<1,4,5,10,11,13,14>

DIRECTORY

P.8 P.3

1. Abstract directory structure

directory ::= accounting-part entry-list1
entry-list ::= [entry •••]
entry ::= name access-list object-part
name ::= nonblank-character •••
access-list ::= ~~next-access-key-number~~ implicit-options
[access-pair •••]
access-pair ::= ~~access-key-number~~ options
object-part ::= ownership | hard-link | soft-link
ownership ::= disk-file-spec | directory-spec |
subprocess-descriptor-spec | static-name-tag-spec
object-specifier
~~hard-link~~ ::= disk-file-spec | directory-spec |
subprocess-descriptor-spec | static-name-tag-spec |
dynamic-name-tag-spec | access-key-spec
hard-link::= object-specifier
soft-link ::= directory-spec name access-key-spec

Note: [ ] denote enclosed structure is optional
••• denotes preceding structure may be repeated
| denotes adjacent structures are alternatives

2. Directory actions available to the user

2.1. Basic entry access (a procedure used by several actions)

2.1.1. Parameters:

2.1.1.1. C: A directory (in which an entry is to be accessed)

2.1.1.2. BD: A (packed) string of characters, LJZF in rightmost 56 bits of each word.

2.1.1.3. C: An access-key

2.1.1.4. D: A set of options

2.1.2. Procedure -

2.1.2.1. Scan the directory (2.1.1.1.) for an entry, the characters of whose name are identically those of parameter 2.1.1.2. At most one such entry can exist, and if it doesn't, fail.

2.1.2.2. If the previous step found an entry then scan its access-list for an access-pair with an access-key-number the same as that of parameter 2.1.1.3.'s. Either zero or one such access-pair exists, and if the former is true, generate an error (abort).

2.1.2.3. Finally, perform an option-by-option logical conjunction ("and"), of the options from the access-pair found in the last step with the options of 2.1.1.4., and return the result along with the object-part of the entry found in step 2.1.2.1.

2.2. Get an object from a directory

2.2.1. Parameters:

2.2.1.1. C: A directory (with OB.ACC enabled)

2.2.1.2. BD: A string of characters

2.2.1.3. C: An access-key

2.2.2 Procedure -

2.2.2.1. Set $\partial \leftarrow$ the directory 2.2.1.1.
Set $\nu \leftarrow$ the name 2.2.1.2.
Set $\kappa \leftarrow$ the access-key 2.2.1.3.
Set $\omega \leftarrow$ a set of all-enabled options

2.2.2.2. Perform "Basic entry access" (2.1.) with ∂, ν, κ, and ω as parameters. (This step may fail or abort.)

2.2.2.3. If the object-part returned in the last step is an ownership or hard-link, then use the object-specifier (disk-file-spec, directory-spec, etc.) it contains, along with the options which were also returned, to construct a capability. Return this capability.

2.2.2.4. If the object part returned in step 2.2.2.2. was a soft-link, on the other hand,
Set $\partial \leftarrow$ directory-spec in that soft-link
Set $\nu \leftarrow$ name in that soft-link
Set $\kappa \leftarrow$ access-key-spec in that soft-link
Set $\omega \leftarrow$ options returned in step 2.2.2.2., and go back to step 2.2.2.2.

2.3. Owned object [and ownership entry] creation (disk files, directories, subprocess descriptors, static name tags)

[Owned objects are ones which may be destroyed — it is important that at least one ~~[illegible]~~ directory entry with the power of destruction for each such object exists. Thus creating an owned object results also in the insertion of an ownership entry in a directory; deleting this entry also destroys the object (2.6.1.).]

2.3.1. Non-scratch owned object creation

2.3.1.1. Parameters:

2.3.1.1.1. C: A directory (with OB.? enabled)

2.3.1.1.2. BD: A string of characters (a prospective name)

2.3.1.1.3. (Other parameters to describe the object being created)

2.3.1.2. Procedure -

2.3.1.2.1. The characters of parameter 2.3.1.1.2. must be nonblank. If not, or if the directory of 2.3.1.1.1. contains an entry with the same name, generate an error.

2.3.1.2.2. Otherwise, create the object as specified by the parameter(s) of 2.3.1.1.3.

2.3.1.2.3. Construct an ownership entry with the resulting object specifier, the name 2.3.1.1.2., and an access-list consisting of the implicit access-pair, containing a set of all-enabled options.

