UNISA Chatter – Operating System Concepts: Part 11 … File System Concepts

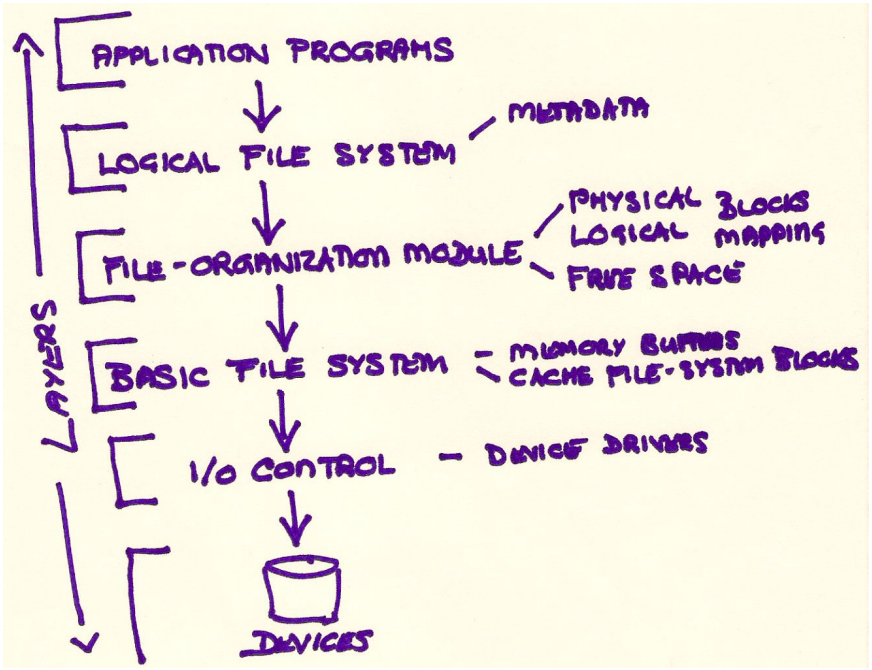
Willy-P. Schaub **13 Jun 2010 6:33 PM** | 0

See UNISA – Summary of 2010 Posts for a list of related UNISA posts. **If you are not studying at UNISA you should probably give this materials summary post a miss :)**

Last time we looked at the file system interface… today we teleport ourselves back into the operating system to explore some of the key concepts.

File System Structure

File systems are typically implemented in a modular or layered structure, which creates logical boundaries.

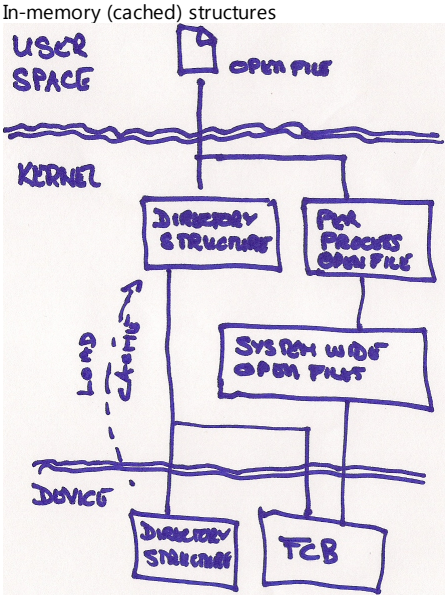


- The lowest level is comprised of the **devices**, such as hard drive, tapes and CD-ROM devices … the hardware or iron layer.
- The I/O control layer consists of **device drivers** and **interrupt handlers** which manage the bridge between the main memory and the devices.
- The basic file system issues **generic commands** to the device drivers to read and write physical blocks of data from the devices. In addition the layer is also responsible for the **caches** that hold the various file system blocks.
- The file organization module is aware of files and their **logical** and **physical blocks**. The layer maps or translates logical to physical and vice versa, as well as tracking **unallocated blocks**.
- Bit Vector is a free-space list, implemented as a bit map or bit vector, which tracks "free" blocks.
- Finally, the logical file system is the interface of the application and manages the **metadata information**, such as file system structures and **file control blocks** (FCB).

