**Morbois Antoine**

**Curtet Quentin**

**Pouget François**

**Le Bissonnais Gautier**

# C++ Rapport TP – EPSI Hadopi

## Spécifications générales

Le but de ce projet est de pouvoir échanger des fichiers au travers d’un réseau local. On peut imaginer qu’il sera utilisé dans un réseau d’entreprise pour faciliter les échanges.

Le client doit donc dès le lancement du logiciel pouvoir synchroniser un répertoire de son ordinateur avec des ordinateurs distants. Pour fonctionner le logiciel doit être démarré sur chacun des postes que l’on souhaite synchroniser. Il n’y a pas de serveur qui centralise les échanges, par conséquent chaque poste devra donc jouer le rôle de serveur et de client.

Ce logiciel sera développer en C++ et devra utiliser un système de threading pour répartir les échanges et optimiser au maximum la vitesse de ces derniers. Les aspects réseau du serons gérer dans les librairies fournis avec le cahier des charges.

Le produit finis sera déployé uniquement sur des Windows x64.

En ce qui concerne le développement, notre équipe utilisera Visual Studio (2012 ou 2013).

Serons utilisé également Google Code comme gestionnaire de version ainsi que Trello pour assigner les différentes tâches du TP.

## Spécifications détaillées

Pour des raisons de simplicité nous utiliserons des Desgin pattern :

1 singleton config :

* nombre envoie simultané au client (nombre de thread)
* nombre réception simultané
* timeout du scan
* adresse du répertoire

1 singleton config-interne :

* 1 tableau avec la liste des clients
* 1 tableau notant les fichiers

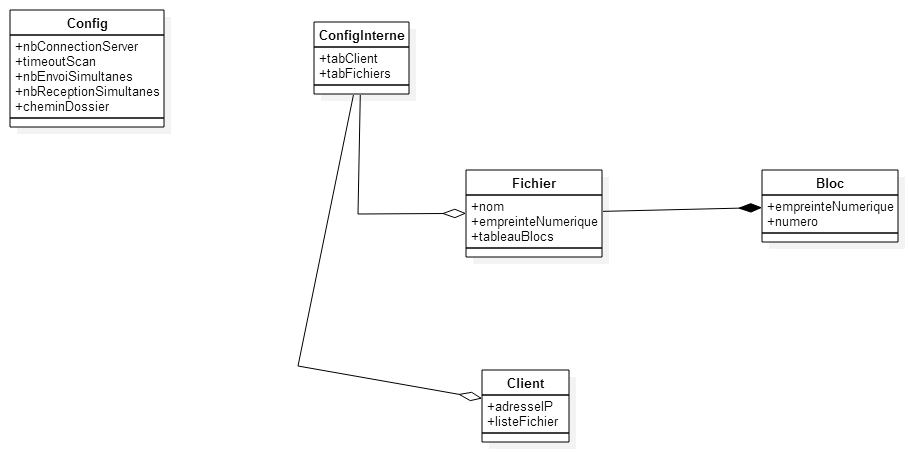
Découpage en différentes listes des tâches

* Transfert de fichier avec l’utilisation des threads (~10h).
* **tableau de fichier** (Synchroniser liste des fichiers présent dans le dossier et dans le tableau contenant le nom et l’empreinte numérique (refresh toutes les 5 min idéal)) (2h)
* Vérification tableau nom + empreinte sur tous les serveurs (2h)
* **Tableau de blocs** (2h) (décomposition de chaque fichier en plusieurs blocs (16k max)
* Mise en attente en cas de surnombre de connexion (optimisation si temps)
* Mise en place de **log élaboré** dans un fichier .txt (thread)
* Mise en place « **log utilisateur** » détaillant quel utilisateur a récupéré quel fichier
* Stop synchronisation si **touche échap** (suppression bloc si fichier non entier) (~1h)
* Paramètre rentré manuellement dans un 1er temps puis demande paramètre avant lancement (30min) => paramètre console
* Même nom + empreinte différente? => la machine venant de se connecter renomme son fichier avant d’importer l’autre. (~2h)

Liste des exceptions possible :

* N’arrive pas à créer le serveur.
* @Todo

Diagramme de classe :



## Bilan de réalisation