# TD n°1 - Processus-Fichiers

## Exercice 1 – Fork loop

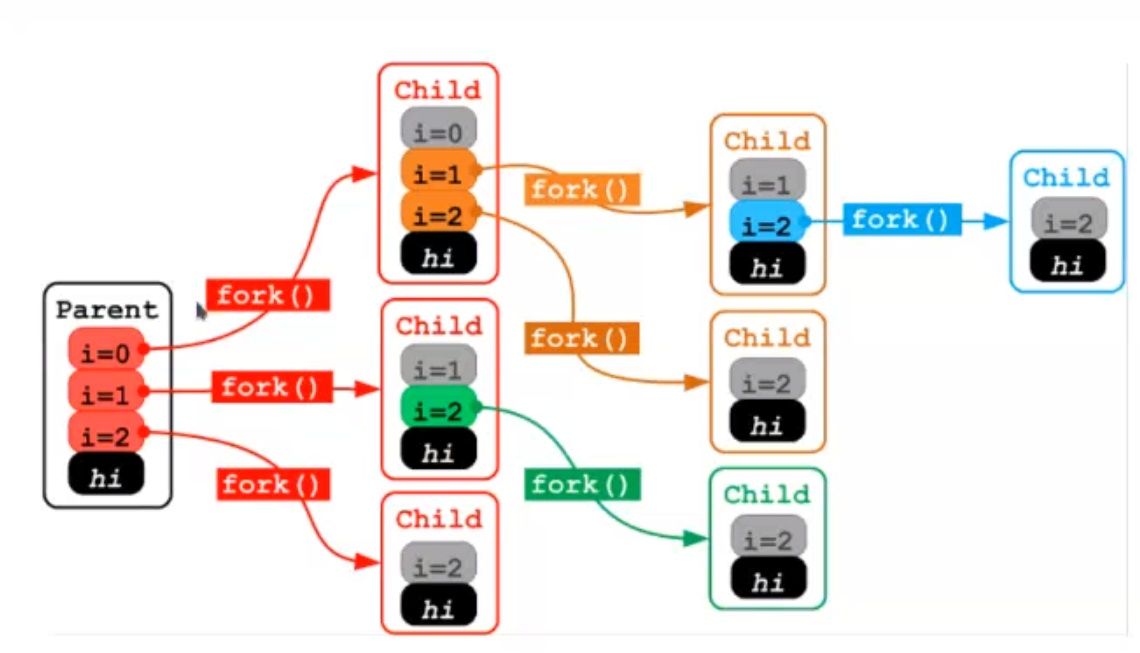
#include <stdio.h>

void main() {  
int i;  
int n=3;  
for(i=0; i<n; i++) {  
 fork();  
 printf(“[%d] [%d] i=%d\n”, getppid(), getpid(), i);  
 }

printf(“[%d] [%d] hi\n”, getppid(), getpid());

}

Il y aura 3 fils car 3 itérations.

Il y aura 3 petits fils.

2^n-1 fils et 2^n processus au total.

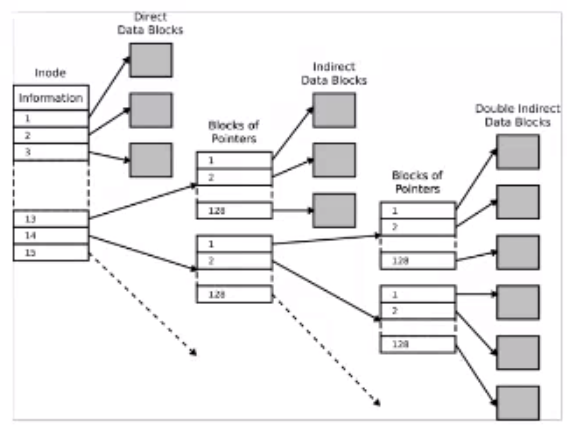
## Exercice 2 – I-Node

1. Combien y a-t-il de pointeurs (de blocs) dans la structure de l’i-node ?  
   15 pointeurs. Car 12+1+1+1 = 15 pointeurs.  
   12 pointeurs directs, 1 pointeur indirect simple, 1 pointeur indirect double et 1 pointeur indirect triple.

### Combien d’adresses peut contenir un bloc de données ? b/4 = 512/4 = 128 adresses.

### Quelle est la taille maximale des fichiers adressables par ext2 ?

b = 0.5 KiO(512 octets)



12\*b+b/4\*b+(b/4)²\*b+(b/4)^3\*b = (12+b/4+(b/4)^2+(b/4)^3)\*b = 1Gio

### Comment augmenter cette taille ?

En augmentant b par exemple = 1024.

### e) On désire créer un fichier contenant un total de 20.000.000 (vingt millions) de caractères (caractères de fin de ligne et de fin de fichier compris) :

### - Quelle est la fragmentation interne totale sur le disque résultant de la création de ce fichier ?

Nombre de blocs : 20.10^6/512 # 39062.5 soit 39062 blocs plein + 1 partiellement rempli au total (256 octets)

20E6 – 39062\*512 = 256 octets utilisés dans le dernier bloc.  
512 – 256 = 256 octets perdus par fragmentation interne.

### - Quelle est la fragmentation externe ?

Pas possible d’y répondre…

### f) Quelle est la proportion de l'espace qui est utilisé pour les données (pointeurs…) par rapport à l'ensemble des blocs destinés au fichier dans le cas d'un fichier de 71 Kio ? On considérera que l'i-node occupe un bloc.

Le nombre de blocs de données nécessaires pour stocker le fichier est 71 Kio / 0,5 Kio = 142 blocs.  
Inode = 1 bloc qui contient 12 adresses.   
Reste 142-12 = 130 blocs.   
Au niveau 1 : 128 blocs.  
Au niveau 2 : 130 – 128 = 2 blocs.

Inode : 1 bloc  
Niveau 1 : 1 bloc  
Niveau 2 : 2 blocs  
Au total : 4 blocs  
Le fichier occupe 142 blocs de données + 4 blocs de « structure de données » = 146.   
Les données occupent donc 142/146 = 97%

### g) Calculez le temps moyen d'accès à chacun des blocs de données pour un fichier de taille maximale. On suppose que les accès sont aléatoires, que le système n’utilise aucun cache des blocs, et que la lecture d'un bloc nécessite un délai d = 6 ms. Nombre maximal de blocs = 2113676 Le temps d’accès pour les blocs directs nécessitent la lecture de l’inode puis le bloc de données = (2\*d). Le premier niveau indirect consiste en l’inode, le bloc indirect puis le bloc de données = (3\*d). Le second niveau indirect consiste à un accès de plus que le précédent = (4\*d). Le troisième niveau = (5\*d).

2\*d\*12  
12 = nombre de blocs  
12 millisecondes pour accéder aux octets de ces blocs.

2\*d\*12+3\*d\*128+4\*d\*128\*128