Handles and Directories

P. J. Denning

For CS471/CS571

Handles

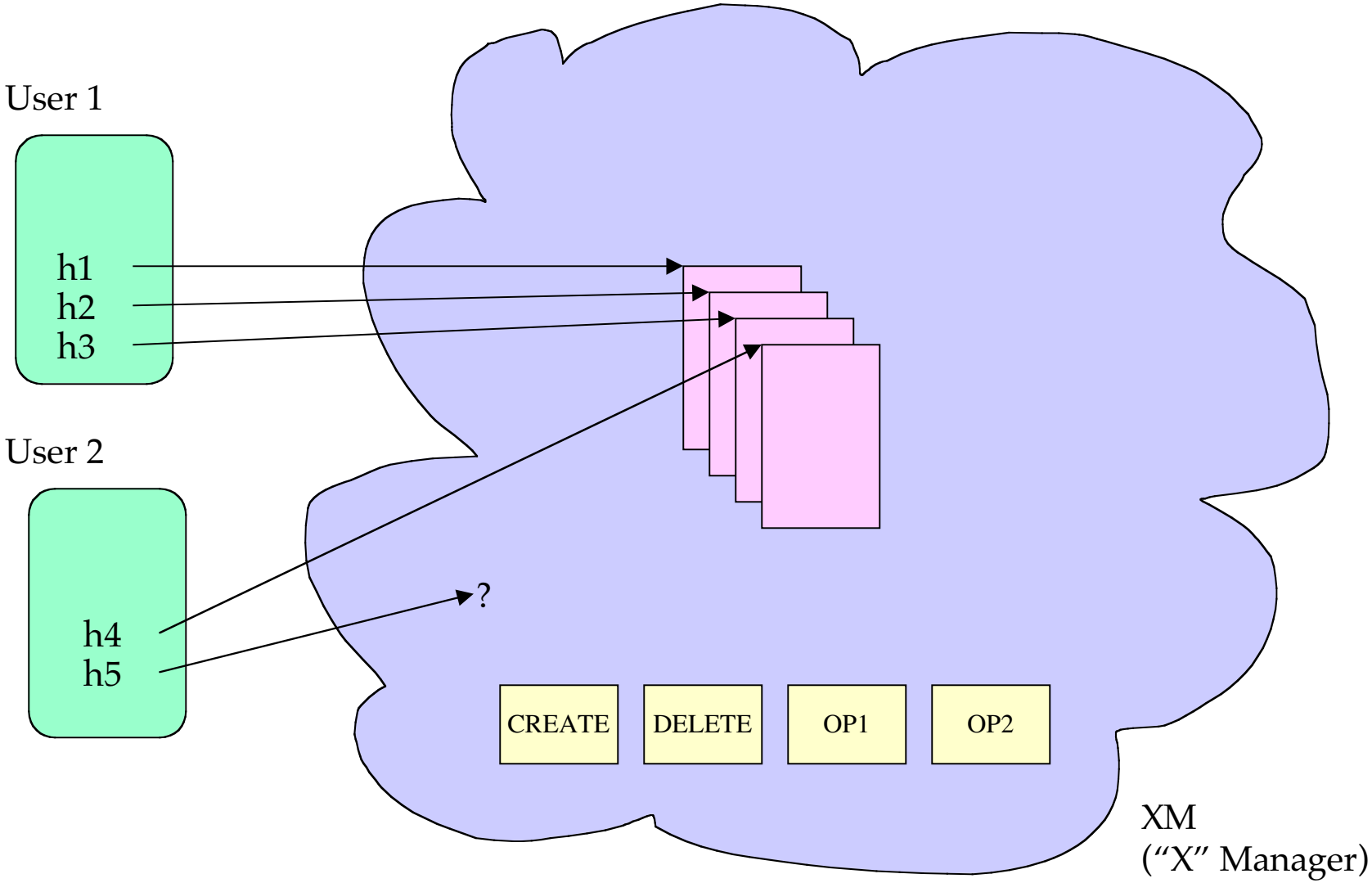
- Handles: system-generated names for objects
- Issued: by object class manager on creating object
- Protected: OS tries to prevent them from being modified
- h = (T, A, id) = (type, access, identifier)
- TYPES: OS object type
  - file
  - device
  - directory
  - virtual machine
  - etc

Diagram (next page) shows users calling on XM, the manager of type X objects.

When XM creates object of type X, it returns a handle to the object. Object itself is in the storage areas controlled by XM. Handle is in user's workspace.

To perform an operation, say OP1, on an object, the user calls the OP1 procedure (in XM) along with the object's handle and parameters of the call.

This is called *information hiding* because the inner details of the XM are hidden from users. Users see only objects as represented by handles. Users can perform only the operations supplied by the type manager. Details of implementation are of no concern to the user.



Handles

- TYPE: code indicating type of object designated by the handle -- e.g., file, device, pipe, virtual machine.
- Checked by type managers to ensure that operations are applied to the proper types of objects.

