# Détails UML

## Client

ClientSocket

* Crée un socket.
* Se connecte au serveur.

Communication (thread)

* Lectuer/Ecriture sur le flux réseau.

Extractor

* Contient un certain nombre de switch (sur la commande string reçue).
* Pour chaque cas, on utilise le JsonMapper pour créer un objet du type désiré.

MessageHandler (autant de méthodes process que de requêtes/réponses possibles)

* Reçoit l’objet désérialisé par le JsonMapper.
* Agit en conséquence (prépare la réponse, récupération des paramètres, réutilisation du JsonMapper, …).
* Appelle la méthode process désirée qui appelle la méthode write() de Communication.

## Serveur

ServerSocket

* Crée un socket réceptionniste.
* Attend passivement que des clients se connectent.
* Lorsqu’un client se connecte, il crée un nouveau thread de type ClientManager dans lequel il passe le socket du client.

ClientWorker (thread)

* Récupère et stocke le socket du client.
* Lecture/Ecriture sur le flux réseau.

Extractor

* Contient un certain nombre de switch (sur la commande string reçue).
* Pour chaque cas, on utilise le JsonMapper pour créer un objet du type désiré.

MessageHandler (autant de méthodes process que de requêtes/réponses possibles)

* Reçoit l’objet désérialisé par le JsonMapper.
* Agit en conséquence (prépare la réponse, récupération des paramètres, réutilisation du JsonMapper, …).
* Appelle la méthode process désirée qui appelle la méthode write() de ClientManager.