Filesystem Document

陳昭成

I wrote my own simple file system on top of a file (virtual disk file)

   - The file = a virtual disk

  -The whole structure of the file system should exist within the virtual disk file.

  -All the storing and removing file operations ONLY affect the file.

  -The file system is able to store binary files, not only text files.

FILE SYSTEM INTERFACE:

Creating, destroying, and accessing the file system

- int myfs_create(const char *filesystemname, int max_size);

- int myfs_destroy(const char * filesystemname);

Provides the following set of functions (API) to allow users to manage files on the virtual disk file

- int myfs_file_open(const char *filename);

- int myfs_file_close(int fd);

- int myfs_file_create(const char * filename);

- int myfs_file_delete(const char * filename);

- int myfs_file_read(int fd, char *buf, int count);

- int myfs_file_write(int fd, char *buf, int count);

What I maintain:

1. FAT:

   - An int array , size=max_size/block size(real storage space) and is initialized to 0 in each column.

   - If the block is vacant, put 0 in the column.

   - If the block is full , put the next block number in the

column.

- If it is the last block of a file and it's occupied but not full, just record -1 !

- Function:

    To record the free blocks and record where the next block of a file is by putting the next block number in the column.

[0]block0	[1]block1	[2]block2	[3]	[4]	[5]

1. Dir table:

    - Dir_struct* type array,

    - Struct member:

    Typedef struct{

        1. filename

        2. location of the first block FAT

        3. location of the last block FAT

        4. file id (fd)=its index of the directory array.

}Dir_struct

- Each Dir_struct*pointer dynamically points to a Dir_struct object when the corresponding file is created.

- Array Size =The maximum allowed number of file in virtual machine.

- Function:用來存各 file 的相關資料,如:檔案名稱,第一個 block 的位置等等,當需要對某個 file 做 read/write/delete 時都需參照此表,才能順利完成.

Att of F0(fd=0)	Att of F1 (fd=1) Is being accessed	Att of F2 (fd=2)	…………………

3.Disk_Buffer:

- An array of type char*.

- Each char*pointer dynamically points to a block-sized character string when the corresponding file is created.

- Array Size=Size of Dir array=The maximum allowed number of file in virtual machine.

- Function:

(1)用來暫存要寫進 file 的內容,當 buffer 裝滿
或 write 結束時,會將其內容寫回 virtual disk 儲存.

(2)當作 read 時,從 virtual disk 中該 file 的第一個 block
的內容放入 buffer 供使用者讀取,若使用者要求的讀取長
度超過該 block 的內容長度,就在將該 file 下一個 block 的
內容讀進 buffer,以此類推.

Disk_Buffer of F0(fd=0)	Disk_Buffer of F1 (fd=1)	Disk_Buffer of F2 (fd=2)	………….

When call create():

　Go through FAT to find a free block for the file.

　Go through Dir table to find a NULL Dir_struct poiner ,and
make that pointer point to a Dir_struct type object to record
the info of the file.

　　Go through Disk_Buffer to find a NULL pointer,and make
that pointer point to a dynamically created char array to act as

the buffer for read and write for the file.

When delete() , check the directory table to get the first block's id and go to that block to get the next id and set the FAT column to 0 according to the traversed id until the next id is -1(reach the last block of this file)!

Virtual Disk overview:

Update FAT and Directory to virtual disk whenever close() and delete() is called!

In Mem

FAT   Dir table   Disk_Buf
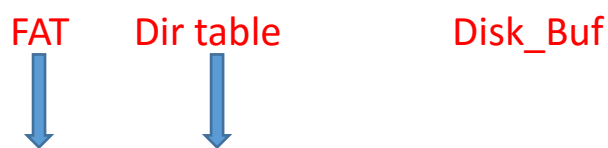
In Virtual Disk

FAT	Dir table	File storage

存了兩個檔名為"ABC"及"DEF"的檔案後,

Dir table 的内容:

```
Dir0's Filename :ABC
0 is Dir0's First Block
6 is Dir0's Last Block
0 is Dir0's ID
Dir1's Filename :DEF
1 is Dir1's First Block
7 is Dir1's Last Block
1 is Dir1's ID
```

FAT 的内容:

```
Read FAT content from VD:  2 7 3 4 5 6 -1 -1 0 0 0 0 0 0 0 0 0 0 0 0
```

The last block of a file has -1 in its next_block_id!