



Projet Système jeu de chevaux

18.10.2022

Par :

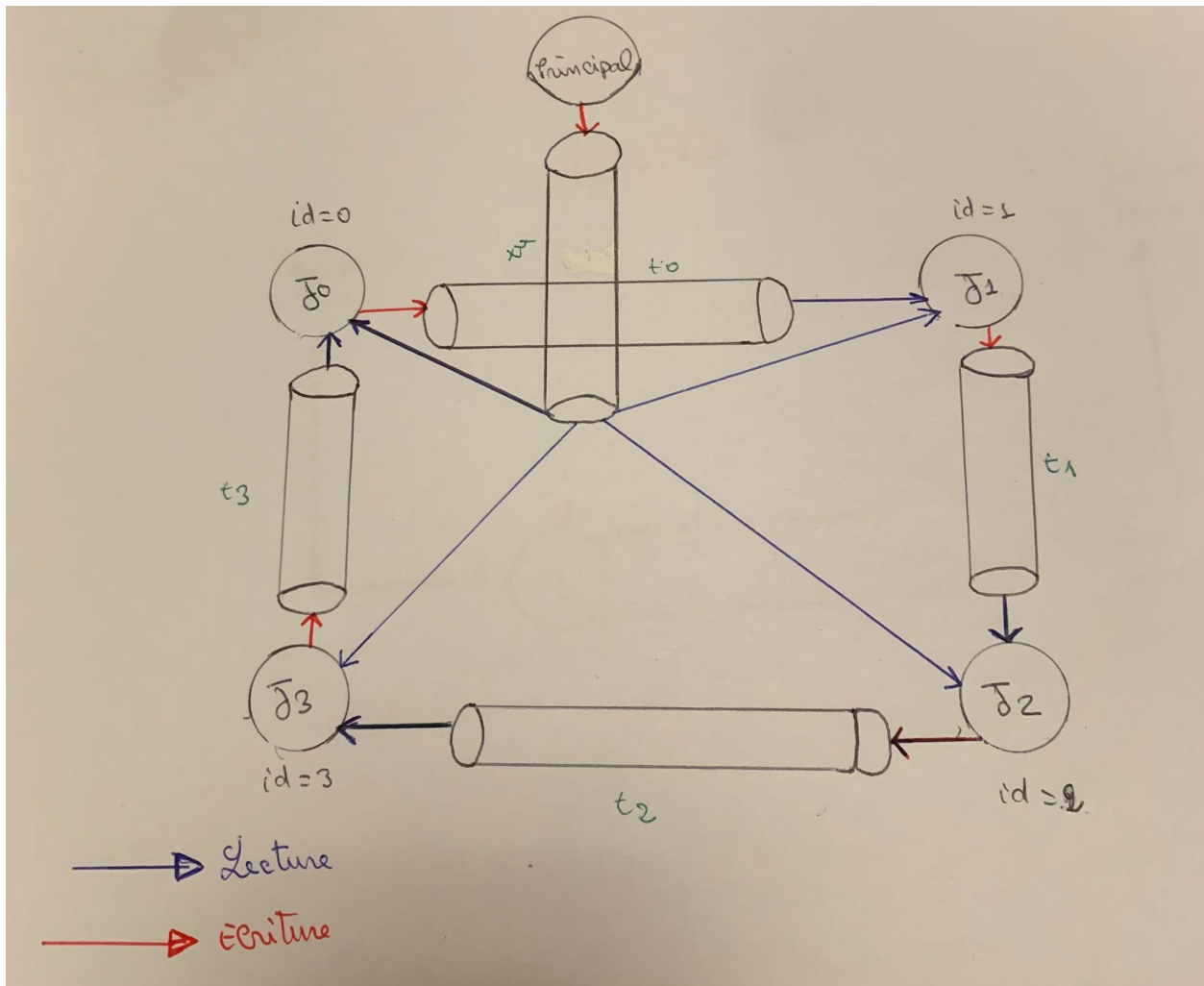
BARRY Mariama & DIALLO Thierno

Introduction

Le but de ce projet est de réaliser un jeu de petits chevaux dans lequel chaque joueur est un processus différent, et que les joueurs communiquent à travers des pipes tout au long d'une partie. L'idée derrière ce jeu est de faire circuler une information entre différents processus fils issus du même processus père avec l'utilisation des tubes d'où la particularité de ce projet.

Les processus doivent être capables de communiquer par tube entre eux, afin de pouvoir se transférer les bonnes informations jusqu'à ce que l'un d'entre eux soit en capacité de la traiter et pour finalement la transférer au processus père.

