Procedure for Recovering From Arbitrary Lost Pages (or Tracks or Volumes)

0. Some general hints:

   a. Do a %BUFFER=32.

   b. $LOG on some permanent file.

   c. Always $LIST the output files from each step so that they will appear in both the conversation buffer and in the $LOG output. This makes it a whole lot easier to back up if something screws up. (Actually, you really <u>don’t</u> want to list either "list3" or "list4", and "listvntd1" and "listvntd2" are listed by the macros.)

   d. $Empty list? . If any steps can be skipped, then there is no risk of having lists of files from the last disaster.

   e. If the disaster has damaged files which are needed for the recovery (or the basic running of the system), then the recovery must be done from a backup system. Many of the macros take a "tables=" parameter to provide a set of alternate tables describing the desired file system. The object deck supplied via this parameter should be just the UMMPS tables, with all disks as NODMGR volumes and no fake devicelist. If you want to also use a non-standard set of file routines, then use the "frtns=" parameter in addition to the tables parameter.

   f. $set ebm=h and etm=h so times will be recorded in the log

   g. Keep a log on paper of what is going on so that the "next shift" programmers will know what has been done.

   h. There are references sprinkled throughout these instructions to files like "list1, list2", etc. These are the names which the macros will use by default. The prefix used by the macros can be set using the "set_default" macro, q.v.

   i. Turn macros on and attach FILE:CMDMACLIB.

1. Reformat and <u>zero</u> all damaged pages (or appropriately redefined alternate pages). One way to accomplish this is by using the VAMREC program. Remember to release the

volume from the Disk Manager before running VAMREC. Another way is to re-dasdi a new volume using the DASDI program and copy the bad pack to the just-dasdi'd pack using the DISKCOPY program. Record the damaged page numbers reported by VAMREC or DISKCOPY for use with VTOCUTIL in step 15.

1.5 Run CHKVTOC and check all the volumes. This provides a basis for comparison on the state of the disk subsystem.

2. Use the VNTD program (T,C-trace the catalog-) to check to see if extent header of catalog was lost. If so, rebuild extent header from DSCB if possible (non-existent program) or restore extent header from filesave tape (currently not saved).

3. Use the FIXEH macro to run the FIX EXTENT HEADER program which reads all catalog pages in each extent to find zeroed catalog pages and rebuilds the record headers. (This program also has the capability for deallocating the zeroed segments in the extent header, but this function is not needed in the disaster recovery process.) If "pre-allocated" parts of the master index, system file catalog or scratch file catalog have been lost, this program should probably rebuild record and segment headers and relink the segments (currently it doesn't).

4. Use the VNTDOUT1 macro to run the VNTD program (V,C-verify the catalog-) to find out which catalog segments have bad pointers (to lost segments) or which catalog segments are no longer chained to some user catalog (because of a lost segment in the chain or a lost master index). This will produce the file "listvntd1", which is the input to the next step.

5. If the output from VNTD in step 4 indicates that any part of the chain of master index catalog segments has been broken, then skip ahead to step 8.

6. Use the LIST0 macro without a "par=" to run the FIX CATALOG program to chain catalog segments back together. This program uses the file "listvntd1" from VNTD (V,C) as input. It looks at the userid and link field in each affected segment to figure out how to chain the segments back together and which segment is the first in the chain. For every resultant chain of orphaned segments, FIX CATALOG attempts to find a home in the catalog for it. This must be the end of a chain of good segments starting at a master index entry since the master index is undamaged. If no

chain of segments for this userid was discovered ruptured by VNTD (in step 4), then the chain of segments must be truly orphaned due to an inopportune system crash. In this case, the segments are verified to be empty of file or sharing descriptors and if so are deallocated in the extent header. If there are descriptors of some sort in the chain, a message is produced identifying the first segment in the chain so that it can be manually re-chained after examination.

This macro generates the "list0" file of userids whose catalog was damaged. This list of userids is input to the AMALCOMP program in step 19. VNTD can be run at this point with the V,C option for verification purposes. No errors should result.

7. Skip ahead to step 11, since the master index needed no fixing.

8. This (and the following two steps) are only used if VNTD indicated in step 4 that the master index was damaged. Use the LIST0 macro with PAR=FMI to recover any corrupted master index segments. (FIX CATALOG will complain if this is not done.) This will ensure that all retrievable portions of the master index are chained back together so that step 10 does not: 1) rechain the master index after it possibly expanded as a result of the disaster making it <u>appear</u> to have been properly terminated when in fact it was damaged, or 2) re-create a master index entry when in fact it may exist on a section of the master index which was orphaned as a result of the disaster and thus was not found by the catalog verify program.

