Séance 2
Entrées-Sorties – FIFO

Les fichiers : appels système

Les fichiers : open

Les fichiers : descripteur de fichier

Les fichiers : read

Les fichiers : write

Les fichiers : close

FIFO : création

FIFO : communication

FIFO : comportement de open

Les fichiers : appels système

Fonctions bibliothèque standard C : fopen, fread, fwrite, fseek, fclose.
Appels système : open, read, write, lseek, close.

Fonctions bibliothèque standard :

- standards (paramètres, comportement)
- plus faciles à utiliser
- utilisent un tampon mémoire

Appels système :

- modèle universel, utilisés aussi pour mécanismes de communication

Les fichiers : open

open (chemin du fichier, mode d'ouverture, droits d'accès) ouverture / création d'un fichier

chemin du fichier (char *) : ex. *"fic1", "../fic2", "/home/…/fic3"*

mode d'ouverture (entier) : conjonction d'indicateurs avec | (OU)
- O_RDONLY : mode lecture seule
- O_WRONLY : mode écriture seule
- O_RDWR : mode lecture et écriture
- O_CREAT : créer le fichier s'il n'existe pas
- O_TRUNC : vider le fichier s'il existe
- O_APPEND : ajout en fin de fichier
- etc

```
// ouverture pour écritures seules,
// créer fichier si n'existe pas, vider si existe
open (…, O_WRONLY |O_CREAT |O_TRUNC, …);
```

droits d'accès (entier) : droits à donner au fichier si création
on peut l'omettre si pas création, ex. open (…, O_RDONLY);

```
open("fic", …,  0640);      // valeur octale : 0640
=>  rw- r-- ---     fic
```

Les fichiers : descripteur de fichier

int open (…);

retour : descripteur de fichier, un entier >= 0 identifiant le fichier, à utiliser pour les lectures, écritures, …
-1 si échec

```
int fd;
fd = open (…);
read (fd, …);
```

Descripteur de fichier :

- a un sens local au processus qui fait open
- est valable seulement pendant la session open/close

Les fichiers : read

*int read (int desc, void *data, int taille)* lecture de données dans un fichier

desc : descripteur du fichier
data : adresse où ranger les données lues
taille : taille à lire, en octets

```
int val;
read (fd, &val, sizeof(val));

float tab[100];
read (fd, tab, 100 * sizeof(float));
```

retour : nb d'octets lus ou 0 si fin de fichier
-1 si échec

rem : taille lue peut être < taille demandée (fin de fichier atteinte)

Les fichiers : write

*int write (int desc, void *data, int taille)*

desc : descripteur du fichier
data : adresse des données à écrire
taille : taille des données, en octets

retour : nb d'octets écrits
-1 si échec

écriture de données dans un fichier

```
int val = 45;
write (fd, &val, sizeof(val));

float tab[] = { 3.4, … };
write (fd, tab, sizeof(tab));
```

Les fichiers : close

int close (int desc) fermeture d'un fichier

desc : descripteur du fichier

retour : 0 ou -1 si échec

FIFO : création

FIFO (ou tube nommé) : un fichier, mais qui a le comportement d'un mécanisme de communication

Création :

```
mkfifo  –m 600  monfifo
```

par une commande

ou

```
mkfifo ("monfifo", 0600);
```

par une fonction

FIFO : communication

émetteur

```
…
fd = open ("monfifo", O_WRONLY);
…
write (fd, …);
…
```

écriture à la fin *lecture au début*

récepteur

```
…
fd = open ("monfifo", O_RDONLY);
…
read (fd, …);
…
```

read :

- bloquant si pas de donnée
- les données lues sont extraites

FIFO : comportement de open

Si open en mode lecture alors qu'aucun processus n'a ouvert en écriture ("pas d'écrivain potentiel") :
open bloque,
est débloqué dès qu'un processus ouvre en mode écriture ("un écrivain potentiel se manifeste").

Si open en mode écriture sans lecteur potentiel : open bloque, est débloqué dès qu'un lecteur potentiel se manifeste.