Partiel JD durée 3h (documents autorisés)

Réseau social

L'objet de ce problème est de créer une application java qui analyse des données d'un réseau social. Les données du réseau social sont de 2 types : message et commentaire. Un message initie un fil de discussion, un commentaire commente un message ou un commentaire.

- Un *message* est défini par les informations suivantes :
  - date : date du message
  - idMessage : est l'identifiant (unique) du message (entier)
  - idUser : est l'identifiant (unique) de l'utilisateur (entier)
  - message : est le contenu du message (chaîne de caractères)
  - user : est le nom de l'utilisateur (chaîne de caractères)
- Un *commentaire* est défini par les informations suivantes :
  - date : date du commentaire
  - idCommentaire : est l'identifiant (unique) du commentaire (entier)
  - idUser : est l'identifiant (unique) de l'utilisateur (entier)
  - commentaire : est le contenu du commentaire (chaîne de caractères)
  - user : est le nom de l'utilisateur (chaîne de caractères)
  - pidCommentaire : est l'identifiant du commentaire que ce commentaire commente (-1 s'il s'agit d'un commentaire à un message)
  - pidMessage : est l'identifiant du message que ce commentaire commente (-1 s'il s'agit d'un commentaire à un commentaire)

L'objectif est de créer un serveur qui calcule en continu les 3 messages postés qui ont les meilleures **valeurs d'importance**. La valeur d'importance est un entier qui détermine l'importance (ou la pertinence, l'actualité, ...) d'un fil de discussion (chaque nouveau message initie un nouveau fil de discussion). Plus la valeur est grande, plus le message est considéré comme important. La valeur d'importance d'un message posté est calculée comme la somme de son score et des scores des commentaires associés. Un commentaire est associé à un message posté s'il s'agit d'une réponse au message ou le commentaire d'un commentaire associé au message posté.

Un score est un entier positif ou nul. Chaque message ou commentaire posté a un score initial de 20, qui décroit de 1 à chaque nouvelle donnée (nouveau message ou nouveau commentaire). Si le score total d'un message atteint 0 alors ce message est considéré inactif, même si d'autres commentaires lui sont ensuite associées. Seuls les messages actifs peuvent être parmi les 3 meilleurs.

Le serveur est censé être connecté à un réseau social et recevoir de lui les messages et commentaires. Pour notre simulation, le serveur va lire les données dans un fichier.

1. Créer les deux classes `Commentaire` et `Message`.
2. Créer un serveur qui doit effectuer les opérations suivantes :
   (a) Un thread lit du fichier `reseauSocial.txt` les messages et commentaires et calcule en continu les 3 meilleurs messages postés (ceux qui ont les plus grandes valeurs d'importance). Une ligne du fichier (un message ou un commentaire) sera lue chaque seconde.
   (b) Un pool de threads sert à répondre aux clients. Si un client se connecte, le serveur envoie au client les 3 messages postés qui ont les meilleures valeurs d'importance (au moment où le client se connecte) sous la forme suivante :

   `idMessage|idUser|idMessage|idUser|idMessage|idUser`
3. Modifier le serveur pour que le thread qui lit les données et calcule les messages de meilleures valeurs d'importance puisse être sur une autre machine virtuelle que les threads qui répondent aux clients (on utilisera le mécanisme RMI).
4. Modifier le serveur de telle sorte que la lecture des fichiers de données se fassent via un gestionnaire de protocoles.
5. Modifier le serveur pour que l'envoi des résultats se fasse dans un format XML.

Questions optionnelles (plus plus de réalisme) :

a. Le score décroît de 1 chaque 30s et non à chaque lecture d'une ligne du fichier.

b. La lecture d'une ligne du fichier se fait après un temps aléatoire compris entre 1 et 3s.

Le fichier `reseauSocial.txt` contient des données qui peuvent vous servir à tester l'application. Une ligne est un enregistrement, soit de message, soit de commentaire, donc dans l'un des formats suivants :

```
idMessage|idUser|message|user||
idCommentaire|idUser|commentaire|user|pidCommentaire|
idCommentaire|idUser|commentaire|user||pidMessage
```