK scheduled at 01/23 0500.0    30
job   ASCY for  M1416  2962 NVLECK scheduled at 01/23 2200.0    30
Notifications:
 GRP01  01/21 2224.6
 GRP06  01/21 1841.9
 GRP08  01/21 2223.5

 GRP01 bad party  word 25
ENBAUM  2531     4      S  2531     0 000001     0          T0109
     0   700     0     20    20     20     20     20
BEGIN SORT.
begin MODUA
GRP01
GRP06
Delete OHAYON  6147      1      S  6147     0 000101     0
     0    25     0      5    10     5      5      0      H
Failed to delete T0424  6147
GRP08
end MODUA
NO DIRECTORY M5806CMFL04GRP08
DELETED THE OLDER TIMUSD TIMACC
DELETED THE OLDER DISKED TIMACC
 C0116  9016 QTA       75 S/B       25
 C0116  9017 QTA       50 S/B       25
 M1416  9324 QTA     1000 S/B     9000
 M1416CMFL04 QTA        0 S/B        1
 M1416CMFL06 QTA     3000 S/B    30000
DSKEDT.FILE. QTA     1000 S/B      100
Extra time  M6177   DM1000000      0      4
Extra time  T0288  1713000000      0      9
Extra time  MFL03TR2TV2000000      0   1632
Extra time  MFL03TR1TV1000000      0      9
Extra time  T0298  3794000000      0      5
Extra time  MFL03 M5779000000      0     28
Extra time  C0032CMLF01000000      0     19
Extra time  M5777CMF102000000      0     20
Extra time  MFL03  6058000000      0      3
Extra time  XXXXX  5422000000      0     21
Extra time  LOTLB   FAP000000      0      4
Extra time  ACROS   FAP000000      0      3

Total $  5555.93 today, $102031.98 this month.
console $  5073.62
no requests.
no mail to M5826  1648  1648
no mail to M6119  4107  4107
no mail to M6119CMFL01  9201
no mail to M6119CMFL01  9202
no mail to M6119CMFL01  9203
no mail to M6119CMFL01  9204
no mail to M6119CMFL01  9205
no mail to M6119CMFL01  9206
no mail to M6119CMFL01  9207
```

4289

```
  cut      13,       9 probs, warned          47
begin MFD check.

DISK ERROR IN DUMP. YOUR CORE STATUS IS RESET.
AUTOMATIC LOGOUT
 MEMORY BOUND ZERO, NO SAVED FILE CREATED.
 M1416   385 LOGGED OUT 01/22/69  023.0 FROM (FIB)
 TOTAL TIME USED =   8.4 MIN.


$
$ the CRANK has bombed out due to hardware trouble.
$
$ we were in the middle of the MFDCHK when it died, so
$ we can just type the remaining commands.
$

mfdchk , udonly , del uaccnt gaf
begin MFD check.

 M5806CMFL04CMFL04  1500

 T0161  3592   250   251
 T0168CMFL01     0    60
 T0196  2863   250   125
 T0266  3516    50    48
 T0289  4810   500   118
 T0368  3352    50    32
 T0375  4952   150   144
 T0416  2928   300   239
 T0424  6147    25     5
 N5178  1961   200   116
 M5962  5929   176   149
 M6004  2814   300   176
 M6321  3166   300    81
 M6398  2759   300   333
 M6631  6484   100    45

MFD count 442 zero 3 k
Match alot 204358 sleep alot 2951 use  1922

UDONLY DONE.

$
$ notice that the directory for m5806 cmf104 seems to be missing.
$ let's check it.
$

listf (att) m5806 cmf104

USER NOT IN M.F.D.  M5806 CMFL04--ATTACH

$
$ it's not there. let's look in HIST FILE to see if it was there yesterda
$

hed
Type: f m5806 cmf104 pf
 M5806CMFL04GRP08 CMFL04                          012069          1
     0      0      0      0      0      0      0      0      0      0
     0      0      0      0      0      0      0      3      0      0
     0      0      0      0      0      0      0      0      0      0
Type: q
```

sleepers

42893

```
$
$ it should have been on, but it looks like it was never created.
$ (NDAYS is 3, but it has never been charged for any UFD).
$ so let's use TAUT to create the directory.
$

taut make m5806 cmf104 1500

$
$ now if we ran MFDCHK again it would not complain about M5806 CMFL04.
$
$ let's clean up some of the "sleepers". the only one it is safe to
$ delete today is the t424 number
$

