

Module 11: File-System Interface

- File Concept
- Access Methods
- Directory Structure
- Protection
- File-System Structure
- Allocation Methods
- Free-Space Management
- Directory Implementation
- Efficiency and Performance
- Recovery

Applied Operating System Concepts 11.1 Silberschatz, Galvin, and Gagne 1999

File Concept

- Contiguous logical address space
- Types:
 - Data
 - * numeric
 - * character
 - * binary
 - Program

Applied Operating System Concepts 11.2 Silberschatz, Galvin, and Gagne 1999

File Structure

- None - sequence of words, bytes
- Simple record structure
 - Lines
 - Fixed length
 - Variable length
- Complex Structures
 - Formatted document
 - Relocatable load file
- Can simulate last two with first method by inserting appropriate control characters.
- Who decides:
 - Operating system
 - Program

Applied Operating System Concepts 11.3 Silberschatz, Galvin, and Gagne 1999

File Attributes

- **Name** – only information kept in human-readable form.
- **Type** – needed for systems that support different types.
- **Location** – pointer to file location on device.
- **Size** – current file size.
- **Protection** – controls who can do reading, writing, executing.
- **Time, date, and user identification** – data for protection, security, and usage monitoring.
- Information about files are kept in the directory structure, which is maintained on the disk.

Applied Operating System Concepts 11.4 Silberschatz, Galvin, and Gagne 1999

File Operations

- create
- write
- read
- reposition within file – file seek
- delete
- truncate
- **open(F_i)** – search the directory structure on disk for entry F_i , and move the content of entry to memory.
- **close (F_i)** – move the content of entry F_i in memory to directory structure on disk.

Applied Operating System Concepts 11.5 Silberschatz, Galvin, and Gagne 1999

File Types – name, extension

File Type	Usual extension	Function
Executable	exe, com, bin or none	ready-to-run machine-language program
Object	obj, o	compiled, machine language, not linked
Source code	c, p, pas, 177, asm, a	source code in various languages
Batch	bat, sh	commands to the command interpreter
Text	txt, doc	textual data documents
Word processor	wp, tex, rtf, etc.	various word-processor formats
Library	lib, a	libraries of routines
Print or view	ps, dvi, gif	ASCII or binary file
Archive	arc, zip, tar	related files grouped into one file, sometimes compressed.

Applied Operating System Concepts 11.6 Silberschatz, Galvin, and Gagne 1999

Access Methods

- Sequential Access
 - read next*
 - write next*
 - reset*
 - no read after last write*
(rewrite)
- Direct Access
 - read n*
 - write n*
 - position to n*
 - read next*
 - write next*
 - rewrite n*

n = relative block number

Applied Operating System Concepts 11.7 Silberschatz, Galvin, and Gagne 1999

Directory Structure

- A collection of nodes containing information about all files.

The diagram illustrates a directory structure. At the top, a cloud-like shape labeled 'Directory' contains five circles representing nodes. Arrows point from each of these nodes down to a corresponding rectangular box labeled 'Files'. The boxes are labeled 'F 1', 'F 2', 'F 3', 'F 4', and 'F n' from left to right.

- Both the directory structure and the files reside on disk.
- Backups of these two structures are kept on tapes.

Applied Operating System Concepts 11.8 Silberschatz, Galvin, and Gagne 1999

Information in a Device Directory

- Name
- Type
- Address
- Current length
- Maximum length
- Date last accessed (for archival)
- Date last updated (for dump)
- Owner ID (who pays)
- Protection information (discuss later)

Applied Operating System Concepts 11.9 Silberschatz, Galvin, and Gagne 1999

Operations Performed on Directory

- Search for a file
- Create a file
- Delete a file
- List a directory
- Rename a file
- Traverse the file system

Applied Operating System Concepts 11.10 Silberschatz, Galvin, and Gagne 1999

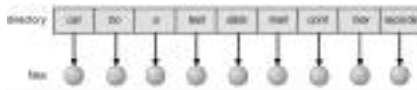
Organize the Directory (Logically) to Obtain

- Efficiency – locating a file quickly.
- Naming – convenient to users.
 - Two users can have same name for different files.
 - The same file can have several different names.
- Grouping – logical grouping of files by properties, (e.g., all Pascal programs, all games, ...)

Applied Operating System Concepts 11.11 Silberschatz, Galvin, and Gagne 1999

Single-Level Directory

- A single directory for all users.

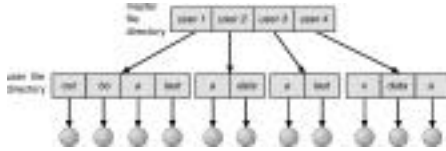


- Naming problem
- Grouping problem

Applied Operating System Concepts 11.12 Silberschatz, Galvin, and Gagne 1999

Two-Level Directory

- Separate directory for each user.