2.3.2. Scratch owned object creation
2.3.2.1. Parameters:
2.3.2.1.1. C: A directory (with OB.? enabled)
2.3.2.1.2. BD: A string of characters
2.3.2.1.3. D: An access-key number
2.3.2.1.4. (Other parameters to describe the object being created)
2.3.2.2. Procedure - as in the non-scratch case (2.3.1.2.) except:
2.3.2.2.1. Give the implicit options in the access-list only the power of destruction (OB.DSTRY).
2.3.2.2.2. Place a set of all-enabled options paired with the access-key number of 2.3.2.1.3.y in the access-list, also. (which may not be the null access-key-number)

2.4 Link entry creation

2.4.1. Create a hard-link entry in a directory
2.4.1.1. Parameters:
2.4.1.1.1. C: A directory (with OB.CLE enabled)
2.4.1.1.2. BD: A string of characters
2.4.1.1.3. AC: A disk-system-implemented object
2.4.1.2. Procedure -
2.4.1.2.1. The characters of 2.4.1.1.2. must be nonblank. If not, or if the directory 2.4.1.1.1. contains an entry with the same name, generate an error.
2.4.1.2.2. Otherwise, construct a hard-link entry in this directory with parameter 2.4.1.1.2. as a name

and with the object-specifier from the capability 2.4.1.1.3. Put the options from this capability in the implicit access-pair of the new entry.

2.4.2. Create a soft-link entry in a directory
2.4.2.1. Parameters:
2.4.2.1.1. C: A directory (with OB.CLE enabled)
2.4.2.1.2. BD: A string of characters
2.4.2.1.3. C: Another directory (with OB.INSL enabled ?)
2.4.2.1.4. BD: Another string of characters
2.4.2.1.5. C: An access-key
2.4.2.2 Procedure - as in the hard-link case (2.4.1.2.) except:
2.4.2.2.1. Construct an entry containing the directory-spec, name, and access-key-spec of parameters 2.4.2.1.3. through 2.4.2.1.5. (rather than the single object-specifier in step 2.4.1.2.2.)
2.4.2.2.2. Place a set of all-enabled options in the implicit access-pair.

2.5. Entry modification

2.5.1. Add an access-pair to an access-list
2.5.1.1. Parameters:
2.5.1.1.1. C: A directory (with OB.AAP enabled)
2.5.1.1.2. BD: A string of characters
2.5.1.1.3. C: An access-key
2.5.1.1.4. D: An access-key number

2.5.1.1.5. D: A set of options

2.5.1.2. Procedure -

2.5.1.2.1. Perform a "Basic entry access" (2.1.) upon parameters 2.5.1.1. through 2.5.1.1.3. and 2.5.1.1.5.

2.5.1.2.2. If the access-list of the entry found in the last step doesn't contain an access-pair with the same access-key number as parameter 2.5.1.1.4, then add such a pair to this access-list, containing the options which were returned in the last step. Otherwise generate an error.

error if 2.5.1.1.9 = 0

2.5.2. Delete an access-pair from an access-list

2.5.2.1. Parameters:

2.5.2.1.1. C: A directory (with OB.DAP enabled)

2.5.2.1.2. BD: A string of characters

2.5.2.1.3. D: An access-key number

IF 2.5.2.1.3 =

2.5.2.2. Procedure - If parameter 2.5.2.1.3. is in an access-pair in the access-list of an entry in 2.5.2.1.1. named by parameter 2.5.2.1.2., then remove this access-pair from the access-list, otherwise fail.

error if not found key

2.5.3. Rename an entry in a directory

2.5.3.1. Parameters:

2.5.3.1.1. C: A directory (with OB.RENM enabled)

2.5.3.1.2. BD: A string of characters (the old name)

2.5.3.1.3. BD: Another string of characters (the new name)

2.5.3.2. Procedure - If the directory 2.5.3.1.1. contains an

or if this string is not a legal name, generate an error. Otherwise, if the directory contains an entry named by parameter 2.5.3.1.2

entry named by parameter 2.5.3.1.2., then change this entry's name to parameter 2.5.3.1.3., else fail

2.6. Entry [and owned object] destruction

2.6.1. Remove an ownership entry from a directory
2.6.1.1. Parameters:
2.6.1.1.1. C: A directory (with OB.DOE enabled)
2.6.1.1.2. BD: A string of characters
2.6.1.1.3. C: An access-key
2.6.1.2. Procedure –
2.6.1.2.1. Perform the "Basic entry access" (2.1.), using parameters 2.6.1.1.1. through 2.6.1.1.3. and a set of options with only OB.DSTRY enabled. If this option is not enabled in the options which are returned, or if the entry found is not an ownership, generate an error.
2.6.1.2.2. Remove the entry found in the last step from the directory 2.6.1.1.3.
2.6.1.2.3. Destroy the object designated by the object-specifier returned in step 2.6.1.2.1. (It may already have been destroyed.)

