0) **open and close, special**

For objects of all kinds, the following generalities are true.

1) A global open count is maintained for each object. This is the number of processes holding the object open.

2) A local open count is maintained in each process for each object it holds open.

The global open count for an object is increased and decreased by 1 each time a local open count for the object drops from 0 or reaches 0, respectively.

3) For some kinds of objects, there is a different capability representing the open form of the object from that representing the closed form of the object.

For these objects, closing is sometimes done by presenting a capability for the open form and sometimes by presenting a capability for the closed form.

4) Whenever a discs process is restarted, the global open count is decremented for all objects which have a non-zero local open count.
I) directory

(These are low-level objects)

A) creation

[See object creation of document of 4/12/70]

Additional parameters would be:

i) max size (wd, rd size, others?)

ii) whether this directory is to have a directly associated accounting entry and, if so, how much of each associated item to store per entry.

B) open action

does not return own capability but does insure that the low-level file representing the directory is open (i.e., if a 0-level file, all in ECS). Hence two open directories represented by the same capability are the closed directory.

C) close action

applied to the same capability as the closed directory.

After enough close the low-level file representing the directory will be moved back to the close (global open count = 0).

Notes: all directory actions may be performed on closed directories. It will be temporarily opened for the action.
II) disk files

(These are low level disk objects)

A) creation

additional parameters (over object creation of 4/17/70)
would be only shape.

B) open action

This returns a new kind of capability (pos file)

C) close action

see Bruce for what kind of capability needed here
III) subprocess descriptors

(These are low-level disk objects)

A) creation

See a special document on subprocess descriptors
(a new one not yet ready)

B) open

Similar to directories, this just causes the low-level file
representing the subprocess descriptor to be brought
into core. No new capability is returned.
All actions may be performed by the user on an
unopened descriptor. It will be temporarily opened
for the action.

C) close

Just present the same capability

Note: For directories and subprocess descriptors, the
open action just makes repeated references to the
object more efficient.
IV. Access Keys

Each access key is just a number represented in the directory by that number represented in its capabilities by that number.

Access keys may not be opened and closed.

Creation requires no additional parameters. Creation does not make an ownership entry. This access key may not be destroyed.
IV) **Global ecs object**

A) **Implementation**
   
   a c-list containing the capabilities for the objects and a file containing a unique name for each of the objects

B) **Directory representation**
   
   the index in the c-list of A) along with the corresponding unique name in the file of A)

C) **Creation**
   
   (will be used only by system routines)

   supply a capability for an ecs object (with destruction bit on)

   The capability will be placed in an empty slot in the c-list of A), and a new unique name will be associated with it and the result placed in the directory

   (This creates an ownership entry. This permits destruction, which removes the object from the c-list of A) and resets the unique name in the file of A)

D) **Open**
   
   returns the capability from the c-list of A), option bits and the option bits of the cloned form of the object.
   (also unique name is checked)

E) **close**
   
   (not applicable)
VI) eis-goodie (do not remember new name)

A) Implementation

a clist containing capabilities for open objects and a hashtable, hashed on eis-goodie unique name, pointing to corresponding c-list location and containing global open count.

B) Creation

Just produces a new eis-goodie unique name which is placed in directory. Does not make an ownership entry.

C) Open

2 kinds of open

1) Open an eis-goodie with a presented eis object

f-return if the unique name of the eis-goodie already in the host table of A)
error if the destruction is not an in the capability for the presented object. Otherwise, the unique name is entered in the host table, a free slot is found in the c-list and the capability for the presented eis object is placed in the c-list. Also the host table entry is made to point to the slot in the c-list and the global open count becomes 2.

2) Open an eis-goodie, presenting no eis object.

f-return if the unique name of the eis-goodie
is not in the hash table
otherwise; bump the open count as needed
and return the capability from the indicated
place in the c-1st option bits masked by the
option bits of the capability for the "closed"
form of the object.

(1) close

must present a capability for the closed form of the object.
If global count goes to zero, the entry is removed from
the hash table and the corresponding file object
is destroyed.