Conception



Afin de répondre au besoin attendu par le cahier de charge (le sujet du projet), nous avons tout d'abord structuré notre projet en 3 dossiers, Projet_0, Projet_1 et Jeu:

Projet_0

Dans cette partie nous avons procédé à la création de nos différents processus

Projet_1

Pour cette partie essentielle du projet nous avons établi la communication entre les différents processus pour y arriver nous avons imaginé une structure applicative et de données permettant de réaliser cette communication.

Jeu

Cette partie qui consiste à implémenter le jeu en question, viendra en dernier lieu une fois que nous avons le `Projet_1` qui est fonctionnel.

Nous sommes donc partis du constat que les pipes sont des objets qui permettent l'échange de données notamment des tableaux d'octets entre processus. Nous avons tout d'abord défini les flux de données qui passent par chacun des pipes. De cette façon nous en somme arrivés au constat suivant :

- La donnée est lancée par le parent c'est à dire le processus père au premier de ses fils
- Les processus fils communiquent entre eux et ceux jusqu'à ce que la donnée soit transmise à tous les fils
- Lorsqu'ils peuvent répondre, ils envoient leur réponse au parent.

Découpage des entités et en fonctionnalité et définition des rôles dans ce projet.

Nous avons opté pour découper le plus possible notre code en fonctionnalité dans le but de le rendre le plus lisible possible. De cette façon, nous avons d'abord tenté d'identifier les grandes parties du code pour les séparer dans plusieurs fichiers c différents.

La fonction communication est l'une des plus importantes car c'est dans cette fonction que nous avons défini le modèle de communication entre les processus.

La fonction `creat_fork` son rôle est de créer nos processus, sa boucle continue à tourner tant qu'on a pas atteint le nombre de processus souhaité par l'utilisateur.

La fonction `joue` sa particularité est le fait d'initialiser le jeu avec la génération d'une valeur sur un intervalle défini.

Réalisation

Dans cette partie, nous détaillerons un peu plus notre démarche lors de la création du code. Ainsi que les éventuels problèmes auxquels nous avons été confrontés.

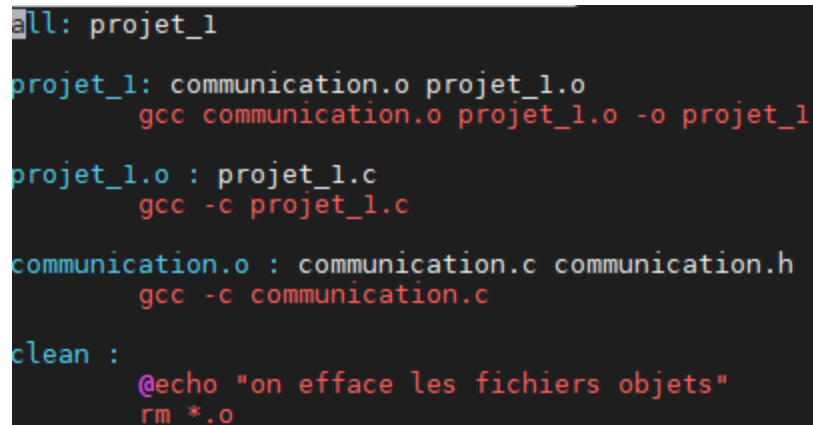
Dans le `project_0` nous avons un fichier source c qui contient une fonction `creat_fork()` qui crée `N` processus fils.

Dans le `project_1` : Création de `N` fils puis mise en place d'un cycle de communication par tube, par le biais de la fonction communication énoncée plus haut.

Pour chacun de ces dossiers, nous avons créé un Makefile.

Ce Makefile est le même modèle de partout sauf bien évidemment on a pas les mêmes fichiers sources dans les 3 dossiers.