2.6.2. Remove a (hard- or soft-) link entry from a directory
2.6.2.1. Parameters:
2.6.2.1.1. C: A directory (with OB.DLE enabled)
2.6.2.1.2. D: A string of characters
2.6.2.2. Procedure – If the directory 2.6.2.1.1. contains an

entry named by parameter 2.6.2.1.2., then if this entry is a link entry delete it from the directory, otherwise generate an error. If, on the other hand, no such entry exists, fail.

2.7. Display a directory

[This action will allow an abbreviation scheme to be implemented above the directory system.]

3. Directory actions for the load-/dump-/reaver procedure(s)

3.1. Get the nth LLD (low-level-disk-file) object (disk file, directory, or subprocess descriptor) in an ownership entry

3.1.1. Parameters:

3.1.1.1. C: A directory (with OB.GOD? enabled)

3.1.1.2. D: n (n = 1 means the first LLD object)

3.1.2. Procedure – If parameter 3.1.1.2. is non-positive, generate an error, otherwise if the directory 3.1.1.1. owns at least n LLD objects, return a capability for the nth, else tail.

3.2. Get the nth ownership entry of type directory

3.2.1. Parameters:

3.2.1.1. C: A directory (with OB.GOD? enabled)

3.2.1.2. D: n (n = 1 means the first directory object)

3.2.2. Procedure – As in 3.1.2., except only directories are counted.

3.3. Get the nth LLD object-specifier in a directory

3.3.1. Parameters:

3.3.1.1. C: A ~~directory~~ disk file, assumed in directory format

3.3.1.2. D: n

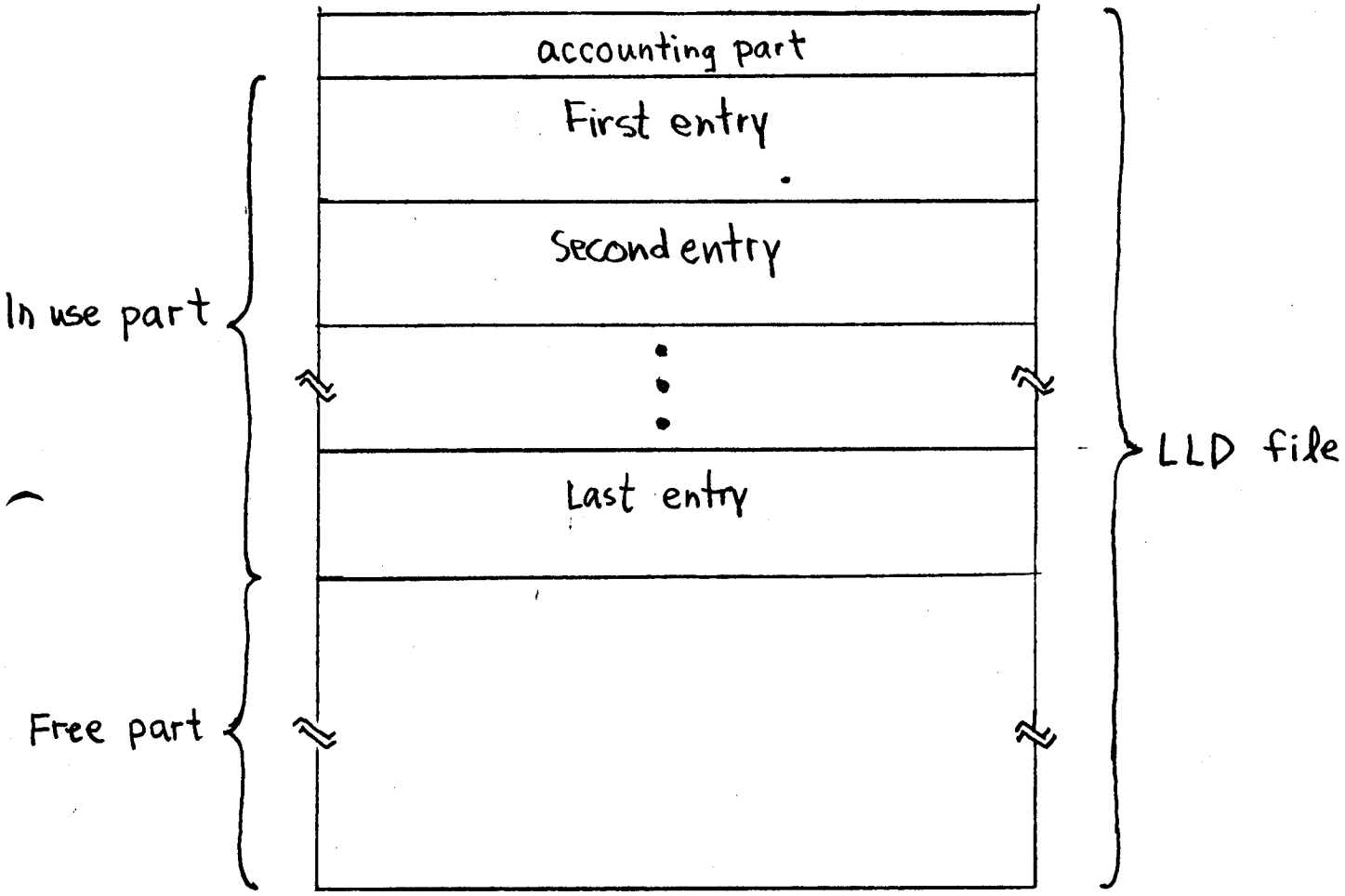
3.3.2. Procedure – If parameter 3.3.1.2. is non-positive, generate an error, otherwise if the directory 3.3.1.1. contains at least n LLD object-specifiers (including