File System Puzzles

- **Boot control block** … contains information needed by the operating system to boot.
  - Boot block in UFS
  - Partition boot sector in NTFS
- **Volume control block** … contains volume information such as size of blocks, number of blocks.
  - Superblock in UFS
  - Master file table (MFT) in NTFS
- **Directory structure** … used to organize files.
  - Includes file names and inode numbers on UFS.
  - Included in MFT in NTFS.

- In-memory (cached) structures



  - **Mount table** … contains information about mounted volumes.
  - **System wide open file table** … contains a copy of the file control block (FCB) of each open file.
  - **Per process open-file table** … contains pointer to system tables for open files for the associated process.

Directory Puzzles

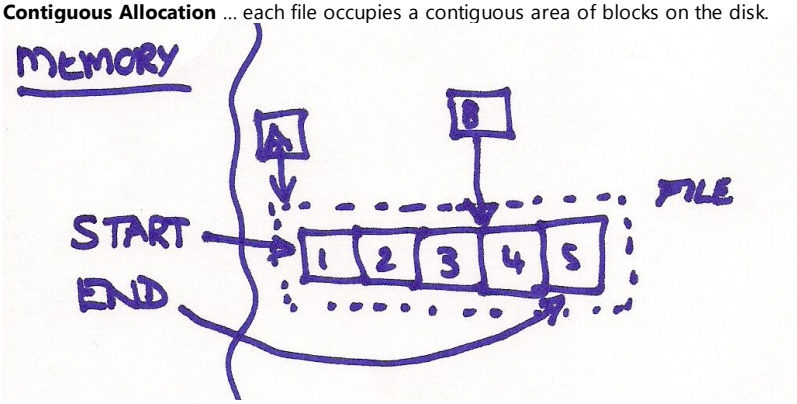
Two of the possible directory allocation and directory management algorithms are:

- **Linear list** implementation … linear list of file names with pointers to data blocks.
  - Advantage
    - Simple
  - Disadvantage
    - Finding an entry requires a linear (slow) search
    - Using sorted or tree data structures can improve performance
- **Hash table** implementation … hash table is used for the data structure.
  - Advantage
    - Quicker to search
  - Disadvantage
    - Need to provision for hash table collisions
    - Hash tables are generally fixed size

Allocation Strategies

lastly, let's summarise some of the main allocation algorithms:

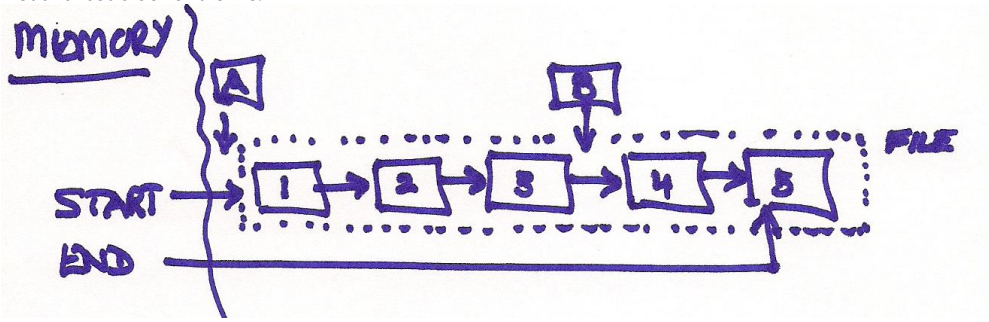
- **Contiguous Allocation** … each file occupies a contiguous area of blocks on the disk.