9. Use the VNTDOUT1 macro to again find out which catalog segments have bad pointers or which segments are no longer in some user catalog. This produces a new version of the "listvntd1" file for use by FIXCAT in the next step.

10. Use the LIST0 macro with PAR=LMI to again run FIX CATALOG, this time to fix the user catalogs. This will add userids to the "list0" file which was produced by step 8. Since PAR=LMI has been specified, FIXCAT may create an entirely new master index entry if the master index for a userid was lost. (If it has to recreate a master index entry, it calls a special entry to CRECAT, (RECRECAT) in the file system to do such.)

    The VNTD program can be run at this point with option V,C to verify that the catalog segment chaining is now fixed.

11. Use the VNTDOUT2 macro to run the VNTD program to verify the affected user catalogs to find out what sharing descriptors are no longer pointed to by file descriptors and which file descriptors point to lost sharing descriptors. This uses the V,U,*SYS and V,U,*ALL options.

12. Use the LIST2 macro to run the FIX SHARING DESCRIPTOR program which reads the output from VNTD (file "listvntd2") and zeros sharing descriptors not pointed to by file descriptors. (These files will get their catalog information restored if the DSCB is OK or get completely restored if DSCB is bad. The AMALCOMP program will discover this fact.) This program will also zero the chain pointer in the last good sharing descriptor of any good file descriptor and add the name of the file to the file "list2". This file contains the names of files which should have sharing information restored from the filesave tapes.

    The LIST2 macro also sorts list2.

    VNTD may be run again at this point with the V,U,*ALL option for verification purposes. No errors should result.

13. If you have "only" lost one or more entire volumes then create an empty "list1" file and skip ahead to step 17. Otherwise proceed to fix the VTOCs on the affected volumes via steps 14 through 16.

14. Run the VTOCUTIL program using the VTOCUTIL macro.

    14a. If you know a PAT page has been damaged, use the FINDDSCBS and FIX option to rebuild the PAT. The FINDDSCBS and FIX option will succeed reliably if (and ONLY if) bad PAT pages are zeroed. It should ONLY be used if you know a PAT page has been damaged.

    14b. If VTOCUTIL indicates that there are problems with the DSCB chains for any file (or if a DSCB page was zeroed in step 1), use the FIX option to deallocate any DSCB’s that have lost a Type E or Type F somewhere in their chain. VTOCUTIL will also update the PAT to reflect the data pages reclaimed due to deallocated DSCB’s. (The AMALCOMP program will note these files as being in the catalog but missing from the volume, so they won’t be lost without a trace.)

15. Use the the LIST1 macro to run the VTOCUTIL program to determine which files were affected by the damage to the records discovered in step 1. For each affected volume, enter the volume name to VTOCUTIL. VTOCUTIL should find no

errors on the volume. (Step 14 should have fixed them.) Then enter the damaged pages as PAGE commands, as instructed by the macro. This produces the file "list1", a list of files and DSCB-E locations which need their data restored. If the output from VTOCUTIL indicates that any *IPLAREA file has been damaged, then it should be replaced by using the *IPLINIT program.

Use the SORTLIST1 macro to remove any duplicates (and any *IPLAREA or *??ASTER.CATALOG?? files) and sort the "list1" file.

16. Use the VALIDATELIST1 macro to find out which files on "list1" are still consistent. Note that this must be done from the id MTS, since validate requires you to have read access to the file. You should also $SET FILEREF=OFF before doing this, to prevent the references from being recorded. Remove the consistent files from "list1" by manually editting the file. These are files which only had an unused data page damaged and are still valid. I think the VALIDATE program cannot be run from a backup system. If some consistent file is not removed from "list1", then it will be unnecessarily restored.

17. Use the LIST3AND4 macro to run the CATLIST program to produce "list3" (a list of all non-scratch files in the catalog) and to run the VTOCLIST program to produce "list4" (a list of all non-scratch files in the VTOCs). Note - if you are running on the damaged system no files should be created between the production of "list3" and "list4".

18. Determine whether the two key online filesave directories were affected by the disk problem. (These files are named "RSTR:FILEDIR.MAS" and "RSTR:TAPEDIR".) If not, proceed with step 19.