the directory-spec within a soft-link), return the nth one, else fail.

3.4. Replace the nth LLD object-specifier in a directory
3.4.1. Parameters:
3.4.1.1. C: A ~~directory (with OBJ.GOD~~ enabled')
3.4.1.2. D: n
3.4.1.3. D: An object-specifier for a LLD object
3.4.2. Procedure - As in 3.3.2. except instead of returning the object-specifier, replace it with parameter 3.4.1.3.

4. Concrete Directory Structure

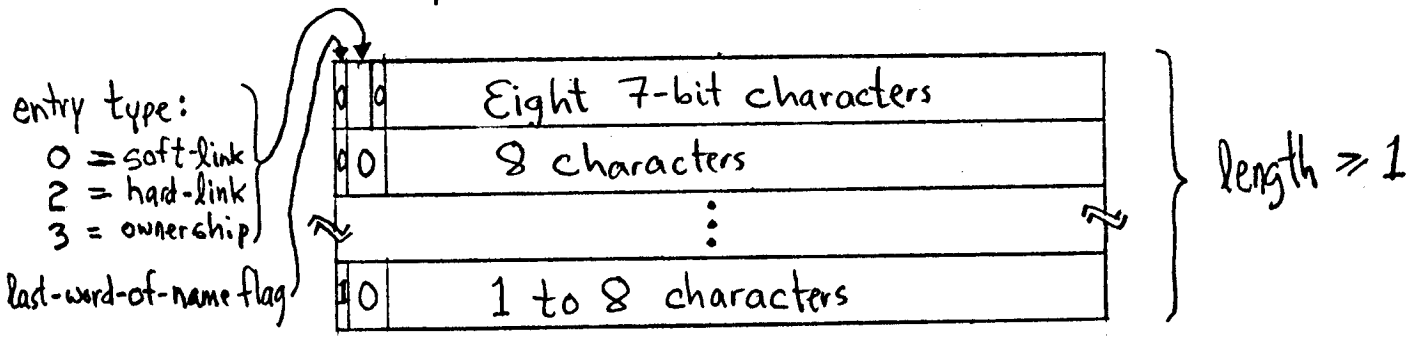
4.1. The directory



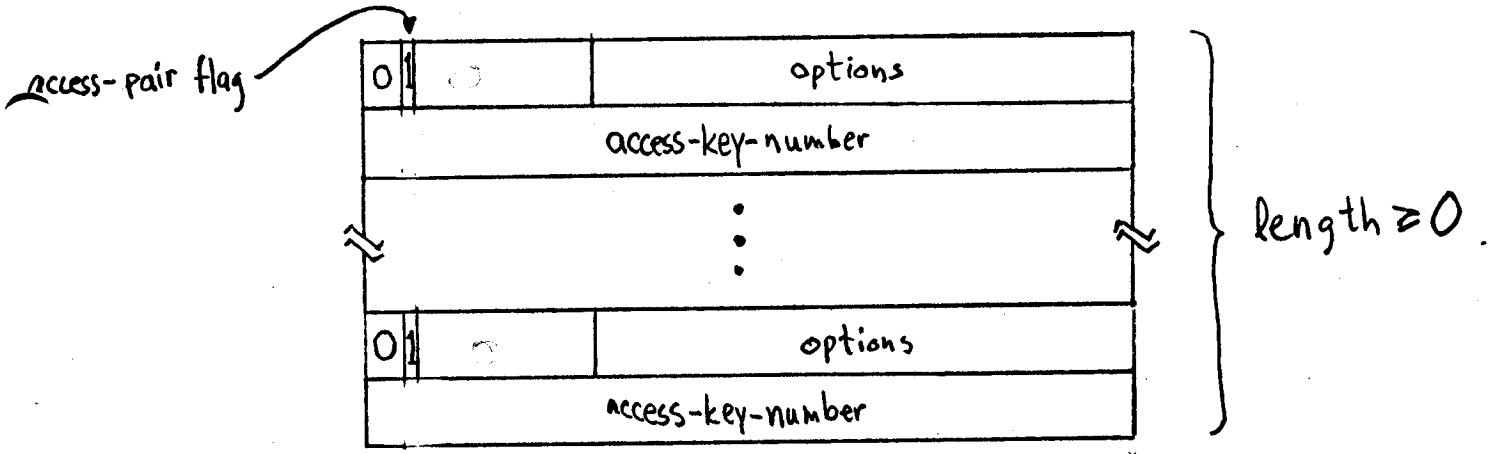
