Rapport Projet OS partie 1

Realisé par: Kevin Issa et Arthur Installé
matricule Kevin: 000514550
matricule Arthur:000495303
email: kevin.issa@ulb.be et arthur.installe@ulb.be

Octobre 2022

Table des matières

1. Introduction:

Après avoir appris le fonctionnement matériel d'un ordinateur, la prochaine étape est d'en savoir plus sur le système d'exploitation et les méthodes plus avancées pour créer un programme.

C'est dans ce but que s'inscrit ce projet, sous couvert du thème de la base de données, nous appliquons un moyen d'exécuter en parallèle plusieurs parties du même programme grâce aux processus et de les faire communiquer entre eux avec des pipes. Dès lors, nous allons vous expliquer dans ce rapport notre manière de procéder ainsi que nos choix de conception et les difficultés rencontrées.

2. Présentation du projet:

Ce projet consiste en une base de données d'étudiants où l'utilisateur peut choisir des requêtes grâce à des commandes terminales telles qu'insérer un étudiant, si son id n'est pas déjà présent dans la base de données, de sélectionner une série d'étudiants correspondant à un filtre champ=valeur, d'en enlever suivant le même filtre ou de mettre à jour leurs informations grâce aux filtres champ=valeur(appel à select) champ modifié = valeur modifiée.

3. Problématique du projet:

3.1. Organisation du travail de groupe:

Étant donné qu'il s'agit de notre premier travail en groupe, un problème que l'on a eu était qu'étant donné que nous avions des méthodes de programmation différentes et des habitudes différentes lors du choix des noms de variables, lors de la mise en commun de fonctions liées, la relecture fut difficile et nous avons alors pris conscience du problème de compréhension du code qui s'est posé.

Dès lors, nous avons décidé de demander à des personnes du milieu professionnel, comment ils travaillent en équipe. Ainsi, nous avons séparé notre github en branche pour chaque fonctionnalité du projet sur lequel nous travaillons qui seront à la fin merge sur le main. De plus, nous avons posé des règles de programmation lorsque nous codons, telles que nommer les variables de telle manière, laisser des commentaires pour expliquer la logique derrière les fonctions et prévenir lorsque l'on travaille sur une branche et lorsque l'on commit et push quelque chose afin d'éviter les cas d'overwrite lorsque nous travaillons sur la même branche.

3.2. Choix du type de programmation:

Le projet étant à la base en C puis changé en c++ suite à un changement de consignes, 2 choix s'offraient à nous. Étant donné que nous avions commencé en C nous pouvions continuer comme cela et ne pas prendre le risque de devoir modifier des fonctions

pour la partie 2 ou passer en c++ et profiter dès lors de la facilité qu'apporte le langage comparé au C (vecteur,string,référence,etc).

Dès lors, nous avons décidé de rester en C tout en utilisant quelques avantages du c++. En effet, nous avons décidé d'uniquement utiliser ces avantages dans le fichier querys afin de nous faciliter un peu le codage tout en pouvant facilement repasser à du C si la partie 2 du projet l'exige. les appels systèmes étant en C, nous avons trouvé plus logique de rester en C.

Par conséquent, nous n'avons pas utilisé les classes de c++ car cela demanderait de modifier profondément les fichiers fournis en prenant beaucoup de temps à faire cela sans avoir beaucoup de gains.

3.3. Problème d'implémentation des querys:

Lors de l'implémentation des querys, une des difficultés durant le parcours de la base de données est qu'il fallait trouver une méthode pour savoir quel champ l'utilisateur a utilisé comme filtre. La méthode simple étant de faire une série de if, else if

Cependant, cette méthode n'est pas très belle en termes de codage. Dès lors, nous avons cherché une méthode alternative qui permettrait de raccourcir le code sans le rendre plus dur à lire. Par conséquent, une des méthodes possibles serait d'utiliser un switch case, mais étant donné que les switch cases de c++ sont limités aux int et que nous n'ayons trouvé aucune meilleure méthode que le if, else if, nous avons codé les querys, select, delete et update de cette manière, même si nous pouvions utiliser des opérateurs logiques, nous trouvions cela moins lisible.

3.4. Les pipes:

En effet, utiliser les pipes de la bonne manière fut compliqué au début, car nous pensions que le père devait envoyer l'information de la requête aux fils et que les fils renvoyaient l'information traitée au père pour que cela puisse marcher. De plus, nous avions mal compris comment utiliser les pipes avec plusieurs processeurs. Dès lors, nous nous sommes retrouvés avec 8 pipes.

C'est lors, de la discussion de nos idées avec un autre groupe, que nous nous sommes aperçus que nous avions mal compris le fonctionnement attendu. Ainsi, nous avons réduit le nombre de pipes de 8 à 4 et nous avons modifié le fonctionnement des fils afin que le père n'ait pas besoin d'attendre la réponse du fils via une lecture, ce qui facilite grandement le code.

Cependant, plus tard dans le projet, transaction nous a obligé a revenir à 8 pipes car les fils doivent communiquer au père qu'ils ont fini leur tâche.

3.5. Problèmes de compatibilité Mac OS, linux

Effectivement, même si linux et mac os sont tous les deux sous unix, leur compilateur étant légèrement différents, plusieurs problèmes furent découvert. En effet, à plusieurs moments, le compileur de mac indiquait une erreur qu'il n'y a pas sous linux. Par exemple

pour la fonction sprintf, sous C++, mac avertit que la fonction est obsolète et qu'elle n'est présente uniquement pour une compatibilité avec C alors que linux compile sans problème.

De plus, lors d'une erreur venant d'un new associé a un malloc dans la même fonction, linux indiquait une allocation fault alors que mac compilait sans problème. Dès lors, nous avons fait en sorte que le code puisse compiler sans problème sur les deux machines.

3.6. Problème important avec les tests:

Lors de la phase de test de nos fonctions, chaque query fonctionne en prenant en tant que base, la base de données des tests fournis mais lorsque l'on passe les tests avec la commande, ceux-ci ne fonctionnent plus alors que lorsque l'on rentre nous même les mêmes tests, cela ne pose aucun problème. Cependant par manque de temps, nous n'avons pas pu régler le problème.

4. Conclusion:

En conclusion, ce projet nous a permis de mettre en pratique ce que nous avons vu durant les travaux pratiques et le cours même si cela été difficile au début, mais plus important, il nous a permis de découvrir une nouvelle méthode de travail qu'est le travail de groupe.