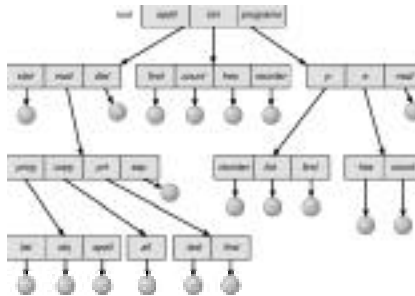
- Path name
- Can have the same file name for different user
- Efficient searching
- No grouping capability

Applied Operating System Concepts

11.13

Silberschatz, Galvin, and Gagne 1999

Tree-Structured Directories



Applied Operating System Concepts

11.14

Silberschatz, Galvin, and Gagne 1999

Tree-Structured Directories (Cont.)

- Efficient searching
- Grouping Capability
- Current directory (working directory)
 - `cd /spell/mail/prog`
 - `type list`

Applied Operating System Concepts

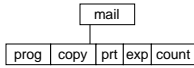
11.15

Silberschatz, Galvin, and Gagne 1999

Tree-Structured Directories (Cont.)

- Absolute or relative path name
- Creating a new file is done in current directory.
- Delete a file
`rm <file-name>`
- Creating a new subdirectory is done in current directory.
`mkdir <dir-name>`

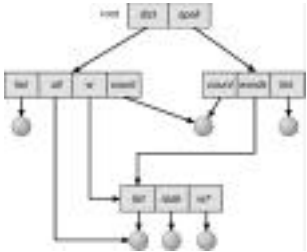
Example: if in current directory `/spell/mail`
`mkdir count`



- Deleting "mail" deleting the entire subtree rooted by "mail".

Acyclic-Graph Directories

- Have shared subdirectories and files.



Acyclic-Graph Directories (Cont.)

- Two different names (aliasing)
 - If `dict` deletes `list` dangling pointer.
- Solutions:
- Backpointers, so we can delete all pointers.
 - Variable size records a problem.
 - Backpointers using a daisy chain organization.
 - Entry-hold-count solution.

Access Lists and Groups

- Mode of access: read, write, execute
- Three classes of users

a) owner access	7	RWX
		1 1 1
b) groups access	6	RWX
		1 1 0
c) public access	1	RWX
		0 0 1
- Ask manager to create a group (unique name), say G, and add some users to the group.
- For a particular file (say *game*) or subdirectory, define an appropriate access.


```

owner  group  public
  \    /      /
   chmod 761 game

```
- Attach a group to a file


```

chgrp G game

```

Applied Operating System Concepts 11.22 Silberschatz, Galvin, and Gagne 1999

File-System Structure

- File structure
 - Logical storage unit
 - Collection of related information
- File system resides on secondary storage (disks).
- File system organized into layers.
- *File control block* – storage structure consisting of information about a file.

Applied Operating System Concepts 11.23 Silberschatz, Galvin, and Gagne 1999

Contiguous Allocation

- Each file occupies a set of contiguous blocks on the disk.
- Simple – only starting location (block #) and length (number of blocks) are required.
- Random access.
- Wasteful of space (dynamic storage-allocation problem).
- Files cannot grow.
- Mapping from logical to physical.


```

      Q
     /
LA/512
     \
      R

```

 - Block to be accessed = ! + starting address
 - Displacement into block = R

Applied Operating System Concepts 11.24 Silberschatz, Galvin, and Gagne 1999

Linked Allocation

- Each file is a linked list of disk blocks: blocks may be scattered anywhere on the disk.

block =

pointer

Applied Operating System Concepts11.25Silberschatz, Galvin, and Gagne 1999

- Allocate as needed, link together; e.g., file starts at block 9

Applied Operating System Concepts11.26Silberschatz, Galvin, and Gagne 1999

Linked Allocation (Cont.)

- Simple – need only starting address
- Free-space management system – no waste of space
- No random access
- Mapping

$$LA/511 \begin{cases} Q \\ R \end{cases}$$

- Block to be accessed is the Qth block in the linked chain of blocks representing the file.
- Displacement into block = $R + 1$
- File-allocation table (FAT)* – disk-space allocation used by MS-DOS and OS/2.

Applied Operating System Concepts11.27Silberschatz, Galvin, and Gagne 1999

Indexed Allocation

- Brings all pointers together into the *index block*.
- Logical view.

index table

Applied Operating System Concepts 11.28 Silberschatz, Galvin, and Gagne 1999

Example of Indexed Allocation

Applied Operating System Concepts 11.29 Silberschatz, Galvin, and Gagne 1999

Indexed Allocation (Cont.)