A directory is ~~either~~ of fixed size (~~number of~~ in words) and resides on a one-level file of one of the four standard sizes, ~~or~~ it is expandable, and resides ~~on a multilevel~~ file of an ~~standard~~ arbitrary shape. In any case, all free space is at the end of the file (~~and empty blocks of expandable directories are destroyed.~~ do not exist).

4.2. The directory entry

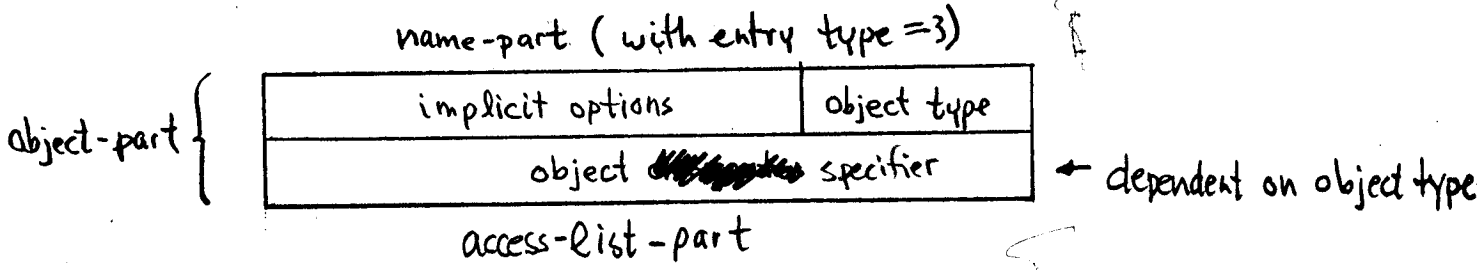
4.2.1. The name-part



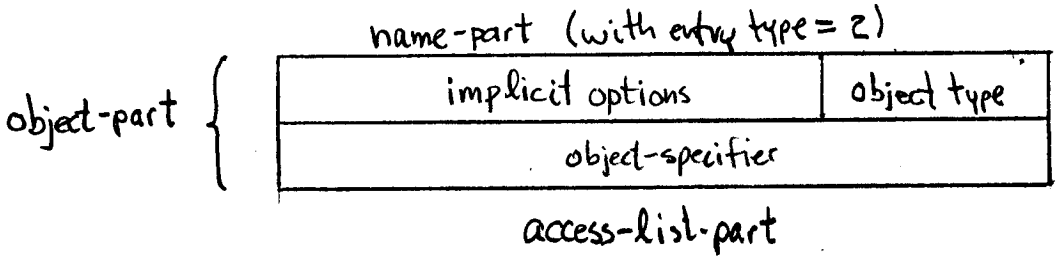
4.2.2. The access-list-part



4.2.3. Ownership entry



4.2.4. Hard-link entry



4.2.5. Soft-link entry