delall t424 6147 , call delmfd t424 6147 , e#yes
*

$
$ all done for today
$

delete $$f#$fib output , tut

Drum:  98.2% used, 7 left.
Disk:  84.6% used, 24620 left.
Cylinders:  74.1% used, 517 left.
```

# ENTRY IN UACCNT TIMACC OR GAF

| NAME | PROG | PARTY | STBY | LINMUL | UNIT | RCODE | GROUP | PASS | (PROB) | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DRUM | DISK | TAPE | T1 | T2 | T3 | T4 | T5 | REQNO | DEPT | DATE | DATEF | | |

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| word | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 |
| number | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

GROUP and UNIT have blank right
RCODE has leading zeroes
NAME is left-justified

PARTY is party line group no.
STBY allow standby if non-zero
UNIT is unit group
RCODE has leading zeroes
GROUP is allocation group

PROB does not appear in UACCNT TIMACC

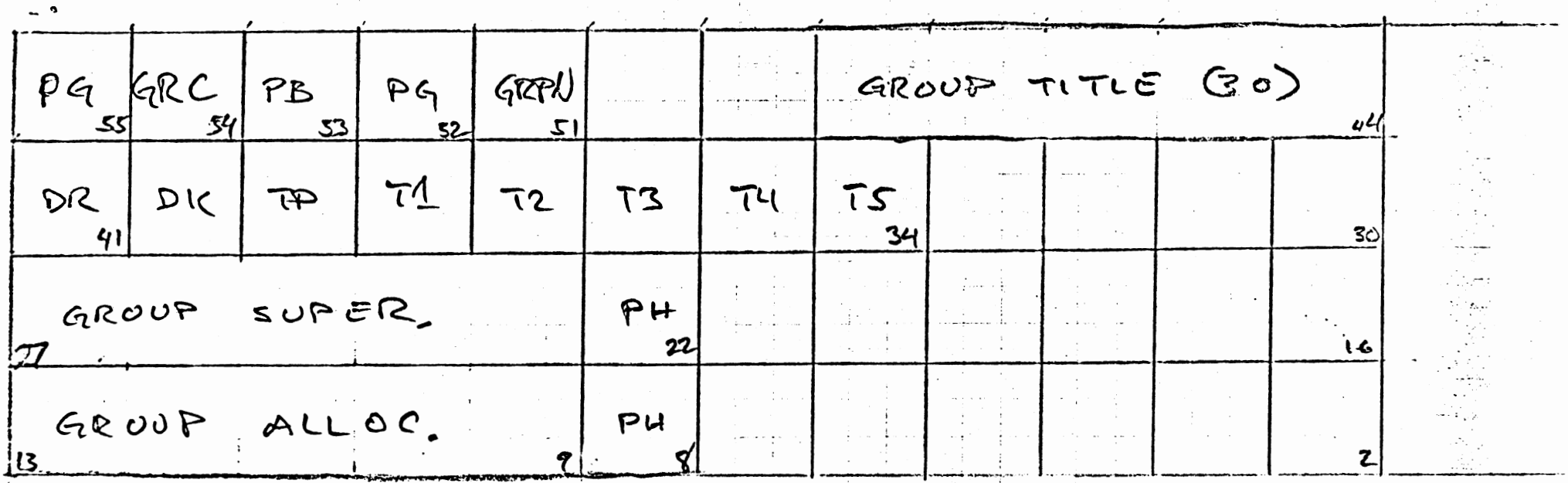
# HIST FILE — USER ENTRY

| 55 PROG BCD | PROG BCD | GRP BCD | FULL NAME (30 CHRS) BCD | | | 35 | | DATE ON BCD | DATE OFF BCD | CODE BCD | 44 MDED ISECS BCD |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 41 Y1 I-SECS | Y2 | Y3 | Y4 | Y5 | Y6 IMINS | M1 ISECS | M2 | M3 | M4 | M5 | 30 M6 IMINS |
| 27 W1 I | W2 | W3 | W4 | W5 | W6 IMINS | 21 TREC I | ND I | REC TODAY I | TNFL I | MSUM F$ | 16 YSUM F$ |
| 13 D1 I | D2 | D3 | D4 | D5 | D6 IMINS | 7 TA1 IMINS | TA2 I | TA3 | TA4 | TA5 IMINS | 2 DSKQ I |

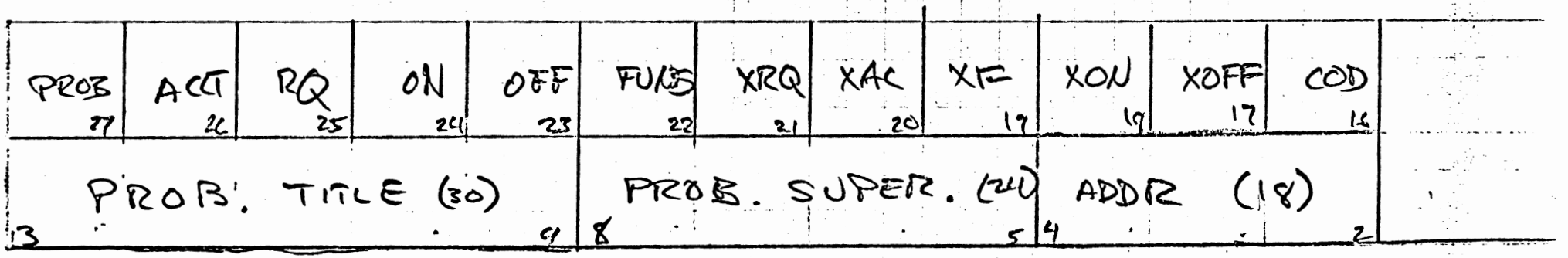
*CODE = 0 - normal user ; = 1 - never logged in ; = 2 - dropped from UACCNT

# ACCOUNT TRAILER

| 55 PROG BCD | 54 ACCT# BCD | 53 GRP BCD | 52 BCD | 51 F $ | 50 SP F $ | 49 BCD | 48 BCD | 47 BCD | 46 BCD | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 41 Y1 F$ | Y2 | Y3 | Y4 | Y5 | Y6 | YDK | YUF | 33 YAC | YS | YDED | YMSC |
| 27 M1 F$ | M2 | M3 | M4 | M5 | M6 | MDK | MUF | 19 MAC | MS | MDED | MMSC |
| 13 D1 F$ | D2 | D3 | D4 | D5 | D6 | DDK | DUF | 5 DAC | DS | DDED | DMSC |




HEADER




PROB. ENTRY

# GROUP LIMIT

## TIMUSD TIMACC

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PB | PG | NAM | | | | | | | | | | |
| LLD | LLT | UNIT | | T1 | T2 | T3 | T4 | T5 | CT | | | |

## DISKED TIMACC

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PB | PG | TT | INP | PRT | TP | DPU | BPU | PLT | CY | | | |

seconds. TT >, ≥

## DISK USAGE

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PB | PG | GRP | NAM | ALOT I | USED I | UR I | NF I | NL I | NZ I | | NOUFD (≠0) | |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PB | PG | GRP | FUNC | DOLS | CTS | | | | | | | |

TODAY RQST

## Identification

RECUR SAVED

## Purpose

A modified FIB command with no argument checking, added features, and no setting of FIBBIT, for use with administrative pre-scheduled FIB jobs. Used in the CRANK.

## Usage

R RECUR JOB TLIM-DAY- -TIME- -PROB PRG NM-

where "JOB", "TLIM", "DAY" are like those for the FIB command (AH. 1.03).

The job is scheduled for "PROB PRG NM" if given, otherwise the command user.

DAY may have three additional values:

'DAY' the job will be deferred for 1 day

'WEEK' the job will be deferred for 1 week

'MONTH' the job will be deferred until the start of next month.

## Identification

CAPTUR

## Purpose

CAPTUR combines all GAF's mentioned in CURENT GAFS in C116 CMFLO1 into a file called UACCNT GAF. Each GAF is checked against the group's LIMIT file and bad entries are corrected.

## Usage

R CAPTUR -(AT) prob prog- -global- -MASH- -(GRP) grpn-

The file CURENT GAFS in prob prog is examined. If "(AT)" is emitted, prob prog is T116 CMFLO1.

CAPTUR prints a list of all groups mentioned in CURENT GAFS, and then reads each GAF in turn into core and checks it. Each entry is checked to see that it is legal, and corrective action is taken for any bad entries or bad items in an entry. Figure 1 shows the possible errors and the action taken on each. For each error, CAPTUR prints

GRPXX bad code word x

followed by the entry, and then asks what to do unless global is supplied. Global may be "ALLYES" OR "ALLNO".

CAPTUR also totals each group and checks the sums against the LIMIT for each group. If "MASH" is specified, groups which are over-allocated are not written into UACCNT GAF.

If the GAF is okay, it is written into UACCNT GAF, preceded by a 28-word group separator with the group name in word 27.

If a GAF is missing, CAPTUR will skip it. If CURENT GAFS is missing or empty, CAPTUR prints "No modifications" and calls CHNCOM.

Use of the "(GRP)" argument allows forcing the notification of one group. CURENT GAFS will not be read in this case.

| WORD | ITEM | DISCREPANCY | ACTION TAKEN: ALWAYS | YES | NO |
|---|---|---|---|---|---|
| 27 | NAME | BEGINS WITH BLANK | DROP ENTRY | | |
| 26 | PROGNO | NOT CMFLXX AND >9999 | | LEAVE | DROP ENTRY |
| 25 | PARTY | NOT "0", "1", OR IN LIMIT | | LEAVE | SET TO "0" |
| 24 | STNBY | NOT "S" | | LEAVE | SET TO "S" |
| 23 | UFD | " . " | NO ACTION | | |
| 22 | UNIT | NOT "0" | | LEAVE | SET TO "0" |
| 21 | RCODE | UNAUTHORIZED BIT | | LEAVE | SET TO BEST |
| 20 | GROUP | NOT "0" | SET TO "0" | — | |
| | PASS | — | NO ACTION | | |
| 18 | PROBNO | NOT ASSIGNED TO THIS GROUP | | LEAVE | DROP ENTRY |
| 17 | blank | NOT BLANK | | LEAVE | SET TO BLANK |
| 16 | blank | " | | LEAVE | SET TO BLANK |
| 15 | blank | " | COMMENT ONCE | | |
| 14 | blank | " | " | | |
| 13 | DRUM | DOESN'T BEGIN WITH BLANK | DROP ENTRY | | |

CAPTUR

Identification

MODUA

merge corrections into time-accounting file.

Usage

R MODUA - GLOBAL -

UACCNT GAF will be merged into UACCNT TIMACC and a new copy of UACCNT TIMACC created. If GLOBAL is supplied, it may be either "YES" or "NO" - It is taken as a GLOBAL answer for all add and drop questions asked by MODUA.

Groups with no change are simply copied.

The name of a group to be changed is printed, and merging begins. MODUA always adds or deletes entries in UACCNT TIMACC; the questions asked refer to additions or deletions in the MFD.

## Identification

SWEEP

check disk usage

## Purpose

SWEEP reads UACCNT TIMACC and runs through the entire directory hierarchy. For each directory, it sets record quota if necessary, and reads the U.F.D. to determine number of records used, number of files, number of links, and number of zero entries. This information is written into a file called DISK USAGE.

## Usage

R SWEEP

Identification

ATIME

Purpose

ATIME updates the master accounting file HIST FILE by combining information from the old HIST FILE, UACCNT TIMACC, DISKED TIMACC, and DISK USAGE. It also uses the prices from SYSTEM PRICES to compute the charges for each user and the total charges for each account.

