Published: 07/14/67
(Supersedes: BG.8.03, 03/02/67)

Identification

Directory Supervisor, Special User Primitives
C.A. Cushing

Purpose

The directory supervisor provides the primitives for manipulating directory entries and decides the permission needed to carry out the requested operation from the intent of the caller.

Primitives

Certain primitives of directory supervisor are callable only by special users (certain back up and administrative procedures). These primitives also have one of four intents (read, execute, write, append) with respect to the given directory. The following is a list of these primitives and their normal intent.

1. setlimits (write)
2. setsystrap (write)
3. setdtd (write)
4. getentry (read)
5. putentry (write)
6. setretrieve (write)

1. setlimits

The primitive setlimits is called only by certain administrative procedures. This primitive sets the two storage limit parameters in a given branch (see BH.1.01 for a description of these limits).

```
call setlimits(dir, entry, hilimit, lolimit, code);

dcl  dir char(*), /*path name of a directory*/

     entry char(*), /*name of an entry in dir*/

     hilimit fixed bin(17), /*number indicating the desired
                              upper device bound*/

     lolimit fixed bin(17), /*number indicating the desired
                              lower device bound*/

     code fixed bin(17); /*if non-zero, it represents the code
                           of an error detected by the basic
                           file system*/
```

The user needs the write permission in the directory containing the branch to which entry points in order to change these two items in the branch. The findbranch primitive is called to find this branch. The storage limit parameters are changed to the given values and the date-and-time-branch-modified item is updated to the current date and time. Segment control is notified of the modification to the directory containing this branch through the primitive dirmod and control is returned to the caller.

2. setsystrap

The primitive setsystrap is called only by certain administrative procedures. This primitive sets the system trap procedure and argument list in a given branch. As a result of setting a system trap on a branch, any initial attempt to reference the branch will cause the given system trap procedure to be called.

```
call setsystrap (dir, entry, flag, trapsw, trap, code);

dcl flag fixed bin(2), /*a two-bit flag such that
                         if the left-most bit is
                            ON, replace the trap switch in the
                                branch with trapsw or if
                            OFF, ignore trapsw argument
                         if the right-most bit is
                            ON, replace the trap procedure and
                                argument list with trap or if
                            OFF, ignore the trap argument*/

      trapsw bit(1), /*new setting for the trap switch
                       argument in the branch*/

      trap char(*); /*new setting for the trap procedure
                      name and argument list in the branch*/
```

(NOTE: In order to delete the system trap item in a branch set the right bit ON in flag and set trap equal to null. In order to make a new trap effective for those who may have the segment active currently, the trap switch item must be set ON in the branch regardless of its current value.)

The user needs the write permission in the directory containing the branch to which entry points in order to change the system trap and/or switch in that branch. The findbranch primitive is called to find the branch to which entry effectively points. If the left (right)-most bit in flag is ON, the trap switch (trap procedure) item in the branch is changed to trapsw (trap). If the trap switch item was set ON and the segment to which the branch points is active, then the branchmod primitive in segment control is called. This call is made in order to deactivate the segment and force those processes using the segment to refind the branch pointing to it and thus spring this trap.

The date-and-time-branch-modified item is set to the current date and time, the branch is unlocked and segment control is notified of the modification to the directory containing this branch through the primitive dirmod. Control is then returned to the caller.

3. setdtd

The primitive setdtd is called only by the backup system. This primitive changes the date-and-time-last-dumped item in a given entry.

```
call setdtd (dir, entry, dt, code);

dcl dt bit(72); /*date and time to be put into the
                  date-and-time-dumped item in entry*/
```

The user needs the write permission to change the date-and-time-last-dumped item in an entry.

The findentry primitive is called to find entry. The date-and-time-last-dumped item in entry is replaced by dt, the entry is unlocked and control is returned to the caller.

4. getentry

The primitive getentry is called only by the backup system. This primitive returns the entire contents of a requested entry to the caller.

```
call getentry (dir, entry, count, date, cdsw, space, entryp,
     code);
```

```
dcl count fixed bin(17), /*total number of branch (link)
                           slot numbers in dir if entry
                           is a branch (link)*/

    date bit(72), /*date and time the above count was
                    last updated*/

    cdsw fixed bin(1), /*a switch when =1 indicates that
                         only the above two arguments
                         are wanted*/

    space area((*)), /*area in which the contents of
                       entry will be stored*/

    entryp ptr; /*pointer to the structure containing
                  these contents in space*/
```

The user needs the read permission in dir in order to get the contents of entry.

If the caller requests the contents of the CACL of dir or the count and date only then the primitive getdirseg of segment control is called to make dir a known segment. If the count and date only are desired they are found and returned to the caller. If the CACL is desired, its contents are packed into space and entryp is set to point to the beginning of these contents in space. Control is then returned to the caller.

If the caller requests the contents of a branch or link in dir, then the findentry primitive is called to find it. The contents of this entry are packed into space and entryp is set to point to the beginning of these contents in space. If entry is a branch (link) the count and date associated with branches (link) are returned also. The entry is then unlocked and control is returned to the caller.

5. putentry

The primitive putentry is called only by the backup system. This primitive replaces the entire contents of an entry which were originally obtained by a call to getentry.

```
call putentry (dir, entry, count, date, cdsw, space,
     entryp, code);
```

The user needs the write permission in dir to replace the contents of entry.

If the caller requests to replace the contents of the CACL in dir or only the count and date then getdirseg is called to make dir a known segment. If the CACL is to be replaced the entryp pointer is used to find the beginning of the information in space which is to replace the contents of the CACL in dir. If the count and date are to be replaced and count is greater than the total number of branches (links), then the branch (link) slot table must be increased to a size equal to count. If count is less then the count it is to replace, then those entries in dir which have slot numbers greater than count are deleted.

If the caller requests to replace the contents of a link or a branch and the associated count and date, the given count and date are replaced first as stated above. The findentry primitive is called to find the requested entry and the contents of this entry are replaced by the information given in space. The file pointer and current length items are not replaced in branches. If there is a naming conflict when trying to put the given names into entry, i.e., another entry in dir has this name, then that entry with the earliest date-and-time-branch-modified item will have all its names replaced by the name "reload.error.(some unique identifier)". Eventually the reloader will fix this entry name unless the tape, on which the information belonging to this entry was stored, is bad.

6. setretrieve

The primitive setretrieve is called only by the backup system. This primitive sets the retrieval trap and trap switch items in a given branch. If the retrieval trap switch is ON this implies that the segment to which the given branch points will be retrieved from the dump tapes by the retrieval trap procedure when the segment is referenced.

call setretrieve (dir, entry, flag, trapsw, trap, code);

See the discussion of setsystrap.