TP 2 Répertoire accessible par internet

L'objectif de ce TP est de mettre à disposition sur internet une base de données d'adresses, application client / serveur multi-utilisateurs utilisant les outils de communication réseau TCP/IP de l'environnement Java.

1. Le répertoire d'adresses

Le code Java de l'application « Répertoire d'adresses Internet » est disponible depuis moodle. Cette application est composée des types d'objets suivants :

- La classe `Personne.java` implante la notion d'adresse internet. Une personne se caractérise par son nom, son adresse mail, l'adresse web de sa page personnelle et un champ d'information.
- L'interface `Repertoire.java` décrit la notion de répertoire d'adresses. Ce type abstrait fournit des opérations pour ajouter, modifier, retirer, chercher et lister les personnes contenues dans la base de données.
- La classe `IHM.java` propose une interface graphique pour manipuler un répertoire d'adresses. Celle-ci se compose d'un ensemble de boutons pour appliquer les opérations sur le répertoire d'adresses, d'une fiche pour afficher ou saisir les caractéristiques d'une personne et de la liste des personnes contenues dans le répertoire.

2. La répartition de l'application

Le protocole de communication entre ces deux processus est de type Client / Serveur. Chaque fois que le processus client veut effectuer une opération sur le répertoire alors il envoie alors une requête vers le serveur TCP. Le serveur applique l'opération sur le répertoire et retourne une réponse au processus client. Précisez le protocole de communication entre les processus sachant que l'interface `Repertoire.java` est définie comme suit :

```
package repertoire;

public interface Repertoire
{
  // Ajouter une personne dans le répertoire.
  public boolean ajouterPersonne (Personne personne);

  // Modifier une personne dans le répertoire.
  public boolean modifierPersonne (Personne personne);

  // Retirer une personne du répertoire.
  public boolean retirerPersonne (String nom);

  // Rechercher une personne dans le répertoire.
  public Personne chercherPersonne (String nom);

  // Lister les noms des personnes.
  public String[] listerPersonnes ();
}
```

Pour définir ce protocole de communication, vous devez lister les différents messages de requête et de réponse échangés entre les deux processus et définir les données transportées par chacun de ces messages.

3. L'implantation du protocole de communication

Différents types de données (entiers, booléens et chaînes de caractères) doivent être transportés durant les échanges de messages entre le processus client et le processus serveur. Les lignes suivantes donnent les instructions Java permettant d'envoyer et de recevoir ces types de données :

```
// Création d'un flux de lecture de données binaires.
java.io.DataInputStream fluxIn = new java.io.DataInputStream(unInputStream);
// La lecture de données sur un flux java.io.DataInputStream
int unEntier = fluxIn.readInt ();
boolean unBooleen = fluxIn.readBoolean();
String uneChaine = fluxIn.readUTF();
// Création d'un flux d'écriture de données binaires.
java.io.DataOutputStream fluxOut = new java.io.DataOutputStream(unOutputStream);
// L'écriture de données sur un flux java.io.DataOutputStream
fluxOut.writeInt (unEntier);
fluxOut.writeBoolean(unBoolean);
fluxOut.writeUTF(uneChaine);
```

Pour plus d'informations sur la lecture et l'écriture de données, consultez la documentation en ligne des classes `java.io.DataInputStream` et `java.io.DataOutputStream`.

4. Pour aller plus loin

Le protocole de communication entre les processus précédemment défini ne permet de gérer qu'un seul carnet d'adresses par serveur. Modifiez ce protocole de communication afin de permettre à un serveur de gérer plusieurs carnets d'adresses. L'idée sous-jacente est de transporter dans les requêtes un identifiant désignant le carnet sur lequel on désire appliquer une opération. Modifiez ensuite votre serveur et votre client en conséquence. Ensuite, définissez le protocole de communication qui permettrait à un processus client de créer, supprimer, rechercher et lister les carnets d'adresses présents dans un serveur. Il faut définir une nouvelle interface ListeRepertoires, implanter cette interface dans le serveur, implanter une nouvelle interface graphique et les classes gérant le protocole. Ajoutez une phase d'authentification : seuls les utilisateurs déclarés dans le serveur pourront manipuler les répertoires.

Attention, soignez la conception de votre client. On doit pouvoir l'utiliser depuis une autre classe Java, ce que vous aurez à faire lors du prochain TP. Vous avez à rendre le client et le serveur pour la dernière version du protocole.