Usage

R ATIME-TIME- - MONTH - -WEEKLY- -CLEAR- -n-

All arguments are optional. If TIME, MONTH, WEEKLY, or CLEAR is specified last, DISK USAGE will not be read and storage and access charges will not be made. The CLEAR argument zeros the yearly totals, the WEEKLY argument zeros the week totals, and the MONTH argument zeros the monthly totals. If n is specified, it causes ATIME to charge for n days of storage and access (this is useful when catching up after a crash).

ATIME first reads in SYSTEM PRICES and DISKED and TIMUSD TIMACC into core. If the TIMACC files are too large, an error message is printed and EXIT is called. The program then passes through HIST FILE and UACCNT TIMACC, writing a new HIST FILE. If a user appears in UACCNT but not in HIST, a new entry in HIST is made for him and if necessary a new account trailer is created. If a user appears in HIST but not in UACCNT, the HIST entry is marked with a code showing that the user has been deleted. For each user entry, the core copy of the usage files is searched, and if any usage is found, the net change since ATIME ran last for each shift is computed by the formula

delta = max (O, usage file - month total from HIST)

This delta is added to the monthly, weekly, and yearly totals. (Note that losing TIMUSD in the middle of the month will still charge as much usage as possible.)

If storage and access are being charged, the file DISK USAGE is read to find an entry for the user, and his total disk usage, number of days on the system, and total UFD usage are increased.

Each user's usage is "priced out" and his yearly and monthly dollar totals are increased by his charge for the day. Problem-number totals are also accumulated so that when the account trailer is reached, the dollar totals can be increased.

Console charge is handled in a special fashion so that it may decrease if the user uses more CPU. The console charge is not added into the dollar totals for each user except at the end of the month, although it is added into the month total in the account trailer.

## Identification

HRQ

process accounting requests.

## Usage

R HRQ

HRQ reads a file called TODAY RQST and sorts it. It then makes a pass through HIST FILE, making requested changes. Changes are of the following form:

ADJ sign DOLS CTS

adjust "spent" field in HIST trailer

RESET

set "spent" field to MSUM

DEBIT DOLS CTS

CREDIT DOLS CTS

charge or credit user. Modify total month and year sums, and accumulate in "miscellaneous charges" field.

## Identification

TR1

## Purpose

Reads the history file and produces one time report file for each administrative group. Used in the CRANK.

## Usage

R TR1 or

R TR1 GRPXX

where GRPXX specifies a single group report to be produced.

## Operation

The program reads "HIST FILE" and creates a report for each group. The file "GRPXX LIMIT" is read for each group to get group names. The "GRPXX TIMRPT" file written contains:

1. Heading

   for each user:
2. User PROBNO, PROGNO, full name
3. For each shift, allotted, used, percent, delta
4. Console time delta
5. Disk allottment, use, percent

Last:

6. Totals for entire group, as in 2-5 above.

If an error occurs, the program attempts to move on through "HIST FILE" to the next group.

"(IOH)" is not loaded. All I-O is done with the BF package.

Users who have been dropped from UACCNT TIMACC are skipped.

## Identification

CUT

## Purpose

CUT checks for requisitions out of funds or past their cutoff. It rewrites UACCNT TIMACC, setting flags in word 20 of each user record if the user requisition is out of funds or past its termination date, and sends URGENT MAIL to all users who are receiving cutoff.

## Usage

R CUT -NOMAIL- -NWRITE-

All arguments are optional and may occur in any order. If NOMAL is specified, no URGENT MAIL will be sent.

CUT reads UACCNT TIMACC, the GRPXX LIMIT files, and HIST FILE. It writes a new copy of UACCNT, a summary listing called DAILY SUMMRY, and may rewrite a GRPXX LIMIT file (unless NWRITE is specified) if the primary requisition has reached its cutoff date and if there is a secondary requisition which is valid.

Identification

KLUDGE

Purpose

Manages those aspects of permission to use the ESL scope which must be kept in the master time-accounting file.

Usage

R KLUDGE -'(PRNT)'-

Operation

The file "KLUDGE FILE" is read, and a list of the problem and programmer numbers of authorized display users is made. "UACCNT TIMACC" is then read and a new copy created. For each user authorized to use the display, the following is done:

1. The permission bit ($040_8$) is set on in his restriction code.
2. A duplicate entry is created, after his normal entry, with all data the same except
   a) Party group is 20
   b) Unit group is 2
   c) Name is a "Star Name", that is, the last letter of the name is replaced by a star. A short name is filled out with stars to make six characters.

Users with any of these attributes, who do not occur in the authorization file, have the authorization removed.

"KLUDGE FILE" must be a card-image file, with the first two items on each line the problem and programmer numbers of a user. These items must be separated by blanks.

The program comments on leftover authorizations if "(PRNT)" is specified, and gives statistics telling how many entries were read and written at end-of-job.

Identification

MFDCHK

find differences between UACCNT TIMACC and MFD

Usage

R MFDCHK -Nl- -HUSH- -Kilpar-

The program reads the MFD and compares it to UACCNT TIMACC. Unless "HUSH" is specified, lines are printed describing the differences as follows:

for each user in Nl TIMACC but not in MFD

PROB PROG NAME DISKQ

for each ufd-less user in Nl TIMACC and MFD

PROB PROG NAME DISKQ UFDNM

a space is then printed

for each directory in MFD but not in UACCNT

PROB PROG DISKQ DISKU

Summary lines are printed at the end, giving the number of MFD entries, MFD zero entries, Kludge users, the total allocation for "matches", and the total allocation and use for "sleepers".

If Nl is not specified, UACCNT TIMACC will be read.

Kilpar allows automatic cleanup of the MFD. It may have one of four values:

KILL attempts DELMFD

ASK ask whether to attempt DELMFD

MASH DELALL, DELMFD

ASKl ask whether to DELALL, DELMFD

The "KILL" and "ASK" options will succeed only if the directory is empty.

## Identification

UDONLY SAVED

## Usage

R UDONLY

This program is equivalent to the following RUNCOM:

