Développement système sur Linux
Echanges via une mémoire partagée

1 DESCRIPTION

L'objectif de ce projet est de développer un mécanisme d'échange de données entre deux applications LECTEUR et ECRIVAIN sous Linux en utilisant le mécanisme de « Shared Memory » en langage C. L'application LECTEUR est fournie, il faut développer l'application ECRIVAIN.

Le projet manipule les concepts de :
- Formats de données binaires
- Mémoire partagée.

Le projet doit être réalisé par binôme, le code source fera l'objet d'une évaluation.

Un rendu du code est demandé sur Campus sous forme d'archive. Seul le code source C devra être rendu (pas d'envoi de binaires ni des fichiers objets générés). L'archive devra être préfixée par le nom de chaque étudiant composant le groupe, chaque nom séparé par un '_' (ex : « Martin_Dupont_SHM.zip »).

2 REALISATION

Les développements reposent sur les éléments système vus en cours et sur les mécanismes de mémoire partagée dont les points d'entrées sont donnés dans l'encadré ci-après.

Le concept de mémoire partagée permet à plusieurs processus d'accéder à un même segment de mémoire. C'est un moyen simple et rapide pour échanger des données entre processus : les données sont simplement placées à un endroit accessible par les différents processus.

Les principales fonctions de manipulation de la mémoire partagée sont :

```
int shm_open(const char *name, int oflag, mode_t mode);
void *mmap(void *addr, size_t length, int prot, int flags, int fd, off_t offset);
int munmap(void *addr, size_t length);
int shm_unlink(const char *name);
```

L'édition de liens nécessite l'ajout de l'option « `-lrt` ».

Un soin particulier sera apporté à la libération des ressources.

2.1 LECTEUR & ECRIVAIN

Un LECTEUR de données en SHM est fourni sous forme de fichier exécutable ubuntu, compatible WSL sous Windows. Ce lecteur est capable de recevoir plusieurs types de données:
- Zero string
- Int Little Endian et Big Endian
- Short Little Endian et Big Endian
- Struct {short ; int ;} Little Endian

Il est demandé de développer un ECRIVAIN de données qui fournit chacun des types traités par le LECTEUR. La série de donnée fournies sera préférentiellement faite de manière interactive.

2.2 FORMAT DE LA SHM

La zone de mémoire SHM est organisée de la manière suivante :

- 1er octet = N° de code de ce qui suit (entre 1 et 7)
- Octets suivants = donnée proprement dite

Les types de données correspondant aux codes connus du LECTEUR sont les suivants :

1. Zero string
2. Int Little Endian
3. Int Big Endian
4. Short Little Endian
5. Short Big Endian
6. Struct {short ; int ;} Little Endian
7. Fin du LECTEUR

2.3 PROTOCOLE DES ECHANGES

La SHM est remise à zéro par le LECTEUR à chaque fois qu'il a traité une donnée, l'octet de code est donc à 0 dès l'initialisation et le LECTEUR surveille sa valeur à 1 Hz.

L'ECRIVAIN doit remplir les octets de donnée en fonction de ce qu'il veut envoyer, puis il positionne l'octet de code entre 1 et 6 pour signaler qu'il y a une donnée à lire et le type de cette donnée.

L'octet de code à 7 provoque la fin du LECTEUR

Le nom de la SHM à ouvrir est donné par le LECTEUR à son lancement, ainsi que les informations de protocole ci-dessus.

2.4 FORMAT D'AFFICHAGE DE L'ECRIVAIN

Le LECTEUR affiche les données numériques au format hexadécimal.

La donnée envoyée par l'ECRIVAIN doit être visualisée à l'identique dans le LECTEUR, quelque soit le format.