STI 1ère année – Programmation Système

TD 3: Système de fichier

J.-F. Lalande
inspiré des excellents TDs de Yann Busnel

1 Système de fichiers

Voici ci dessous un système de fichier simplifié proposé par Yann Busnel. L'idée de ce TD est d'analyser ce système de fichier et de recoder les fonctions d'accès simples.

1.1 Description

Chaque bloc de données fait 1024 octets. Si les données a ranger ne remplissent pas un bloc entier, un caractère spécial pourra servir de délimiteur. Le numéro d'i-noeud du répertoire racine / est 0. Dans chaque i-noeud, on a représenté seulement le type de fichier, la taille du fichier, le nombre de liens physiques sur ce fichier et la liste des numéros des blocs de données contenant les données du fichier (on a en général plusieurs blocs, car le fichier dépasse 1024 octets). Les blocs de données des répertoires contiennent des paires (nom du fichier, numéro d'i-noeud). Les blocs (ou parties de blocs) non utilisées contiennent des valeurs inconnues.

Bitmap des i-noeuds :

1	1	1	1	0	1	0	1	0	0

Bitmap des blocs de données :

1	1	0	1	1	1	1	1	0	0

Table des i-noeuds :

No d'i-noeud	0	1	2	3	4	5	6	7	8	9
Type	rep.	rep.	fichier	rep.		fichier		fichier		
Taille	1024	1024	1900	1024		16		20		
Nb liens	4	2	1	2		1		1		
Num bloc	0	1	3, 5	4		7		6		

Blocs de données :

0	1	2	3	4	5	6	7	8	9
. 0	. 1		aaaaa	. 3	bbbbb	12345	bonjour		
.. 0	.. 0		aaaaa	.. 0	bbbbb	67890	bonjour		
toto 1	a.txt 2		aaaaa	c.txt 7	bbb	67890			
tata 3	b.txt 5		aaaaa						

Exercice 1 Dessinez l'arborescence correspondant à ce système de fichier. Quelle place est disponible (en octets) pour stocker les données de nouveaux fichiers ? Quelle est la place perdue (en octets) du fait de la fragmentation interne ?

Exercice 2 Quel est le contenu de chacun des fichiers ?

Exercice 3 Combien de blocs faut il lire sur le disque (et lesquels) pour connaitre le numero d'i-noeud du fichier a.txt ?

Exercice 4 Pour chacune des séquences de commandes ci-dessous, vous indiquerez ce qui a changé dans le système de fichier par rapport à son état initial (on suppose que le système est dans l'état initial avant l'exécution de chaque séquence de commandes).

1. *mv /toto/b.txt /tata*
2. *mv /toto /tata*
3. *rm /toto/b.txt*
4. *cp /toto/a.txt /tata*
5. *ln −s /toto/b.txt /toto/bb.txt*
6. *cat /tata/c.txt >> toto/a.txt*

1.2 Implémentation

L'objet de cette section est de tenter d'implémenter un système de fichier directement en mémoire (on évite, pour simplifier le passage par le disque). L'idée est de reproduire le fonctionnement de la table des i-noeuds et des blocs de données lorsque l'on souhaite faire des opérations basiques sur des fichiers (imprimer, déplacer, copier, etc...).

Vous trouverez dans le listing ci-dessous une implémentation du système de fichier décrit précédemment (à récupérer sur le serveur enseignement).

Listing 1 – fs.h

```
#define TMAX 100

typedef struct inode {
   int type; // Type du fichier: 0=rep 1=fichier
   int taille; // Taille en octets
   int nb_liens; // Nombre de lien
   int * num_bloc; // Table des numéros de blocs
} inode;

int bitmap_inode[10];
int bitmap_blocs[10];
char blocs[10][TMAX];
inode table[10];
```