```
Call CHFILE  UACCNT  TIMACC 125 OLD  UACCNT

  COMBIN * VACCNT  TIMACC  UACCNT

  UPDATE 2 UACCNT  TIMACC
```

However, it is much faster and doesn't ask questions.

Identification

DSCREP

discrepancy check on HIST FILE

Usage

R DSCREP -HNM-

The file HNM FILE is checked for discrepancies. (If HNM is omitted, it is set to HIST.) SYSTEM PRICES is read to obtain the current prices, and then HIST FILE is scanned. Each user entry is priced out, and if the total differs from MSUM (word 17), a comment of the form

UC = XX FE = YY PROB PROG GRP

is typed, where XX is DSCREP's total, and YY is the HIST value. The computed totals are also accumulated, and if they differ from the MSUM field in the trailer, a comment of the form

*ERR PB PG MYT = X1 SIGM = X2 SP = X3, SY = X4

is typed, where X1 is DSCREP's total, and X2 is MSUM (word 18) of the trailer. X3 is the "charges-this-rec" figure from HIST and X4 is the year sum.

Identification

TR2 SAVED
TR3 SAVED

Purpose

TR2 and TR3 produce weekly and monthly time reports, respectively.

Usage

R TR2 - DATE d- -READ hf- -WRITE of- or

R TR3 -DATE d- -READ hf- -WRITE of-

All arguments are optional and may appear in any order. The programs read "hf" FILE" and write "of TIMRPT". If "READ" is omitted, hf will be "HIST". If "WRITE" is omitted, TR2 will write "WEEKLY TIMRPT" and TR3 will write "MNTHLY TIMRPT".

The LIMIT files for each group are also read. Writing is done with DWRITE.

## Identification

MONEY

write monthly bills

## Usage

R MONEY -READ- -NOPROB- -date-

All arguments are optional, in any order. If "NOPROB" is specified, PROB MONEY and PROB TIME will not be produced. If "READ" is specified, MONEY will call READ DATA to accept modifications to internal parameters, such as prices and the date. An unrecognized argument is taken as the date.

MONEY reads SYSTEM PRICES to get the prices before looking at its arguments. After picking up its arguments, it examines DATE to see if it is the first of the month, and if so backs the date up one day and prepares to write PROB TIME and PROB MONEY. (These reports are not produced unless it is the first of the month.)

MONEY then opens all output files and runs through HIST FILE, writing a report into GROUP TIME, GROUP MONEY, and SUMMRY MONEY (and the PROB Files if required). It reads GRPXX LIMIT files to get allocations, group titles, supervisor names, and problem-number information. User entries are priced out and the totals checked against the totals in the account trailers. A comment is typed and placed in GROUP MONEY. Miscellaneous charges are read from ADJ FILE.

If a limit-file entry is extra or missing, a comment is typed.

When all of HIST FILE has been processed, a summary page is written on GROUP MONEY and GROUP TIME, and any leftover entries in ADJ FILE are typed out. MONEY then closes all output files and exits via CHNCOM.

Identification

ADJFIX

create ADJ FILE

Usage

ADJFIX is run once a month, just before MONEY, to create ADJ FILE from HIST RQST. ADJFIX reads HIST RQST and discards all but DEBIT and CREDIT items. It keeps all these in core and sorts them into proper order for merging with HIST.

R ADJFIX

If there are too many debits and credits, ADJFIX will print an error message and call EXIT.

## Identification

ACDS

prepare accounting cards

## Usage

R ACDS HIST INSTAL SEQ VOU DATE

ACDS reads HIST FILE and the GRPXX LIMIT files, and prepares one card for each account-requisition pair with a non-zero balance. These cards are written into OUT FILE. The arguments are:

| | |
|---|---|
| HIST | name 1 of HIST FILE |
| INSTAL | "IPC" |
| SEQ | batch number |
| VOU | voucher number |
| DATE | in form MMDDYY |

Identification

HED

edit HIST FILE

Usage

R HED -N1-

N1 will be HIST if omitted. HED will open N1 FILE for reading, and print "Type:" to indicate that it is ready for editing requests.

More than one request can be given on a line. The requests are:

1. PRINT

   Print top line of current entry

2. PRINTF

   Print full current entry in the format of HIST FILE.

3. NEXT

   Go to next entry

4. INSERT (same args as ADD, below)

   Insert before current entry

5. ADD args

   Add after current entry. There are three options:

   a) ADD F NN

      The file NN FILE will be added

   b) ADD U PROB PROG GRP

      A user entry will be added. HED will ask for the user full name

   c) ADD A PROB GRP

      An account trailer for PROB will be added

6. DELETE

   The current entry will be deleted.

7. SPLITU FN

   The current entry will be written into FN FILE and deleted from the file.

8. SPLITA FN

All entries with PROB the same as the current entry, including the current entry, will be written into FN FILE and deleted from N1 FILE.

9. CHANGE C1 V1 ... Cn Vn **

Thange the current entry. Each item identified by Ci will have its value set to Vi. If one of the Ci specifies a text field (NAME), the Vi is ignored and the program will ask for the value on a separate line. Numbers are whole numbers: to set the fractional part of a dollar figure, use the form +.xx to add the cents after setting the whole-number part. If "*p" is typed instead of **, the entry will be typed out.

The charge codes are: (x may be 1-6, y may be 1-5, and z may be 1-9).

| User | |
|---|---|
| Yx | Year sums |
| Mx | Month sums |
| Wx | Week sums |
| Dx | Daily use |
| TAy | Time allotted |
| DISKQ | Disk quota |
| DU | Disk use |
| TREC | Disk total |
| ND | Days on |
| TNFIL | UFD use |
| COD | code |
| DED | disk |
| ON | date on |
| OFF | date off |
| PROB | probno |
| PROG | progno |
| GRP | group |
| YSUM | year dollar sum |
| MSUM | month dollar sum |
| NAME | full name |
| HELP | summary of codes |

| Account | |
|---|---|
| YS | Year dollar sum |
| MS | Month dollar sum |
| DS | Daily dollar sum |
| Yz | Year dollar charge |
| Mz | Month dollar charge |
| Dz | Daily dollar charge |
| SP | Charges this req |
| PROB | probno |
| GRP | group |
| HELP | summary |

10. FILE

N1 FILE will be finished up and closed out

11. QUIT

HED will exit without filing

Note that there is no "TOP" request. This is because HIST FILE is usually greater than 100 records, and so it shouldn't be too easy to make multiple passes.

The message "Past it" means that a search has failed, but we have discovered this without reaching EOF. Usually, a search failure leads to the message

EOF. Type "F" to file, "Q" to quit.

Editing requests to HED can be abbreviated as follows:

| | |
|---|---|
| PRINT | P |
| PRINTF | PF |
| NEXT | N |
| INSERT | I |
| ADD | A |
| DELETE | D |
| SPLITU | SU |
| SPLITA | SA |
| CHANGE | C |
| FILE | FL |
| QUIT | Q |

Edit Master System Files
EDUA

Purpose
To permit Project MAC system administrators to edit files which constitute the central part of the data base for a CTSS system-administration facility.

Usage

R EDUA NAME1 -NAME2- -REQUEST args-

NAME1 is the primary name of the accounting file to be edited. The secondary name must be "TIMACC."

NAME2 TIMACC will be created. If NAME2 is "*" or omitted, NAME2 will be the same as NAME1.

REQUEST if specified, is the first editing request to be executed by EDUA.

EDUA is a special editing program for use with files of the form of DACCNT TIMACC. It is a descendant of ALOCAT (MAC-A-242). Many of the useful features of other file-editing programs operating on CTSS have been incorporated into EDUA, but the special requirements of allocation-file editing dictated the creation of a special editing program.

We may envision a "window" which allows EDUA to see one entry in the file, the current entry, corresponding to a particular user or to a common file. Requests may be issued to modify or delete the current entry, to make a new entry the current entry, to find a particular entry and make it the current entry, or to perform various utility operations.

When EDUA is resumed, it opens NAME1 TIMACC for reading and NAME2 TIMACC for writing, executes the REQUEST given on the command line if any, and then prints "Type:" and waits for editing request lines. A description of each request follows.

Editing Requests

1. C -PROB NAME- -CODE1 VAL1- -CODE2 VAL2- ...
causes the entry PROB NAME to be changed. If PROB NAME are omitted, the current entry will be modified.

EDUA: "Type changes:" (only if no pairs given)
user: pairs of words, the first word being a code for the item in the entry, and the second being the new value. An asterisk ("*") causes exit from the "C" request. If the program is in verify mode, the program will type "Change:" every time it waits for a pair, and the modified entry will be printed on exit from the request.

codes:

| | |
|---|---|
| IRUN, DISK, TAPE | for record quotas |
| T1, T2, T3, T4, T5, | for time allotments |
| NAME | name |
| PROG | programmer number |
| PROB | problem number |
| PARTY | party line group |
| RCODE | user restriction code |
| PASS | Password |
| UNIT | unit group |
| STNBY | standby indicator |
| UFD | home directory |
| GROUP | cutoff flag |
| SG | subgroup |

The second word in a PASS pair may be anything: it is not read. The program will type "Password" and turn the printer off when it receives this pair, so that the user may type the password without its appearing on the console.

An illegal code will cause an error message to be printed, and no change will be made.

2. I PROB NAME PROG DISK T1 T2 T3 T4 -T5- -UFD-
A PROB NAME PROG DISK T1 T2 T3 T4 -T5- -UFD-
"A" causes an entry for PROB NAME to be inserted after the current entry. "I" causes an entry for PROB NAME to be inserted before the current entry. If the problem number of the current entry is not equal to PROB, the program will give a choice between inserting a problem number header entry first, and ignoring the request.

EDUA: "progn:" (if PROG not given)
user: the programmer number assigned by the Information Processing Center for this user.

EDDA: "Password"
user: password for this user. (The program turns the printer off for typing this word.)

EDDA: "Disk t1 t2 t3 t4 t5" (if none given)
user: six numbers separated by spaces, representing the disk allotment and time allotments or shifts 1-5. — If T5 is not specified it will be set to 0.

Standard values are inserted for PARTY, DRUM, TAPE, and PCCDE. These may, of course, be modified with the 'C' request.

If the program is in the verify mode, the new entry will be printed.

3. D -PROB NAME-
causes the entry for PROB NAME to be deleted. If PROB NAME is omitted, the current entry will be deleted. If the program is in verify mode, the entry being deleted is printed. The 'D' request automatically reads the next entry in the file: that is, it incorporates an automatic 'N' request (see below).

4. P
causes two lines representing the current entry, to be printed. These are not in the form of the file; in particular, the password does not print. The lines printed have the form

