Arthur CHEVALIER

Samuel DA SILVA

Compte rendu intermédiaire :

### Avancement du projet :

Nous sommes actuellement en train de finir la partie gestion des flux. Il reste à implémenter la gestion de la requête unsubscribe et créer les test fonctionnels.

### Description succincte des différents packages et classes les composant :

* Package App :

Il possède toutes les classes assurant la gestion l'ajout et la lecture de messages grâce aux requêtes :

* + MessageDataBase assure le stockage des messages, mais aussi de les trier en fonction de plusieurs paramètres.
  + Message représente un Message avec son ID.
  + Subscription s’occupe de la gestion des abonnements et transmission des messages vers les abonnés
* Package Clients :

il regroupe toutes les classes de clients:

* + Client implémenente des fonctions de bases pour envoyer des requêtes à un serveur
  + Publisher publie des messages avec un nom d'utilisateur prédéfinis. Hérite de Client
* Package Servers :

il rassemble les différents serveurs :

* + Counter est compteur adapté à l’utilisation par des threads et retournant un entier unique à chaque appel de sa méthode
  + Simple Serveur est serveur simple TCP acceptant une requête et renvoyant la réponse associée à cette requête
  + NonConnectedServer est plus complexe, permettant le multi-thread à l’aide d’un executor et gérant plus de requêtes
  + MicroBlogCentral gère les flux d’abonnements et les requêtes d’abonnements en plus des précédentes.

* Package Requests :

il comporte les différentes classes permettant d'interpréter et de former des requêtes valides :

* + Request permet d'identifier les paramètres, le header et le corps du requête et de récupérer plusieurs informations sur celle-ci.
  + RequestInterpreter permet de récupérer le paramètre est l'identifiant de celui-ci.
  + RequestMaker permet de créer une requête correcte à partir de deux strings.
  + RequestFactory renvoie un objet de type Request en fonction de la chaine de caractères reçus à la suite la connexion. Cette classe implémente le pattern Factory.
  + PUBLISHRequest vérifie le bon format d’une requête PUBLISH. Si celle-ci est valide, la requête est exécutée, le message est publié dans la base de données.

# UML du projet :

# 