BRISSARD Théo
BERTINEAU Théo

Rapport projet C avancé

Introduction

Dans le cadre de l'UE Programmation C Avancé nous avons dû réaliser un projet mettant en application une grande partie de ce que nous avons vu au cours du semestre.

Le but du projet est de mettre en place un système de communication au travers d'un Démon, qui attend que des Clients le contactent pour leur rendre des Services.
Les Services sont des programmes lancés par le Démon alors que les Clients sont des programmes indépendants.
Il y a une mécanique de communication entre le Démon et les Clients dans un premier temps puis entre les Clients et les Services par la suite.

Grâce aux indications et aux fichiers fournis, il nous a été beaucoup plus simple de nous repérer et de réaliser ce projet.

Organisation du code

Nous allons ici résumer la liste des fichiers et leurs buts.

- demon.c : attend un client et lance les services à travers une séquence de communication définie
- utils_demon.c : une liste de fonctionnalités utilisées par le demon
- utils_services.c : une liste de fonctionnalités utilisées par les services et les clients
- config.c : utilisé par le demon afin savoir si le service demandé peut être lancé
- fichier.c : package personnel pour gérer les fichiers (ouverture, fermeture, ect…) certaines des fonctions de ce package ne sont adaptées au projet
- lireecrire.c : package personnel pour écrire et lire dans un fichier déjà ouvert

Dans CLIENT
- client_exit.c : pour donner l'ordre au demon de s'arrêter
- clientBinarisation.c : lance le service Binarisation
- clientCompression.c : lance le service Compression
- clientMiroir.c : lance le service Miroir
- clientSomme.c : lance le service Somme

Dans CONFIG
- test_config.c : pour tester si config.c fonctionne

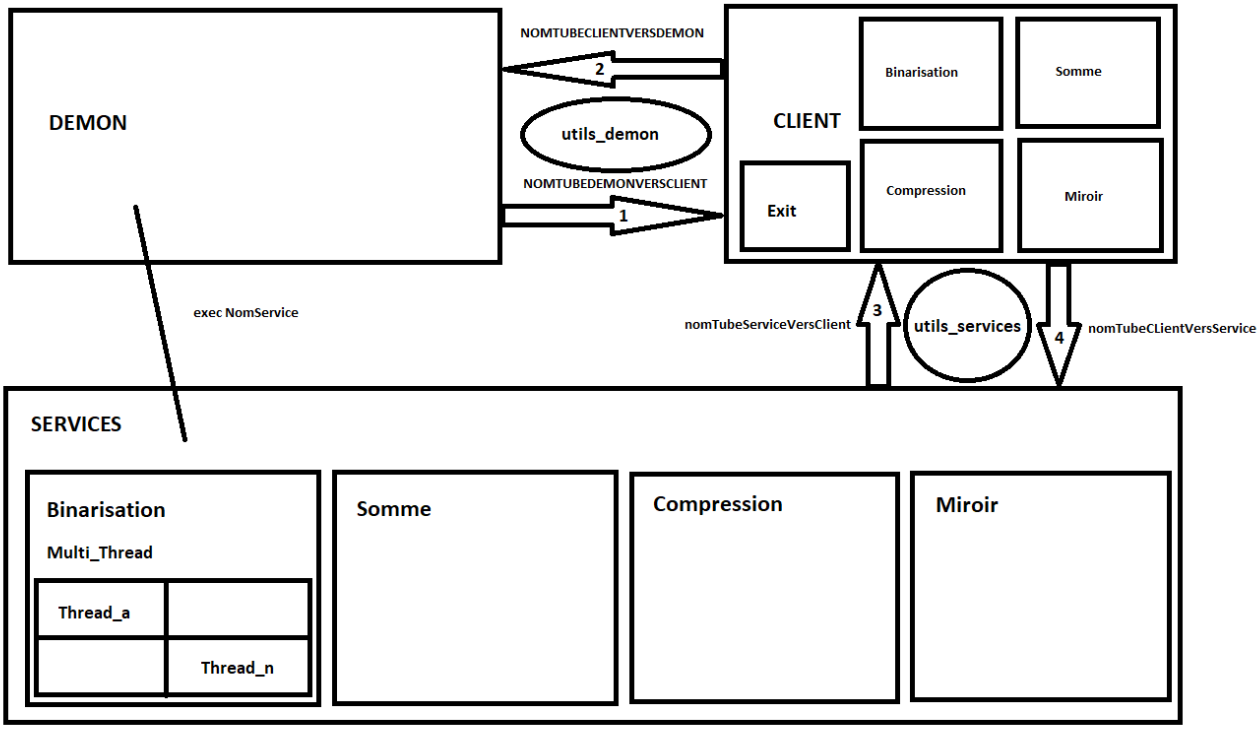
Dans IMAGE
- Image.c : pour manipuler une image

Dans MYASSERT
- myassert.c : pour détecter les erreurs proprement

Dans SERVICES

- binarisation.c : convertit une image de nuance de gris en image binarisé
- compression.c : renvoie une chaine de caractère sans les voyelles
- miroir.c : renvoie une chaine de caractère dans l'ordre inverse
- somme.c : renvoie la somme de nombres flottants

Protocoles de communication



Fonctionnement des tubes :

1. Est toujours en attente qu'un Client fasse une demande, dès que c'est le cas, le tube n'est plus accessible en attendant que le Client ait sa demande traitée
2. Est toujours en attente qu'un Client fasse une demande, dès que c'est le cas, le tube n'est plus accessible en attendant que le Client ait sa demande traitée
3. Envoi le résultat du service
4. Envoi les informations nécessaires pour réaliser le service

1 et 2 sont créé par le demon dans sa phase d'initialisation, ils sont protégés par un mutex
3 et 4 sont créé par le demon une fois qu'un client a pris contact avec lui

Problèmes rencontrés

- Gérer l'arborescence des dossiers du projet dans le code
- Pour config.c, cahier des charges trop flou

Utilisation de nos clients

- clientSomme prend en paramètre [vos float]
- clientMiroir prend en paramètre [votre string]
- clientCompression prend en paramètre [votre string]
- clientBinarisation prend en paramètre [PATH de l'image] [nbThread] [seuil]
- client_exit ne prend pas de paramètre