    18a. Check whether the master filesave directory, "RSTR:FILEDIR.MAS" has been damaged. If not, skip to step 18b. If the master filesave directory, "RSTR:FILEDIR.MAS" has been damaged, check to see whether the file "RSTR:FILEDIR.NEW" has also been damaged. If it hasn’t, $RENAME RSTR:FILEDIR.NEW AS RSTR:FILEDIR.MAS to get the previous version. (The file save merge program leaves the previous version of the master filesave directory in RSTR:FILEDIR.NEW after it has built a new version.) If neither "RSTR:FILEDIR.MAS" or "RSTR:FILEDIR.NEW" is available, the previous version of "RSTR:FILEDIR.MAS" must be restored off of a file save tape, using *RST. (Maybe we should avoid the possibility of losing these directories by saving copies of them on particular tapes after each run of the file save merge program.)

If FSTEST is used to move either directory onto the test pack, SPUNCH must be specified @I@¯TRIM.

18b. Check whether the filesave tape directory, "RSTR:TAPEDIR" has been damaged. If not, skip to step 18c. If it has, the most recent version of this file must be restored from a file save tape using *RST. Next, it must be determined which tapes have been written by filesave since the last time the directories were saved (this is determined by scanning back over the operators logs). The tape directory entries for these tapes must be deleted for the proper running of the REGENERATE program in step 18c. (To delete the entry for a tape, just delete the line in the file corresponding to the tape's tape number, e.g. line .057 has the entry for tape number 57.)

18c. Find the file save tapes written by file save since the time which the file and/or tape directories were restored from. Then run the REGENERATE program with the options appropriate to the disaster. REGENERATE will rebuild tape directory entries for these tapes if PAR=TAPE is supplied, file directory entries for the files saved on these tapes if PAR=FILE is supplied, or both by giving PAR=BOTH.

18d. If PAR=FILE or PAR=BOTH have been specified, REGENERATE will create files of the name RSTR:FILEDIRnn which must be merged with the old master file restored in step 12a. This should be done by running the file save MERGE program with the option PAR=RECONSTRUCT. (This inhibits MERGE from declaring that files are destroyed if they are not present in the catalog. Since the catalog is being reconstructed at this point, and some files are lost but will be restored later, declaring them destroyed at this point would be wrong.) After MERGE has finished, proceed with step 13.

19. Use the AMALCOMP macro to run the AMALCOMP program. This program compares lists zero through four and the online filesave directory (RSTR:FILEDIR.MAS). It produces lists five through nine, which are the input to the subsequent steps. If you do not want to restore from the most recent tape versions, (perhaps because the disk problem occurred before the most recent file save), then you should specify a time= parameter on the amalcomp macro. It has to be in a form acceptable to the PLUS time and date routines, for example "Sep 17 10:45". For the details of what AMALCOMP does, see the decision table presented later under the heading "File information lost and final file status".

19a. Use SORTLIST8 if there are a lot of files to be restored. (See note below)

20. $Signon to RSTR. Create an empty file to use as a checkpoint file, say "checkpoint". Use the RESTORELIST8 macro with UNIT0=checkpoint to run the FAST RESTORE program to restore data and/or catalog for affected files. This reads "list8" from the AMALCOMP program and asks that the appropiate filesave tapes be mounted. For efficiency, this program uses a special entry to CREATE, (RSTRCRE) which 1) does not initialize page 1 of the file and 2) which returns the page map buffer.

    Note: If you have lost a very large number of files, say several volumes worth, it may be faster to use multiple streams to do the restoration. Use the SORTLIST8 macro to sort "list8" into tape order. Then split "list8" into several pieces at tape boundaries and use the FASTRESTORE macro for each stream.

Time passes ... for any disaster worth its salt.

21. At this point you should be able to run on the production system.

22. Use the RECATLIST6 macro to run the RECATALOG program to recatalog files from scratch using "list6" from AMALCOMP. This program also fixes the file type appropriately.

23. Use the DESTROYLIST5 macro to run CALLDR to destroy files with lost DSCB or data using "list5" from AMALCOMP. CALLDR will generate an error message for files whose DSCB was lost, which should be ignored.

24. Use the RELDSKLIST7 macro to run CRELDSK to destroy files on "list7" from AMALCOMP. These files have lost catalog and data (but not DSCB) and no file save information is extant, or these files were uncataloged before the disaster.

25. Use the ACCUPDATE macro to update users disk accounting.

Additional Notes:

1) There is a program (*FILES) which takes as input "list9" from AMALCOMP and tells a user how she personally was affected by all of this.

2) Files lost without our knowledge are those created after

the last filesave which lost both DSCB and catalog. (In general, we know about the userid and can tell the user to beware, unless: (1) we lost the user’s master index entry; (2) we lost all of the user’s catalog segments; and (3) there are none of that user’s files in the file save directory.)