  - Advantages
    - Easy and quick access
    - Great if files are accessed sequentially and files are small
  - Disadvantages
    - Finding space for a new file
    - External fragmentation
  - Inserting a block at the beginning
    - 1x write of the new block
    - 5x reads of the existing file
    - 5x writes to the new file
    --> 11x I/O operations in total

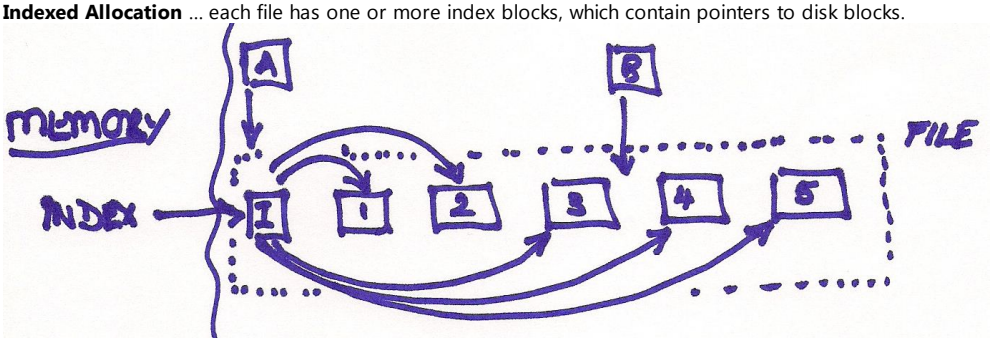
- Inserting a block in the middle (after 3) … assuming we have enough space after the file
  - Go to 4th location, where new block will be inserted
  - 2x reads of the existing file
  - 1x write of new block
  - 2x writes to the new file
  --> 5x I/O operations in total

- **Linked Allocation** … each file consists of a linked list of disk blocks. The directory contains a pointer to the first and last block of the file.



  - Advantages
    - File size need not be known up front
    - Finding space for new or growing files is easier
    - Great for large file that are accessed sequentially
  - Disadvantages
    - Effective only for sequential access files
    - Pointers require space, which result in lost data space area
    - Reliability … loose one pointer and file is corrupt
  - Inserting a block at the beginning
    - 1x write of the new block, with pointer to original first block
    - Update directory information to point to new block
    --> 1x I/O operations in total
  - Inserting a block in the middle (after 3)
    - 3x reads of the existing block
    - 1x write of new block, with pointer to block 4
    - 1x write of block 3 with new pointer to new block
    --> 5x I/O operations in total

- **Indexed Allocation** … each file has one or more index blocks, which contain pointers to disk blocks.



  - Advantages
    - No excessive external fragmentation
    - File size need not be known up front
    - Finding space for new or growing files is easier
    - Great for files are are large and accessed randomly
  - Disadvantages
    - Pointer overhead, as with linked allocation
  - Inserting a block at the beginning
    - 1x write of new block
    - Update index block in memory
    --> 1x I/O operations in total
  - Inserting a block in the middle (after 3) - 1x write of new block
    - Update index block in memory
    --> 1x I/O operations in total

- **Linked scheme** … linked index blocks, in other words, it is indexed allocation with two or more index blocks.
- **Multi-Level Index** … first index block points to multiple index blocks.
  - Maximum size of files = (block size / pointer size) * blocksize
  - Example:
    - 1 block = 8192, 1 pointer = 4 bytes and using two level index:
      --> (8192/4) = 2048 index entries
      --> 2 level = 2048 * 2048 = 4,194,304
      --> 4,194,304 * 8192 = 34.4 GB

Other Notes

- Recovery
  - Consistency checker … system program (chkdsk) that validates and repairs disk structures inconsistencies
  - Full backup … backup of all files from disk
  - Incremental backup … backup of all files that have changed
- Mounted File Systems
  - Mounting the same file system in multiple locations could result in multiple paths to the same file … confusing
- VFS Layer
  - Virtual file system
  - Introduces a layer of indirection in the file system, separating file system generic operations from the actual implementation.

The next batch of explorations will focus on security and "protection" … may the lock-down begin.

Blog - Links