```
PROB NAME PROG PARTY PCODE UFD
DRUM DISK TAPE T1 T2 T3 T4 T5 SG
```

If UFD is the same as PROG, it is not printed.

5. F PROB NAME
causes the entry PROB NAME to be located in the file, and made the current entry. If the program is in verify mode, the entry is printed.

6. N
causes the program to move down one entry in the file. If the program is in verify mode, the new entry is printed.

7. V ON
V OFF
controls the setting of the verify switch. In "verify" mode, entries are printed when they are found or modified. In "verify off" mode, they are not printed, and the "TYPE" comment is suppressed. The program starts in verify mode.

8. T
causes the program to close and reopen the file, so that the current entry is the first entry in the file.

9. FILE
causes the old file to be replaced with the edited copy.

10. Q
causes the program to exit, forgetting the changes given on the last pass through the file. This is useful if a mistake in editing has been made.

11. HELP
causes a summary of requests to be printed.

12. PT
causes the full entry to be printed in the form of the file.

Ncte<u>s</u>

1. The message "ECF" means that the program cculd not find the entry it was lccking fcr, perhaps because cf a mistyped request cr dropped character. A "T" request may te issued at this pcint. (NCTE THAT "T" FIIES CHANGES!)

2. The "F" request will accept a "*" for either PROB cr NAME. "F * NAME" will find the first cccurrence of NAME in the file. "F PRCE *" will find the problem header fcr PRCE. "F" and similar requests search the part of the file belcw the current entry.

3. Eoth the name and the programmer number of a ccmmcn file must te CMFLxx, where 'xx' is the ccmmcn file number. 99 common files are pcssible.

4. The "C", "F", and "D" requests will accept a prcgrammer number instead cf NAME.

5. In general, EIUA will accept as much as pcssible cr cne line, and ask for the rest. Passwcrds are an excepticn tc this rule, since they are read by the LCGIN subrcutire GTPASS.

6. Cn successful return frcm any search except that fcr "* *", EIUA will print the current group number if it has changed since the last request.

Restriction

EIUA may te used cnly by system administrators.

78733

## Identification

REQ

modify LIMIT files

## Purpose

REQ provides an editing facility for the files named GRPXX limit. It enables the program user to modify the requisition data for legal problem numbers in a group, and to modify control information in the header of the LIMIT file.

## Usage

R REQ FUNC ARGS

In general, REQ will read as many arguments as possible from the command line, and ask for the rest, so that one could say just "R REQ" and receive the response "func:".

Each function code and its appropriate arguments is described below:

1. LIMSET g C1 V1 ... Cn Vn *

   sets GRPg LIMIT file header.

   The following codes are legal

| | |
|---|---|
| DRUM | drum limit (usually O) |
| DISK | disk limit |
| TAPE | tape limit (O, or 999999) |
| T1 - T5 | shift limits |
| PARTY | private party group |
| GRC | "best" RCODE (usually 201) |
| ALPB | allocator probno (C0116) |
| ALPG | allocator progno (9OXX) |
| GN | group number |
| NAME | group name (30 chrs) |
| SUPER | supervisor name (24 chrs) |
| SUPPH | supervisor phone |
| ALCTR | allocator name (24 chrs) |
| ALPH | allocator phone |

The items ID, TO, and ADDR do not have an associated Vi; REQ will ask for their values on a new line.

4. PRINT grp prob

   The entry for "prob" will be printed.

5. DELETE grp prob

   The entry for "prob" will have its code set to 2.

6. ADDON GRP PROB XRQ XACCT XFUNDS XON XOFF

   A supplementary requisition, to take effect when the current one reaches its cutoff date, will be added to the entry for PROB.

7. DEBIT GRP PROB DOLS CTS
   CREDIT GRP PROB DOLS CTS

   An entry will be made in TODAY RQST to charge or credit the problem number, REQ will ask for an explanation of the charge (24 chrs) on a separate line. HRQ will pick up the entry at midnight and accumulate it in the miscellaneous field of HIST.

8. RESET GRP PROB

   Enters a request in TODAY RQST to have the "SP" field of HIST set to "MS" - that is, to have "charges this req" set equal to "charges this month".

9. ADJ GRP PROB SIGN DOLS CTS

   Enters a request in TODAY
   RQST to modify the "SP" field in HIST up or down.

After a NEW, MODIFY, or ADDON, REQ asks "OK" and, if "yes" is given, files the LIMIT. If "no" is given, it enters a MODIFY request.

The value of FUNDS must be either "OPEN" or a number in dollars. Dates must be 6 digits in the form MMDDYY.

The values of DISK, DRUM, TAPE, T1, T2, T3, T4 and T5 in a LIMSET may be either numbers or amounts like "+20" (no space is allowed) to increment the current value by 20. GROUP may be either GRPnn or just nn.

Summary

| func | abbr | args |
|---|---|---|
| LIMSET | L | grp C1 V1 ... Cn Vn * |
| NEW | N | grp prob req off funds |
| MODIFY | M | grp prob C1 V1 ... * |
| PRINT | P | grp prob |
| DELETE | D | grp prob |
| ADDON | | grp prob req acct funds on off |
| DEBIT | | grp prob dols cts |
| CREDIT | | grp prob dols cts |
| ADJ | | grp prob sign dols cts |
| RESET | | grp prob |

Identification

SPY SAVED

Usage

R SPY PRØB PRØG

If the user is not logged in, SPY prints a message and loops sleeping and looking for him.

When PRØB PRØG logs in, SPY prints a message of the form

PRØB PRØG line X CONSØLE

and calls CHNCOM. (1). If no commands are stacked, and "START" is typed, SPY will patch the ADØPT vector to make the console attached to the program an IIØØ slave of PRØB PRØG's console. (Key 22 must be set.) If PRØB PRØG logs out, SPY returns to waiting for him.

Identification

SCANUA

Check time-accounting files for privilege code.

Usage

R SCANUA -N1- -N2-

The file N1 N2 is scanned for privilege codes containing specified bits, and entries found are printed.

If N1 is omitted, its value is UACCNT.

If N2 is omitted, its value is TIMACC.

Response: ENTER CODE

User: Octal digits representing bits in a user restriction code (see Section AG. 7.03) which are to be searched for. Any entry which matches one or more of the specified bits is printed.

Identification

DELALL

Purpose

DELALL attaches to a user directory and deletes all the files in it. It is used to clean out "sleeper" directories.

Usage

R DELALL PROB PROG PROB1 PROG1 ...

The contents of all directories specified are deleted. No error messages are printed if files can't be deleted or if the directory does not exist.

Identification

KKUT

Purpose

KKUT prints a list of problems nearing cutoff or cut off.

Usage

R KKUT -LOUD-

KKUT scans through HIST and the LIMIT files. If LOUD is specified, a line of the form

GRPXX PROB reason

is printed for each problem nearing cutoff or cut off. Summary statistics are printed at the end.

Identification

SUS

Sorted Usage Summary

Usage

R SUS arg -n-

arg may be

| | |
|---|---|
| NT | number of temporary records |
| NF | number of files |
| NL | number of links |
| NZ | number of zeros (holes) |
| TE | total UFD length |
| ALOT | record quota |
| USE | record use |
| UR | record use counting temps |

SUS reads DISK USAGE and sorts it in core on arg. The top n entries are printed. If n is omitted, it is 15.

8/18/66



## Identification

Administrative utility command
ADMIN

## Purpose

ADMIN is a special privileged command which performs several utility functions for system administrators.

## Usage

In the following discussion, "USER" may be
1) Problem and programmer number of a logged-in user
2) The line number (1-34) of a logged-in user
3) Null, taken to be the line of the command user

The general form of a use of the ADMIN command is:

ADMIN FUNCT -ARGS-

FUNCT is one of ten code words which determines the particular utility function which will be selected. The interpretation of the arguments (if any) depends on the function chosen.

The available functions are:

| | |
|---|---|
| KILL | Send a user an automatic logout |
| PB | Set up a guaranteed percentage for a user |
| SETTAU | Set a user's time allotted and time used |
| WRITE | Type on a user's console |
| ERASE | Remove a file directory entry from the M.F.D. |
| PRNTON | Print on the on-line printer |
| DELTEM | Delete all temporary files in a directory |
| RDTIMU | Find out how much time a user has used |
| CNDRLA | Reset used-time figures to zero |
| SPY | Spy on a user |

A short description of each function and the arguments which it requires follows.

'KILL' USER

If USER is logged in, a message of the form

M1416 385 ON 100041 KILLING PROB PROG

is printed on-line, and USER recieves a "WAIT" message followed by an automatic logout.

'PB' PCT USER

If USER is logged in, his guaranteed minimum percentage (PB) is set to PCT. A message of the form

M1416 385 ON 100041 ST PB PROB PROG TO PCT

is printed on-line.

'SETTAU' S ALLOT USD USER

If USER is logged in, his TAU-vector entry for shift S is set to show ALLOT minutes allotted and USD minutes used. Either ALLOT or USD may be "*", indicating no change. A message of the form

M1416 385 ON 100041 ST TU S PROB PROG TO ALLOT USD

is printed on-line.

'WRITE' USER

If USER is logged in, the command will respond "TYPE." The line typed will be written on USER's console, after resetting his output buffers.

'PRNTON'

Response: TYPE.
Command user: Any message

An identifying line is printed on-line, followed by the successive lines typed by the command user. An input line consisting of only a carriage return causes the on-line printer to eject a page and causes the command to terminate.

Alternate form:

'PRNTON' '(MESS)' N1 N2

The file N1 N2, which must be a card-image file, is printed on-line, preceded by an identifying line and followed by a page eject.

'ERASE' PROB PROG

If an entry for PROB PROG is found in the M.F.D., its use will be set to zero by a call to ALLOT, and it will be deleted by a call to DELMFD.

'DELTEM' PROB PROG

All files in the directory PROB PROG which have the temporary mode bit will be deleted.

'RDTIMU' USER

The time allotted to and used by USER will be typed out in the following form:

```
NAME    PROB  PROG
  1       A1    U1
  2       A2    U2
  3       A3    U3
  4       A4    U4
  5       A5    U5
NOW ON UNIT xxxxxx LINE xxxx
```

If PROB PROG is given for a user who is not logged in, NAME will be blank and the last line will be

FIRST LOGIN D1 T1 LAST LOGOUT D2 T2

'CNDRLA'

An on-line message of the form

M1416 385 ON 100041 RESET TUSED

will be printed, and then the time-used portion of all TAU-vector entries will be set to zero. This function is used once a month, when the file TIMUSD TIMACC is deleted, in order to make sure that users who are logged in have their used times reset.

The items NAME, SUPER, and ALCTR do not have a Vi associated; REQ will ask for their value on a separate line. Typing "*" ends the string of Ci Vi and causes the header to be typed, after which REQ asks "OK". A "yes" will file the LIMIT, and a "no" will cause REQ to ask for more Ci Vi sequences.

2. NEW grp prob acct req off funds

REQ will add a new problem number entry to the limit file. It will ask for the problem name, the problem supervisor name, and his address on separate lines.

3. MODIFY grp prob C1 V1 ... Cn Vn *

REQ will accept change code and value pairs until a "*" is found. The codes are:

| | |
|---|---|
| PROB | problem number |
| ACCT | account number |
| REQ | requisition number |
| ON | date on |
| OFF | date off |
| FUNDS | funds |
| XRQ | next requisition number |
| XACCT | next account number |
| XFUNDS | next funds |
| XON | next date on |
| XOFF | next date off |
| CODE | code (1 = active, 2 = deleted) |
| ID | problem name (30 chr) |
| TO | problem supervisor (24 chr) |
| ADDR | supervisor address (18 chr) |

'SPY' USER

If USER is logged in, a message of the form

USER FOUND PROB PROG LINE xxxx

is typed, and the command calls CHNCOM.(1). If no other command is stacked and START is typed, the command patches the ADOPT array in core A so that USER's input and output appear on the console of the command user; that is, the command user's console is made and IOOO slave of USER.

If PROB PROG for a user not logged in is supplied, the command sleeps waiting for him to log in. If he does, execution proceeds as above.

Restriction

The ADMIN command may be used only by designated system administrators, who must have a privilege code of 137(8) or better.

(END)

Identification

TAUT

utility functions for administration

Usage

To create a file directory:

R TAUT MAKE PROB PROG -RQ-

The directory PROB PROG is created. If RQ is given, the record quota is set to RQ.

To set authorship on a GAF after modifying it:

R TAUT UNLOCK GRPXX

GRPXX GAF in C116 90XX is set to mode 124 with author 90XX.

Identification

MOVER SAVED

Move contents of one directory to another.

Usage

R MOVER FMPB FMPG TOPB TOPG -'(NSTP)'- -(CHGE)-
-'(STOP)'-

Moves files and links from directory

FMPB FMPG to Directory TOPB TOPG.

Print entries which won't move.

If (STOP) is supplied, program goes dead in FMPB FMPG. Otherwise, it will re-attach to user directory.

If (CHGE) is supplied, the program will attempt to change the mode of any file which will not move to O. (STOP) or (NSTP) must be supplied if (CHGE) is used.

The name of any file which won't move is printed.

Identification

SETDMO SAVED

Purpose

Set a user into a party group, for purposes of authorizing demonstration access.

Usage

R SETDMO PROB PROG NAME -GROUP-

The program switches to COMFIL 2 and searches for the specified user in UACCNT TIMACC, writing a new copy. The entry for the user has its party line group set to GROUP.

If GROUP is not specified, it is set to '2'.

## Group Allocation Tools

The following pages document the subsystem of programs available to group allocators and provide some information concerning the contents of a GAF from the group supervisors point of view.

CTSS Resource-Usage Accounting and Allocation

Responsibility for allocation of a group's total resources among the members of the group, and decisions as to who in the group may have access to the computer, reside with the group supervisor. Each group supervisor may log in on the system-administration problem number as the group allocator for his group. This user may divide a gross allocation agreed upon between the group supervisor and the administration among the users in the group.

The group supervisor, logged in on the group allocator number, communicates his decisions concerning allocation to the system-administration facility by maintaining a Group Allocation File (GAF). A special-purpose editing program, ALOCAT, has been provided to facilitate this file-maintenance process. Limits on the total resources which may be distributed to users in a group are also kept on the disk in a file called GROUP LIMIT. Also listed in this file are the problem numbers available for assignment in the group.

Whenever a GAF has been modified, the program NOTIFY is used to inform the system-administration facility that a change has occurred, thus implicitly requesting corresponding changes to the system master accounting file.

Once each day, notifications are collected, and any GAF's which have been changed are merged into that master accounting file, thus implementing the resource-allocation decisions of the group supervisors. This process is sometimes referred to as "turning the crank".

Reports of computer time used will be sent to group supervisors in two forms: a daily report written in the file directory of the group allocator (under the name GRPXX TIMRPT), and a weekly report sent through Institute mail.

## Definitions of Terms

Authorization to use the CTSS system consists, in operational terms, of an entry in a special table in memory (the "User Account File") which is accessible only to the time-sharing supervisory program and to a single user who is responsible for overall system administration. Each entry in the file consists of a unique label and items which establish limits on disc storage occupancy and computer time use, plus other control information.

The pertinent items in a User Account File entry or GAF entry are described below:

1) Problem number(*): Historically a "task" or "job" number. Three or four digits preceded by a letter. Examples: M6119, T0123. (To be assigned from a block of numbers provided by Headquarters and listed in the Group Limit file.)

2) Programmer number(*): Unique to an individual; may be thought of as a representation of the user's name for internal system use. Up to four digits. Must be unique within a group of entries bearing identical problem

numbers. This number is permanently assigned to an individual by the M.I.T. Information Processing Services Center. Users who do not have a programmer number should contact the Center to obtain one.

3) Programmer name: A six- (or fewer) character representation of the user's surname. By convention, the last six characters of names exceeding six. This item has no significance inside the system (programming number is always used); it is provided only for convenience in logging in. Must be unique within a block of entries bearing identical problem numbers. Examples: SMITH, ENBERG (Rosenberg).

4) Password: A critical item. The password feature of the system is the only means of limiting access to authorized users. Security of passwords should be carefully maintained. In general, only one person in a group should have access to the Group Allocation File. Up to six letters and/or digits.

5) Record quotas: Three numbers expressing the maximum number of secondary storage "records" (blocks of 432 computer words, or about 30 card images) a user may occupy on each device. There are three devices:

DRUM - Presently, all drum storage is used by the supervisor. It is not available to users.

DISK - 80,000 records of disk storage are available. Typical quotas are 100 to 300.

TAPE - Users with a tape quota may write tapes in a special format. All users may reach such tapes. Tape storage is available to users on a limited basis. Special arrangements with Headquarters are necessary.

6) Time allotment: The number of minutes of computer (processing) time allotted to a user within the accounting period (one month). A separate time allotment is included for each "shift" of computer operation. Shifts are defined as follows:

| | Time | Shift Number |
|---|---|---|
| | 0800-1800 | 1 |
| Mon. - Fri. | 1800-2400 | 2 |
| | 2400-0800 | 3 |
| | Sat. 0800-Mon. 0800 | 4 |

Shift 5 represents "foreground-initiated background" (FIB) use. It allows users to request jobs to be run on a deferred basis, when the user is not at this console. See section AA.1.03 of the CTSS Programmer's Guide.

7) Party group: Each user is assigned by his administrative group supervisor to one of three so-called "party-line groups" (abbreviated "party group") for purposes of system access control. (Note the ambiguous use of the term "group". "Party group" will be specifically distinguished from "administrative group" where necessary.)

The three party groups are:

S - standby one

C - common

P - private

Assignment to party group S means that a user will always be assigned a "standby line", regardless of the number of other users logged in. He may never log in while the maximum number of users (at present, 30) is logged in. While logged in he is always subject to a "bumping".