- Need index table
- Random access
- Dynamic access without external fragmentation, but have overhead of index block.
- Mapping from logical to physical in a file of maximum size of 256K words and block size of 512 words. We need only 1 block for index table.

$$LA/512 \begin{cases} Q \\ R \end{cases}$$

- Q = displacement into index table
- R = displacement into block

Applied Operating System Concepts 11.30 Silberschatz, Galvin, and Gagne 1999

Indexed Allocation – Mapping (Cont.)

- Mapping from logical to physical in a file of unbounded length (block size of 512 words).
- Linked scheme – Link blocks of index table (no limit on size).

$$LA / (512 \times 511) \begin{cases} Q_1 \\ R_1 \end{cases}$$

- Q_1 = block of index table
- R_1 is used as follows:

$$R_1 / 512 \begin{cases} Q_2 \\ R_2 \end{cases}$$

- Q_2 = displacement into block of index table
- R_2 displacement into block of file:

Applied Operating System Concepts

11.31

Silberschatz, Galvin, and Gagne 1999

Indexed Allocation – Mapping (Cont.)

- Two-level index (maximum file size is 512^2)

$$LA / (512 \times 512) \begin{cases} Q_1 \\ R_1 \end{cases}$$

- Q_1 = displacement into outer-index
- R_1 is used as follows:

$$R_1 / 512 \begin{cases} Q_2 \\ R_2 \end{cases}$$

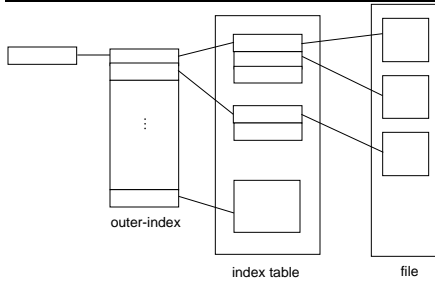
- Q_2 = displacement into block of index table
- R_2 displacement into block of file:

Applied Operating System Concepts

11.32

Silberschatz, Galvin, and Gagne 1999

Indexed Allocation – Mapping (Cont.)



Applied Operating System Concepts

11.33

Silberschatz, Galvin, and Gagne 1999

Free-Space Management (Cont.)

- Need to protect:
 - Pointer to free list
 - Bit map
 - * Must be kept on disk
 - * Copy in memory and disk may differ.
 - * Cannot allow for block[*j*] to have a situation where bit[*j*] = 1 in memory and bit[*j*] = 0 on disk.
 - Solution:
 - * Set bit[*j*] = 1 in disk.
 - * Allocate block[*j*]
 - * Set bit[*j*] = 1 in memory

Applied Operating System Concepts 11.37 Silberschatz, Galvin, and Gagne 1999

Directory Implementation

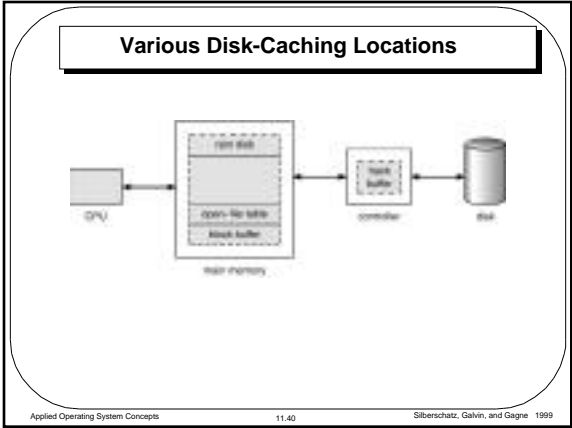
- Linear list of file names with pointer to the data blocks.
 - simple to program
 - time-consuming to execute
- Hash Table – linear list with hash data structure.
 - decreases directory search time
 - *collisions* – situations where two file names hash to the same location
 - fixed size

Applied Operating System Concepts 11.38 Silberschatz, Galvin, and Gagne 1999

Efficiency and Performance

- Efficiency dependent on:
 - disk allocation and directory algorithms
 - types of data kept in file's directory entry
- Performance
 - disk cache – separate section of main memory for frequently used blocks
 - free-behind and read-ahead – techniques to optimize sequential access
 - improve PC performance by dedicating section of memory as virtual disk, or RAM disk.

Applied Operating System Concepts 11.39 Silberschatz, Galvin, and Gagne 1999



Recovery

- Consistency checker – compares data in directory structure with data blocks on disk, and tries to fix inconsistencies.
- Use system programs to *back up* data from disk to another storage device (floppy disk, magnetic tape).
- Recover lost file or disk by *restoring* data from backup.

Applied Operating System Concepts 11.41 Silberschatz, Galvin, and Gagne 1999