3) This procedure has the disadvantage that it may restore some files which were destroyed since the last online filesave. This can only happen if catalog segments associated with that user ID has been damaged or if master index has been lost, however.

4) Losing a whole volume causes no particular problems (other than the amount of information lost). If MTS001 is lost one would have to initialize an empty master index. If other extents of the catalog are lost the extent headers must be rechained.

APPENDIX

Programs:

1. VNTD - catalog verification;
   GUSER=input commands
   SPRINT=output

2. VTOCUTIL - PAT/DSCB verification and fixing;
   SCARDS=input commands
   SPRINT=output

3. FIXEH - fix extent header
   no input or output

4. FIXCAT - fix catalog segments and master index
   SCARDS=VNTD output from V,C
   0=list of userids whose catalog was affected (list 0)

5. FIXSD - fix sharing descriptor
   SCARDS=VNTD output from V,U,*ALL or V,U,....
   SPUNCH=list of files which lost some sharing information (list 2)

6. CATLIST - list files in the catalog
   SERCOM=errors
   0=list of files in the catalog

7. VTOCLIST - list files in the VTOCs
   SERCOM=errors
   0=list of files in the VTOCs

8. AMALCOMP - generates input for FASTRESTORE.
   reads
   0=list 0
   1=list 1
   2=list 2
   3=list 3
   4=list 4
   5=RSTR:FILEDIR.MAS (or a copy of it)
   writes
   15=list 5
   16=list 6
   17=list 7
   18=list 8
   19=list 9

9. FASTRESTORE - restore data and/or recatalog file from filesave tapes.
   SCARDS=input, list 8 produced by AMALCOMP
   0=checkpoint (both read and added to - initially should be assigned to an empty file)

10. RECATALOG - recatalog files from scratch

SCARDS=list 6 output from AMALCOMP

11. CALLDR - call DESTRYR to destroy files
    SCARDS=list 5 output from AMALCOMP

12. CRELDSK - call RELDSK to destroy uncataloged files.
    SCARDS=massaged list 7 from AMALCOMP

13. *VALIDATEFILE - Validate internal consistency of files
    GUSER=file name list, each line having a file name and options for that file (probably only ZEROCHECK for disaster recovery use)
    SERCOM=error messages

<u>Auxiliary Programs</u>:

1. REGENERATE - file save file and tape directory rebuilding
2. MERGE - file save file directory updating
3. *FILES - informational program to allow users to determine which of their files were affected by a disk disaster and print out a bulletin concerning the disaster.

<u>Macros</u>:

accupdate: runs *accrestore to update users' disk accounting. Uses as input suitable massaged data from lists 5, 6, 7, and 8.

amalcomp : runs amalcomp program to read lists zero through four and the online filesave directory and produce lists five through nine.
fdir= online filesave directory
time= time of disaster

destroylist5 : destroys the files on list 5 by running the CALLDR program. It asks for confirmation.

fastrestore : runs the fast restore program.
tables= alternate tables
frtns= alternate file routines
scards= input list of files to restore
unit0= checkpoint file

fixeh : runs the fix extent header program.
tables= alternate tables
frtns= alternate file routines
par= null to fix up the record headers, DEALLOCATE to deallocate incorrect segments. For disaster recovery, you should not specify this parameter.

list0 : runs the FIXCAT program which fixes segment chain pointers and produces "list0", the list of userids with damaged catalogs. It reads "listvntd1", produced by the vntdout1 macro.
tables= alternate tables
frtns= alternate file routines
par= nothing (if no master index damage), FMI to fix master index, LMI to fix user catalogs when there has been master index damage.

list1 : runs the VTOCUTIL program to produce "list1", the list of files which have damaged data pages.

list2 : runs the FIXSD program to produce "list2", the list of files which have damaged sharing information. It also fixes any invalid sharing descriptor pointers. It reads "listvntd2", produced by the vntdout2 macro. This also sorts "list1" and removes any duplicates.
tables= alternate tables
frtns= alternate file routines

list3and4 : runs the CATLIST program to produce "list3", the list of all permanent files in the catalog, and runs the VTOCLIST program to produce "list4", the list of all permanent files in the VTOCs. It also sorts these lists.
tables= alternate tables
frtns= alternate file routines

recatlist6 : recatalogs the files on "list6" by running the RECAT program. It asks for confirmation.

reldsklist7 : releases the disk space for files on "list7" by running the CRELDSK program. It asks for confirmation.

restorelist8 : restores the files on list "list8" by running the fast restore program.