Listing 2 – fs.c

```
#include "stdlib.h"
#include "string.h"
#include "stdio.h"
#include "fs.h"

int main() {
   int i;
   int bitmap_inode_tmp[10] = {1,1,1,1,0,1,0,1,0,0};
   int bitmap_blocs_tmp[10] = {1,1,0,1,1,1,1,1,0,0};

   /* Début du modèle */

   // Recopie des deux tables de bitmaps
   for (i=0; i<10; i++) {
      bitmap_inode[i] = bitmap_inode_tmp[i];
      bitmap_blocs[i] = bitmap_blocs_tmp[i];
   }

   // Remplissage de la table des i-noeuds
   table[0].type = 0; table[1].type = 0;
   table[2].type = 1; table[3].type = 0;
   table[5].type = 1; table[7].type = 1;
   table[0].taille = TMAX;
   table[1].taille = TMAX;
   table[2].taille = TMAX+5;
   table[3].taille = TMAX;
   table[5].taille = 16;
   table[7].taille = 20;
   table[0].nb_liens = 0; table[1].nb_liens = 1;
   table[2].nb_liens = 1; table[3].nb_liens = 2;
   table[5].nb_liens = 1; table[7].nb_liens = 1;
   table[0].num_bloc = (int *)calloc(2, sizeof(int));
   *(table[0].num_bloc) = 1;
   *(table[0].num_bloc+1) = 0;
   table[1].num_bloc = (int *)calloc(2, sizeof(int));
   *(table[1].num_bloc) = 1;
   *(table[1].num_bloc+1) = 1;
   table[2].num_bloc = (int *)calloc(3, sizeof(int));
   *(table[2].num_bloc) = 2;
   *(table[2].num_bloc+1) = 3;
   *(table[2].num_bloc+2) = 5;
   table[3].num_bloc = (int *)calloc(2, sizeof(int));
   *(table[3].num_bloc) = 1;
   *(table[3].num_bloc+1) = 4;
   table[5].num_bloc = (int *)calloc(2, sizeof(int));
   *(table[5].num_bloc) = 1;
   *(table[5].num_bloc+1) = 7;
   table[7].num_bloc = (int *)calloc(2, sizeof(int));
   *(table[7].num_bloc) = 1;
   *(table[7].num_bloc+1) = 6;

   // Remplissage des données des blocs
   strcpy(blocs[0],". 0\n.. 0\ntoto 1\ntata 3\n");
   strcpy(blocs[1],". 1\n.. 0\na.txt 2\nb.txt 5\n");
   for (i=0; i<TMAX-1; i++) {
      strcpy(blocs[3]+i,"a");
   }
   strcpy(blocs[4],". 3\n.. 0\nc.txt 7\n");
   for (i=0; i<20; i++) {
      strcpy(blocs[5]+i,"b");
   }
   strcpy(blocs[6],"123456789067890\n");
   strcpy(blocs[7],"bonjour\nbonjour\n");

   /* Fin du modèle */

   // Fonctions à implémenter:

   // Ecrit le contenu du fichier de l'i-noeud 2 à l'
         écran
   print(2);
   printf("\n");

   // Recupérer un numéro d'i-noeud pour un nom
         de fichier
   // dans un répertoire dont on donne le numéro d'i
         -noeud
   int numero_inoeud = get_num_inode(1, "a.txt");
   printf("Numéro de l'i-noeud pour a.txt dans le
         répertoire d'i-noeud 1: %i\n",
         numero_inoeud);
}
```

Exercice 5 Etudiez ce code. Comment est implémenté la liste des numéros de blocs pour un i-noeud dont le fichier est scindé en plusieurs blocs ? (par exemple, l'i-node 2).

Exercice 6 Codez la fonction ***void*** *print(****int*** *num_inoeud* qui imprime à l'écran le fichier de l'i-noeud numéro *num_inoeud*.

Exercice 7 Codez la fonction ***int*** *get_num_inode(****int*** *num_inode_root,* ***char*** *∗ name)* qui renvoie le numéro de l'i-noeud pour le fichier "name" qui se trouve dans le répertoire dont l'i-noeud est *num_inode*.

Exercice 8 Les courageux pourront réfléchir à des fonctions de déplacement (mv), copie (cp), etc...