Voir image ci dessous



```
all: projet_1

projet_1: communication.o projet_1.o
    gcc communication.o projet_1.o -o projet_1

projet_1.o : projet_1.c
    gcc -c projet_1.c

communication.o : communication.c communication.h
    gcc -c communication.c

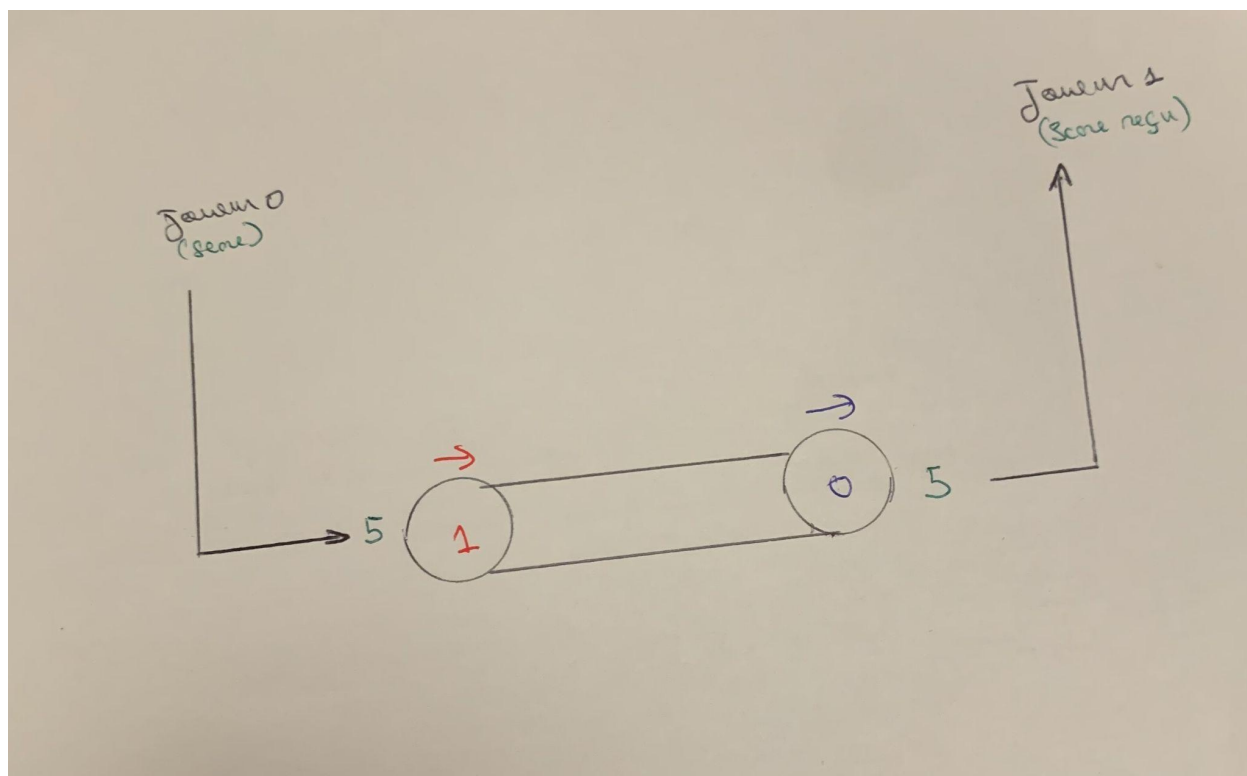
clean :
    @echo "on efface les fichiers objets"
    rm *.o
```

Quant à la réalisation du jeu

Nous avons mis en place le principe du jeu par la création d'une fonction **joue** qui prend en paramètre le joueur courant, dont le rôle est de lancer un dé dont les valeurs sont comprises de 1 à 6 et renvoie la valeur obtenue dudit dé lancé.

Nous avons ensuite fait en sorte que les joueurs qui sont des processus comme on l'avait indiqué dans l'introduction, puissent transmettre une information après chaque tour du joueur courant dont l'information initiale vient du processus parent. Cette première information est reçue par le premier fils qui à son tour renvoie à son frère processus suivant l'information sur son score réalisé après avoir joué, ainsi ce dernier continue avec le prochain joueur jusqu'à ce que le tour des joueurs soit effectué.

Un exemple d'illustration de notre protocole d'échange entre le joueur 0 et le joueur 1 dont le joueur 0 transmet son score de 5 au joueur 1 voir (image).




Remarque sur les problèmes rencontrés :

Pour la partie jeu, nous avons réussi juste à faire un tour avec un pion pour chaque joueur. Dans notre conception du jeu nous avons prévu que pour chaque joueur il devait faire un tour complet avec un score maximale que nous avons fixé à 60 pour qu'il soit l'heureux gagnant d'une partie de jeu .

Pour illustrer nos dires voir ci-dessous un exemple d'exécution.

```
----- lancement du jeu -----  
Joueur 0 a obtenu : 3  
je suis le fils0  
mon score 3  
fils0 a lu la valeur :2  
Joueur 1 a obtenu : 2  
mon score 2  
je suis le fils1  
fils1 a lu la valeur 3  
Joueur 2 a obtenu : 4  
mon score 4  
je suis le fils 2  
fils 2 a lu la valeur 2  
Joueur 3 a obtenu : 5  
mon score 5  
je suis le fils3  
fils3 a lu la valeur 4
```

Conclusion



L'objectif final de ce projet était de nous faire découvrir et utiliser les fonctions avancées de programmation système sur Linux en C. D'élaborer notre propre moyen de communication entre différents processus afin de nous faire réfléchir sur la façon d'organiser les données d'une application ainsi que leur cycle de vie.

Il nous a fallu de longue réflexion et une étude attentivement de la question afin de concevoir une solution à la fois faisable techniquement et qu'on pense répondre à ce qui est attendu de nous . C'est donc au moyen de nos différentes réflexions et analyses que nous avons pu réaliser les différentes étapes de ce projet bien qu'on a malheureusement pas pu achever totalement le jeu.