Assignment to party group C means that a user competes with all users so assigned (from all administrative groups) for one of the "primary lines" in this common party group. If he fails to obtain a primary line and fewer than the maximum number of users are logged in, the user may log in on a standby line (subject to bumping).

Assignment to party group P means that a user competes for a primary line only with those other users from his own administrative group who are also assigned to P. If he fails to obtain a primary line, the same conditions as under C apply. The availability of party group P places under the control of the administrative group supervisor means of guaranteeing system access to a subset of his users.

The distinction between primary and standby lines lies in the "bumping" privilege. A user who has access to a primary line is permitted by the system to log in, possibly bumping (forcing an automatic log out of) some standby user (if necessary to keep the total number of users below the limit). If more users are assigned to a party group than the number of primary lines it contains, then the lines are assigned on a first-come, first-served basis.

8) Restriction code: Associated with each user there is a user restriction code, which defines the actions a user may perform. For most groups, this code will always be '000001'. A complete explanation of the code is available in the CTSS Programmer's Guide, Section AG.7.03. A restriction code bit is available, enabling group supervisors to create a so-called restricted user. The 200's bit of RCØDE, if on, prevents a user from using any of the "disk-loaded" commands--that is, he may use only

LOGIN, LOGOUT

RESUME, RESTOR, CONTIN, RECALL, R

SAVE, MYSAVE

START, RSTART

USE, PM, STOPAT, TRA, PATCH, STRACE, FAPDBG

All other commands will be "not found".

For all practical purposes, such a user may only resume saved files, and the particular saved files in his directory determine completely what use he can make of the time-sharing system.

The restricted-user feature will be useful for authorizing student or other highly limited or specialized uses of the computer without offering its full generality. Links to many of the public commands as SAVED files, such as LISTF SAVED, will be placed in public file. Therefore, setting up a restricted user would typically consist of the following sequence:

a) create a non-restricted user in the GAF, and NØTIFY.

b) the next day, log in on the number, and set up the links necessary to give the user the desired privileges.

c) change the RCØDE of the user to "201" or "200" in the GAF, change the password, and NØTIFY.

d) the restricted user is now set up.

Notes: 1) Imbedded blanks are not permitted in any item.

2) The problem number/programmer number (marked with (*) above) considered together constitute the unique label of each authorized-user access to the system. Use caution in changing either or both, since:

a) uniqueness must be maintained

b) such a change effectively deletes the user whose entry is altered, together with all his files, from the system and creates a completely new user entry.

Modify Group Allocation Files
ALOCAT

Purpose
To permit Project MAC group supervisors to create and edit files which constitute a part of the data base for a CTSS system-administration facility. These files (GAF's; see below) express the group supervisor's decisions as to how the computer resources allotted to his group are to be distributed to group members.

Usage

R ALCCAT NAME1 -REQUEST args-

NAME1 is the primary name of the allocation file to be edited. The secondary name must be GAF (Group Allocation File)

REQUEST if specified, is the first editing request to be executed by ALCCAT. As many of its arguments as desired may be specified on the command line.

ALCCAT is a special editing program for use with GAF's. Many of the useful features of other file-editing programs operating on CTSS have been incorporated into ALOCAT, but the special requirements of allocation-file editing dictated the creation of a special editing program.

We may envision a "window" which allows ALCCAT to see one entry in the file, the current entry, corresponding to a particular user or to a common file. Requests may be issued to modify or delete the current entry, to make a new entry the current entry, to find a particular entry and make it the current entry, or to perform various utility operations.

An entry named GROUP LIMIT must be present in the group allocator's file directory. This will be a link to the system administrator's directory. The contents of GROUP LIMIT define the amount and kind of resources which the allocation group has been authorized to distribute among its members. The file contains

1. a list of allowable problem numbers for this group.
2. the total record quotas and time allotments authorized for this group.
3. the special party line group (if any) assigned to this group (See section AH.1.01 of the CTSS manual.)
4. an octal number in the form of a user restriction code, with 1-bits wherever a privilege bit may be assigned by the group supervisor. (See section

AG.7.03 of the CTSS manual.)

5. the problem and programmer number of the group allocator. This must match the identification of the user running ALOCAT.

The GAF entries will be checked against the information in GROUP LIMIT. Three kinds of errors are possible:

a) Format errors - an entry in the file is irretrievably wrong. It will be dropped.
b) Convention errors - certain required information is not present. It will be inserted.
c) Restriction errors - an entry contains an item which is unauthorized for this group. A standard value will be inserted if possible. Illegal problem numbers are commented upon, but not fixed.

In addition, the total allotments for time and records in the file will be checked against the authorized totals in GROUP LIMIT, and a comment itemizing those resources which exceed the limit will be printed

a) when the file is first read
b) whenever a summary is requested
c) when the file is closed.

When ALOCAT is resumed, the file GROUP LIMIT is read in, and then the whole GAF is read into core, sorted, and checked. If REQUEST was specified on the command line, it is executed. ALOCAT then says "Type: " and waits for editing request lines. A description of each request follows.

Editing Requests

1. C -PROB NAME- -CODE1 VAL1- -CODE2 VAL2- ...
causes the entry PROB NAME to be changed. If PROB NAME are omitted, the current entry will be modified.

ALCCAT: "Type changes:" (only if no pairs given)
user: pairs of words, the first word being a code for the item in the entry, and the second being the new value. An asterisk ('*') causes exit from this request. If the program is in verify mode, the program will type "Change:" every time it waits for a pair, and the modified entry will be printed on exit from the request.

| codes: | | |
|---|---|---|
| | DRUM, DISK, TAPE | for record quotas |
| | T1, T2, T3, T4, T5, | for time allotments |
| | NAME | name |
| | PROG | programmer number |
| | PROB | problem number |
| | PARTY | party line group |
| | RCODE | user restriction code |
| | PASS | Password |
| | SG | subgroup (for allocator use) |
| | UFD | Home directory |

The second word in a PASS pair may be anything: it is not read. The program will type "Password" in red and turn the printer off when it receives this pair, so that the user may type the password without its appearing on the console.

An illegal code will cause an error message to be printed, and no change will be made.

2. A PROB NAME PROG DISK T1 T2 T3 T4 -T5- -UFD-
causes an entry for PROB NAME to be created. The file is first searched to insure that an entry for PROB NAME does not exist.

ALCCAT: "Progn:" (if PROG not given)
user: the programmer number assigned by the Information Processing Center for this user.

ALCCAT: "Password"
user: password for this user. (The program turns the printer off for typing this word.)

ALCCAT: "Disk t1 t2 t3 t4 t5" (if none given)
user: six numbers separated by spaces, representing the disk quota and the time quotas on the 5 shifts. If T5 is not specified, it is set to zero.

If "DISK" is specified as zero, ALOCAT assumes you are creating a ufd-less user, and asks

ALLOCAT: "Ufd"

user: the programmer number of the user's home directory. if the directory PROG UFD does not exist when the user tries to log in, he will be unable to log in. If the user being created is to have his own directory, but you wish to have him have a record quota of zero, simply type his programmer number in response to this question.

Standard values are inserted for PARTY, DRUM, TAPE, and ACCDE. These may, of course, be modified with the 'C' request.

If the program is in the verify mode, the new entry will be printed.

3. D -PROB NAME-
causes the entry for PROB NAME to be deleted. If PROB NAME is omitted, the current entry will be deleted. If the program is in verify mode, the entry being deleted is printed. The 'D' request automatically reads the next entry in the file: that is, it incorporates an automatic 'N' request (see below).

4. P
causes two lines representing the current entry, to be printed. These are not in the form of the file; in particular, the password does not print. The lines printed have the form