Handles

- ACCESS: code whose i-th bit enables i-th operation of the type manager
- E.g., if read is 4th command of 6 file-system commands, a handle enabling read would look like
  - (F, 000100, id)

Handles

- IDENTIFIER: a system-chosen number that distinguishes an object from all others
- If system includes multiple machines on network, the number is (machine id, local id)
- User does not choose the id
- User stores handles in own workspace

Requirements on id's

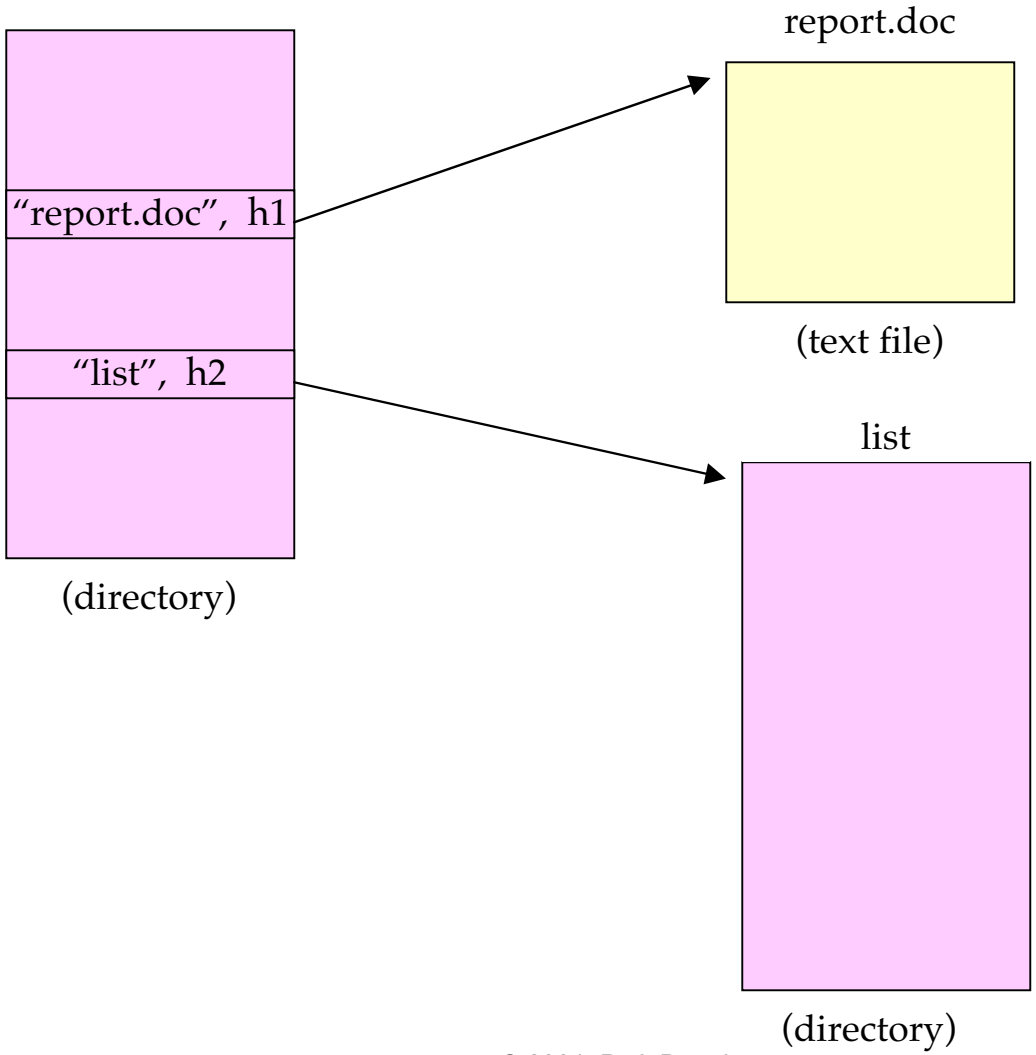
- Unique in space: no two objects of the same type have same id
  - What errors if not met?
- Unique in time: ids are not reused; just issued and used for one object only
  - What errors if not met?

Handle Management

- Handles given to user, stored in user space
- Protection: can handles be protected from tampering or modification?
- Management: can user assign local names to handles?
- DIRECTORY SYSTEM: provides protection and management for handles

Directories

- A directory is a table associating strings with handles
- Strings: user-chosen names
- (Handles: system-chosen unique ids)
- h = SEARCH(d, “string”)
- ENTER(d, “string”, h) --- no duplicates
- REMOVE(d, “string”)
- Directory hierarchies: ENTER(d1, “string”, d2) makes directory d2 subordinate of d1



Directories

- Directory tree: begins with root (Unix /)(DOS \)
- Pathname: sequence of directory names beginning with root and ending with object that need not be directory
  - /usr/local/info/mail/pjd
  - \windows\system\programfiles
  - hard disk:desktop:applications
- Extend SEARCH to accept pathnames for "string"
- Pathnames automatically unique in space (why?) but not in time (why?)

Directories

- VM component: current directory cd
- String cd automatically prefixed to a string generated by the VM for a directory search
- Another VM component: path
- Path is one or more directory-pathnames; SEARCH extended so that SEARCH("string") uses all the directories of the path as search places until a match on string is found
- Simplifies searching
- *Directories are not files* -- may be implemented as files (usual method), RAM tables, or database tables.