Rapport de Projet

Auteurs :
Valdrin Salihi, Guillaume Pierre

Professeurs Encadrant :
G. Subrenat, Y. Bertrand,
N. Courilleau, D. Meneveaux

Table des matières

Organisation du code :

Fichiers Source Principaux :

- orchestre.c :

Le fichier orchestre.c va servir au sein de ce projet lors de sa création lancer tous les services dont on aura besoin ainsi que d'une association de paire tubes nommés prévus à la communication entre chaque client et l'orchestre et de même entre client et service. Ainsi qu'il y aura une communication unidirectionnelle via un tube anonyme de l'orchestre vers chaque service. En matière de communication, l'orchestre va aussi créer des sémaphores afin de pouvoir harmoniser l'échange entre toutes les autres entités afin qu'il n'y ai pas d'erreur dans chaque processus.

- client.c :

Le fichier client.c lui s'occupera d'effectuer des requêtes à l'orchestre et si ce dernier va accepter, il va alors le rediriger via un processus qui sera expliqueé dans la suite vers le service voulu afin d'effectuer l'action demandé et enfin enverra la réponse à l'orchestre pour passer à la suite.

- service.c :

Le fichier service.c est créer lorsque l'orchestre.c sera lancer. Le but de ce dernier va être d'effectuer une action que l'on aura paramétrée au préalable (exemple : recherche d'un maximum, une addition entre deux variables, etc..) lorsque l'orchestre lui aura donné la permission de le faire. Une fois l'action effectuée, le service enverra la réponse via un tube nommé en destination du client.

Organisation du code :

Fichiers Sources des relations :

- client_orchestre.c/.h :
Les fonctions utiles dans ce fichier entre le client et l'orchestre vont être la création de sémaphore ainsi que la création et la fermeture des tubes nommés.

- client_service.c/.h :
La partie client_service va contenir la structure « cS » entre le client et le service permettant le bon fonctionnement de la relation ainsi que l'initialisation de la structure.

- orchestre_service.c/.h :
Les fichiers orchestre_service.c et .h vont contenir une structure du nom d'« oS » permettant la relation entre l'orchestre et les services. Il y aura aussi la création d'un sémaphore à l'intérieur et la création du premier service.

Fichiers Sources annexes :

- tubesem.c/.h :
Ces deux fichiers ont été rajoutés par nos soins au projet. Ils intègrent des fonctions nécessaire aux manipulations des tubes nommés comme anonymes ainsi que la manipulation des sémaphores. En plus de cela, à chacune des fonctions des ces deux fichiers on y intègrent directement des assert améliorés (grâce au myassert du projet) afin de tester comme demander le bon déroulement de chacune de ces fonctions.

Organisation du code :

Fichiers Sources annexes :

- service_somme.c/.h :
Le fichier service_somme est le premier service demandé dans ce projet servant à récupérer des floats d'un client et d'en faire la somme.

- service_compression.c/.h :
Le service_compression lui est le second service permettant d'effectuer une compression un peu particulière. Tout d'abord, il ne fonctionne que sur des chaînes de caractères sans présence de chiffre. Son action va être de prendre la chaîne de caractères que le client lui donne et de la modifier de telle sorte qu'au sein de cette nouvelle chaîne, chaque caractère est précédé d'un chiffre indiquant combien de fois il a été présent depuis le début de la chaîne.

- service_maximum.c/.h :
Le dernier service est celui du maximum, il est censé faire une recherche multi-threads dans un tableau de float que le client va lui envoyé. Le principe est que chaque thread va s'occuper d'une partie du tableau est va y indiquer le maximum trouvé, si le maximum trouvé est supérieur à celui du thread précédant alors il va l'écraser sur la variable partagé par tous les threads.

- client_somme.c/.h :
Le fichier client_somme et l'autre face de la pièce entre ce dernier et le service_somme. Son rôle va être de lui envoyer les données (plus précisement des floats) via une paire de tubes nommés et d'en recevoir le résultat afin de pouvoir l'envoyer par la suite à l'orchestre.

- client_compression.c/.h :
Le client_compression va dans la même idée que le précédent envoyer des données à son partenaire (service_compression) et attendra la réponse du service. Les données qu'il va envoyer seront des chaînes de caractères.

Organisation du code :

Fichiers Sources annexes :

- client_maximum.c/.h :
Dans la même veine que ses deux compères, client_maximum va envoyer des floats au service_maximum et recevoir à la fin de l'opération le résultat.

- client_arret.c/.h :
Ce fichier va servir de cas d'arrêt pour les requêtes du client.

- config.c/.h :
Le fichier config ne va pouvoir interagir seulement avec l'orchestre, il aura pour but de donner des informations à l'orchestre comme le nombre disponibles et le nom des services.

Protocoles de communication :

Sémaphore :

Nous utilisons deux sémaphores, le premier entre en section critique lorsqu'il doit gérer la communication de plusieurs clients avec l'orchestre. Le second sémaphore permet de prévenir l'orchestre de la fin d'un service.

Tubes :

- Tubes nommés :
Deux tubes nommés permettent la communications entre le Client et l'Orchestre dans les deux sens. Puis, nous avons 6 tubes nommés pour la communications entre chaque services et le client.

Protocoles de communication :

Tubes :

- Tubes anonymes :
Nous avons un tube anonyme pour la communication entre l'orchestre et le service, seul l'orchestre communiquera envers le service. Le service ne fera que réceptionner les données de l'orchestre.

Déroulement :

Orchestre – Client :

L'Orchestre créée les deux tubes nommées et le sémaphore pour s'assurer d'une bonne communication. Ensuite, il ouvre les deux tubes, puis attend la demande d'un client.

Le client de son côté ouvre les tubes et récupère le sémaphore. Il entre en section critique pour qu'un seul client soit géré à la fois. Il envoie aussi une demande à l'orchestre après il analyse la demande du client et traite cette demande. Si, tous se passe bien l'orchestre envoie les données nécessaires au client.

Le client reçoit les données et envoie à l'orchestre que tous c'est bien déroulé et ferme les communications et sort de la section critique.

L'orchestre reçoit du client que tout s'est bien passé et ferme les communications avec le client.

Service - Client :

Après avoir communiqué l'orchestre, le client ouvre ses communications avec le service et envoie un mot de passe attend que le service vérifie ce mot de passe. Le service vérifie que le mot de passe soit correcte, si tout est bon il envoie au client un code d'acceptation.

Le client reçoit le code, puis lance la fonction de service en lien avec le numéro de service et un code qui assure le bon dérouler. Notre service, lance sa propre fonction de service, après la réception du code il ferme ses communications avec le client.