sortfile : sorts a file.
in= input file (must be a single line file)
out= output file
keystart= start column of key (default 1)
keylen= length of key
lrecl= record length (default 255)
blksize= block size (default lrecl)
par= other sort parameters

sortlist0 : sorts "list0". It also removes duplicates.

sortlist1 : sorts "list1". This removes duplicates and *IPLAREA and anything matching "*??ASTER.CATALOG??".

sortlist8 : sorts "list8", the list of files to be restored, into tape sequence order. This is handy if you want to do multiple restore streams.

validate : runs the line file validation program.
guser= input file names
par= one-shot file name

validatelist1 : validates each file on "list1" with the ZEROCHECK option. This does not update "list1", that has to be done manually.

vntd : runs the VNTD program.
tables= alternate tables
frtns= alternate file routines

vntdout1 : produces "listvntd1" by running the VNTD program with input V,C.
tables= alternate tables
frtns= alternate file routines

vntdout2 : produces "listvntd2" by running the VNTD program with input V,U,*ALL and V,U,*SYS.
tables= alternate tables
frtns= alternate file routines

vtocutil : runs the VTOCUTIL program.

<u>Files</u>:

```
listvntd1 : output from VNTD V,C
      Output from VNTD
      Input to FIXCAT

listvntd2 : output from VNTD V,U,*ALL V,U,*SYS
      Output from VNTD
      Input to FIXSD

list0 : Userids which lost catalog segments
      Output from FIXCAT
      Input to AMALCOMP

list1 : Files to have data restored - data lost, DSCB ok
      Output from VTOCUTIL
      Input to AMALCOMP

list2 : Files to have sharing information restored
      Output from FIXSD
      Input to AMALCOMP

list3 : Files in the catalog
      Output from CATLIST
      Input to AMALCOMP

list4 : Files in the VTOCs
      Output from VTOCLIST
      Input to AMALCOMP

list5  :  Files  whose  data  was  lost  and must be destroyed by
           calling DESTRYR
      Output from AMALCOMP
      Input to CALLDR

list6 : Files whose FD was lost and must be recataloged
      Output from AMALCOMP
      Input to RECAT

list7 : Uncataloged files which  must  be  destroyed  by  calling
           RELDSK.
      Output from AMALCOMP
      Input to CRELDSK

list8  : Files to restore (data and/or DSCB and/or FD and/or SD).
           This list also contains location of files on  filesave
           tapes.
      Output from AMALCOMP
      Input to FASTRESTORE

list9 : Files and userids affected
      Output from AMALCOMP
      Input to *FILES
```

Decision Table for AMALCOMP Program:

In this table Y means yes, - means no, and ? means don’t care.

cat bad	lost data	lost si	in cat	in vtoc	on tapes	lost users	mult vers	action
Y	Y	-	-	Y	Y	?	-	fd si data
-	Y	-	-	Y	Y	Y	-	fd si data list0
Y	Y	-	-	Y	-	?	-	note reldsk
-	Y	-	-	Y	-	Y	-	note reldsk
?	Y	-	Y	Y	Y	?	?	data
?	Y	-	Y	Y	-	?	?	destroy
Y	Y	Y	Y	Y	Y	?	?	si data
Y	Y	Y	Y	Y	-	?	?	destroy
Y	-	Y	Y	-	Y	?	?	si dscb data
Y	-	Y	Y	-	-	?	?	destroy
Y	-	Y	Y	Y	Y	?	?	si
Y	-	Y	Y	Y	-	?	?	note si
?	-	-	Y	Y	?	?	?	nothing
Y	-	-	-	-	Y	?	-	fd si dscb data
Y	-	-	-	-	-	?	-	note cat
-	-	-	-	-	Y	Y	-	fd si dscb data list0
-	-	-	-	-	Y	-	-	nothing
?	-	-	Y	-	Y	?	?	dscb data
?	-	-	Y	-	-	?	?	destroy
Y	-	-	-	Y	Y	?	-	fd si
Y	-	-	-	Y	-	?	-	recat
-	-	-	-	Y	Y	Y	-	fd si list0
-	-	-	-	Y	-	Y	-	recat list0
-	?	-	-	Y	?	-	-	reldsk
?	?	-	-	Y	?	?	Y	reldsk

Note that the correct functioning of this program depends on a subtle use of the "Mult Vers" test - namely that if there are multiple versions which appear only in the VTOCs, then the first one will NOT have the Mult Vers test true. Also if there is a version in the catalog, then Mult Vers will be on for all versions and further there will not be multiple versions in the catalog.