Examen Java et Systèmes Distribués (1h45)

Exercice 1 – Questions de cours (8pts)

Répondre aux questions suivantes :

1- Expliquer la différence entre un middleware MOM et RMI.

Dans le cadre de RMI,

2- Citer les trois composants élémentaires de l'architecture. Donner, respectivement, le rôle de chaque composant
3- Donner graphiquement l'architecture élémentaire en spécifiant les interactions, dans l'ordre, entre ses différents composants
4- Dans le processus de développement d'une application RMI, il existe deux niveaux de compilation. Donner ces deux niveaux en expliquant leur utilité

Exercice 2 – Sockets et RMI (12pts)

Nous souhaitons développer une application distribuée pour déterminer si un nombre entier **n** est premier ou pas.

Pour cela, nous rappelons que **n** est premier s'il vérifie que, $\forall (m^2 \leq n), alors: n\%m \neq 0$.

1- Ecrire, en java, une méthode **boolean EstPremier(int n)** permettant de dire si un nombre est premier ou pas. (2pts)

Nous souhaitons distribuer l'appel à cette méthode **EstPremier** avec le comportement suivant :

- Côté client :
  - Un affichage demandera de donner un entier
  - L'utilisateur saisira un entier
- Ensuite, côté serveur :
  - Le serveur affichera l'entier saisi
  - Il fera appel à la méthode EstPremier et renverra le résultat au client
- Ensuite, côté client
  - Le client affichera un message disant si le nombre est premier ou pas en se basant sur le retour du serveur

Partie Sockets :

2- En vous inspirant du code en Annexe 1, donner les codes sources des éléments de l'application en considérant la technologie java des sockets (2pts)

3- Donner, dans l'ordre, les étapes de compilation et d'exécution (2pts)

Partie RMI

4- En vous inspirant du code en Annexe 2, donner les codes sources des éléments de l'application en considérant la technologie java RMI (2pts)

5- Donner, dans l'ordre, les étapes de compilation et d'exécution (2pts)

6- Modifier le code du client pour faire 10 invocations successives. (1pt)

7- Dire si le reste du code doit être modifié ou pas en argumentant votre réponse. (1pt)

Annexe 1

- **Fichier Client.java**

```
public class Client
{
public static void main(String args[]) throws Exception
{
InetAddress IA=InetAddress.getLocalHost();
Socket S=new Socket(IA,Integer.parseInt(args[0]));
BufferedReader BR = new BufferedReader(new InputStreamReader(S.getInputStream()));
PrintWriter pred = new PrintWriter( S.getOutputStream());
...
S.close();
}}
```

- **Fichier Serveur.java**

```
public class Serveur
{
public static void main(String args[]) throws Exception
{
ServerSocket SS= new ServerSocket(Integer.parseInt(args[0]));
Socket S=SS.accept();
BufferedReader BR = new BufferedReader(new InputStreamReader(S.getInputStream()));
PrintWriter PW = new PrintWriter(S.getOutputStream());
...
S.close();}}
```

Annexe 2

- **Fichier HelloInt.java**

```
public interface HelloInt extends Remote
{
public void SayIt() throws RemoteException;
}
```

- **Fichier Hello.java**

```
public class Hello extends UnicastRemoteObject implements HelloInt {
                    public Hello() throws RemoteException {
                    super();
                    }
public void SayIt()throws RemoteException {
System.out.println("bonjour");
}
}
```

- **Fichier Client.java**

```
public class Client{
public static void main(String[] args)
{
try {
        Remote remote_h = Naming.lookup("rmi://localhost:1099/hello");
        if (remote_h instanceof HelloInt)
                {
                        ((HelloInt) remote_h).SayIt();
                }
}
catch (Exception e) {
}
}}
```

- **Fichier Serveur.java**

```
public class Serveur{
public static void main(String[] args) throws Exception {
Hello h = new Hello();
Naming.bind("rmi://localhost:1099/hello",h);
System.out.println("C'est fait!");
}}
```