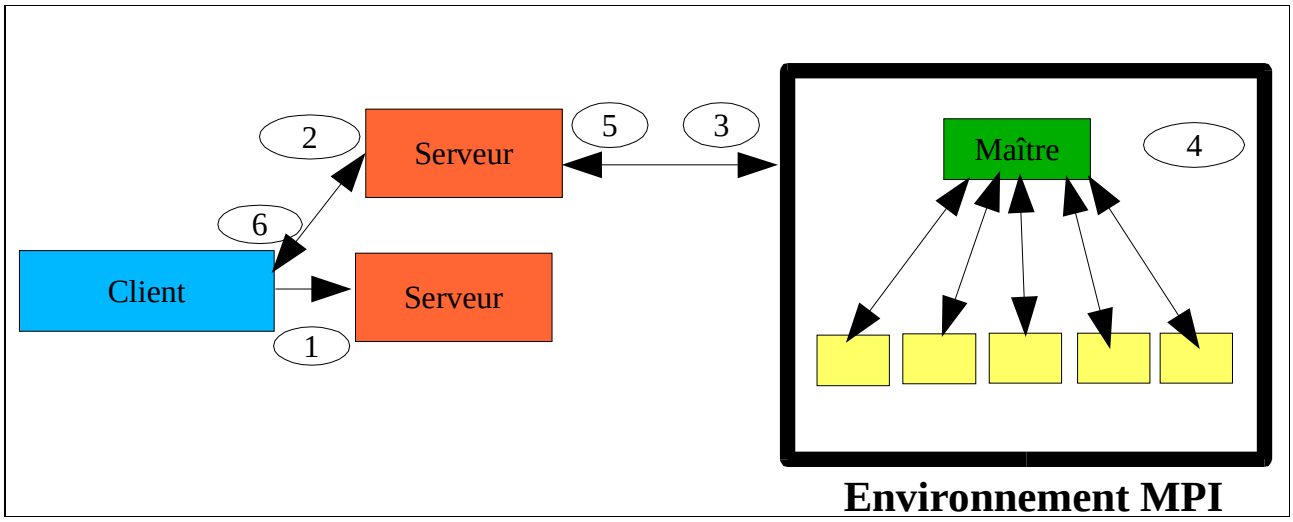
Description de la mise en oeuvre du dispatcher de catalogues.

Pour distribuer le travail aux différents noeuds, nous avons mis en oeuvre un dispatcher afin d'optimiser les performances du cluster. En effet, un client envoi une liste de travaux à effectuer et le dispatcher réceptionne ces travaux et les effectue sur les différents noeuds. Lorsque tous les travaux sont terminés, les résultats collectés sont envoyés au client dans le même ordre que les requêtes correspondantes reçues précédemment.

Protocole de fonctionnement lors d'une requête :

1. Le client envoi une demande de connexion.
2. Le serveur accepte cette nouvelle connexion et se clone à l'aide d'un fork, afin de continuer d'accueillir de nouveaux clients et de traiter les requêtes formulées par le client.
3. Pour traiter les requêtes, le serveur lance un environnement mpi et lui envoi les travaux à effectuer. Cet environnement facilite la mise en oeuvre d'une application au sein d'un cluster. Dans notre cas, nous avons choisi une organisation des processus assez simple qui est constituée d'un maître qui distribue les travaux et d'esclaves qui les exécutent. Pour traiter les requêtes, les esclaves lancent l'application du catalogues dont la requête en fait l'objet puis recueillent les résultats.
4. Le maître distribue les requêtes aux différents esclaves et récupère leurs résultats.
5. Lorsque le maître a collecté tous les résultats, il les renvoi dans le même ordre que celui de la réception des requêtes correspondantes au serveur qui les a reçu de son client.
6. L'environnement mpi se termine ainsi que le serveur qui traitait le client.

Schéma récapitulatif de ce protocole :



Initialisation du maître :

Lors du lancement de l'environnement mpi , le maître lit le fichier «file_dispatcher.conf» afin de localiser les différents catalogues situés dans les différents noeuds du cluster.

Structure du fichier file_dispatcher.conf :

[Fichier_initialisation] → [noeud]*

[noeud] →*nom_du_noeud* : *chemin_accès_application, liste_ou_se_trouve_les_catalogue*; (saut de ligne sauf pour le dernier catalogue)

Exemple de fichier :

USNO_A2 : pmm.exe, /home1/catalogues/USNO-A2/bin/pmm.exe, cocat2.u-strasbg.fr, cocat3.u-strasbg.fr;

UCAC1 : ucac1, /home1/catalogues/UCAC1/bin/ucac1, cocat2.u-strasbg.fr;

Installation du dispatcher au sein du cluster :

- Créez un utilisateur car une protection évite l'utilisation d'un environnement mpi avec le compte root.
- Loggez vous au compte de cet nouvel utilisateur et décompresser le fichier source_dispatcher.tar dans le HOME directory.
- Reloggez vous en root et téléchagez les différents catalogues sur le serveur.
- distribuez les catalogues dans les différents noeuds à l'aide de la commande «cpcp».
- Editez pour chaque noeud le fichier /etc/bashrc afin d'y introduire les variables d'environnements utilisées par les différents applications des catalogues.

Exemple de variable d'environnement dans une fichier /etc/bashrc :

```
############################################################
#Variable d'environnement des programmes des catalogues.#
############################################################


#Variables du catalogue GSC1.1.
export GSCDAT11=/home1/catalogues/GSC1.1
export GSCBIN11=/home1/catalogues/GSC1.1/bin


#Variables du catalogue GSC1.2.
export GSCDAT12=/home1/catalogues/GSC1.2
export GSCBIN12=/home1/catalogues/GSC1.2/bin


#Variable du catalogue GSC2.2.
export GSC2root=/home1/catalogues/GSC2.2
```

```
#Variable du catalogue GSC_ACT.
export GSCDAT_ACT=/home1/catalogues/GSC_ACT
export GSCBIN_ACT=/home1/catalogues/GSC_ACT/bin

#Variable du catalogue UCAC1.0.
export UCACroot=/home1/catalogues/UCAC1/bindat

#Variable du catalogue USNO-A1.
export PMM1root=/home1/catalogues/USNO-A1

#Variable du catalogue USNO-A2.
export PMM2root=/home1/catalogues/USNO-A2

#Variable du catalogue USNO-B1.
export USNOBroot=/home1/catalogues/USNO-B1

#Variable du catalogue 2MASS.
export MASSroot=/home2/catalogues/2MASS/data

#Variable du catalogue 2MASSI.
export root_2MASS=/home2/catalogues/2MASSI/data

#Variable du catalogue APM.
export APMroot=/home2/catalogues/APM/bindat

#Variable du catalogue DENIS.
export DENIS_root=/home2/catalogues/DENIS/data

#Variable du catalogue DENIS-P.
export denisROOT=/home2/catalogues/DENIS-P/data
```

Configuration du système afin que le serveur démarre lorsque cocat1 boot :

Pour cela, j'ai écris une application dont les sources sont situées dans le répertoire source_lanceur au sein du HOME directory du root. J'ai placé son exécutable dans le répertoire /usr/local/bin.

Puis dans le fichier /etc/inittab :

Mettre en commentaire la commande :

```
1:2345:respawn:/sbin/mingetty tty1
```

Puis, ajouter la commande :

```
1:3:respawn:/usr/local/bin/lanceur /dev/tty1 "/home/nis/nicaisse/cocat/bin/server"
```