

## T.P. 5 : Verrous

**Exercice 1** On désire créer et gérer une liste du personnel d'une entreprise. Cette liste sera stockée dans un fichier Unix sous forme de liste de structure. La structure comprendra les informations suivantes :

- Numéro d'identification,
- nom
- prénom,
- date de naissance,
- nom du service,
- qualité,
- salaire.

Un client pourra mettre à jour, insérer, supprimer ou rechercher des informations dans cette liste.

Créer une fonction par opération demandée.

```
int maj(identifiant, nom-du-champ, valeur);
int insere(nom, prenom, date-de naissance, ...);
int supprime(identifiant);
int recherche(nom-du-champ, valeur);
int liste();
```

Les fonctions `insere` et `recherche` retournent un numéro d'identifiant. Les autres renvoient 0 si tout s'est bien passé, -1 sinon.

D'autre part, il sera possible de lancer plusieurs clients de façon concurrente. Vous devrez tenir compte de ce paramètre et gérer les problèmes d'accès concurrents à l'aide des verrous POSIX disponible sous Unix.

Vous utiliserez pour ce faire la fonction :

```
int fcntl(desc, commande, verrou);
```

où `verrou` est un pointeur sur une structure `struct flock`.

```
struct flock
{
    short int l_type; /* F_RDLCK, F_WRLCK ou F_UNLCK */
    short int l_whence; /* SEEK_SET, SEEK_CUR ou SEEK_END */
    off_t l_start; /* position relative du debut p.r.a l_whence */
    off_t l_len; /* longueur: 0 <==> jusqu'à fin de fichier */
    pid_t l_pid; /* pid du processus auquel appartient le verrou */
};
```

`commande` est parmi

**F\_SETLK** Demande de pose non bloquante de verrou.

**F\_SETLKW** Demande de pose bloquante de verrou. Si il existe déjà un verrou bloquant, le processus est endormi jusqu'à ce qu'il n'y est plus de verrou incompatible ou que l'appel soit interrompu.

**F\_GETLK** Test d'existence d'un verrou incompatible avec le verrou donné. S'il n'existe pas, le champ `l_type` de `verrou` vaut `F_UNLCK` sinon, `verrou` est rempli avec les informations sur ce verrou incompatible.