TP Noté

Le Nalinec Tibère  
Pedretti Zack

Toussaint Jarod

Préambule :

Pour résoudre ce problème, nous pouvons utiliser la topologie en étoile.

Le service central centralise le calcul global. Il répartie les petits calculs vers les clients et récupère les résultats renvoyés par les clients.

Lorsqu’un nouveau client se connecte, le service central lui envoie un bout du calcul.

Partie « Le tracé de rayon » :

Question 2 :

|  |  |
| --- | --- |
| Taille de l’image | Temps de calcul (ms) |
| 16x16 | 35 |
| 32x32 | 68 |
| 64x64 | 151 |
| 128x128 | 383 |
| 256x256 | 761 |
| 512x512 | 1914 |
| 1024x1024 | 6072 |
| 2048x2048 | 21591 |
| 4096x4096 | 88071 |

Question 3 :

Vous pouvez retrouver ci-dessous les lignes qu’on a modifié afin de pouvoir afficher deux parties de l’image :

On calcule d’abord deux images différentes avec scene.compute() et on les affiche avec disp.setImage().

int x0 = 0;

int y0 = 0;

int x1 = largeur/2;

int y1 = hauteur/2;

int l = largeur/2;

int h = hauteur/2;

//Image en bas à gauche

Image image = scene.compute(x0, y0, l, h);

//Image en haut à droite

Image image2 = scene.compute(x1, y1, l, h);

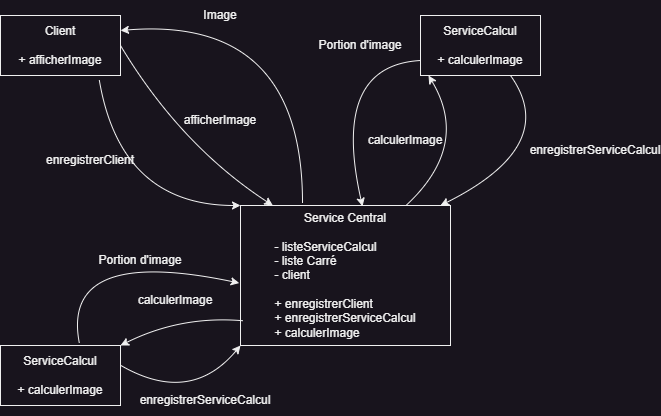
// Affichage de l'image calculée

disp.setImage(image, x0, y0);

disp.setImage(image2, x1, y1);

Partie « Accélérons les choses » :

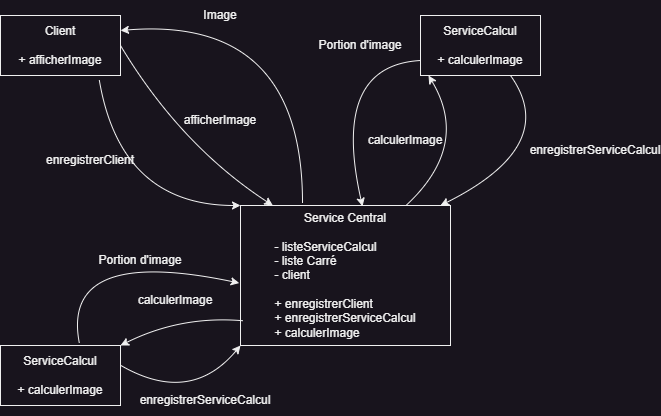
Schéma :



1. Le processus fixe serait le service central (noté SC sur le schéma) et les processus éphémères seraient les services RayTracing qui calculent l’image.
2. Une fois les services RayTracing envoyé au service central, celui-ci leur attribue des portions à calculer sous la forme d’un point de coordonnées x y et la largeur et la hauteur du carré.

Si l’on veut que les calculs se fassent en parallèle, il faut dans un premier temps décomposer l’image principale en portions de même largeur et hauteur, par exemple, une image qui sera décomposer en carrés de 64x64, et ensuite le serveur central aura une boucle for qui attribue des pixels pour chaque service RayTracing. Il faut ensuite intégrer au programme des threads sur les méthodes qui partagent ces portions à calculer aux services de calcul. Les threads servent à ne pas mettre en pause les autres opérations du service central, et ainsi faire les calculs en parallèle.  
  
Pour ce faire, on possède un tableau de coordonnées déjà calculées. A chaque fois que l’on souhaite démarrer un calcul sur un service de calcul, le programme recherche une coordonnée non calculée. Une fois que la coordonnée est trouvée, la méthode qui appelle un service de calcul est exécutée sur un thread pour paralléliser les opérations.

Description de l'architecture complète (schéma) de votre application ainsi que les rôles de tous ses composants :



code des interfaces utilisées (le reste étant dans le dépôt) :



diagramme de séquences incluant tous les acteurs (processus) qui illustre l'ordre des étapes lors d'une exécution complète :