```
PROB NAME PROG PARTY ACCDE UFD
DRUM DISK TAPE T1 T2 T3 T4 T5 SG
```

(UFD will be blank if it is the same as PROG; that is, if the user is not ufd-less.)

5. F PROB NAME
causes the entry PROB NAME to be located in the file, and made the current entry. If the program is in verify mode, the entry is printed.

6. N
causes the program to move down one entry in the file. If the program is in verify mode, the new entry is printed.

7. V ON
V OFF
controls the setting of the verify switch. In "verify" mode, entries are printed when they are found or modified. In "verify off" mode, they are not printed, and the "Type" comment is suppressed. The program

starts in verify mode.

8. S
causes a summary of the allocation file to be printed. Only totals for disk allocation and shifts 1 through 5 are shown. The file is checked against "GROUP LIMIT".

9. FILE -NEWNAME-
sorts the file, checks totals against GROUP LIMIT, and files the edited GAF as NEWNAME GAF (or as NAME1 GAF, if NEWNAME was not specified).

10. Q
causes the program to exit without making any changes to the GAF. This is useful if a mistake in editing has been made.

11. HELP
causes a summary of requests to be printed.

Notes

1. 'F' and similar requests will search the whole file, not just the portion below the current entry.

2. If the named file is not found when the program is started, ALCCAT assumes that the GAF is to be created, and activates an 'A' request.

3. Both the name and the programmer number of a common file must be CMFLxx, where 'xx' is the common file number. 99 common files are possible.

4. ALCCAT makes no distinction between upper-case and lower-case letters on input.

5. The "F", "C", and "D" requests, when searching, will accept either programmer name or programmer number.

6. The entire GAF is held in core while ALCCAT is operating. ALCCAT sorts the GAF when it is first read in, and again when the GAF is written out.

7. In general, the command will accept as much as possible from the typed line, and ask for the rest. Thus you may type an "A" request like

   a m123 Jones 2040 100

   ALCCAT will type "Password" to ask for the password, and then type "t1" to ask for the shift 1 time allotment. If you typed two numbers, it would then type "t3" to ask for shift 3, and so on. Passwords are an exception to this rule, because they are read by a special program.

8. The comment "ILLEGAL PROBNO: TXXXX" will be printed for each GAF entry with a problem number which is not authorized for your group. These entries are counted in the total kept by ALCCAT, and so may put you over your group's limit.

Restriction

ALCCAT is the only editing program which should be used on a GAF. Ignoring this restriction may have serious consequences, since the GAF's are basic components of a system-administration facility for CTSS. ALCCAT is the means of insulating group supervisors from changes in this facility, such as GAF format changes, which are of only internal significance.

In particular, the use of the ED command to edit a GAF, because of the automatic sequence-numbering behavior of ED, will produce an unusable GAF.

Request GAF processing

NOTIFY

## Purpose

To notify the system-administration facility of CTSS that a GAF has changed, thus implicitly requesting corresponding changes to the time-accounting files.

## Usage

R NOTIFY

NOTIFY will write several words into CURENT GAFS in common file 1 to indicate to the system-administration facility that the GAF should be processed. The total allocations in the GAF are checked against GROUP LIMIT, and the program will refuse to notify if the GAF is over any limit.

## Restriction

The system-administration facility reads only CURENT GAFS in T116 CMFL01, so that group allocations must be done while logged in under problem number T116 (C116 at IPSC).

Change passwords

PW

## Purpose

Enables a group supervisor to create new passwords for every member of an allocation group.

## Usage

R PW N1 N2 N3

response: TYPE MAGIC PHRASE (36 or fewer characters)

user: any phrase (the printer is turned off)

response: END OF PW.

N1 N2 is the file to be read

N3 N2 will be created, with every password replaced by a "pronounceable" six-letter word.

N3 must not be the same as N1

## Operation

The 36-character phrase typed is used to initialize a random-number generator. Then the file "DIGRAP HTABLE", a file giving the relative frequency, in English, of each pair of letters, is read. This file should be linked to in T116 CMFL01 (C116 at IPSC). The file N1 N2 is read, record by record, and the password field is replaced by a randomly-generated combination of letters.

## Discussion

It is recommended that group leaders change all passwords for their group at least once a month. This program provides a simple way of accomplishing this.

Sort account file

SORTUA

## Purpose

Since ALOCAT must place additions at the end of the file, a GAF undergoing modification frequently gets out of sort. While this presents no problem to the system-administration facility, many group leaders prefer to keep their GAF's in sort for readability. This program sorts files in GAF format.

## Usage

R SORTUA NAME1

The file NAME1 GAF will be sorted on problem and programmer numbers, with low numbers at the beginning.

Identification

GAFSUM

Summarize Group Allocation Files

Usage

R GAFSUM -FILNAM-

The file FILNAM GAF will be summarized. If FILNAM is omitted, the program will ask for the name.

response: 'D' FOR DETAIL, 'S' FOR SUMMARY.

user: The letter "D" will select the detail mode of operation, in which every entry satisfying the search criterion (see below) is printed and summarized. Summary lines will be printed after all detail lines. The letter "S" suppresses the individual lines and prints only the totals.

response: GIVE FIELD AND CONTENTS FOR TOTAL

user: FIELDIDENT CONTENTS

This pair of words selects a search criterion. Only those entries satisfying the criterion will be summarized.

The field identification codes are:

| | |
|---|---|
| PROB | total on given problem number |
| PROG | total on given programmer number |
| NAME | total on given name number |
| PARTY | total on given party-line assignment |
| RCODE | total for all entries containing any bit specified |
| * | total for all entries |

response: detail lines, if any, in the form

PROB NAME PROG PARTY RCODE DISK T1 T2 T3 T4 T5 TOTALS

followed by summary lines, giving

1. count of entries meeting the criterion
2. disk allotment total
3. time allotment totals in minutes, and in hours and hundredths

the program then types

'D' FOR DETAIL, 'S' FOR SUMMARY, 'F' FOR NEW FILE

"D" and "S" are as before; "F" will cause the program to ask for the first name of the new file. The file name "END" will cause exit from the program.